ISSN 0135-1710

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Ministry of Education and Science of Ukraine
Kharkiv National University of Radio Electronics

Автоматизовані системи управління та прилади автоматики

Management Information System and Devices

Випуск 187

Всеукраїнський міжвідомчий науково-технічний збірник

Заснований 1965 р.

Затверджений до друку
Науково-технічною радою
Харківського національного університету радіоелектроніки
(протокол № 11 від 10 грудня 2025 р.)

ЗАСНОВНИК
Харківський національний університет радіоелектроніки

АДРЕСА РЕДАКЦІЇ:
Україна, 61166, м. Харків, проспект Науки, 14
Інформаційний сайт: https://asu-pa.nure.ua
E-mail редколегії: asu@nure.ua

Issue 187

All-Ukrainian Interdepartmental Scientific and Technical Collection

Founded in 1965

Approved for publication
by the Scientific and Technical Council
of the Kharkiv National University of Radio Electronics
(Protocol No. 11 d.d. 10 December 2025)

ESTABLISHER
Kharkiv National University of Radio Electronics

EDITORIAL OFFICE ADDRESS:
14 Nauky Ave., Kharkiv, Ukraine, 61166
Information site: https://asu-pa.nure.ua
E-mail of the editorial board: asu@nure.ua

Ідентифікатор медіа R30-03874
Витяг з реєстру суб'єктів у сфері медіареєстрантів від 25.04.2024 № 1410

Журнал включено до Переліку наукових фахових видань України, в яких можуть публікуватися результати дисертаційних робіт на здобуття наукових ступенів доктора й кандидата наук, наказом Міністерства освіти і науки України від 16.07.2018 № 775 (додаток 7)

Харків – 2025

ЗМІСТ

CONTENTS

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

INTELLIGENT INFORMATION TECHNOLOGY

Влах-Вигриновська Г. І., Кромкач В. О.
Вибір методики оцінювання точності для завдань аналізу статичних сцен на основі згорткових нейронних мереж (UA) 5

Vlakh-Vyhrynovska H., Kromkach V.
Choosing an accuracy assessment method for static scene analysis tasks based on convolutional neural networks

Гулевич М. В.
Оцінювання ефективності методу формування тестових сценаріїв для C++ бібліотек на основі агента з Q-навчанням (EN) 20

Hulevych M.
Evaluation of the effectiveness of the test scenarios forming method for C++ libraries based on a Q-learning agent

Гребеннік І. В., Коваленко О. А.
Адаптивна нейронечітка система виведення для сортування об'єктів поштових відправлень у конвеєрному потоці (UA) 47

Grebennik I., Kovalenko O.
Adaptive neuro-fuzzy inference system for sorting postal items in a conveyor stream

Єнгаличев С. О., Семенов С. Г.
Експериментальні дослідження методу планування задач у розподілених інформаційних системах з огляду на невизначеність метаданих (EN) 63

Yenhalychev S., Semenov S.
Experimental studies of the task planning method in distributed information systems taking into account metadata uncertainty

Канарський Є. О., Орєхов О. О.
Експертне оцінювання критеріїв якості людино-машинних інтерфейсів доповненої реальності (EN) 87

Kanarskyi Y., Orekhov O.
Expert evaluation of quality criteria for human-machine interfaces in augmented reality

Лапін М. О., Бохан К. О.
Навчання за кількома прикладами (*few-shot*) графової моделі нейронної мережі без використання зворотного поширення помилки (EN) 103

Lapin M., Bokhan K.
Few-shot learning of a graph-based neural network model without backpropagation

Слуцкін М. В., Вовк О. В.
Аналіз інформаційних технологій управління комунікаціями поліграфічного підприємства (EN) 123

Slutskin M., Vovk O.
Analysis of information technologies for communication management in a printing company

Філатов В. О., Черненко М. В.
Підтримка завдань користувача в інформаційних системах на основі агентних технологій (EN) 142

Filatov V., Chernenko M.
User task support in information systems based on agent technologies

Шматко О. В., Гамаюн І. П., Коломійцев О. В.
Гібридна модель машинного навчання для класифікації програмних помилок у хмарних *SaaS*-застосунках (EN) 156

Shmatko O., Gamayun I., Kolomiitsev O.
Hybrid machine learning model for classifying software bugs in SaaS cloud applications

КОМП'ЮТЕРНІ СИСТЕМИ, МЕРЕЖІ ТА ВБУДОВАНІ ТЕХНОЛОГІЇ

COMPUTER SISTEMS, NETWORKS AND BUILT-IN TECHNOLOGIES

Главчев М. І., Главчев Д. М., Панченко В. І.
Дослідження ефективності впровадження *open-source*-рішень для моніторингу й балансування навантаження в мікросервісних застосунках (EN) 182

Glavchev M., Hlavchev D., Panchenko V.
Evaluating the effectiveness of open-source solutions for monitoring and load balancing in microservice applications

ПРОГРАМНІ СИСТЕМИ ТА БЕЗПЕКА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ | **SOFTWARE SYSTEMS AND INFORMATION TECHNOLOGY SECURITY**

Абакумов А. І., Харченко В. С. Комбінований метод оцінювання безпеки кіберактивів БпЛА з використанням IMECA-процедур і тестування на проникнення (EN) | ***Abakumov A., Kharchenko V.*** Combined method of uav cyber assets security assessment by use of procedures imeca and penetration testing — 200

Петренко О. С., Петренко О. Є., Бідун А. К., Островський З. Н. Оцінювання рівня та пріоритизації багатопараметричних загроз із використанням алгоритму нечіткої логіки Мамдані першого типу (EN) | ***Petrenko O., Petrenko O., Bidun A., Ostrovskyi Z.*** Model for assessing the level and prioritizing multi-parameter threats using the Mamdani fuzzy logic algorithm of the first type — 220

Прокопович-Ткаченко Д. І., Рибальченко Л. В., Черкаський О. В., Черкаський Д. О., Зубченко Н. С. Ітераційні фрактальні відображення як засіб посилення ентропії та надійності процесів формування криптографічних ключів (UA) | ***Prokopovych-Tkachenko D., Rybalchenko L., Cherkaskyi O., Cherkaskyi D., Zubchenko N.*** Iterative fractal mappings as a means of enhancing entropy and reliability of cryptographic key generation processes — 234

Чуприна А. С., Репіхов В. М. Опорна модель превентивного супроводження програмного забезпечення (EN) | ***Chupryna A., Repikhov V.*** Reference model for preventive software maintenance — 254

Штангей С. В., Мельнікова Л. І., Марчук А. В., Лінник О. В., Гребенюк Є. А. Адаптація алгоритмів багатокласової класифікації для виявлення типу мережевої атаки в маршрутизаційних протоколах з використанням структурованих баз даних (EN) | ***Shtangey S., Melnikova L., Marchuk A., Linnyk O., Hrebeniuk Y.*** Adaptation of multiclass classification algorithms for identifying network-attack types in routing protocols using structured databases — 278

СИСТЕМНЕ МОДЕЛЮВАННЯ ТА ОБЧИСЛЮВАЛЬНІ МЕТОДИ | **SISTEM MODELING AND COMPUTATIONAL METHOD**

Ковальчук Д. А., Жила О. В. Геометрія вимірювань, моделі сигналів і алгоритми відновлення радіозображень в радарах із синтезованою апертурою, що використовують безперервні ЛЧМ-сигнали (EN) | ***Kovalchuk D., Zhyla O.*** Measurement geometry, signal models and agorithms for radio image recovery in synthesized aperture radars using continuous LFM signals — 299

Пащенко Р. Е., Марюшко М. В. Вибір розмірів "вікна" в процесі розрахунку фрактальних розмірностей космічних знімків сільськогосподарських земель (EN) | ***Pashchenko R., Mariushko M.*** Selection of "window" sizes when calculating fractal dimensions of agricultural land space images — 315

Сітніков В. І., Лященко В. О. Методи зберігання й оброблення великих даних у завданнях виявлення дезінформації (EN) | ***Sitnikov V., liashchenko V.*** Methods for efficient big data storage and processing in disinformation detection tasks — 324

АЛФАВІТНИЙ ПОКАЖЧИК — 338 — **ALPHABETICAL INDEX**

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

УДК 004.8 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.005

Г. Влах-Вигриновська, В. Кромкач

ВИБІР МЕТОДИКИ ОЦІНЮВАННЯ ТОЧНОСТІ ДЛЯ ЗАВДАНЬ АНАЛІЗУ СТАТИЧНИХ СЦЕН НА ОСНОВІ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Об’єктом вивчення є методики кількісного оцінювання точності й надійності прогнозів згорткових нейронних мереж у завданнях аналізу статичних сцен, зокрема семантична сегментація та монокулярне метричне оцінювання глибини. Використані теоретичні, аналітичні та емпіричні наукові **методи дослідження**: порівняльний аналіз, синтез, систематизація, експериментальне моделювання тощо. **Актуальність роботи** зумовлена тим, що традиційні метрики оцінювання точності не завжди беруть до уваги особливості завдань аналізу статичних сцен – дисбаланс класів, локалізацію та малі об’єкти, шум чи змінне освітлення. Це знижує точність результатів і потребує впровадження гібридних метрик, граничноорієнтованих і метрик кількісного оцінювання невизначеності (UQ) для забезпечення надійності й безпеки систем. **Метою дослідження** є обґрунтування й вибір найбільш ефективної та доцільної методики оцінювання точності для завдань аналізу статичних сцен на основі згорткових нейронних мереж способом порівняння й систематизації наявних метрик, аналізу їх переваг і обмежень у різних класах завдань і розроблення інтегрованого фреймворку для підвищення якості оцінювання. Для досягнення окресленої мети необхідно виконати такі завдання: провести порівняльний аналіз традиційних метрик; дослідити сучасні підходи й вибір релевантних метрик і протоколів для конкретних класів завдань; розробити концептуальний гібридний фреймворк, що забезпечує повну валідацію моделі, зважаючи на перекриття, геометричну точність і калібрування впевненості. Унаслідок дослідження сформульовано висновки. Для аналізу статичних сцен оптимально комбінувати такі метрики: *accuracy* й *F*1 – з метою класифікації, *IoU* і *mAP* – для детекції. Найбільш ефективні – *mAP* для складних сцен і для детекції малих об’єктів. Запропоновано гібридний фреймворк, що забезпечує повну валідацію моделі, покриваючи загальний об’єм, геометричну якість і надійність. Цей фреймворк поєднує гранично-орієнтовані метрики для забезпечення геометричної точності та методологію кількісного оцінювання невизначеності для калібрування впевненості та локалізації помилок. Це розв’язує проблему невідповідності між високою точністю моделей та обмеженістю стандартних метрик валідації. Перехід до *Boundary IoU* та метрик відстані, зокрема *Hausdorff Distance*, забезпечить масштабно-збалансовану та значно вищу чутливість до помилок на контурах, слугуватиме інструментом для виявлення катастрофічних локальних геометричних відхилень. Концептуальний фреймворк стимулює розроблення більш надійних і точних архітектур ЗНМ.

Ключові слова: комп’ютерний зір; семантична сегментація; детекція об’єктів; згорткові нейромережі; метрики оцінювання.

1. Вступ

Швидкий розвиток архітектур згорткових нейронних мереж (ЗНМ) і їх широке застосування в завданнях аналізу зображень (семантична сегментація, детекція, класифікація сцен) призвів до ситуації, коли результати різних робіт важко порівнювати через розбіжності в наборах даних, метриках і протоколах оцінювання. Автори сучасних досліджень 2022–2025 рр. наголошують на необхідності стандартизації підходів до

оцінювання, інтеграції метрик для калібрування й оцінювання невизначеності, а також у *cross-dataset*-бенчмарках, що моделюють реальні доменні зсуви.

Точність оцінювання відіграє важливу роль, оскільки статичні сцени часто містять складні елементи, такі як малі об'єкти, шум або варіативне освітлення. Проблема полягає в тому, що традиційні метрики оцінювання, наприклад загальна точність, не завжди беруть до уваги особливості завдань, як-от дисбаланс класів або локалізацію об'єктів.

Згідно з останніми тенденціями перехід від класичних метрик до спеціалізованих, зокрема *mAP* та *IoU*, стає необхідним для об'єктивного порівняльного оцінювання моделей ЗНМ (*CNN*). Це особливо актуально в умовах швидкого розвитку глибокого навчання, де нові архітектури вимагають адаптованих методів оцінювання для забезпечення високої продуктивності й узагальнення.

Особливо високі вимоги до точності висуває монокулярне метричне оцінювання глибини (*MMDE*). Тоді як традиційні методи часто прогнозують лише відносну глибину, *MMDE* прагне до отримання карт глибини з абсолютною шкалою. Цей перехід є необхідним для забезпечення геометричної узгодженості, що дає змогу надійно розгортати системи в реальному світі без додаткового калібрування. З огляду на критичні застосування, де висока точність і надійність є життєво важливими (наприклад, у самокерованих транспортних засобах або медичній діагностиці), методики оцінювання продуктивності ЗНМ мають бути не менш досконалими, ніж самі моделі [1].

2. Аналіз літературних джерел і постановка проблеми дослідження

Останні дослідження щодо оцінювання *CNN* для аналізу статичних сцен зосереджені на вдосконаленні метрик, зважаючи на особливості завдань, зокрема на детекцію малих об'єктів у складних сценах чи класифікацію зображень. Наприклад, у роботі [2], присвяченій детекції малих цілей у складних статичних сценах, автори пропонують мережу *IIHNet*, оцінюючи її за допомогою прецизії, повноти, *mAP* і *AP*50, демонструючи покращення на 20–70 %, якщо порівнювати з базовими моделями, такими як *Faster RCNN*.

Аналогічно в дослідженні впливу архітектурних факторів *CNN* на класифікацію зображень віддаленого зондування впроваджено метрики загальної точності (*OA*), кількості параметрів та *FLOPs*, показано, що збільшення глибини мережі покращує семантичне навчання, але надмірна глибина знижує *OA* [3].

Серед загальних метрик для *CNN* в завданнях аналізу зображень виокремлюють *accuracy, precision, recall, F1-score, mAP* і *IoU* як основні для класифікації, детекції та сегментації.

Літературний огляд точності в *CNN* для віддаленого зондування виявив перехід від традиційних метрик (*OA, Kappa*) до *DL*-специфічних, таких як *F1-score* (використовується в 59 % досліджень для бінарних завдань) та *mAP* (29 %), з акцентом на *P-R*-криві для завдань з порогами. У дослідженні класифікації сцен на датасеті *Places2* застосовується *top*-5 *accuracy* як основна метрика через неоднозначність міток, з *ResNet*-34 досягаючи 38.57 % *top*-1 *accuracy* [4–6]. Дослідники візуальних методів пояснюваності *CNN* упроваджують нові метрики, зокрема *Average Drop* та *Insertion / Deletion scores*,

для оцінювання правильності пояснень, пов’язаних із точністю моделі в завданнях класифікації зображень. Унаслідок вивчення останніх наукових праць можемо виокремити детальні дослідження калібрування та невизначеності, розвідки, що аналізують помилки метрик і їх інтерпретацію в реальних завданнях, серед яких огляди з калібрування [7], великі систематичні дослідження невизначеності та *UQ* [8], якості наборів даних і нових *ReM*-версій *COCO* [9], а також статті, присвячені помилковим інтерпретаціям метрик у зображеннях. Загалом тенденції 2022–2025 рр. вказують на інтеграцію метрик пояснюваності з традиційними для кращого розуміння поведінки мереж.

У сфері семантичної сегментації постійний розвиток енкодер-декодер архітектур спрямований на підвищення точності, особливо на стиках об’єктів. Дослідники розробляють методи, які посилюють внутрішню узгодженість об’єктів і забезпечують "загострення границь". Існують також підходи, що поєднують глибокі ЗНМ з умовно випадковими полями (*CRF*) для ефективного захоплення контекстуальних зв’язків між семантичними ознаками й ознаками глибини, що демонструє прагнення до покращення точності на рівні моделювання [10].

У контексті оцінювання глибини прогрес від відносної до абсолютної метричної глибини (*MMDE*) є вирішальним для забезпечення геометричної узгодженості, необхідної для 3D-реконструкції та навігації. Оскільки ці завдання вимагають високоточного просторового оцінювання, методи валідації мають забезпечувати виявлення будь-яких геометричних неточностей, що можуть призвести до збоїв у реальних системах [11].

Основний недолік полягає в *boundary un-awareness* (нечутливість до границь). Цей ефект зумовлений тим, що різниця в точному позиціюванні або формі границі несуттєво впливає на загальний бал, доки площа перекриття залишається незмінною. Традиційні метрики не здатні адекватно оцінювати моделі, оптимізовані для роботи зі складними структурами й дрібними гранулярними елементами.

Недосконаліть традиційних метрик перекриття для об’єктивного оцінювання геометричної точності ЗНМ, поряд із зростанням потреби в кількісному оцінюванні надійності прогнозу, формує ключову проблему дослідження. Необхідність удосконалення методик оцінювання точності аналізу статичних сцен на основі ЗНМ визначається двома основними напрямами, які потрібно виконати одночасно, а саме:

- розроблення або адаптація метрик, які забезпечують чутливість до граничних помилок і топології, водночас замінити або доповнити обмежені метрики перекриття;
- інтеграція методології кількісного оцінювання невизначеності (*UQ*) для оцінювання надійності та калібрування впевненості.

3. Мета й завдання дослідження

Метою цієї роботи є обґрунтування та вибір найбільш ефективної та доцільної методики оцінювання точності для завдань аналізу статичних сцен на основі згорткових нейронних мереж способом порівняння та систематизації наявних метрик, аналізу їх переваг і обмежень у різних класах завдань і розроблення інтегрованого фреймворку для підвищення якості оцінювання.

Для досягнення окресленої мети необхідно розв’язати такі завдання:

- провести порівняльний аналіз переваг і обмежень ключових метрик, таких як *accuracy, precision, recall, F1-score, IoU* та *mAP*, у контексті різних архітектур *CNN*;

- дослідити сучасні підходи й обрати релевантні метрики й протоколи для конкретних класів завдань;

- розробити концептуальний гібридний фреймворк, що забезпечує повну валідацію моделі з огляду на перекриття, геометричну точність і калібрування впевненості.

4. Виклад основного матеріалу

Аналіз статичних сцен за допомогою *CNN* передбачає завдання класифікації, сегментації та детекції об’єктів, де оцінка точності є критичною. Основні методики оцінювання можна поділити на метрики, орієнтовані на класифікацію (наприклад, *accuracy, precision, recall, F1-score*), і метрики, орієнтовані на локалізацію (наприклад, *Intersection over Union* (*IoU*) та *mean Average Precision* (*mAP*)) з адаптаціями для конкретних архітектур *CNN*.

Рис. 1. Метрики оцінювання точності

Традиційні метрики достатні для збалансованих наборів даних у завданнях класифікації, проте незбалансовані або просторово складні сцени в статичному аналізі вимагають гібридних протоколів, що містять як глобальні, так і попіксельні оцінки [12]. *Accuracy* вимірює частку правильних прогнозів серед усіх зразків, забезпечуючи простий глобальний індикатор продуктивності. Її перевага полягає в простоті та інтерпретації для збалансованих завдань класифікації з використанням архітектур, таких як *ResNet*, де вона ефективно оцінює загальну надійність моделі в маркуванні статичних сцен (наприклад, міських середовищ). Однак *accuracy* дає збій у незбалансованих наборах даних, типових

для аналізу сцен, де домінантні класи (наприклад, фонові пікселі) завищують показники, маскуючи невдачі на рідкісних об'єктах (як приклад, пішоходи).

Precision (TP/(TP + FP)) і *Recall* (TP/(TP + FN)) розв'язують окреслені проблеми, зосереджуючись на продуктивності позитивного класу. Зокрема *precision* перевершує в мінімізації помилкових тривог у завданнях виявлення з *YOLO* або *Faster R-CNN*, тоді як *recall* забезпечує всебічне охоплення елементів сцени. Обмеження передбачають чутливість до порогів упевненості та незбалансованості класів, що може призводити до компромісів у реальному часі статичного виявлення [13].

F1-score – це гармонійне середнє між *precision* та *recall*, що балансує їх, використовується в 40–59 % досліджень для мультикласових завдань. Метрика *F1-score* пропонує стійкість для незбалансованих статичних сцен у класифікаторах або детекторах *CNN* і є особливо корисною для оцінювання варіантів *U-Net* у семантичній сегментації, де коефіцієнт *Dice* (еквівалент *F1* для бінарних випадків) кількісно оцінює перетин, досягаючи високих показників у наборах даних міського керування. Однак *F1-score* ігнорує просторові нюанси, обмежуючи його корисність в архітектурах, що вимагають точності меж, таких як *CNN* на базі розширених конволюцій (наприклад, *DeepLab*).

IoU оцінює просторовий перетин (перетин / об'єднання) між передбаченими й реальними регіонами, що є критичним для сегментації в статичних сценах. Обмеження передбачають нечутливість до помилок меж і помилкових позитивів поза регіонами перетину, роблячи *IoU* менш ідеальним для завдань з пріоритетом на виявлення.

mAP агрегує середню точність по класах та порогах *IoU*, перевершуючи у виявленні об'єктів з *YOLOv8*, де фіксує варіації масштабів у статичних сценах (наприклад, *mAP* 0.88 у порівняльних дослідженнях). Недоліком *mAP* є обчислювальна інтенсивність і залежність від вибору порогу, що може недооцінювати моделі в розріджених сценах.

У табл. 1 систематизовано інформацію про метрики, які найчастіше використовуються, їх переваги й недоліки.

Таблиця 1. *Порівняльна характеристика традиційних метрик*

Метрика	**Застосування**	**Переваги**	**Недоліки**	**Рекомендоване застосування в архітектурах CNN**
Accuracy	Класифікація	Проста, інтуїтивна; добре для збалансованих даних	Вразлива до незбалансованості; маскує помилки в рідкісних класах	*ResNet* для класифікації сцен (наприклад, *ADE20K*)
Precision	Детекція	Мінімізує помилкові позитивні; корисна для уникнення помилкових тривог	Ігнорує помилкові негативні; залежить від порогу	*YOLO/Faster R-CNN* для виявлення в спостереженні

Кінець таблиці 1

Метрика	**Застосування**	**Переваги**	**Недоліки**	**Рекомендоване застосування в архітектурах CNN**
Recall	Детекція	Забезпечує виявлення всіх об'єктів; критична для повноти	Може призводити до багатьох помилкових позитивних	*U-Net* для сегментації в медичному аналізі
F1-score	Сегментація	Балансує *precision* та *recall*; стійка до незбалансованості	Ігнорує просторові деталі; припускає рівну важливість	Гібридні моделі для імбалансованих наборів
IoU	Локалізація	Вимірює просторову точність; стійка до імбалансу класів	Нечутлива до помилок меж; не для класифікації	*HRNet/PSPNet* для семантичної сегментації
mAP	Комплексні сцени	Агрегує по класах та порогах; всебічна для багатокласових завдань	Обчислювально складна; залежить від вибору *IoU*	*YOLOv8* для виявлення об'єктів у статичних сценах

Вибір метрик також залежить від архітектури *CNN*, яка використовується для аналізу статичних сцен. Наприклад, для завдань класифікації зображень, де потрібно визначити, до якого класу належить зображення, часто використовують *accuracy, precision, recall* та *F1-score*.

Для завдань сегментації зображень, де потрібно розділити зображення на різні ділянки, застосовують *IoU* та інші метрики, основані на піксельній точності.

Для завдань детекції об'єктів, де потрібно виявити та локалізувати об'єкти на зображенні, використовується *mAP*.

Для класифікації в статичних сценах (наприклад, маркування сцен з *ResNet*) найбільш релевантні *accuracy* та *F1-score*, доповнені *precision/recall* для імбалансу; протоколи передбачають крос-валідацію на наборах даних, наприклад *ADE20K*. Завдання виявлення об'єктів (наприклад, *YOLO* на наборах типу *COCO* для статичних зображень) пріоритизують *mAP* з порогами *IoU*, даючи змогу аналізувати помилки через криві *precision-recall* для критичних застосувань, як-от спостереження.

Семантична сегментація (наприклад, *U-Net* на *CamVid*) рекомендує *mIoU* та *Dice/F1* з протоколами, що містять точність переднього плану для класів, орієнтованих на безпеку, та час інференсу для реального часу.

Гібридні протоколи, які поєднують *mAP/IoU* для паноптичних завдань, з'являються для всебічного розуміння сцен, забезпечуючи узгодження метрик з обмеженнями розгортання, наприклад *FPS* в автономних системах [14, 15].

У табл. 2 подано інформацію щодо використання метрик і протоколів оцінювання для різного класу завдань.

Таблиця 2. *Релевантні метрики та протоколи для конкретних класів завдань*

Клас завдання	**Рекомендовані метрики**	**Протоколи оцінювання**	**Приклади архітектур *CNN***
Класифікація зображень	*Accuracy, F1-score, Precision/Recall*	Крос-валідація, аналіз імбалансу на *ADE20K*	*ResNet, VGGNet*
Виявлення об’єктів	*mAP, IoU, Precision-Recall* криві	Агрегація по класах, пороги *IoU* 0.5–0.95 на *COCO*-подібних наборах	*YOLO, Faster R-CNN*
Семантична сегментація	*mIoU, F1/Dice, Foreground Accuracy*	Попіксельне оцінювання, час інференсу на *CamVid/Cityscapes*	*U-Net, HRNet*

Щоб об’єктивно оцінити якість контурів було розроблено метрики, зосереджені на геометричних властивостях границь, а не на об’ємному перекритті.

Boundary IoU (BIoU) є ключовим показником, спрямованим на подолання низької чутливості стандартного *IoU* до помилок границь, особливо для великих об’єктів. Замість того, щоб розглядати всі пікселі маски, *BIoU* обчислює перекриття-над-об’єднанням лише для пікселів, розташованих у граничному регіоні (*Boundary Region*).

Основні переваги *BIoU* полягають у його здатності надавати кількісний градієнт, який безпосередньо покращує якість граничної сегментації, а також у його збалансованій реакції на помилки незалежно від розміру об’єкта. *BIoU* зменшує упередженість, яку демонструє стандартний *IoU* щодо великих структур [16].

Метрики на основі відстаней, як-от *Hausdorff Distance* (*HD*), *Average Symmetric Surface Distance* (*ASSD*) і *Normalized Surface Distance* (*NSD*), забезпечують оцінку геометричної розбіжності між контурами. Синергетичне використання *BIoU* (як стабільного індикатора загальної якості контуру) та *HD* (як індикатора катастрофічних локальних помилок) гарантує повний контроль геометричної точності.

Для досягнення надійності в критичних системах оцінювання точності має бути доповнене оцінюванням впевненості моделі. *UQ* є обов’язковим елементом для валідації надійності та безпеки. Кількісне оцінювання невизначеності дає змогу встановити, наскільки добре ймовірнісні прогнози моделі (впевненість) відповідають її фактичній точності (калібруванню).

Надійність механізмів *UQ* здебільшого залежить від обраної метрики.

Серед ключових метрик калібрування та локалізації помилок можна виокремити такі:

Expected Calibration Error (ECE) – стандартна метрика калібрування – кількісно оцінює узгодженість між оголошеною впевненістю моделі та її фактичною точністю. Адаптація *ECE* для сегментації є необхідною для валідації надійності прогностичних імовірностей;

– *Predictive Entropy* (предиктивна ентропія) значно перевершує інші метрики *UQ*, як-от варіація чи взаємна інформація, з погляду як калібрування (*ECE-label*), так і здатності до локалізації помилок (*Uncertainty-Error overlap*). Висока ентропія в певному регіоні сцени чітко вказує, де саме прогноз є найменш надійним.

– *Area Under the Sparsification Error curve (AUSE)* оцінює ефективність самого механізму *UQ*; вимірює, наскільки успішно оцінки невизначеності можуть бути використані для ідентифікації та відкидання найбільш невпевнених пікселів або патчів, підвищуючи цим загальну надійність системи. Застосування *AUSE* підтверджує, чи є механізм *UQ* практично корисним для розроблення механізмів фільтрації та відмови.

5. Обговорення результатів дослідження

Комплексне оцінювання точності ЗНМ для аналізу статичних сцен вимагає паралельного використання кількох груп метрик. Запропоновано гібридний фреймворк, який забезпечить повну валідацію моделі, покриваючи загальний об’єм, геометричну якість та надійність:

Overlap (*IoU, Dice Score*) – базова оцінка загального покриття.

Boundary / Geometry (*BIoU, HD, ASSD*) – гарантування геометричної точності та виявлення граничних помилок.

Reliability / Calibration (*ECE, Predictive Entropy, AUSE*) – оцінювання впевненості та локалізації помилок.

Цей підхід розв’язує проблему узгодження цілей: моделі, які оптимізуються під функції втрат, що містять гранично-орієнтовані метрики (наприклад, *Generalized Surface Loss*), будуть належним чином винагороджені в процесі валідації за допомогою *BIoU* та *HD*.

BIoU забезпечує більш справедливу й масштабно-збалансовану оцінку продуктивності, що є критично важливим у роботі з гетерогенними сценами, де об’єкти сильно різняться за розміром.

Інтеграція *UQ*, особливо через метрики, які здатні до локалізації помилок, є необхідною для підвищення надійності систем.

Predictive Entropy завдяки своїй високій надійності дає змогу точно ідентифікувати ділянки сцени, де прогноз є найменш впевненим.

Предиктивна ентропія вимірює ступінь "чистоти" розподілу ймовірностей класу в кожному пікселі. Що вища ентропія, то більш невизначений прогноз моделі щодо того, до якого класу належить піксель. Ентропія ефективно використовується для локалізації помилок. Гібридний фреймворк оцінювання – це багатовимірний інструментарій, призначений для забезпечення повної картини продуктивності ЗНМ за межами простого відсотка перекриття. У табл. 3 відтворено взаємодію компонентів запропонованого гібридного фреймворку.

Таблиця 3. *Взаємодія компонентів запропонованого фреймворку*

Метрика	Що вимірює?	Вирішувана проблема	Ключова роль у фреймворку
IoU/Dice	Об’ємне перекриття	Базовий рівень точності	Установлення загального успіху
Boundary IoU (BIoU)	Точність контуру	Нечутливість *IoU* до границь	Геометрична якість, збалансована оцінка
Hausdorff Distance (HD)	Максимальна помилка	Виявлення локальних катастроф	Індикатор безпеки та наявності викидів
Predictive Entropy	Локальна невизначеність	Локалізація ненадійних зон	Забезпечення надійності, механізм відмови
ECE	Калібрування впевненості	Недовіра до ймовірностей	Валідація довіри до прогнозів моделі

Нижче наведений рекомендований протокол для репрезентативного оцінювання моделей аналізу статичних сцен.

1. Чітко задокументувати препроцесинг, аугментації та розмітку; зберегти *seed*-и й опублікувати код.

2. Обирати набір метрик відповідно до завдання: для сегментації – *mIoU + boundary F-score + Pixel Accuracy*; для детекції – *COCO mAP(0.5:0.95) + AP@0.5 + recall@K + analysis by object sizes*.

3. Побудувати довірчі інтервали (*bootstrapping*) для ключових метрик.

4. Провести *cross-dataset evaluation*, якщо мета – узагальнення в змінених доменах.

5. Оцінити калібрування (*ECE, NLL*) і в разі потреби застосувати методи калібрування (*temperature scaling*) або UQ (*ensembles, MC Dropout*).

6. Провести *robustness checks* (шум, змінене освітлення, геометричні трансформації) і, якщо необхідно, *adversarial robustness* аналіз.

Етапи оцінювання, їх основна мета й ключові метрики оцінювання моделей відповідно до рекомендованого протоколу подано в табл. 4.

Таблиця 4. *Етапи оцінювання, їх основна мета й ключові метрики оцінювання моделей*

№	Етап оцінювання	Основна мета	Ключові дії / метрики
1	**Відтворюваність та документація**	Забезпечення можливості повторного отримання результатів	* Документація: препроцесинг, аугментації, розмітка
			* Фіксація всіх *Seed*-ів (*NumPy, ML framework*)
			* Публікація Коду
2	**Релевантний набір метрик**	Об’єктивне оцінювання якості моделі відповідно до задачі	* Сегментація: *mIoU, Boundary F-score, Pixel Accuracy*
			* Детекція: *COCO mAP (0.5:0.95), AP@0.5, Recall@K, AP_S, AP_M, AP_L*
3	**Довірчі інтервали (*Bootstrapping*)**	Оцінювання статистичної значущості та варіативності результатів	* Побудова 95 % довірчих інтервалів для ключових метрик методом ***Bootstrapping***

Кінець таблиці 4

№	Етап оцінювання	Основна мета	Ключові дії / метрики
4	***Cross-Dataset Evaluation***	Оцінювання узагальнення моделі на нові домени	* Тестування моделі на **зовнішньому, *OOD*** *(Out-of-Domain)* наборі даних
5	**Калібрування та *UQ***	Оцінювання надійності передбачуваних імовірностей	* **Оцінювання:** *ECE (Expected Calibration Error), NLL*
			* **Методи (за потреби):** *Temperature Scaling, Ensembles, MC Dropout*
6	***Robustness Checks***	Оцінювання стійкості до природних і ворожих спотворень	* **Природна *Robustness*:** тестування на шум, змінене освітлення, геометричні трансформації
			* ***Adversarial Robustness*:** аналіз стійкості до ворожих атак (напр., *PGD*)

Цей протокол забезпечує чіткий і всебічний підхід до оцінювання моделей аналізу сцен за межами лише метрик точності (*accuracy / mAP*) і беручи до уваги статистичну значущість (*CI*), узагальнення (*cross-dataset*), калібрування (*ECE*) та стійкість (*robustness*).

Результати використання цього протоколу для оцінювання моделі детекції об’єктів *YOLOv8-L* на наборі *COCO* подано в табл. 5.

Таблиця 5. *Результати використання протоколу для оцінювання моделі детекції об’єктів*

№	Етап оцінювання	Деталі реалізації / метрики	Результати	Висновок
1	**Відтворюваність**	Модель: *YOLOv8-L*. Аугментації: *Mosaic, Flip, MixUp, HSV*	Фіксація Global_Seed = 101. Код навчання опубліковано	Висока відтворюваність
2	**Набір метрик**	*COCO mAP (0.5:0.95), AP@0.5, Recall@100*. Аналіз за розмірами.	***mAP:*** 49.8 %. ***AP@0.5*:** 68.2 %. ***AP_S (Small)*:** 31.5 %.	Добра загальна продуктивність, але слабкість до малих об’єктів
3	**Довірчі інтервали**	*Bootstrapping* (500 ітерацій) для *COCO mAP*	95 % *CI* для *mAP*: **[49.1 %, 50.5 %]**	Результат статистично стабільний
4	***Cross-Dataset***	Тестування на ***Open Images V6*** (*OOD*-набір)	*mAP* на *Open Images*: **38.9 %**. (падіння на 10.9 п.п.)	Модель чутлива до **зміни домену** (*Domain Shift*)
5	**Калібрування**	Оцінювання *ECE* та *NLL*. Застосування *Temperature Scaling*	***ECE* до:** 6.2%. **ECE після:** 4.1 %. ***NLL*:** 0.75	Початково перекалібрована, успішно виправлено ***Temperature Scaling***

Кінець таблиці 5

№	Етап оцінювання	Деталі реалізації / метрики	Результати	Висновок
6	***Robustness Checks***	Вплив шуму, зміни освітлення та *adversarial* атак	**Падіння *mAP*:** шуми (-6.7 %), контраст (-8.3 %), ***PGD Attack*** (-48.3 %)	Дуже вразлива до ***Adversarial Attacks*** і зміни контрасту

Приклад використання протоколу оцінювання для завдання семантичної сегментації з використанням моделі *U-Net* на аерофотознімках наведено в табл. 6.

Таблиця 6. *Результати використання протоколу для оцінювання моделі семантичної сегментації*

№	Етап оцінювання	Деталі реалізації / метрики	Результати	Висновок
1	**Відтворюваність**	Модель: *U-Net* (*VGG*-16 як кодувальник). Препроцесинг: нормалізація *RGB* до $[0, 1]$	*Seed: Global_Seed* = 888. Аугментації: випадкове обертання, випадкова зміна масштабу (0.8–1.2)	Забезпечено чітке документування архітектури та аугментацій
2	**Набір метрик**	*mIoU, Boundary F-score, Pixel Accuracy.* Аналіз за класами: вода, рослинність, пісок, будівлі	*mIoU* (загальний): 75.3 %. *Boundary F-score:* 70.1 %. *mIoU* (пісок): 89.5 %. *mIoU* (будівлі): 60.2 %	Хороша загальна продуктивність, але слабкість у сегментації будівель (менші та складніші форми)
3	**Довірчі інтервали**	*Bootstrapping* (700 ітерацій) для метрики *mIoU* (будівлі)	95 % *CI* для *mIoU* (будівлі): [58.5 %, 61.9 %]	Результат для складного класу є статистично значущим
4	***Cross-Dataset***	Тестування на OOD-наборі *Potsdam* (аерофотознімки з іншого міста / камери) з мапінгом класів	*mIoU* на *Potsdam*: 65.9 %. (падіння на 9.4 п.п.)	Модель чутлива до змін у кольоровій гамі та роздільній здатності зображень іншого домену
5	**Калібрування**	Оцінювання *ECE* та *NLL*	*ECE*: 5.1 %. *NLL*: 0.45	Модель має помірне перекалібрування. Необхідне пост-хок калібрування (напр., *Temperature Scaling*)
6	***Robustness Checks***	Тестування на: симуляція хмарності (зміни яскравості та контрасту), стиснення *JPEG* (висока якість)	Падіння *mIoU:* симуляція хмарності (-4.8 %), стиснення *JPEG* (-1.1 %)	Чутлива до затемнення / зміни контрасту, що імітує хмарну погоду

Цей приклад демонструє, як протокол може бути застосований для оцінювання моделі сегментації аерофотознімків, виявляючи її слабкості у сегментації малих об’єктів (наприклад, будівель, зелених насаджень тощо) та чутливість до погодних умов (хмарності).

6. Висновки й перспективи подальших досліджень

Порівняння методик демонструє, що для аналізу статичних сцен оптимально комбінувати метрики: *accuracy* та *F1* для класифікації, *IoU* і *mAP* для детекції. Найефективніша *mAP* для складних сцен і для детекції малих об’єктів.

Перспективи передбачають розроблення гібридних метрик з елементами пояснюваності для кращого розуміння помилок *CNN*.

Дослідження також підтвердило недоліки використання традиційних метрик перекриття (*IoU, Dice Score*) для об’єктивного оцінювання геометричної точності ЗНМ в аналізі статичних сцен через їх нечутливість до граничних помилок (*boundary un-awareness*s).

Удосконалення методик оцінювання вимагає переходу до гібридного фреймворку, основаного на двох аспектах:

- гранична точність – упровадження *Boundary IoU (BIoU)* та метрик відстані (*HD, ASSD*) забезпечить необхідну чутливість до помилок на контурах, що є життєво важливим для високоточних геометричних застосувань, таких як *MMDE*;
- надійність – інтеграція кількісного оцінювання невизначеності (*UQ*) з пріоритетом *Predictive Entropy* та *ECE*, є обов’язковою для валідації безпеки та калібрування впевненості моделі.

Запропонований гібридний фреймворк забезпечує більш повну й об’єктивну картину продуктивності ЗНМ, об’єднуючи оцінювання геометричної якості та калібрування впевненості. Упровадження гранично-орієнтованих метрик стимулює розроблення архітектур і функцій втрат, які безпосередньо покращують точність контурів, оскільки ці зусилля будуть належним чином винагороджені. Це підвищує загальну надійність систем комп’ютерного зору в критичних доменах.

Подальші дослідження можуть будуть присвячені адаптації метрик до реального часу й інтеграції з великими моделями, як *Vision Transformers*, розробленню гібридних функцій втрат, що безпосередньо оптимізують гранично-орієнтовані метрики (наприклад, *BIoU Loss* або *Loss*-функції на основі поверхневої відстані), для забезпечення максимальної узгодженості між навчанням і валідацією.

Перспективними також є дослідження та уніфікація *UQ*-фреймворків, зокрема розроблення метрик, які поєднують просторову локалізацію невизначеності з чутливістю до границь (*Boundary UQ*) для підвищення надійності оцінювання критичних об’єктів.

References

1. Aalst, J., Maruccio, F., Simoẽs R., Janssen, T., Wolterink, J., Ooijen, P., Brouwer, Ch. (2025), "Reliability of uncertainty quantification methods for deep learning auto-segmentation in head and neck organs at risk", *Physics in Medicine and Biology*, No. 20. DOI: https://doi.org/10.1088/1361-6560/ae110c
2. Fu, Y., Li, X., Hu, Z. (2021), "Small-Target Complex-Scene Detection Method Based on Information Interworking High-Resolution Network", *Sensors*, No. 21(15). DOI: https://doi.org/10.3390/s21155103
3. Chen, F., Tsou, J. (2022), "Assessing the effects of convolutional neural network architectural factors on model performance for remote sensing image classification: An in-depth investigation", *International Journal of Applied Earth Observation and Geoinformation*, No. 112. DOI: https://doi.org/10.1016/j.jag.2022.102865
4. Maxwell, A., Warner, T., Guillén, L. (2021), "Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies – Part 1: Literature Review", *Remote Sensing*, No. 13. DOI: https://doi.org/10.3390/rs13132450
5. Dugăeșescu, A., Florea, A. (2025), "Evaluation and analysis of visual methods for CNN explainability: a novel approach and experimental study", *Neural Computing and Applications*, No. 37, P. 14935–14970. DOI: https://doi.org/10.1007/s00521-025-11282-7
6. Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., Parmar, M. (2024), "A review of convolutional neural networks in computer vision", *Artificial Intelligence Review*, No. 57, P. 99. DOI: https://doi.org/10.1007/s10462-024-10721-6
7. Wang, Ch. (2023), "Calibration in Deep Learning: A Survey of the State-of-the-Art". DOI: https://doi.org/10.48550/arXiv.2308.01222
8. Gawlikowski, J., Tassi, C., et al. (2023), "A survey of uncertainty in deep neural networks. Artificial Intelligence Review". DOI: https://doi.org/10.48550/arXiv.2107.03342
9. Singh, Sh., Yadav, A., Jain, J., Shi, H., Johnson, J., Desai, K. (2024), "Benchmarking Object Detectors with COCO: A New Path Forward". DOI: https://doi.org/10.48550/arXiv.2403.18819
10. Arulananth, T., Kuppusamy, P., Ayyasamy, R., Alhashmi, S., Mahalakshmi, M., Vasanth, K., Chinnasamy, P. (2024), "Semantic segmentation of urban environments: Leveraging U-Net deep learning model for cityscape image analysis", *PLoS ONE*, No 19 (4). DOI: https://doi.org/10.1371/journal.pone.0300767
11. Zhang, J. (2025), "Survey on Monocular Metric Depth Estimation", *Computer Vision and Pattern Recognition*. DOI: https://doi.org/10.48550/arXiv.2501.11841
12. Nemavhola, A., Chibaya, C., Viriri, S. (2025), "A Systematic Review of CNN Architectures, Databases, Performance Metrics, and Applications in Face Recognition", *Information*, No 16 (2), P. 107. DOI: https://doi.org/10.3390/info16020107
13. Classification metrics guide. (2025), "Accuracy vs. precision vs. recall in machine learning: what's the difference?". DOI: https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall
14. Khan, S., Mazhar, T., et al. (2024), "Comparative analysis of deep neural network architectures for renewable energy forecasting: enhancing accuracy with meteorological and time-based features", *Discover Sustainability*, No. 5, P. 533. DOI: https://doi.org/10.1007/s43621-024-00783-5

15. Rayed, M., Islam, S., Niha, S., Jim, J., Kabir, M., Mridha, M. (2024), "Deep learning for medical image segmentation: State-of-the-art advancements and challenges", *Informatics in Medicine Unlocked*, No 47. DOI: https://doi.org/10.1016/j.imu.2024.101504
16. Cheng, B., Girshick, R., Dollár, P., Berg, A., Kirillov, A. (2021), "Boundary IoU: Improving Object-Centric Image Segmentation Evaluation", *EEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: https://doi.org/10.1109/CVPR46437.2021.01508

Received (Надійшла) 01.11.2025
Accepted for publication (Прийнята до друку) 09.12.2025
Publication date (Дата публікації) 28.12.2025

Відомості про авторів / About the Authors

Влах-Вигриновська Галина Іванівна – кандидат технічних наук, доцент, Національний університет "Львівська політехніка", доцент кафедри комп'ютеризованих систем автоматики Інституту комп'ютерних технологій, автоматики та метрології, Львів, Україна; e-mail: halyna.i.vlakh-vyhrynovska@lpnu.ua; ORCID ID: https://orcid.org/0000-0003-4429-1578

Кромкач Владислав Олександрович – Національний університет "Львівська політехніка", аспірант кафедри комп'ютеризованих систем автоматики Інституту комп'ютерних технологій, автоматики та метрології, Львів, Україна; e-mail: vlad.kromkach@gmail.com, ORCID ID: https://orcid.org/0009-0001-5608-5715

Vlakh-Vyhrynovska Halyna – PhD (Engineering Sciences), Associate Professor, Lviv Polytechnic National University, Associate Professor at the Department of Computerized Automation Systems, Institute of Computer Technologies, Automation and Metrology, Lviv, Ukraine.

Kromkach Vladyslav – Lviv Polytechnic National University, Postgraduate Student at the Department of Computerized Automation Systems, Institute of Computer Technologies, Automation and Metrology, Lviv, Ukraine.

CHOOSING AN ACCURACY ASSESSMENT METHOD FOR STATIC SCENE ANALYSIS TASKS BASED ON CONVOLUTIONAL NEURAL NETWORKS

The object of the study is the quantitative assessment of the accuracy and reliability of convolutional neural networks predictions in static scene analysis tasks, including semantic segmentation and monocular metric depth estimation. Theoretical, analytical and empirical scientific research **methods** were used: comparative analysis, synthesis, systematization, experimental modeling, and others. **The relevance of the research** is due to the fact that traditional accuracy assessment metrics do not always take into account the specifics of static scene analysis tasks - class imbalance, localization and small objects, noise or variable lighting. This reduces the accuracy assessment of the results and requires the implementation of hybrid metrics, boundary-oriented metrics and uncertainty quantification (UQ) to ensure the reliability and security of the systems. **The aim of the research** is to substantiate

and select the most effective and appropriate accuracy assessment method for static scene analysis tasks based on convolutional neural networks by comparing and systematizing existing metrics, analyzing their advantages and limitations in different classes of tasks, and developing an integrated framework to improve the quality of assessment. To achieve this goal, the following tasks are solved in the article: to conduct a comparative analysis of traditional metrics; to study modern approaches and select relevant metrics and protocols for specific classes of tasks; to develop a conceptual hybrid framework that provides full model validation, taking into account overlap, geometric accuracy and confidence calibration. As a result of the study, the following conclusions were made: for the analysis of static scenes, it is optimal to combine the metrics: *accuracy* and *F1* – for classification, *IoU* and *mAP* – for detection. The most effective are *mAP* for complex scenes and for the detection of small objects. A hybrid framework is proposed that provides full validation of the model, covering the total volume, geometric quality and reliability. This framework combines boundary-oriented metrics to ensure geometric accuracy and the methodology of quantitative uncertainty assessment for confidence calibration and error localization. This solves the problem of the mismatch between the high accuracy of models and the limitations of standard validation metrics. The transition to *Boundary IoU* and distance metrics, in particular *Hausdorff Distance*, will provide scale-balanced and significantly higher sensitivity to errors on the contours, and will serve as a tool for detecting catastrophic local geometric deviations. The conceptual framework stimulates the development of more robust and accurate CNN architectures.

Keywords: computer vision; semantic segmentation; object detection; convolutional neural networks; evaluation metrics.

Бібліографічні описи / Bibliographic descriptions

Влах-Вигриновська Г. І., Кромкач В. О. Вибір методики оцінювання точності для завдань аналізу статичних сцен на основі згорткових нейронних мереж. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 5–19.
DOI: https://doi.org/10.30837/0135-1710.2025.187.005

Vlakh-Vyhrynovska, H., Kromkach, V. (2025), "Choosing an accuracy assessment method for static scene analysis tasks based on convolutional neural networks", *Management Information Systems and Devises*, No. 4 (187), P. 5–19. DOI: https://doi.org/10.30837/0135-1710.2025.187.005

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.415.2:004.415.53 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.020

M. Hulevych

EVALUATION OF THE EFFECTIVENESS OF THE TEST CASE FORMING METHOD FOR C++ LIBRARIES BASED ON A Q-LEARNING AGENT

Effective optimization of test cases (TC) is a prerequisite for improving the effectiveness of regression testing of C++ libraries. **The subject of the study** is methods for forming (optimizing) TC for C++ libraries. **The purpose of the work** is to evaluate the effectiveness of the method of forming TC for C++ libraries based on a Q-learning agent. **Research tasks:** to improve the mathematical model of a Q-learning agent to increase the efficiency of forming TC for C++ libraries in conditions of high sparsity of the state space of the Q-learning agent; to investigate the influence of the parameters of the improved Q-learning agent model on its behavior under such conditions; to consider the possibility of minimizing the formed TC by the method of their formation using the delta-debugging algorithm for TC minimization; to evaluate the effectiveness of the proposed method and compare it with known TC optimization methods**. Research methods.** The work uses the Monte Carlo Tree Search method, the classical mathematical model of Q-learning, the delta debugging algorithm for TC minimization, and the greedy algorithm for TC optimization. The effectiveness of the proposed method was evaluated on two open-source C++ libraries using statistical analysis of 100 mathematical simulations of the configurations of the methods under study. **Results achieved:** the effectiveness evaluation indicates that the proposed method provides the following average values of TC optimization effectiveness coefficients for C++ libraries: the coverage retention ratio is up to 1.225, the test suite compression ratio is up to 0.86, and the test suite execution time reduction ratio is up to 0.74. It has been established that, when compared with the greedy optimization algorithm, the delta debugging minimization algorithm, and the optimization method based on Monte Carlo Tree Search, the proposed method significantly improves the efficiency of TC formation (optimization) for C++ libraries. **Conclusions.** The improved mathematical model provides generalization of the experience of the Q-learning agent between similar test cases and increases the efficiency of their formation in conditions of high sparsity of the state space of the Q-learning agent. Thus, the results of evaluating the method of forming TC for C++ libraries based on a Q-learning agent confirm its feasibility in solving the problem of forming (optimization) of TC for C++ libraries, which makes it possible to reduce the length of TC in the test suite without loss of the branch coverage of the C++ library code being tested and to reduce the execution time of TS. Further research will be devoted to the formation of TC for C++ libraries based on a deep reinforcement learning agent.

Keywords: test case optimization; software testing; Q-learning; Q-table; delta debugging; code coverage; minimization; C++; agent; reinforcement learning.

Introduction

Software testing is a key element of software quality assurance. The testing process uses test suite (TS) consisting of test cases (TC) for interaction with the corresponding software interfaces of C++ libraries and their components.

TC for C++ libraries and C++ libraries are developed in parallel, reflecting the practical features of using the API (Application Programming Interface, API) of the tested code. They accumulate throughout the software development life cycle and gradually cover a significant part of the functionality, as well as being the main source for verifying the correct

behavior of the output software product when changes are made to the code base. As the volume of such test cases increases, the cost of regression testing increases in both time and computational terms.

C++ software libraries are used in large projects for image processing, file system operations, data analysis, etc. In such conditions, changes to the C++ library code lead to the need to re-execute a significant amount of TC in redundant TS to detect software regression defects.

The execution time of C++ library TS can be reduced by optimizing the TC in the original TS. Thus, the task of TC optimization (Test Case Optimization, TCO) without loss of coverage is a relevant task that allows reducing testing costs, improving the informativeness of TC execution results, and, as a result, ensuring the effectiveness of regression testing.

Previous studies have proposed a method for forming TC for C++ libraries based on a Q-learning agent, which reduces the size of original TS without loss of branch coverage of the code [1]. To justify the effectiveness of the proposed method, it is advisable to conduct a comparative analysis with relevant TC optimization methods.

Analysis of literature

Over the past decade, a number of review papers have been published that provide a fundamental understanding of existing TC optimization methods and criteria for evaluating their effectiveness. Systematic reviews [2–4] provide a basis for understanding the task of TC optimization, covering a broad classification of TC optimization methods, as well as the features of software environments for researching TC optimization methods.

The importance of preserving the semantics of the original test suite, forming adequate criteria for evaluating effectiveness, and adhering to the methodological foundations of researching TC optimization methods is emphasized separately.

The task of TS optimization consists of three interrelated tasks: TCS (Test Case Selection, TCS), TSP (Test Suite Prioritization, TSP), and TSM (Test Suite Minimization, TSM) [5]. The purpose of TCS is to determine a subset of TC that need to be re-executed after changes in the program code of the library being tested. The purpose of TSP is the ordering of test cases in order to maximize the speed of software defect detection. TSM removes redundant TC from the TS while maintaining testing efficiency, thereby reducing the time required to execute the TS.

Machine learning is used to optimize TS [6]. In particular, clustering algorithms are effectively used to reduce redundancy in TS and prioritize the TC [7].

They are capable of operating based on the performance characteristics of C++ library tasks, which makes them useful for optimizing TC for C++ libraries without formal specifications (documented requirements for C++ library behavior). Reinforcement learning can also be effectively applied in these conditions [8].

An important factor in evaluating the effectiveness of TC optimization methods is the classification into adequate and inadequate methods, as formulated in [9]. According to this classification, adequate (coverage-preserving) TC optimization methods are those that ensure the preservation of the full level of testing effectiveness relative to the original TS.

After optimization, an adequate method guarantees that all structural (behavioral) constructs of the software that were tested by the original test cases remain covered by the new test cases.

Thus, the adequacy characteristic is critically important in terms of the suitability of the TC optimization method for regression testing, where the loss of the test case's ability to detect defects in the software under test is unacceptable.

In contrast, non-coverage-preserving methods allow for a partial loss of testing effectiveness in order to achieve higher test case compression rates. Inadequate methods can significantly reduce the amount of test coverage by reducing the ability to detect software defects, makingthem unsuitable for TC optimization for highly reliable software systems or libraries.

The need for adequate TC optimization methods is also confirmed by an analysis of the peculiarities of testing C++ libraries [10], which focuses on the following aspects:

– the absence of formal API specifications in open-source C++ libraries, which increases the role of methods capable of working in such conditions;

– the need to analyze the dependencies between test case instructions during execution in order to optimize the TC effectively;

– the complexity of the execution environment of C++ software, which necessitates flexible methods for forming TC for C++ libraries.

The author's previous work [1] provides a thorough overview of TC optimization methods – delta debugging, dynamic slicing, integer linear programming (ILP) TC optimization methods, methods based on classification and clustering, as well as hybrid methods.

To expand the analytical context of the previous work and clarify the role of individual methods in modern TC optimization research, Table 1 provides a comparative description of recent studies of TC optimization methods in different classes of problems.

Table 1. *Comparative characteristics of studies of TC optimization methods in different classes of problems*

Research	Method class	Research features	Research results
ATM method [11]	Heuristic method of TSP based on evolutionary search	Representation of TC in the form of abstract syntactic trees, search based on TC similarity.	Detection of up to 0.82 software defects when executing 50 % of TS.
GA-based Test Suite Minimization (TSM) [12]	Minimization of TS based on a genetic algorithm	An analysis of the configuration of vehicle parameters based on a genetic algorithm was conducted.	The possibility of finding a balance between maintaining coverage and reducing the execution time of the TS has been demonstrated.
Metaheuristic Fault Detection [13]	Metaheuristics for detecting software defects.	An analysis was conducted on particle swarm optimization, ant colony optimization, cuckoo search, firefly algorithm, etc.	Ability to work effectively without prior knowledge of software behavior. Low reproducibility of results.

Continuation of the Table 1

Research	**Method class**	**Research features**	**Research results**
EA-based prioritization [14]	Evolutionary method of prioritizing TC.	An analysis of the effectiveness of the method was performed with an increase in the volume of input data in large projects.	Effective allocation of time for testing software modules, which increases the efficiency of software testing.
RL+GA Hybrid [15]	A hybrid method based on reinforcement learning and a genetic algorithm.	The possibility of using genetic algorithms to configure the input parameters of a Q-learning agent was investigated.	Acceleration of policy convergence when training an agent on pre-configured input data based on a genetic algorithm.
General Monte Carlo Tree Search (MCTS) study [16]	Monte Carlo Tree Search	An analysis of the applicability of the Monte Carlo method for various classes of problems, including modifications of the method and hybrid configurations, was conducted.	The method is highly effective, but special improvements are needed for different classes of problems.
Monte Carlo Tree Search input parameters fuzzing [17]	Prioritization of input data for training deep neural networks based on Monte Carlo Tree Search method	Testing model as a decision-making process. Investigation of large input data search spaces for TC.	The results prove the effectiveness of the method and show an increase of up to + 30 % in code coverage compared to basic methods.
Monte Carlo Tree Search + UCT [18]	Monte Carlo Tree Search method with upper confidence bound	A classical algorithm based on the Upper Confidence Bound (UCB) is proposed. A classical combination of the MCTS method with the UCB algorithm is proposed.	A fundamental algorithm that laid the foundation for modern methods based on Monte Carlo Tree Search.
Greedy TSM [19]	Greedy method of TS minimization	Research on minimization of TS based on a greedy algorithm depending on the size of input TS and test requirements.	The results obtained prove the possibility of compressing TS to 50–75 % of the original size while maintaining coverage.
Greedy TSM [20]	Greedy method of TSM	Proposed two-criteria optimization of TS based on a greedy algorithm.	The effectiveness of the algorithm has been proven. The results obtained indicate a significant compression of TS while maintaining test requirements.
Greedy TSM [21]	Greedy method of TSM	Analysis of the effectiveness of the method based on mutation testing.	The possibility of compressing TS by an average of 70 % with high software defect detection capability has been proven.
Greedy TSP [22]	Greedy method of prioritizing TS	Analysis of the application of the greedy method for prioritizing TS in regression testing.	The results show a significant reduction in TS while maintaining the ability to detect software defects.

Continuation of the Table 1

Research	Method class	Research features	Research results
Delta Debugging TSM [23]	Delta debugging method for minimizing TS.	A classic delta debugging algorithm, ddmin, is proposed.	Significant reduction of input data without losing the ability to reproduce defects.
Delta Debugging for Fault Localization [24]	Delta debugging method for software defect localization.	Application of DD for localization of test cases leading to reproduction of specified software defects.	Significant reduction in input data without losing the ability to reproduce defects.
Hierarchical Delta Debugging [25]	Hierarchical modification of the delta debugging method for minimizing TS.	Hierarchical version of DD for structured scenarios.	Reduction of the time required to perform TS minimization while maintaining the effectiveness of reduced test cases.
Probabilistic Delta Debugging [26]	Stochastic modification of the delta debugging method for TS minimization.	A proposed probabilistic model based on the delta debugging algorithm and the representation of code as an abstract syntax tree.	The method considers the syntactic relationships between elements and the results of previous tests, which makes it possible to reduce the average processing time by almost 27 % and reduce the size of the TS by 3.4 times compared to existing methods.

To evaluate the effectiveness of the proposed method of TC forming based on a Q-learning agent, it is advisable to use those basic methods that:

– do not require large training data sets or pre-trained models describing the expected behavior of the software;

– can be applied to optimize TC for testing C++ libraries.

Greedy TC optimization algorithms remain the most effective and stable for minimizing TS. Studies [19–22] show that TS minimization methods based on greedy algorithms provide a significant reduction in TS size (up to 50–75 %) while maintaining the ability to detect software defects and preserving testing efficiency.

TC optimization methods based on the delta debugging algorithm (classical admin, hierarchical and probabilistic modifications) [23–26] have stable results in the task of minimizing TC with an increase in the volume of input data. They ensure effective localization of minimal test cases subset while maintaining the ability to reproduce software defects. However, these methods only optimize the structure of existing test suite without forming new test cases at the API call level.

Search methods on the Monte Carlo tree, including those based on the UCT algorithm [17–19], combine research and the use of acquired knowledge, making them relevant for the formation of TC in large action spaces.

At the same time, the application of the Monte Carlo method of TC optimization for C++ libraries requires adaptation: mechanisms are needed to verify the validity of the formed TC, determine the rational depth of the search tree, and limit the input data space.

Other groups – evolutionary algorithms, metaheuristics, and hybrid methods [13–16] – demonstrate high efficiency on large input data space but are not effective in the task of optimizing TC for C++ libraries in the absence of a predefined software behavior model (without formal specifications).

Therefore, for a comparative assessment of the effectiveness of the proposed method, the following were selected:

– Greedy TC optimization algorithm;
– Delta debugging algorithm for TC minimization;
– Monte Carlo Tree Search method.

Thus, the choice of these methods as basic ones is justified from both a theoretical and practical point of view.

Purpose and objectives

The purpose of the study is to evaluate the effectiveness of the method of forming TC for C++ libraries based on a Q-learning agent.

To achieve this goal, the following tasks are set in the study:

1. To improve the mathematical model of the Q-learning agent to increase the efficiency of TC formation for C++ libraries in conditions of high sparsity of the state space of the Q-learning agent.

2. To investigate the influence of the parameters of the improved Q-learning agent model on the agent's behavior in such conditions.

3. To consider the possibility of minimizing the obtained TC using the proposed TC formation method based on the delta-debugging algorithm for TS minimization.

4. To evaluate the effectiveness of the proposed method in a unified TC formation environment, based on the original test suites of two open-source C++ libraries. Use the classic greedy TC optimization algorithm, the delta debugging TC minimization algorithm, and the TC optimization method based on Monte Carlo Tree Search as the main methods for comparison.

Main part

1. Classic Q-learning agent model

The task of forming TC for C++ libraries without API specifications can be formalized as a Markov Decision Process (MDP) [27], where an agent builds a test case step by step, choosing the next action from the permissible API space:

$$\langle S, A, P(s'|s,a), R(s,a), \gamma \rangle,$$

where S is the set of states corresponding to the state s; A is the set of permissible actions in the state s; $P(s'|s,a)$ is the probability function of transition to a new state after performing

the action a in the state s; $\mathrm{R}(\mathrm{s},\mathrm{a})$ is the reward function provided for performing an action a in the state s; $\gamma \in (0,1)$ is the discount factor.

A test case of length $t \in \mathrm{N}$, which is formed as follows:

$$\mathrm{TC}^{\mathrm{t}} = \mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_{\mathrm{t}}, \ \mathrm{a}_{\mathrm{i}} \in \mathrm{A}, \tag{1}$$

consists of a sequence of a C++ library API calls.

The classic Q-learning algorithm belongs to the class of reinforcement learning methods, which allow the agent to sequentially improve its action selection policy based on experience gained during an interaction with the environment. The agent's goal is to maximize the expected total reward by approximating the optimal action utility function:

$$\mathrm{Q}^{*}(\mathrm{s},\mathrm{a}) = \max_{\pi} \mathrm{E}\left[\sum_{\mathrm{t}=0}^{\infty} \gamma^{\mathrm{t}} \times \mathrm{R}(\mathrm{s}_{\mathrm{t}}, \mathrm{a}_{\mathrm{t}})\right], \tag{2}$$

where π is the action selection policy.

The formula considers the randomness of transitions between agent states by maximizing the mathematical expectation of the discounted sum of rewards. During training, Q-values are updated according to Bellman's rule:

$$\mathrm{Q}(\mathrm{s}_{\mathrm{t}}, \mathrm{a}_{\mathrm{t}}) \leftarrow \mathrm{Q}(\mathrm{s}_{\mathrm{t}}, \mathrm{a}_{\mathrm{t}}) + \alpha \times \left[\mathrm{R}(\mathrm{s}_{\mathrm{t}}, \mathrm{a}_{\mathrm{t}}) + \gamma \times \max_{\mathrm{a}'} \mathrm{Q}(\mathrm{s}_{\mathrm{t+1}}, \mathrm{a}') - \mathrm{Q}(\mathrm{s}_{\mathrm{t}}, \mathrm{a}_{\mathrm{t}})\right], \tag{3}$$

where $\alpha \in (0,1]$ is the learning rate multiplied by $\delta^{t} = [\ldots]$ – the TDE (Temporal Difference Error, TDE). The agent sequentially updates Q-values in the Q-table, learning to distinguish useful actions (which increase the coverage or efficiency of the test case) from uninformative or redundant ones.

The process of training an agent with reinforcement can be presented in the form of a diagram shown in Fig. 1.

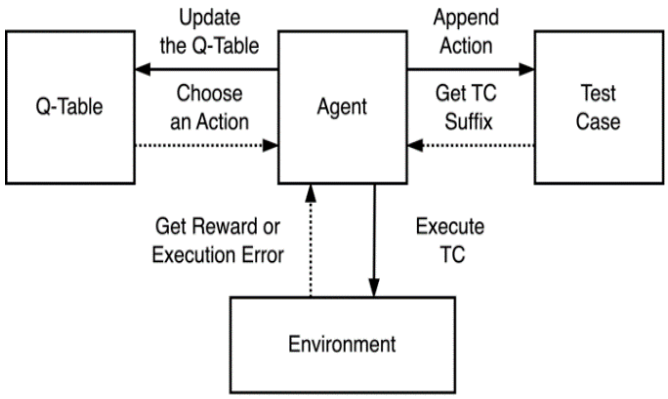


Fig. 1. Diagram of the reinforcement learning process for a Q-learning agent

The agent interacts with the environment sequentially, selecting actions based on the Q-table, updating it based on the results of the test case execution and the rewards received. Each interaction cycle includes selecting an action, forming a partial test case, executing the action in the environment, receiving a reward or execution error, and updating the corresponding Q-values in the Q-table.

The classic Q-learning agent model allows the agent to be gradually trained in effective TC construction strategies. However, the application of the classic Q-learning agent model to solve the TC forming (optimization) problem for C++ libraries has a number of limitations:

– *explosion of the state space dimension.* The space of all possible API call sequences grows exponentially depending on the complexity of the C++ library being tested. This approach leads to high sparsity of the agent's state space in the Q-table – a large number of states occur rarely or not at all, which complicates the generalization of the agent experience;

– *lack of generalization between similar states.* Classical Q-learning stores Q-values for each specific state and does not take into account the structural similarity between agent states that have common call suffixes. This approach can reduce the stability and speed of learning.

2. Improving the classic Q-learning agent model

The shortcomings of the classical Q-learning agent model, such as the explosion of the state space dimension and the lack of generalization between similar agent states, are eliminated by introducing an improved agent model based on a mixed Q-value estimation. To reduce computational complexity, the state s_t is restricted by introducing a test case suffix function.

Thus, let there be a current sequence of API calls of the tested library at step t that corresponds to a partially formed test case (1), and let $k \in \mathrm{N}$ be the length of the call history.

Then, in the previous model [1], the agent's state can be determined based on the last k actions using the test case suffix function as follows:

$$s_t^k \leftarrow \begin{cases} \varnothing, & if\ t = 0 \\ a_1, \ldots, a_t & if\ 0 < t \le k \\ a_{t-k+1}, \ldots, a_t & if\ t > k \end{cases} \tag{4}$$

Unlike the classic single-layer Q-table, in which each state is stored separately, the proposed model uses a multi-layer structure with hierarchical merging of test case suffixes. Each partially formed test case TC^t has a set of suffixes calculated according to (4), which can be represented as follows:

$$S_{suf} = \left\{ s_t^i \mid i = 1,..,k \right\}. \tag{5}$$

This approach allows us to interpret the Q-table as a multilayer table:

$$Q(s,a) = \left\{ Q(s,a)^{(0)}, Q(s,a)^{(1)}, \ldots, Q(s,a)^{(k)} \right\}, \tag{6}$$

where $Q(s,a)^{(0)}$ is the evaluation for the initial state of the agent, $Q(s,a)^{(k)}$ is the evaluation for the suffix of the test case of depth k .

The concept of the spatial influence of TC suffixes is based on the theory of sequence generalization in reinforcement learning [28].

When updating Q-values according to (3), the agent distributes the TDE not only to the current state, but also to all its suffixes. Updates for each suffix depth level $i = 1,..,k$ are performed with an exponential decrease in weight:

$$Q\left(s_t^i, a_t\right) \leftarrow Q\left(s_t^i, a_t\right) + \alpha \times \lambda^{k+1-i} \times \delta^t, \tag{7}$$

where $\lambda \in (0,1)$ is the decay coefficient of shorter TC suffixes influence.

The proposed rule for updating a multilayer Q-table is based on the classical principles of reinforcement learning described in [29], which assume exponential decay of the influence of past states on the parameter. The use of λ discounting in the structural dimension of shorter TC suffixes, as opposed to time discounting of rewards (eligibility traces), is based on the concept of state aggregation [30], which allows generalizing Q-values for similar states.

When selecting an action $a \in \mathsf{A}$, the Q-value estimate for available actions is calculated as the weighted average of Q-values calculated for all suffixes of the current state:

$$\tilde{Q}(s,a) = \left. \sum_{i=1}^{k} w_i \times Q\left(s_t, a_t\right)^{(i)} \middle/ \sum_{i=1}^{k} w_i \right., \tag{8}$$

w_i is the weight coefficient of the suffix i, which is determined by the following formula:

$$w_i = \left(N_i + \xi\right)^{\beta} \times \lambda^{k+1-i}, \tag{9}$$

where N_i is the number of updates of the Q-table for the suffix i, ξ is a small stabilizing term, and $\beta \in (0,1]$ is the experience amplification coefficient.

Thus, formula (8) describes the mechanism of generalizing Q-values in the proposed multilayer Q-table. The weights w_k combine the frequency of Q-table visits for TC suffixes and exponential decay, which reduces the influence of shorter TC suffixes. This approach allows the agent to use information from previous states.

Procedure 1 for multi-layer generalization of Q-values based on current state suffixes is shown in Fig. 2.

Procedure 1. Multi-layer generalization of Q-values based on current state suffixes **(BlendQ-Table)**

Input data:
- Q – Q-table
- A – set of available actions
- s_t – current state of the agent
- λ – decay coefficient of shorter suffix influence
- β – experience amplification coefficient

Output data:
- $\tilde{Q}$ – table of weighted average Q-values

Procedure body:

if $\mathsf{A} = \varnothing$ ***then return None***
local $w_{sum} \leftarrow \varnothing, w_a \leftarrow \varnothing, \tilde{Q} \leftarrow \varnothing$
for each $a \in \mathsf{A}$ ***do***
for $i = 1$ ***to*** $Len\left(s_t\right)$ ***do***
local $s_t^i \leftarrow \text{Equation\_4}(TC, i)$
local $s_t^i \leftarrow \text{Equation\_4}(TC, i)$
if $Q\left(s_t^i, a\right) != 0$ ***then***
local $N_i \leftarrow CountUpdates\left(Q, s_t^i\right)$
local $w_i \leftarrow \left(N_i + 10^{-6}\right)^{\beta} \cdot \lambda^{k+1-i}$
$w_a(a) \leftarrow w_a(a) + w_i \cdot Q\left(s_t^i, a\right)$
$w_{sum}(a) \leftarrow w_{sum}(a) + w_i$
end if
end for
if $w_{sum}(a) > 0$ ***then***
$\tilde{Q}(a) \leftarrow w_a(a) / w_{sum}(a)$
else
$\tilde{Q}(a) \leftarrow 10^{-6}$
end if
end for
return $\tilde{Q}$

Fig. 2. Pseudocode of the procedure for multi-layer generalization of Q-values based on suffixes of the current state

Weighted averaging by Q-table visit frequency comes from [31, 32], which considers the role of visit frequency in stabilizing Q-values and preventing overestimation.

During training, the agent applies an ε-greedy policy in which actions are selected based on multi-layer averaging of Q-values calculated for all suffixes of the current state according to equation (8). The probability of selecting an action $a_t \in \mathrm{A}(s_t)$ in state s_t is defined as:

$$\mathrm{a}_t = \begin{cases} \arg\max\limits_{\mathrm{a}' \in \mathrm{A}(\mathrm{s}_t)} \tilde{\mathrm{Q}}(\mathrm{s}_t, \mathrm{a}'), & (1-\varepsilon), \\ \mathrm{rand}\ \mathrm{A}(\mathrm{s}_t) & \varepsilon, \end{cases} \tag{10}$$

where $\varepsilon \in (0,1)$ is the coefficient of stochasticity of the action space exploration.

Thus, formulas (8), (9), and (10) describe the modified ε-greedy policy of the agent, in which the choice of action is made based on the averaged Q-values for all state suffixes. This policy combines the simplicity of classical Q-learning with the generalization of a multi-layer Q-table, increasing the stability of learning and the accuracy of test case construction.

Algorithm 1 for agent training based on Q-learning is shown in Fig. 3, in which, unlike the classical version, the TDE is distributed among all suffixes of the current state.

Algorithm 1. Agent training algorithm with multi-layer Q-table update

Input data:
- Q – initial Q-table
- TC_0 – original test case
- $k \in \mathrm{N}, k > 0$ – call history length
- $\alpha_0 \in (0,1]$ – initial learning rate
- $\alpha_{final} \in (0,1]$ – final learning rate
- $\varepsilon_0 \in (0,1)$ – initial exploration probability
- $\varepsilon_{final} \in (0,1)$ – final probability of investigation
- $\gamma \in (0,1)$ – discount factor
- $N_{ep} \in \mathrm{N}$ – number of learning episodes
- $N_{retries} \in \mathrm{N}$ – maximum number of retries
- λ – decay coefficient of shorter suffixes influence
- β – experience amplification coefficient

Initial data:
- Q – trained Q-table.

Algorithm body:

1. ***for*** $ep = 1$ ***to*** N_{ep} ***do***
2. ***local*** $TC \leftarrow \varnothing$
3. ***local*** $\mathrm{A}_{ep} \leftarrow \mathrm{GetActionSpace}(TC_0)$
4. ***local*** $C_{orig} \leftarrow GetCoverage(TC_o)$
5. ***local*** $C_t \leftarrow \varnothing$
6. ***local*** $\alpha_{ep} \leftarrow \mathrm{Linear\ Decay}(N_{ep}, ep, \alpha_0, \alpha_{final})$
7. ***local*** $\varepsilon_{ep} \leftarrow \mathrm{LinearDecay}(N_{ep}, ep, \varepsilon_0, \varepsilon_{final})$
8. ***local*** $R_{total} \leftarrow 0,\ \mathrm{Loss} \leftarrow 0,\ s_t \leftarrow \varnothing, a_t \leftarrow \varnothing$
9. ***while*** $C_t < C_{orig}$ ***and not*** $\mathrm{Over}(\mathrm{A}_{ep})$ ***do***
10. ***local*** $N_{retry} \leftarrow 0$
11. ***local*** $\tilde{Q} \leftarrow BlendQTable(Q, s_t, \mathrm{A}_{ep}, \lambda, \beta)$
12. ***while*** $N_{retry} < N_{retries}$ ***do***
13. $a_t \leftarrow \mathrm{Equation\_10}(\tilde{Q}, s_t, \mathrm{A}_{ep}, \varepsilon_{ep})$
14. $TC \leftarrow TC \shuffle a_t$
15. ***if*** $isValid(TC)$ ***then***
16. $\mathrm{A}_{ep} \leftarrow \mathrm{A}_{ep} / a_t$
17. ***break***
18. ***else***
19. $TC \leftarrow TC / a_t$
20. $N_{retry} \leftarrow N_{retry} + 1$
21. ***end if***
22. ***end while***
23. ***if*** $N_{retry} > N_{retries}$ ***then***

Fig. 3. Pseudocode of the agent learning algorithm based on Q-learning with multi-layer Q-table update *(Beginning)*

Algorithm 1. Agent training algorithm with multi-layer Q-table update

24. ***break***
25. ***end if***
26. $s_{t+1} \leftarrow \text{Equation\_4}(TC,k)$
27. ***local*** $r_t \leftarrow \text{Execute}(TC)$
28. ***local*** $\delta^t \leftarrow r_t + \gamma \times \max Q(s_{t+1}, a') - Q(s_t, a_t)$
29. ***for*** $i = 1$ ***to*** k ***do***
30. $s_t^i \leftarrow \text{Equation\_4}(TC,i)$
31. $Q(s_t^i, a_t) \leftarrow Q(s_t^i, a_t) + \alpha \times \lambda^{k+1-i} \times \delta^t$
32. ***end for***
33. $R_{total} \leftarrow R_{total} + r_t$
34. $\text{Loss} \leftarrow \text{Loss} + \text{L}(Q, r_t, a_t, s_t, s_{t+1}, \gamma)$
35. $s_t \leftarrow s_{t+1}$
36. $C_t \leftarrow GetCoverage(TC)$
37. ***end while***
38. $\text{Log}(\text{Loss}, R_{total})$
39. ***end for***
40. ***return*** Q

Fig. 3. Pseudocode of the agent learning algorithm based on Q-learning with multi-layer Q-table update *(The end)*

Rule (7) is used to update the Q-table, and policy (10) is used to select actions. During learning, the learning rate and exploration probability change according to a linear decay law, which ensures a balance between exploration and exploitation of actions.

The proposed Procedure 1 implements a mechanism of multi-layer generalization of Q-values for a set of available actions. For each suffix of the current state s_t^i, a weight coefficient w_i is calculated, which takes into account the frequency of Q-table updates and the distance to the current state according to (9).

The averaged estimates for each action $\tilde{Q}(a)$ are formed as a normalized weighted average of Q-values across all suffixes. The resulting table is used by the agent to make decisions during training or TC forming according to ε-greedy policy.

Thus, the proposed model provides a hierarchical generalization of the Q-table, allowing the agent to use the knowledge acquired in previous states during forming new sequences of actions. Such a multilayered representation of states increases resistance to high sparsity of the agent's state space and improves the agent's adaptation to heterogeneous API environments.After the training stage, the agent uses the obtained Q-table to form new TC. The pseudocode of Algorithm 2 for forming TC is shown in Fig. 4.

Unlike the learning process, which uses an ε-greedy policy, at the TC formation stage, actions are selected using a softmax policy, which provides a smoother balance between exploration and exploitation of already accumulated knowledge.

The probability of choosing an action $a_t \in \text{A}(s_t)$ in state s_t can be determined as follows:

$$\text{P}(\text{a}_\text{t}|\text{s}_\text{t}) = \frac{\text{e}^{\tilde{\text{Q}}(\text{s}_\text{t},\text{a}_\text{t})/\tau}}{\sum_{\text{a}' \in \text{A}(\text{s}_\text{t})} \text{e}^{\tilde{\text{Q}}(\text{s}_\text{t},\text{a}')/\tau}}, \tag{11}$$

where τ is a temperature parameter that regulates the influence of stochasticity.

In Algorithm 2, the agent sequentially builds a new TC using the trained Q-table. At each iteration, it forms the current state based on the latest calls, performs multi-layer generalization of

Q-values according to Procedure 1, and then selects the next action according to policy (11). To ensure the correctness of the formed test case, a mechanism for rolling back actions and checking the admissibility of transitions is provided.

The algorithm ends when the reference coverage is reached or the action space is exhausted.

Algorithm 2. Algorithm for forming a test case

Input data:

– Q – Q-table
– TC_0 – original test case
– $N_{retries} \in \mathsf{N}$ – max. number of retries
– τ – temperature for softmax action selection
– λ – decay coeff of shorter suffix influence
– β – experience amplification coefficient
– δ – coverage stability tolerance.

Initial data:

– TC – formed test case

Algorithm body:

1. $TC \leftarrow \varnothing$
2. $k \leftarrow \text{Extract}(Q)$
3. $C_{orig} \leftarrow GetCoverage(TC_o)$
4. $\mathsf{A} \leftarrow \text{GetActionSpace}(TC_o)$
5. $C_t \leftarrow 0$
6. ***while*** $C_t < C_{orig}$ ***and not*** $\text{isOver}(\mathsf{A})$ ***do***
7. ***local*** $N_{retry} \leftarrow 0$
8. ***local*** $s_t \leftarrow \text{Equation\_4}(TC, k)$
9. ***local*** $a_t \leftarrow 0$
10. ***local*** $\tilde{Q} \leftarrow BlendQTable(Q, s_t, \mathsf{A}, \lambda, \beta)$
11. ***while*** $N_{retry} < N_{retries}$ ***do***
12. $a_t \leftarrow \text{Equation\_11}(\tilde{Q}, s_t, \mathsf{A}, \tau)$
13. $TC \leftarrow TC \mathbin{\#} a_t$
14. ***if*** $isValid(TC)$ ***then***
15. $\mathsf{A} \leftarrow \mathsf{A} / a_t$
16. ***break***
17. ***else***
18. $TC \leftarrow TC / a_t$
19. $N_{retry} \leftarrow N_{retry} + 1$
20. ***endif***
21. ***end while***
22. ***if*** $N_{retry} > N_{retries}$ ***then***
23. ***break***
24. ***endif***
25. $C_t \leftarrow GetCoverage(TC)$
26. ***if*** $C_t > C_{orig} + \delta$ ***then***
27. ***break***
28. ***endif***
29. ***end while***
30. ***return*** TC

Fig. 4. Pseudocode of the test scenario formation algorithm

Thus, within the scope of the study, model [1] has been extended to improve the stability of learning and generalization of the action selection policy.

The main changes include the following:

– *multi-layer updating of the Q-table:* in the proposed version, the agent distributes the temporal difference error among all suffixes of the current state, rather than just for one suffix of length k, which allows the context of previous partial test case to be taken into account and ensures better policy generalization;

– *averaging of Q-values:* for each state, a weighted generalization of estimates for all suffixes is performed using weight coefficients defined in (9);

– *action selection policy:* during training, an ε-greedy policy is used, taking into account the averaged Q-values, and during the TC formation the softmax policy is used (11).

3. Evaluation of the effectiveness of TC optimization algorithms and methods

The following three basic metrics are used to evaluate the effectiveness of TC optimization algorithms and methods [3]:

1. Retention of Coverage (RC) coefficient – evaluates the increase in original TS branch coverage, reflecting the loss or retention of testing efficiency:

$$RC = Cov(TS_{out}) \Big/ Cov(TS_{orig}), \tag{12}$$

where $Cov(TS_{out})$ is the TS coverage after optimization, $Cov(TS_{orig})$ is the coverage of the original TS;

2. Compression Ratio (CR) – evaluates the reduction in the number of test case instructions:

$$CR = \left(Len\left(TS_{orig}\right) - Len\left(TS_{out}\right)\right) \Big/ Len\left(TS_{out}\right), \tag{13}$$

where $Len\left(TS_{orig}\right)$ is the number of instructions in the original TS, $Len\left(TS_{out}\right)$ is the number of instructions after optimization.

3. Execution Cost Reduction (ECR) – estimates the reduction in time required to execute the test suite:

$$\mathrm{ECR} = \left(\mathrm{T}\left(\mathrm{TS}_{\mathrm{orig}}\right) - \mathrm{T}\left(\mathrm{TS}_{\mathrm{out}}\right)\right) \Big/ \mathrm{T}\left(\mathrm{TS}_{\mathrm{out}}\right), \tag{14}$$

where $T\left(TS_{orig}\right)$ is the time required to execute the original TS, and $T\left(TS_{out}\right)$ is the time required to execute the TS after optimization.

To evaluate the effectiveness of the proposed method, three basic classical methods of TC optimization are used: Greedy Reduction (GR**)**, Delta Debugging (DD), and Monte Carlo Tree Search (MCTS**)**.

Greedy Reduction. The GR method forms a TC by sequentially selecting actions that provide the greatest increase in branch coverage. At each iteration, an action is selected $a^* \in \mathrm{A}$, which maximizes the coverage difference:

$$\mathrm{a}^* = \arg\max_{\mathrm{a}' \in \mathrm{A}} \left(\mathrm{Cov}\left(\mathrm{TC} \oplus \mathrm{a}'\right) - \mathrm{Cov}\left(\mathrm{TC}\right)\right). \tag{15}$$

Greedy optimization methods are widely used in TS minimization problems due to their simplicity of implementation. However, they are prone to getting stuck in local optima and are sensitive to the initial order of actions [21].

Delta debugging. The DD method is based on iterative removal of actions from the TC to eliminate redundancy without losing testing efficiency. At each step, the equivalence of the specified criterion between the initial and reduced TC is checked. In particular, the execution error is preserved [33].

Search in the Monte Carlo tree. The MCTS method constructs a tree of possible test cases, where nodes correspond to states and edges correspond to API actions.

Actions are selected based on the Upper Confidence Bound (UCB**)** criterion [18, 34]:

$$\mathrm{UCT}(\mathrm{v_i}) = \frac{\mathrm{w_i}}{\mathrm{n_i}} + \mathrm{c} \times \sqrt{\frac{\ln \mathrm{N}}{\mathrm{n_i}}}, \tag{16}$$

where w_i is the total reward for node v_i, n_i is the number of visits to node v_i, N is the number of visits to the parent node, and $c > 0$ is a parameter that controls the balance between exploitation and exploration.

After selecting a branch, the TC simulation stage is performed, the reward for the achieved coverage is evaluated, and the reward is redistributed according to the following formula:

$$\mathrm{n_i} \leftarrow \mathrm{n_i} + 1, \quad \mathrm{w_i} \leftarrow \mathrm{w_i} + \mathrm{r_i}, \tag{17}$$

where r_i is the simulation reward.

MCTS effectively explores a large space of possible test cases and gradually refines the policy of action selection, making it suitable for use as the main method of TC optimization.

Research results

1. Environment setup

To evaluate the effectiveness of the methods, two open-source C++ libraries were selected, which differ in scale and structural complexity. The main characteristics are shown in Table 2.

The *lizard* utility was used to analyze the number of lines of code and cyclomatic complexity of the target libraries. The following abbreviations are used in Table 2:

- API – number of public functions of the target C++ library;
- LoC – number of lines in the source code;
- TCN – number of test cases in the test suite;
- IC – total number of instructions in the test suite;
- BC – branch coverage of original test suite;
- Avg.CCN – average cyclomatic complexity of the library functions.

Table 2. *Characteristics of C++ libraries selected for research*

Library	API	LoC	TCN	IC	BC, %	Avg. CCN
Bitmap PlusPlus	33	760	7	71	27	2.5
Hjson	122	3936	57	1259	36.1	5.4

The evaluation was performed on a workstation with an Intel Core i7 processor (6 cores, 2.6 GHz) and 16 GB of RAM.

The libraries and their test suites were compiled using AppleClang 15.0.0 (LLVM 15) with coverage profiling options *(-fprofile-instr-generate fcoverage-mapping*) enabled. The *llvm-profdata* and *llvm-cov* utilities were used to collect and analyze branch coverage.

The parameters for training the Q-learning agent are consistent with the previous implementation described in [1] and ensure stable convergence of the utility function in conditions of high dimensionality of the action space. Table 3 lists the configuration parameters used for all experiments, unless otherwise specified.

Table 3. *Configuration of the Q-learning agent training according to Algorithm 1*

Parameter name	Designation	Value
Number of episodes	N_{ep}	1000
Maximum number of retries	$N_{retries}$	30
Initial probability of investigation	ε_0	1
Final probability of investigation	ε_{final}	0
Initial learning rate	α_0	0.3
Final learning rate	α_{final}	0.1
Discount factor	γ	0.85
Experience amplification factor	β	1

2. Analysis of the impact of parameters of the improved Q-learning agent model

The parameter k determines the length of the call history used in forming the agent's state. To study its impact, modeling was performed on the Hjson and BitmapPlusPlus libraries, where the relative frequency of state-action pairs occurrences in the Q-table during training in dependence to test case suffix length was analyzed.

Fig. 5, *a* shows the dependence of the relative frequency of finding records (suffixes) in the Q-table on the parameter k for the BitmapPlusPlus library.

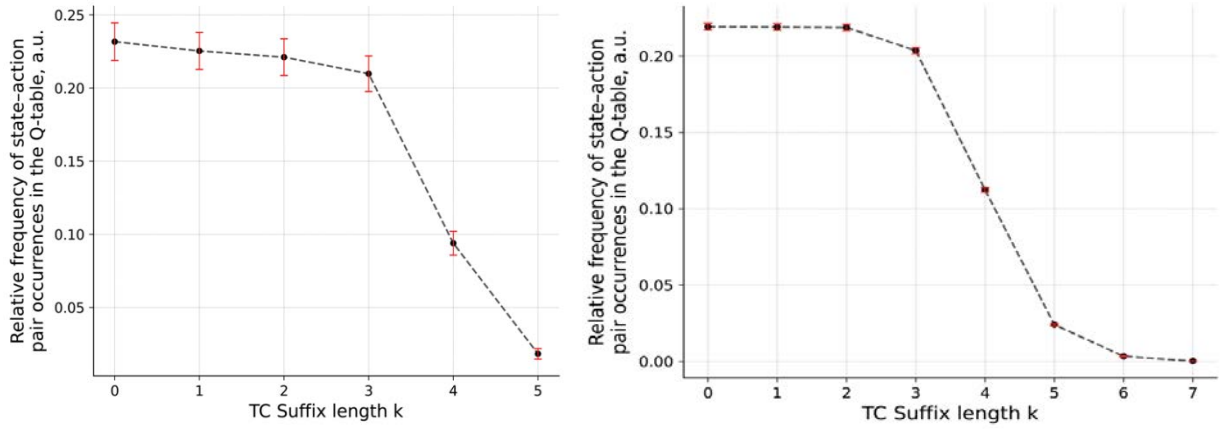
It can be observed that as k increases, the number of records found in the Q-table decreases, which indicates an increase in the sparsity of the agent's state space. For $k \geq 4$, most suffixes occur rarely, which leads to a deterioration in policy generalization.

Fig. 5, *b* shows a similar dependence for the Hjson library, which is characterized by a more branched API call structure. Up to $k \leq 3$, the proportion of states found remains virtually constant. Starting from $k = 4$, there is a sharp drop in relative frequency, and for $k \geq 6$, most suffixes occur very rarely, indicating a loss of the agent's generalization ability, which begins to accumulate unique but uninformative states.

Thus, the optimal choice is the value of the suffix $k = 5$, which provides a compromise between context depth and stability of Q-value updates.

To assess the impact of the parameter λ on learning efficiency, we chose the indicator of the proportion of suffixes found, i.e., the fraction of the discovered state-action pairs in the Q-table, e.g. the states that the agent has already encountered in previous learning episodes to the total number of transitions between the agent's states during training.

This indicator allows us to quantitatively assess the agent's ability to reuse experience when making decisions.

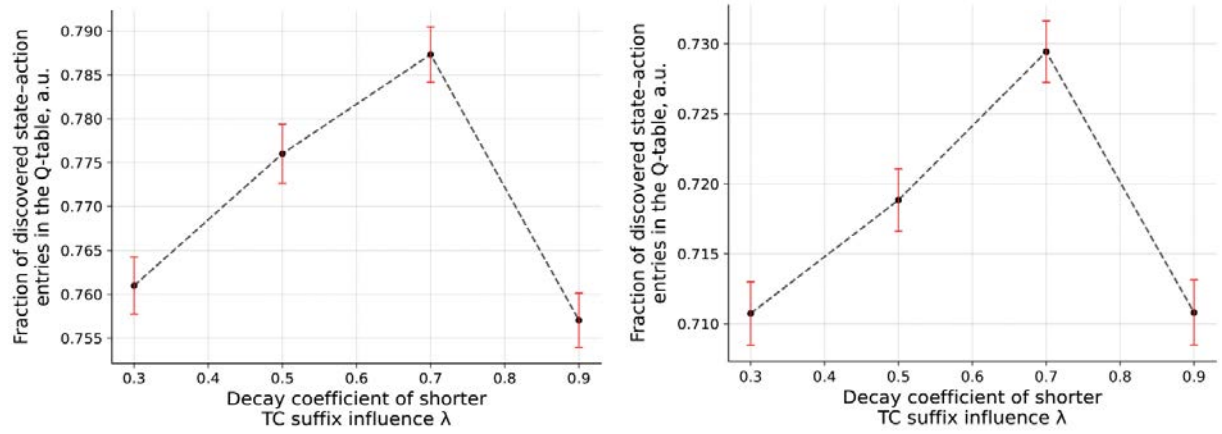


a) b)

Fig. 5. Relative frequency of state-action pairs occurrences in the Q-table during training of the Q-learning agent on the Hjson (b) and BitmapPlusPlus (a) libraries

Figure 6, *a* shows the results for the BitmapPlusPlus library. As λ increases, the proportion of suffixes found initially increases, reaching a maximum at $\lambda = 0,7$, indicating the most efficient reuse of experience. A further increase in λ causes a decrease in the indicator, as the agent begins to rely too much on the experience of shorter suffixes, reducing the number of transitions learned.

A similar trend is observed for the Hjson library (Fig. 6, *b*). An increase in the proportion of suffixes found to $\lambda \le 0,7$ indicates a gradual increase in policy stability and improved consistency of Q-table updates. At $\lambda = 0,9$, the indicator decreases slightly.



a) b)

Fig. 6. Dependence of the average fraction of discovered state-action entries int the Q-table on the coefficient λ during training on the Hjson (b) and BitmapPlusPlus (a) libraries

Thus, a comparative analysis of the two C++ libraries indicates that excessively low values of λ lead to a loss of generalization, while excessively high values lead to a decrease in the use of knowledge from the Q-table. The optimal choice is a decay coefficient for shorter suffixes of TC equal to $\lambda = 0,7$. This value provides the best compromise between exploiting previous experience and exploring new states, which confirms the stability of the policy for different types of C++ libraries.

3. Evaluation of the effectiveness of the TC formation method for C++ libraries based on a Q-learning agent

For comparative analysis, three basic methods and two groups of configurations of the proposed method were used:

– configuration of the classic greedy algorithm for TC formation (GR);

– configuration of the classic delta debugging algorithm for TS minimization (DSL);

– configurations for forming TC using the Monte Carlo tree search method (MCTS1, MCTS2, MCTS3);

– configurations of the TC formation method based on a Q-learning agent (QLB-M1, QLB-M2, QLB-M3);

– configurations of the TC formation method based on a Q-learning agent and a post-processing filter based on the delta debugging algorithm for TS minimization (QLB-MD1, QLB-MD2, QLB-MD3).

For each algorithm configuration, 100 independent mathematical simulations were performed on each target library. The configuration parameters are shown in Tables 4 and 5.

Table 4. *Configurations for TC formation by a Q-learning agent according to Algorithm 2*

Configuration	Temperature, τ	Coefficient, λ	Coefficient, β
QLB-M1	1.5	0.7	1
QLB-M2	3	0.7	1
QLB-M3	5	0.7	1

In the current implementation, the experience amplification factor β is fixed at 1, which corresponds to linear consideration of the update frequency without additional experience amplification.

Thus, the weights of suffixes are determined only by their visit frequency and exponential decay based on the λ parameter.

Table 5. *Configurations for forming TC using the Monte Carlo Tree Search method*

Configuration	Coeff., c	Number of iterations
MCTS1	0.7	200
MCTS2	1.4	200
MCTS3	2.0	200

The distribution of branch coverage metrics for the BitmapPlusPlus library (Fig. 7, *a*) shows that the classical DSL and GR methods consistently maintain the initial coverage level corresponding to the original TS.

The MCTS method demonstrates a moderate increase in average coverage.

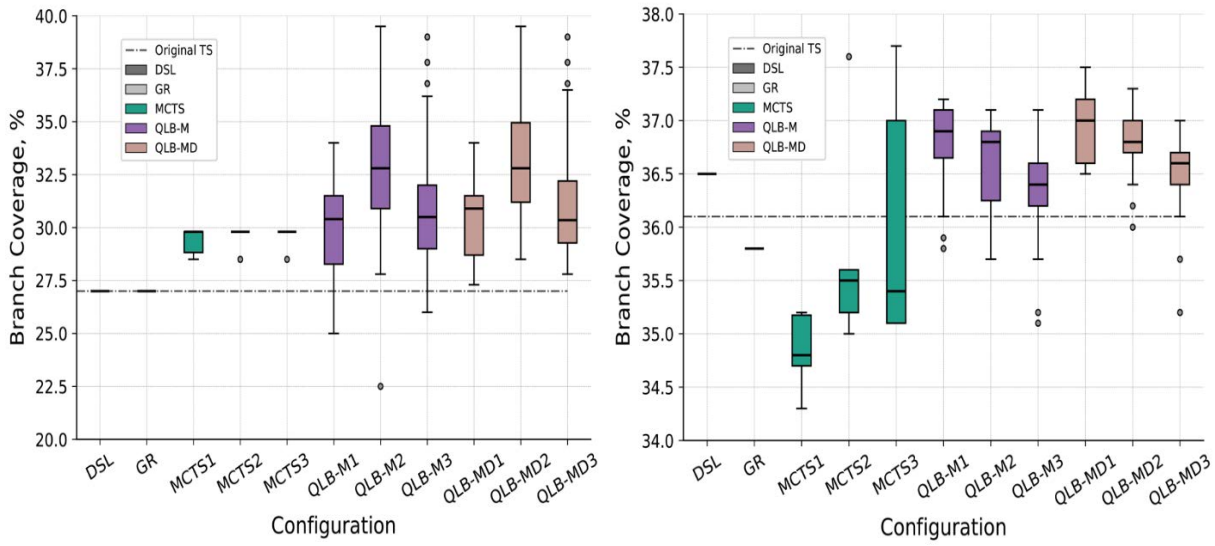
However, it is characterized by limited variability of results due to the stochastic, but not learning, nature of the search.

The proposed QLB-M and QLB-MD configurations provide a significant improvement in performance, with median coverage values exceeding 32 %. This improvement is explained by the ability of the Q-learning agent to generalize the experience of previous episodes, and the subsequent application of delta debugging allows to maintain and increase coverage after reducing the TC.

A similar trend is observed for the Hjson library (Fig. 7, *b*). The classical DSL and GR methods hardly change the initial coverage level (36.1 %), while MCTS provides a non-stable increase to 37 %.

In contrast, the proposed QLB-M and QLB-MD configurations demonstrate a systematic increase in both the median and upper quartiles of the distribution, reaching a peak value of 37.5 %.

Thus, the agent is capable of effectively reproducing complex combinations of API calls in libraries with deep structural dependencies.



a) b)

Fig. 7. Distribution of branch coverage metrics depending on the configuration of the TC optimization method for the BitmapPlusPlus (a) and Hjson (b) libraries**,** where: the dotted line indicates the coverage level of the original TS

Fig. 8 shows the results of comparing the dynamics of branch coverage growth during the execution of TC formed using different optimization methods.

The curves obtained reflect the relationship between the number of instructions executed and the achieved level of branch coverage of the code.

The dotted horizontal line indicates the coverage level of the original TS, and its intersection with the curve of a specific method corresponds to the number of instructions for which adequate compression ratio is maintained without loss of code coverage.

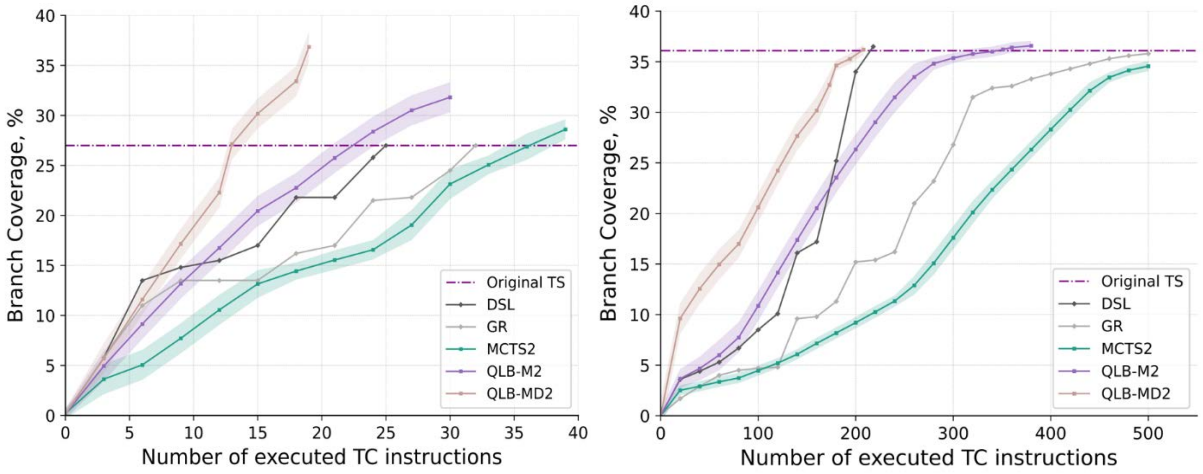


Fig. 8. Dependence of branch coverage on the number of executed TC instructions for different TC optimization methods for BitmapPlusPlus (a) and Hjson (b) libraries

Fig. 9 shows the dependence of the compression ratio on the configuration of TC optimization methods for the BitmapPlusPlus and Hjson libraries.

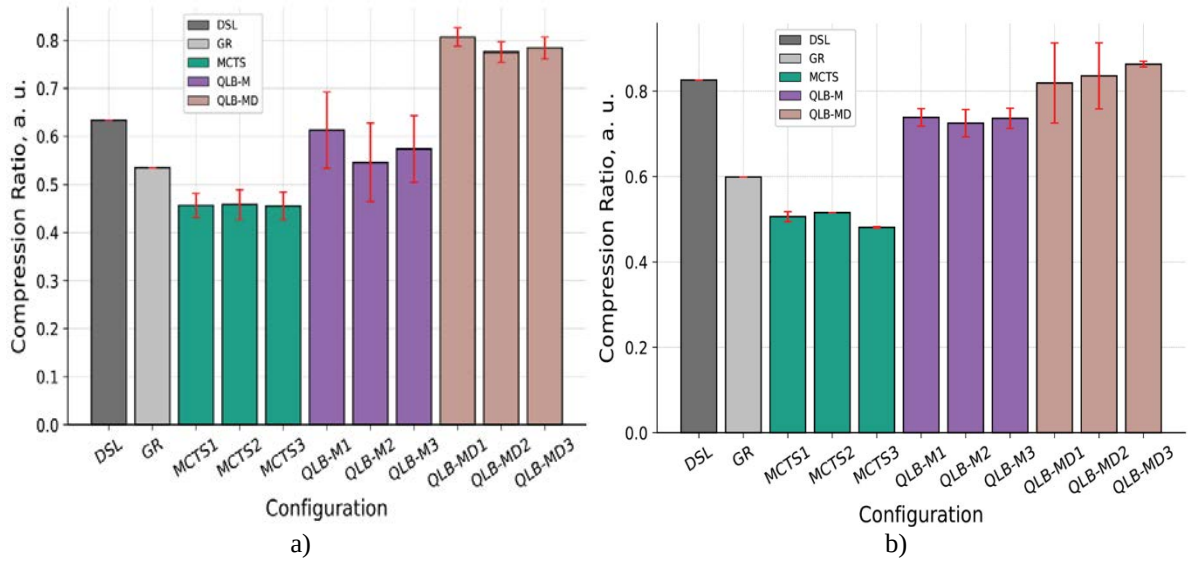


Fig. 9. Compression ratio depending on the configuration of optimization methods for the BitmapPlusPlus (a) and Hjson (b) libraries, where: the red segments on the columns indicate the standard error

For the BitmapPlusPlus library (Fig. 9, *a*), there is a gradual increase in the compression ratio from classical methods (DSL, GR) to the proposed configurations (QLB-M, QLB-MD).

The DSL and GR methods show average values of 0.5–0.6, which corresponds to the basic level of reduction without taking into account the dependencies between API calls.

The MCTS method shows similar results (≈ 0.45), since stochastic TC formation does not guarantee avoidance of redundant actions.

The proposed QLB-M method improves this indicator to 0.55–0.6.

The highest results are achieved in QLB-MD configurations, where the delta debugging algorithm is applied as a post-processing filter after the Q-learning agent formation stage. This approach enables the median compression ratio values exceed 0.8, which indicates a significant reduction of TC without loss of coverage.

For the Hjson library (Fig. 9, *b*), the indicators turned out to be more variable due to the more complex structure of API calls and the greater depth of the call tree.

The basic DSL method shows the highest compression ratio (~ 0.83), while GR is significantly lower (~ 0.6) due to the lack of consideration of dependencies between actions.

Stochastic MCTS configurations remain within the range of 0.48–0.5, reflecting low reduction efficiency without explicit coverage analysis.

QLB-M configurations show a steady increase in compression ratio (≈ 0.7–0.75), while QLB-MD combinations show the highest results (0.82–0.86) with low variance, indicating the stability of the effect after training.

Fig. 10 shows the results of evaluating the average execution time of TS for the BitmapPlusPlus and Hjson libraries.

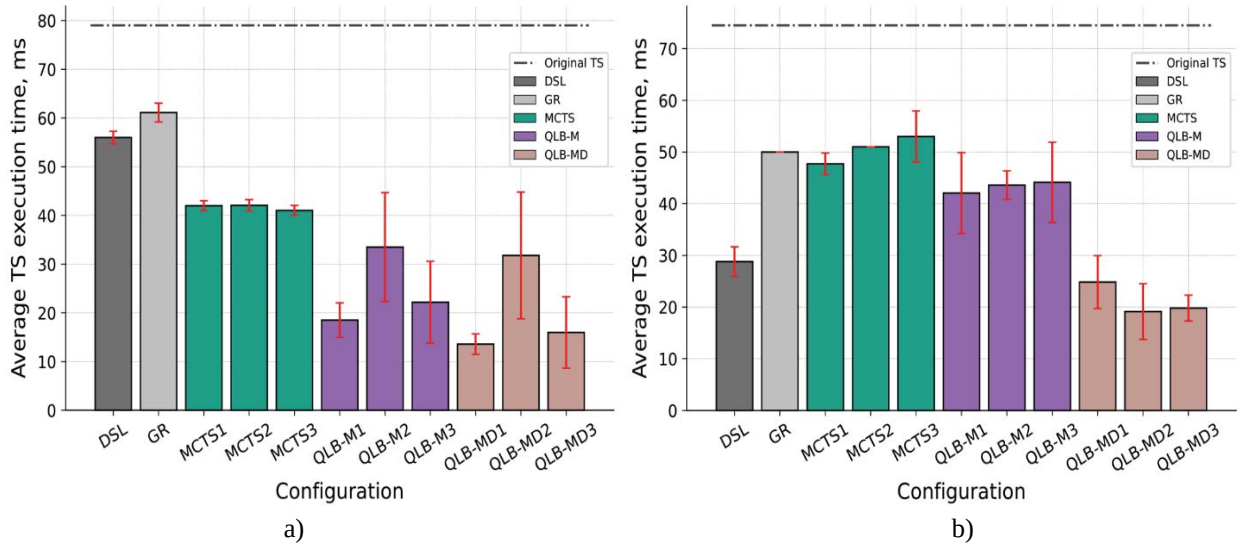


Fig. 10. Average TS execution time depending on the configuration of TC optimization methods for the BitmapPlusPlus (a) and Hjson (b) libraries, where: the red segments on the columns represent the standard error, and the dotted line corresponds to the execution time of the original TS

For the BitmapPlusPlus library (Fig. 10, *a*), there is a clear decrease in execution time from classical methods (DSL and GR) to the proposed ones (QLB-M and QLB-MD).

The basic configurations show the biggest average time (55–60 ms).

The MCTS configurations show a reduction in time to 40–42 ms. However, they do not achieve significant improvements due to the lack of generalized experience.

The QLB-M method provides a further reduction in average time to 20–35 ms due to the ability of the Q-learning agent to build shorter and more focused sequences of actions.

QLB-MD configurations show the highest efficiency, where the combination of learning and

delta debugging reduces the average execution time to 12–30 ms. That is, almost three times less than the basic methods.

For the Hjson library (Fig. 10, *b*), the trend generally remains the same. However, the difference between configurations is less pronounced due to the greater depth of calls and structural complexity of the library.

The DSL method provides the lowest time among the classic basics (~ 30 ms), while GR shows higher values (~ 50 ms), which indicates its low stability when working with more complex C++ libraries.

The MCTS method maintains a time of 45–53 ms with little variation, while QLB-M gradually reduces it to 40–45 ms.

The QLB-MD configuration group provides the lowest results (18–25 ms) while maintaining a high level of coverage and compression ratio, indicating effective coordination of TC optimization processes and reduction of redundant calls in a complex environment.

To evaluate the effectiveness of the methods under study, a comparative analysis of the average TS formation time was performed (Fig. 11).

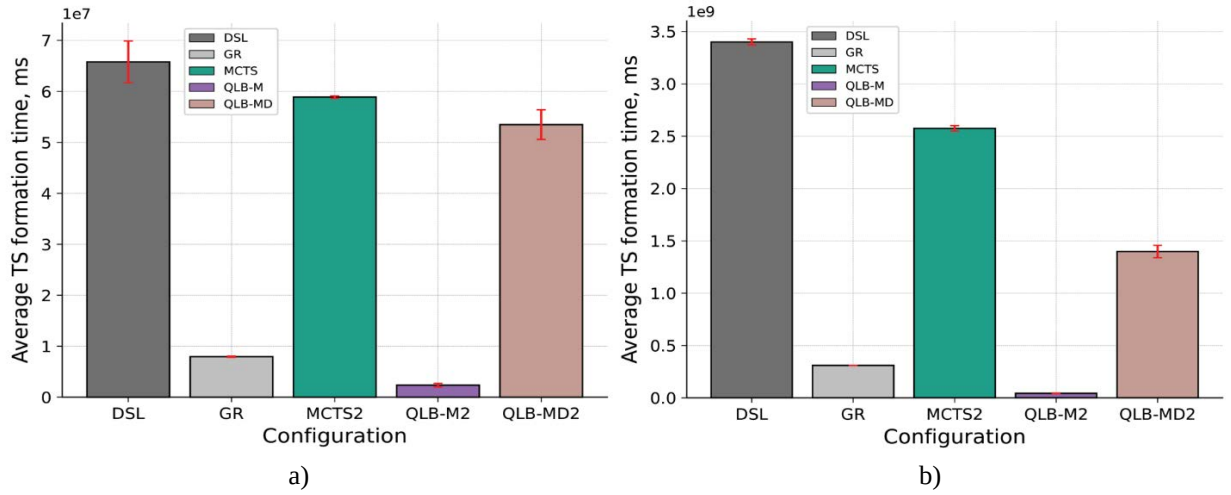


Fig. 11. Average TS formation time for TC optimization methods for BitmapPlusPlus (a) and Hjson (b) libraries

The graphs show that the basic DSL and MCTS2 methods have the highest time costs, while GR shows the fastest formation due to simple heuristics.

The improved QLB-M2 and QLB-MD2 methods show a significant reduction in formation time compared to the basic methods, while maintaining stability and moderate variability of results.

Tables 6 and 7 show the summary results of the evaluation of the effectiveness of the configurations of the studied methods for the C++ libraries BitmapPlusPlus and Hjson.

For the BitmapPlusPlus library (Table 6), the basic DSL and GR methods retain the original coverage (RC = 1), but are characterized by a moderate level of compression (CR = 0.54–0.63) and a low reduction in execution time (ECR ≈ 0.15–0.20).

MCTS search variants increase coverage by ≈ 9 % (to RC ≈ 1.09) and reduce execution time by ≈ 20 %. However, they do not provide a high value of compression ratio.

The proposed QLB-M configurations provide a coverage gain of up to ≈ 1.22, a compression ratio of 0.55–0.61, and a reduction in ECR execution time of ≈ 0.7–0.8.

The highest results were achieved for QLB-MD, where the combination of Q-learning with a delta debugging algorithm filter enables for a ≈ 22 % increase in branch coverage, compression ratio of up to 0.8, and a 70–85 % reduction in ECR execution time.

Table 6. *Evaluation of the effectiveness of the configurations of the studied methods for the BitmapPlusPlus library*

Configuration	RC	CR	ECR
DSL	1	0.63	0.19 ± 0.02
GR	1	0.54	0.15 ± 0.03
MCTS1	1.091 ± 0.02	0.46 ± 0.02	0.23 ± 0.015
MCTS2	1.092 ± 0.02	0.46 ± 0.03	0.23 ± 0.016
MCTS3	1.093 ± 0.02	0.46 ± 0.03	0.21 ± 0.015
QLB-M1	1.116 ± 0.08	0.61 ± 0.08	0.79 ± 0.07
QLB-MD1	1.127 ± 0.08	0.81 ± 0.02	0.85 ± 0.04
QLB-M2	1.215 ± 0.10	0.55 ± 0.08	0.62 ± 0.15
QLB-MD2	1.225 ± 0.10	0.78 ± 0.02	0.71 ± 0.18
QLB-M3	1.153 ± 0.09	0.57 ± 0.07	0.72 ± 0.13
QLB-MD3	1.163 ± 0.09	0.78 ± 0.02	0.78 ± 0.12

For the Hjson library (Table 7), the results are similar, but with less pronounced differences between methods due to the greater complexity of the internal structure of API calls.

Table 7. *Evaluation of the effectiveness of the configurations of the studied methods for the Hjson library*

Configuration	RC	CR	ECR
DSL	1.000	0.83	0.61 ± 0.05
GR	0.982	0.6	0.23 ± 0.00
MCTS1	0.969 ± 0.03	0.51 ± 0.01	0.39 ± 0.03
MCTS2	0.981 ± 0.03	0.52 ± 0.00	0.40 ± 0.005
MCTS3	0.988 ± 0.03	0.48 ± 0.00	0.34 ± 0.07
QLB-M1	1.008 ± 0.01	0.74 ± 0.02	0.41 ± 0.11
QLB-MD1	1.012 ± 0.01	0.82 ± 0.09	0.70 ± 0.07
QLB-M2	1.002 ± 0.01	0.73 ± 0.03	0.40 ± 0.04
QLB-MD2	1.008 ± 0.01	0.84 ± 0.08	0.71 ± 0.074
QLB-M3	0.996 ± 0.01	0.74 ± 0.02	0.42 ± 0.11
QLB-MD3	1.000 ± 0.01	0.86 ± 0.01	0.74 ± 0.039

The main DSL method maintains full coverage (RC = 1) and provides a high compression ratio (CR = 0.83) at ECR ≈ 0.61.

The GR method shows a decrease in coverage (RC = 0.98) and lower efficiency in reducing execution time (ECR ≈ 0.23).

MCTS search configurations achieve a moderate reduction in time (ECR ≈ 0.34–0.40) with stable but slightly lower coverage than the original TS.

QLB-M methods provide a balance between coverage (RC ≈ 1.00–1.01), compression ratio (0.73–0.74), and execution time reduction (ECR ≈ 0.4).

QLB-MD configurations show the highest results: RC ≈ 1.00–1.01, CR ≈ 0.84–0.86, ECR ≈ 0.70–0.74, which confirms the stability and effectiveness of the proposed method in conditions of deeper data structures in complex C++ libraries.

Thus, for both libraries, the QLB-MD configurations are the most effective, providing a balanced increase in coverage (up to + 22.5 %), compression ratio (up to 0.86), and a significant reduction in TS execution time.

To summarize the results of the comparison of methods, radial diagrams were built (Fig. 12), which reflect the relationship between the three main coefficients of effectiveness for the methods under study.

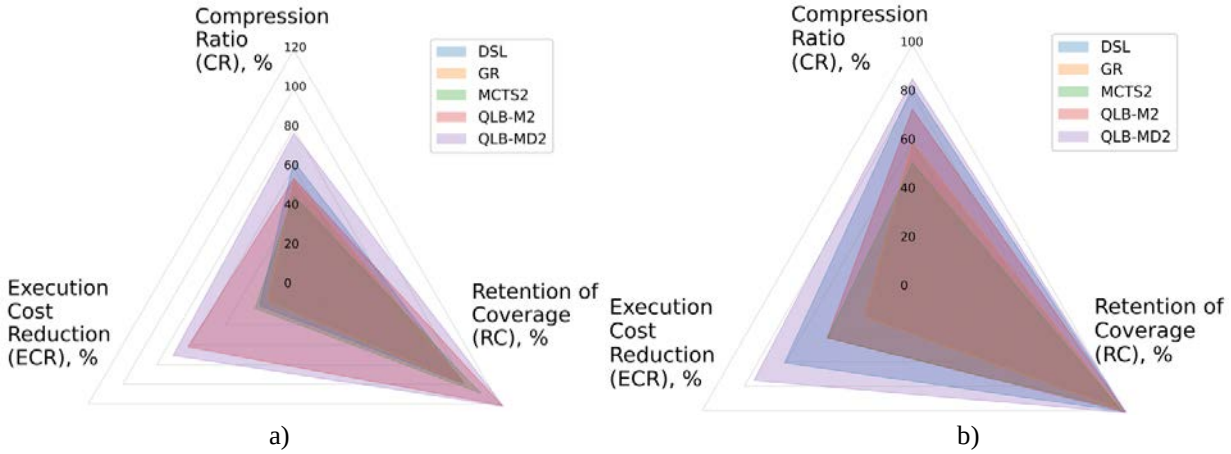


Fig. 12. Radial diagrams of the effectiveness of methods for the BitmapPlusPlus (a) and Hjson (b) libraries

Thus, the combination of the Q-learning mechanism with delta debugging action filtering provides comprehensive optimization of the TC and increases the efficiency of the testing process without losing coverage quality.

Conclusions

Thus, this article evaluates the effectiveness of the method of forming TC for C++ libraries based on a Q-learning agent. The proposed method combines step-by-step formation of function call sequences based on Q-learning with a post-processing filter based on the delta debugging algorithm.

Unlike classical greedy and search-based optimization methods, the proposed agent adaptively takes into account previous experience, balancing between exploring new states and using already known trajectories, which increases the ability to generalize experience and the stability of learning.

Analytical research of the parameters of the mathematical model indicates that the optimal length of the call history k is 5, since with a deeper history there is a sharp decrease in the frequency of finding states in the Q-table.

Research into the parameter of the mathematical model of the decay of shorter suffixes λ shows maximum efficiency at a value of 0.7, when the agent demonstrates the highest proportion of use of accumulated experience, reflecting the system's ability to maintain a balance between stability and flexibility of learning.

A comparative assessment of effectiveness with the main methods showed the advantage of the developed method in all key testing effectiveness criteria, which have the following average values: coverage retention coefficient up to 1.225, compression ratio up to 0.86, and TS execution time reduction ratio up to 0.74.

The improved mathematical model provides generalization of the Q-learning agent's experience between similar TC and increases the efficiency of their formation in conditions of high sparsity of the state space of the Q-learning agent. Unlike the classical mathematical model of Q-learning, in which Q-values are updated only for the current state of the agent (test case suffix), in the proposed model, the temporal difference error is distributed among all test case suffixes with the decay of influence of the shorter suffixes. Thus, according to the results of evaluating the effectiveness of the method of forming TC for C++ libraries based on a Q-learning agent, its effectiveness in solving the problem of forming (optimization) of TC for C++ libraries, which allows reducing the length of original TS without losing the branch coverage of the C++ library code being tested and reducing its execution time.

Further research directions may include the formation of test cases for C++ libraries based on a deep learning agent with reinforcement and the application of the proposed method of forming (optimizing) test cases for software developed using other programming languages.

References

1. Semenov, S., Kolomiitsev, O., Hulevych, M., Mazurek, P., Chernyk, O. (2025), "An Intelligent Method for C++ Test Case Synthesis Based on a Q-Learning Agent", *Applied Sciences*, Vol. 15(15), Art. No. 8596. DOI: https://doi.org/10.3390/app15158596
2. Alian, M., Suleiman, D., Shaout, A. (2016), "Test Case Reduction Techniques – Survey", *International Journal of Advanced Computer Science and Applications,* Vol. 7(5), P. 264–275. DOI: https://doi.org/10.14569/IJACSA.2016.070537
3. Khan, S. U. R., Lee, S., Javaid, N., Abdul, W. (2018), "A Systematic Review on Test Suite Reduction: Approaches, Experiment's Quality Evaluation, and Guidelines", *IEEE Access,* Vol. 6, P. 11816-11841. DOI: https://doi.org/10.1109/ACCESS.2018.2809600
4. Rahman, M., Zamli, K., Kader, M., Sidek, R., Din, F. (2024), "Comprehensive Review on the State-of-the-arts and Solutions to the Test Redundancy Reduction Problem with Taxonomy", *Journal of Advanced Research in Applied Sciences and Engineering Technology,* Vol. 35(1), P. 62–87. DOI: https://doi.org/10.37934/araset.34.3.6287
5. Marappan, R., Raja, S. (2025), "Recent Trends in Regression Testing: Modeling and Analyzing the Critiques in Selection, Optimization, and Prioritization", *National Academy Science Letters.* DOI: https://doi.org/10.1007/s40009-025-01613-6

6. Fontes, A., Gay, G. (2023), "The integration of machine learning into automated test generation: A systematic mapping study", *Software Testing, Verification and Reliability,* Vol. 33(4), e1845. DOI: https://doi.org/10.1002/stvr.1845
7. Sebastian, A., Naseem, H., Catal, C. (2024), "Unsupervised Machine Learning Approaches for Test Suite Reduction", *Applied Artificial Intelligence,* Vol. 38(1), Art. No. 2322336. DOI: https://doi.org/10.1080/08839514.2024.2322336
8. Nayab, S., Wotawa, F. (2024), "Testing and Reinforcement Learning: A Structured Literature Review", *2024 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, Cambridge, United Kingdom, 2024, P. 326-335. DOI: https://doi.org/10.1109/QRS-C63300.2024.00049
9. Coviello, C., Romano, S., Scanniello, G., Marchetto, A., Corazza, A., Antoniol, G. (2019), "Adequate vs. inadequate test suite reduction approaches", *Information and Software Technology,* Vol. 119, Art. No. 106224. DOI: https://doi.org/10.1016/j.infsof.2019.106224
10. Hulevych, M., Kolomiitsev, O. (2025), "Automated Test Generation Techniques for C++ Software", *Control, Navigation and Communication Systems,* Vol. 2(80), P. 102-107. DOI: https://doi.org/10.26906/SUNZ.2025.2.102
11. Pan, R., Ghaleb, T. A., Briand, L. (2023), "ATM: Black-box Test Case Minimization Based on Test Code Similarity and Evolutionary Search", *Proceedings of the 45th International Conference on Software Engineering (ICSE '23)*, Melbourne, Australia, 2023, pp. 1700-1711. DOI: https://doi.org/10.1109/ICSE48619.2023.00146
12. Koitz-Hristov, R., Sterner, T., Stracke, L., Wotawa, F. (2024), "On the suitability of checked coverage and genetic parameter tuning in test suite reduction", *Journal of Software: Evolution and Process,* Vol. 36(8), e2656. DOI: https://doi.org/10.1002/smr.2656
13. Dang, Z., Wang, H. (2024), "Leveraging meta-heuristic algorithms for effective software fault prediction: A comprehensive study", *Journal of Engineering and Applied Science,* Vol. 71, Art. No. 189. DOI: https://doi.org/10.1186/s44147-024-00529-0
14. Rheaf, T. S. (2025), "Prioritization of Modules to Reduce Software Testing Time and Costs Using Evolutionary Algorithms and the KLOC Method", *Journal of Education for Pure Science,* Vol. 15(1). P. 116-131. DOI: https://doi.org/10.32792/jeps.v15i1.658
15. Bai, R., Chen, R., Lei, X., Wu, K. (2024), "A Test Report Optimization Method Fusing Reinforcement Learning and Genetic Algorithms", *Electronics,* Vol. 13(21), Art. No. 4281. DOI: https://doi.org/10.3390/electronics13214281
16. Świechowski, M., Godlewski, K., Sawicki, B., Mańdziuk, J. (2023), "Monte Carlo Tree Search: A review of recent modifications and applications", *Artificial Intelligence Review,* Vol. 56, P. 2497–2562. DOI: https://doi.org/10.1007/s10462-022-10228-y
17. Ye, A., Wang, L., Zhao, L., Ke, J. (2022), "Ex²: Monte Carlo Tree Search-based test inputs prioritization for fuzzing deep neural networks", *International Journal of Intelligent Systems,* Vol. 37(12), P. 11966–11984. DOI: https://doi.org/10.1002/int.23072
18. Kocsis, L.; Szepesvári, C. (2006), "Bandit Based Monte-Carlo Planning", *Lecture Notes in Computer Science (ECML 2006)*, Vol. 4212, P. 282-293. DOI: https://doi.org/10.1007/11871842_29
19. Lin, C.-T., Tang, K.-W., Wang, J.-S., Kapfhammer, G. M. (2017), "Empirically evaluating Greedy-based test suite reduction methods at different levels of test suite complexity", *Science of Computer Programming,* Vol. 150, P. 1-25. DOI: https://doi.org/10.1016/j.scico.2017.05.004
20. Parsa, S.; Khalilian, A. (2009), "A Bi-objective Model Inspired Greedy Algorithm for Test Suite Minimization", *Lecture Notes in Computer Science (FGIT 2009)* Vol. 5899, P. 208–215. DOI: https://doi.org/10.1007/978-3-642-10509-8_24
21. Jehan, S., Wotawa, F. (2023), "An Empirical Study of Greedy Test Suite Minimization Techniques Using Mutation Coverage", *IEEE Access*, Vol. 11, P. 65427–65442. DOI: https://doi.org/10.1109/ACCESS.2023.3289073

22. Putra, A. W., Legowo, N. (2025), "Greedy Algorithm Implementation for Test Case Prioritization in the Regression Testing Phase", *Journal of Computer Science*, Vol. 21(2), P. 290–303. DOI: https://doi.org/10.3844/jcssp.2025.290.303
23. Zeller, A., Hildebrandt, R. (2002), "Simplifying and isolating failure-inducing input", *IEEE Transactions on Software Engineering*, Vol. 28(2), P. 183–200. DOI: https://doi.org/10.1109/32.988498
24. Cleve, H., Zeller, A. (2005), "Locating causes of program failures", *Proceedings of the 27th International Conference on Software Engineering (ICSE '05),* Association for Computing Machinery, New York, USA, 2005, P. 342–351. DOI: https://doi.org/10.1145/1062455.1062522
25. Misherghi, G., Su, Z. (2006), "HDD: Hierarchical delta debugging", *Proceedings of the 28th International Conference on Software Engineering (ICSE '06),* Association for Computing Machinery, New York, USA, 2006, P. 142–151. DOI: https://doi.org/10.1145/1134285.1134307
26. Wang, G., Wu, Y., Zhu, Q., Xiong, Y., Zhang, X., Zhang, L. (2023), "A Probabilistic Delta Debugging Approach for Abstract Syntax Trees", *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE),* Florence, Italy, 2023, pp. 763–773. DOI: https://doi.org/10.1109/ISSRE59848.2023.00060
27. Puterman, M. L. (1994), "Markov Decision Processes: Discrete Stochastic Dynamic Programming", John Wiley & Sons. DOI: https://doi.org/10.1002/9780470316887
28. Singh, S. P. (1992), "Transfer of learning by composing solutions of elemental sequential tasks", *Machine Learning,* Vol. 8, P. 323–339. DOI: https://doi.org/10.1007/BF00992700
29. Sutton, R. S., Barto, A. G. (2018), "Reinforcement Learning: An Introduction", (2nd ed.). MIT Press.
30. Li, L., Walsh, T. J., Littman, M. L. (2006), "Towards a Unified Theory of State Abstraction for *MDPs*", *Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics, 2006*.
31. Watkins, C.J.C.H., Dayan, P. (1992), "Q-learning", *Machine Learning*, Vol. 8, P. 279–292. DOI: https://doi.org/10.1007/BF00992698
32. Hasselt, H. (2010), "Double Q-learning", *Advances in Neural Information Processing Systems (NIPS 2010)*, Vol. 23.
33. Groce, A., Alipour, M. A., Zhang, C., Chen, Y., Regehr, J. (2016), "Cause reduction: Delta debugging, even without bugs", *Software Testing, Verification and Reliability,* Vol. 26, P. 40–68. DOI: https://doi.org/10.1002/stvr.1574
34. Auer, P., Cesa-Bianchi, N., Fischer, P. (2002), "Finite-time Analysis of the Multiarmed Bandit Problem", *Machine Learning*, Vol. 47, P. 235–256. DOI: https://doi.org/10.1023/A:1013689704352

Received (Надійшла) 13.11.2025
Accepted for publication (Прийнята до друку) 08.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Hulevych Mykhailo – National Technical University "Kharkiv Polytechnic Institute", PhD Student, Computer Engineering and Programming Department, Kharkiv, Ukraine;
e-mail: gulevich30misha@gmail.com; ORCID ID: https://orcid.org/0009-0003-8622-3271

Гулевич Михайло Володимирович – Національний технічний університет "Харківський політехнічний інститут", аспірант кафедри комп'ютерної інженерії та програмування, Харків, Україна.

ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ МЕТОДУ ФОРМУВАННЯ ТЕСТОВИХ СЦЕНАРІЇВ ДЛЯ C++ БІБЛІОТЕК НА ОСНОВІ АГЕНТА З Q-НАВЧАННЯМ

Оптимізація тестових сценаріїв (ТС) є необхідною умовою підвищення ефективності регресійного тестування C++ бібліотек. **Предметом дослідження** є методи формування (оптимізації) ТС для C++ бібліотек. **Мета роботи** – оцінити ефективність методу формування ТС для C++ бібліотек на основі агента з Q-навчанням. **Завдання дослідження:** удосконалити математичну модель агента з Q-навчанням для підвищення ефективності формування ТС для C++ бібліотек в умовах високої розрідженості простору станів агента з Q-навчанням; дослідити вплив параметрів удосконаленої моделі агента з Q-навчанням на його поведінку за такими умовами; розглянути можливість мінімізації сформованих ТС методом їх формування завдяки алгоритму дельта-дебаггінг мінімізації ТС; оцінити ефективність запропонованого методу й порівняти з відомими методами оптимізації ТС. **Методи дослідження.** У роботі застосовано метод пошуку на дереві Монте-Карло, класичну математичну модель Q-навчання, алгоритм дельта-дебаггінг мінімізації ТС і жадібний алгоритм оптимізації ТС. Ефективність запропонованого методу оцінено на двох C++ бібліотеках з відкритим вихідним кодом за допомогою статистичного аналізу 100 математичних моделювань конфігурацій методів, які досліджуються. **Досягнуті результати:** оцінка ефективності вказує на те, що запропонований метод забезпечує такі середні значення показників ефективності оптимізації ТС для C++ бібліотек: коефіцієнт збереження покриття становить до 1.225, коефіцієнт стиснення тестового набору (ТН) – до 0.86 й коефіцієнт скорочення часу на виконання ТН – до 0.74. Установлено, якщо порівнювати з жадібним алгоритмом оптимізації ТС, дельта-дебаггінг алгоритмом мінімізації ТС та методом оптимізації ТС на основі пошуку на дереві Монте-Карло, то запропонований метод має суттєве підвищення ефективності формування (оптимізації) ТС для C++ бібліотек. **Висновки.** Удосконалена математична модель забезпечує узагальнення досвіду агента з Q-навчанням між подібними ТС і підвищує ефективність їх формування в умовах високої розрідженості простору станів агента з Q-навчанням. Отже, за результатами оцінювання методу формування ТС для C++ бібліотек на основі агента з Q-навчанням підтверджено його доцільність у розв’язанні задачі формування (оптимізації) ТС для C++ бібліотек, що дає змогу скоротити довжину ТС без втрати гілкового покриття коду C++ бібліотеки, яка тестується, і зменшити час на виконання ТН. Подальші дослідження будуть присвячені формуванню ТС для C++ бібліотек на основі агента глибинного навчання з підкріпленням.

Ключові слова: оптимізація тестових сценаріїв; тестування програмного забезпечення; Q-навчання; Q-таблиця; дельта-дебаггінг, покриття коду; мінімізація; C++; агент; навчання з підкріпленням.

Bibliographic descriptions / Бібліографічні описи

Hulevych, M. (2025), "Evaluation of the effectiveness of the test scenarios forming method for C++ libraries based on a Q-learning agent", *Management Information Systems and Devises*, No. 4 (187), P. 20–46. DOI: https://doi.org/10.30837/0135-1710.2025.187.020

Гулевич М. В. Оцінювання ефективності методу формування тестових сценаріїв для C++ бібліотек на основі агента з Q-навчанням. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 20–46. DOI: https://doi.org/10.30837/0135-1710.2025.187.020

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

УДК 681.5+519.711.2 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.047

І. Гребеннік, О. Коваленко

АДАПТИВНА НЕЙРОНЕЧІТКА СИСТЕМА ВИВЕДЕННЯ ДЛЯ СОРТУВАННЯ ОБ'ЄКТІВ ПОШТОВИХ ВІДПРАВЛЕНЬ У КОНВЕЄРНОМУ ПОТОЦІ

Предметом вивчення є моделі прийняття рішень автоматизованих сортувальних ліній (ASL), якими оснащено перевалкові центри сортування об'єктів поштових відправлень (ОПВ), що належать до логістичної мережі доправлення пошти. **Мета дослідження** – розробити модель прийняття рішень для ASL завантажувальних дверей терміналів перевалкових центрів у вигляді адаптивної нейронечіткої системи виведення (ANFIS). У статті необхідно виконати такі **завдання:** проаналізувати конструктивні особливості обладнання ASL та запропонувати його модифікацію – оснащення дверей терміналів не одним, а трьома завантажувальними лотками; визначити структуру ANFIS відповідно до нечіткої моделі Такагі – Сугено першого порядку; встановити діапазони ваги й габаритів ОПВ, пов'язаних з номерами завантажувальних лотків; визначити критерії для реалізації заданої логіки сортування ОПВ; з'ясувати параметри наборів даних для навчання й тестування ANFIS; провести порівняльне навчання й тестування ANFIS з використанням інструментарію середовища MATLAB з оцінюванням значень відносної точності правильної класифікації ОПВ. **Методи:** системний, аналітичний, комп'ютерного моделювання, методи навчання з архітектурою ANFIS, математичний і статистичний аналіз ефективності навчання. **Досягнуті результати**. Проведено порівняльне навчання ANFIS гібридним методом і методом зворотного поширення помилки. За результатами тестування після 100 епох навчання для реалізації обрано модель ANFIS, що навчена гібридним методом, який забезпечує відносну точність класифікації ОПВ на рівні 96,3 %. **Висновки.** Запропонована модель прийняття рішень ASL у вигляді ANFIS дає змогу реалізувати нечітку класифікацію ОПВ у потоці конвеєра за трьома діапазонами їх вагогабаритних параметрів для реалізації заданої логіки сортування по завантажувальних лотках. Реалізована логіка сортування забезпечує компактне завантаження кузовів вантажівок по секціях і знижує ризики пошкодження ОПВ, коли вони укладаються один на одний.

Ключові слова: об'єкти поштових відправлень; сортування; модель прийняття рішень; автоматизована лінія; нечітка класифікація; адаптивна нейронечітка система виведення.

1. Вступ

Поштова логістика є важливим аспектом бізнесу, що визначає репутацію компаній-виробників у сфері обслуговування клієнтів. Зростання обсягів електронної комерції підвищує вимоги споживачів до швидкості, якості та контролю доправлення товарів поштою. Для доправлення об'єктів поштових відправлень (ОПВ) логістичні компанії створюють транспортну мережу з проміжними центрами сортування посилок (PSC – *Parcel Sorting Centres*) за напрямками. Для організації подібної транспортної мережі розв'язуються складні завдання, пов'язані з контролем потоку ОПВ, плануванням і оптимізацією маршрутів доправлення ОПВ до кінцевих пунктів із використанням проміжних PSC, плануванням графіка прибуття вхідних і вихідних вантажівок на проміжні PSC для розвантаження й завантаження ОПВ після їх сортування за напрямками доправлення.

Однією з важливих проблем у процесі доправлення ОПВ є їх пошкодження під час розвантаження й завантаження, що зумовлено двома факторами. Перший пояснюється тим, що досі не існує узагальненого міжнародного стандарту, який визначає класифікацію ОПВ за їх вагогабаритними параметрами. Це є причиною, чому виробники автоматизованих сортувальних конвеєрів (ASC – *Automated Sorting Conveyor*) і автоматизованих сортувальних ліній (ASL – *Automated Sorting Line*), що використовуються для обладнання PSC, не реалізують сортування ОПВ з огляду на їх ваги й габарити.

Другий фактор зумовлений жорстким графіком роботи PSC. Проміжні PSC [1] працюють цілодобово за планом прибуття вхідних і вихідних вантажівок. Вантажівки завантажуються в чітко встановлені терміни вручну. Посилки, відсортовані за напрямками доправлення, надходять на завантаження потоком. ОПВ завантажуються в кузов вантажівки способом ручного укладання один на одний у кілька шарів. У вантажників немає часу на сортування посилок за їх вагою та габаритами під час завантаження. Такий спосіб завантаження зумовлює ризик пошкодження ОПВ під час їх штабелювання, а також неефективне використання вантажного простору кузова транспортного засобу в процесі його завантаження.

Для усунення проблем, спричинених зазначеними факторами, у працях [2, 3] розроблено моделі прийняття рішення ASL, що застосовують чітку й нечітку логіку сортування ОПВ за їх вагою та габаритами для реалізації двох варіантів логіки завантаження вантажівок шляхом розподілу посилок по завантажувальних дверях терміналів PSC. Недоліками цих моделей є те, що перший варіант логіки завантаження реалізує почергове завантаження вантажівок біля різних дверей терміналів. Це призводить до збільшення часу завантаження й до зниження пропускної здатності PSC. Другий варіант логіки завантаження реалізує завантаження вантажівок ОПВ з однаковим діапазоном ваги й габаритів. Це підвищує ризик їх пошкодження за умови укладання в кілька шарів.

Зазначені недоліки моделей [2, 3] є критичними, тому виникає необхідність проведення досліджень з розроблення моделі прийняття рішень ASL, що дає змогу реалізувати сортування ОПВ за їх вагою та габаритами в конвеєрному потоці з реалізацією секційної логіки завантаження кузовів вантажівок шарами.

Для розроблення й дослідження моделі прийняття рішень ASL обрано модель у вигляді адаптивної нейронечіткої системи виведення (ANFIS – *Adaptive Neuro-Fuzzy Inference System*), яка реалізує задану логіку сортування посилок за вагою та габаритами в конвеєрному потоці.

2. Аналіз досліджень, присвячених удосконаленню логістики центрів сортування посилок

Для сортування об’єктів поштових відправлень PSC обладнуються автоматизованими сортувальними конвеєрами. Конструкція ASC містить автоматизовані сортувальні лінії різного типу. Загалом модель прийняття рішень ASC реалізована як комп’ютерна система управління сортуванням і переміщенням ОПВ на основі інформації штрих-кодів

їх етикеток. Першочерговими завданнями PSC є забезпечення безперервної (цілодобової) та безперебійної роботи ASC відповідно до графіка прибуття вхідних і вихідних вантажівок. Логістичні завдання вдосконалення роботи ASC центрів сортування посилок можна узагальнити за трьома напрямами.

Перший напрям пов'язаний з розробленням алгоритмів для розв'язання завдань підвищення пропускної здатності ASC. Для цього в праці [4] запропоновано генетичний алгоритм, що дає змогу контролювати швидкість вивантаження ОПВ, керувати графіком відправлення вихідних вантажівок і водночас не допускати простоїв. У роботі [5] описано генетичний алгоритм для планування прибуття вхідних вантажівок залежно від фіксованої кількості розвантажувальних дверей для мінімізації часу розвантаження вхідних вантажівок. Для PSC, оснащеного ASC із замкнутим контуром, розроблено гібридні алгоритми, призначені для пошуку оптимального плану прибуття вхідних [6] і вихідних [7] вантажівок відповідно до мети їх розподілу по дверях терміналів. Автори дослідження [8] розглядають завдання розроблення нелінійної моделі планування оптимального завантаження транспортних контейнерів. Контейнер розбивається на підконтейнери. У процесі планування завантаження об'єктів беруться до уваги як їх геометричні параметри (орієнтація об'єктів, мінімально і/або максимально допустимі відстані між об'єктами, комбінаторні властивості розташування об'єктів усередині підконтейнерів), так і обмеження, пов'язані з їх вагою.

Другий напрям пов'язаний з розробленням моделей, спрямованих на мінімізацію часу розвантаження й завантаження посилок, а також часу, необхідного для сортування й доправлення їх до завантажувальних дверей. Для розв'язання цієї проблеми автори праці [1] розробили змішану цілочисельну модель для ASC, яка виконує двоступеневе сортування посилок. Скорочення часу сортування досягається внаслідок мінімізації кількості налаштувань ASC, що вимагають перерозподілу посилок від завантажувальних до розвантажувальних дверей. У роботі [9] запропоновано дві моделі оптимізації пропускної здатності ASC для розподільного центру з огляду на його експлуатаційні обмеження. Перша модель визначає мінімальний маршрут транспортування товарів по конвеєру для конкретного сценарію завантаження-розвантаження. Друга модель зважає на режим роботи конвеєра з максимальною пропускною здатністю та мінімізує всі маршрути товарів для кожного завантаження-розвантаження.

Третій напрям пов'язаний з розробленням імітаційних моделей для ASC з метою проведення досліджень, спрямованих на збільшення його пропускної здатності. Імітаційні моделі реалізують потенційні конфігурації для оснащення ASC різними типами ASL, беручи до уваги їх розміщення щодо розвантажувально-завантажувальних дверей і маршрутів транспортування посилок. Для розв'язання проблеми проєктування компонування кільцевого конвеєра в роботі [10] подано генетичний алгоритм для визначення оптимальної конструкції ASC з метою підвищення його пропускної здатності. Для перевірки алгоритму використовується імітаційна модель.

Автори праці [11] розглядають різні варіанти конструкції сортувальних конвеєрів із замкнутим контуром. Розроблена модель застосовується для оцінювання пропускної здатності конвеєра з різними схемами компонування.

У дослідженні [12] запропоновано детерміновану змішано-цілочисельну лінійну модель для розв'язання питання скорочення часу ручного транспортування посилок способом зміни компонування обладнання конвеєра PSC.

3. Мета й завдання дослідження

Метою статті є розроблення моделі прийняття рішень ASL завантажувальних дверей терміналів PSC у вигляді адаптивної нейронечіткої системи виведення, що здійснює потокову нечітку класифікацію ОПВ за трьома діапазонами їх параметрів для реалізації заданої логіки їх сортування по завантажувальних лотках.

Для досягнення окресленої мети необхідно виконати такі завдання:

– проаналізувати конструктивні особливості обладнання конвеєрів PSC і його можливої модифікації;

– визначити структуру ANFIS відповідно до мети дослідження;

– з'ясувати діапазони ваги й габаритів ОПВ, пов'язані з номерами завантажувальних лотків;

– дослідити логіку сортування ОПВ;

– визначити параметри наборів даних для навчання й тестування ANFIS;

– виконати навчання й тестування ANFIS у середовищі MATLAB.

Навчена ANFIS має проводити нечітку класифікацію посилок за їх вагою та габаритами й визначати номер завантажувального лотка з відносною точністю не менше ніж 95 %.

4. Результати дослідження

4.1. Аналіз конструктивних особливостей обладнання конвеєрів PSC та його можливої модифікації

У спрощеній схемі PSC [1] три конвеєри використовуються для первинного (ASC-1) і вторинного (ASC-A, ASC-B) сортування ОПВ (рис. 1).

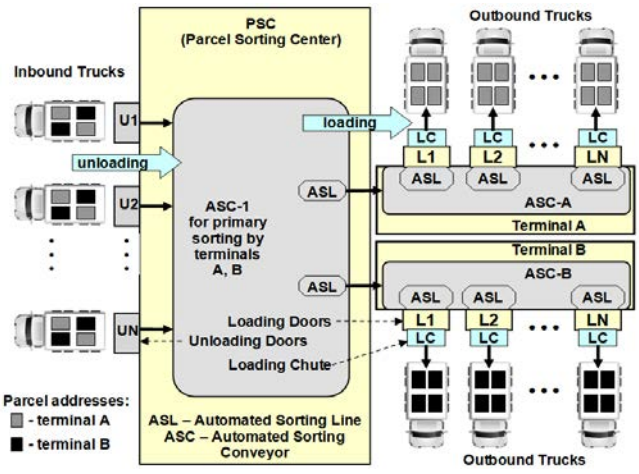


Рис. 1. Спрощена схема PSC

Завдання конвеєрів PSC полягає в розподілі ОПВ у двох напрямках, яким відповідають термінали A і B. Вхідні вантажівки прибувають до розвантажувальних дверей (*unloading door*), позначених як U1, U2, ..., UN. Посилки від розвантажувальних дверей відправляються в ASC-1 для первинного сортування за напрямками доправлення. Із цією метою використовуються два ASL, які зчитують адресу ОПВ з етикетки й визначають, на який з двох конвеєрів, ASC-A (термінал A) або ASC-B (термінал B), посилка буде відправлена. Кінцевою метою сортування на конвеєрах ASC-A і ASC-B є транспортування посилки до завантажувальних дверей (*loading door*) терміналу A або B з певним номером, позначеним як L1, L2, ..., LN. ASL завантажувальних дверей конвеєрів ASC-A і ASC-B визначає адресу посилки за її етикеткою. Якщо вона збігається, ASL відправляє посилку в завантажувальний лоток LC (*Loading Chute*) відповідних дверей. Робочі завантажують вихідну вантажівку посилками з лотків LC.

Недолік конвеєрів ASC-A і ASC-B зумовлений тим, що посилки сортуються попри їх вагогабаритні показники, а ОПВ, відсортовані за напрямками доправлення, транспортуються на завантажувальні лотки потоком. Потокове завантаження ОПВ у кузов вантажівки здійснюється з лотків способом ручного укладання шарами один на одний. Унаслідок того, що конвеєр (потік посилок) зупинити неможливо, у робочих немає часу на сортування ОПВ за їх вагою та габаритами під час завантаження. Такий спосіб укладання ОПВ зумовлює ризик їх пошкодження у процесі завантаження й транспортування, а також неефективне використання вантажного простору кузова вантажівки.

Для усунення зазначеного недоліку всі двері терміналів A і B оснащуються не одним, а трьома завантажувальними лотками, для кожного з яких визначено діапазон ваги й габаритів ОПВ. Завданням моделі прийняття рішень ASL завантажувальних дверей терміналів є виконання нечіткої класифікації ОПВ у потоці на основі їх ваги й габаритів і визначення номера завантажувального лотка LC дверей L, до яких їх потрібно транспортувати.

Варіант логіки завантаження ОПВ в потоці зображений на рис. 2. Двері L терміналів A і B оснащені трьома завантажувальними лотками, позначеними як LC1, LC2, LC3. Завантажувальні лотки призначені для прийому посилок у трьох діапазонах ваги й габаритів. Значення діапазонів збільшуються відповідно до порядку номерів завантажувальних лотків LC.

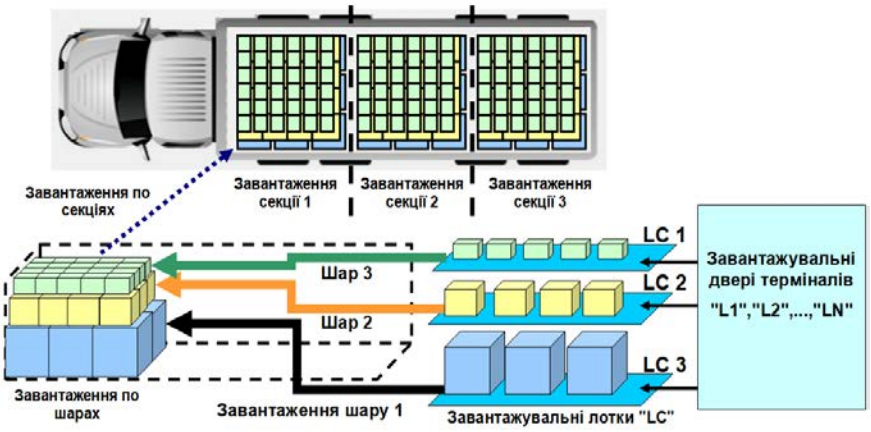


Рис. 2. Логіка завантаження вантажівок

Завантаження здійснюється в три етапи й більше залежно від обсягу вантажного простору автомобіля. На кожному етапі завантаження посилки розташовують у три шари. Перший шар формують з посилок завантажувального лотка LC3 з найбільшим діапазоном ваги й габаритів. Другий шар містить посилки середнього діапазону з лотка LC2, а третій шар – найлегші й найменші посилки з лотка LC3. На другому й наступних етапах завантаження розміщення посилок шарами повторюється. Це забезпечує компактне завантаження вантажного простору автомобіля ОПВ, що надходять у потоці, та знижує ризик їх пошкодження за умови секційного розміщення шарами.

Для реалізації моделі прийняття рішень ASL завантажувальних дверей терміналів обрано ANFIS [13]. З використанням заданих значень вхідних і вихідних наборів даних нечітка модель ANFIS реалізує нечітке визначення номера завантажувального лотка, основаного на заданому методі навчання. На цей час тривають дослідження різних методів навчання ANFIS [14] для підвищення її продуктивності. Універсальність моделі ANFIS зумовлює її широке застосування в різних галузях промисловості [15].

4.2. Визначення структури ANFIS відповідно до мети дослідження

Для розроблення й проведення досліджень використовувався інструментарій створення та навчання ANFIS, реалізований у середовищі MATLAB (версія R2016a-9.0.0.341360). Розроблена в MATLAB нечітка модель ANFIS має чотири входи й один вихід (рис. 3) і відповідає моделі Такагі – Сугено (TS) першого порядку.

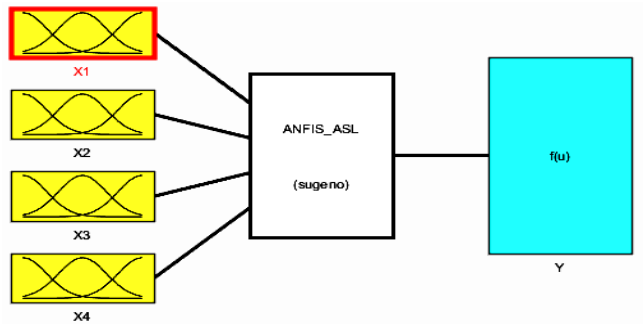


Рис. 3. Нечітка модель ANFIS

Модель ANFIS має чотири входи й один вихід. Вхідними нечіткими параметрами є: x_1 – вага, x_2 – висота, x_3 – ширина й x_4 – глибина ОПВ. Вихідним нечітким параметром є номер завантажувального лотка LC, що позначається як y. Згідно з нечіткою моделлю Такагі – Сугено першого порядку правила подаються в такому вигляді [13]:

$$\begin{gathered} \mathrm{R_k}:\ \mathrm{IF}\left(x_1 \in A_{1q}\right) \text{ and } \mathrm{IF}\left(x_2 \in A_{2j}\right) \text{ and } \mathrm{IF}\left(x_3 \in A_{3m}\right) \text{ and } \mathrm{IF}\left(x_4 \in A_{4s}\right) \to f_k, \\ f_k = \mathrm{c_{0k}} + \mathrm{c_{1k}} x_1 + \mathrm{c_{2k}} x_2 + \mathrm{c_{3k}} x_3 + \mathrm{c_{4k}} x_4, \end{gathered} \tag{1}$$

де R_k – нечітке правило з індексом k, k = 1, 2, 3, ..., K; $A_{i(q,j,m,s)}$ – нечіткі множини, визначені для вхідних параметрів x_i з індексом $i = \{1,\ 2,\ 3,\ 4\}$; q, j, m, s – індекси умов у антецедентах правил R_k, які обираються незалежно від індексу k; f_k – консеквент правила R_k; c_{0k}, c_{ik} –

коефіцієнти полінома першого порядку. Операція визначення ступеня належності чіткої змінної x_i до множини $A_{i(q,j,m,s)}$ в умовах антецедентів правил (1) є знаходженням значення гауссової функції належності:

$$\mu_{Aij}\left(x_i\right) = \exp\left[-\frac{1}{2}\left(\frac{x_i - a_{ij}}{b_{ij}}\right)^2\right], \tag{2}$$

де a_{ij}, b_{ij} – параметри, що визначаються в процесі навчання ANFIS.

Спрощена структура ANFIS, встановлена відповідно до її нечіткої моделі (рис. 3) для трьох правил (1), подана на рис. 4. Кількість правил нечіткої моделі ANFIS уточнюється під час визначення логіки сортування ОПВ.

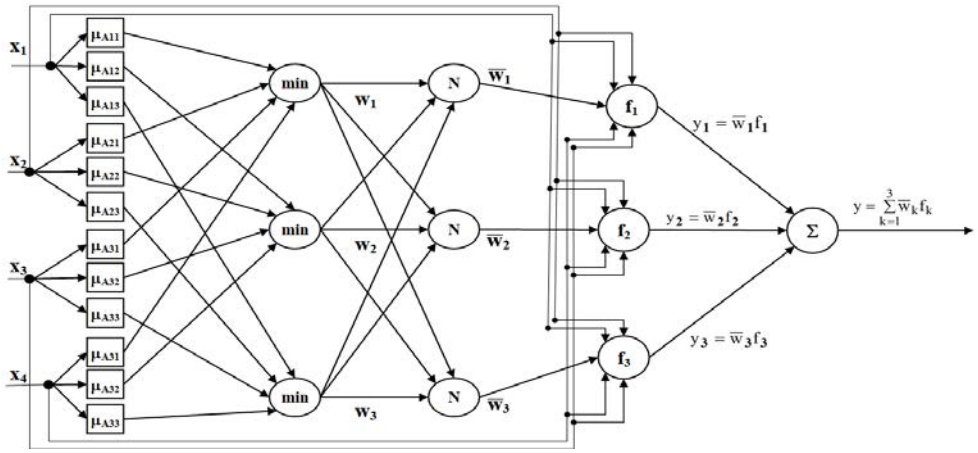


Рис. 4. Структура ANFIS

Структура моделі ANFIS (рис. 4) містить п’ять шарів:

– перший шар відповідає за обчислення значень функції належності $\mu_{Aij}\left(x_i\right)$ для вхідних значень (рівень фазифікації);

– другий шар відповідає за агрегацію ступенів істинності умов кожного k-го правила (1) відповідно до операції T-норми, обчислюючи вагові коефіцієнти w_k. Значення вагового коефіцієнта w_k визначається як мінімальне значення з множини значень функцій належності, розрахованих у першому шарі:

$$w_{\mathrm{k}} = \min\{\mu_{\mathrm{A1j}}\left(\mathrm{x}_1\right), \mu_{\mathrm{A2j}}\left(\mathrm{x}_2\right), \mu_{\mathrm{A3j}}\left(\mathrm{x}_3\right), \mu_{\mathrm{A4j}}\left(\mathrm{x}_4\right)\};$$

– третій шар відповідає за нормалізацію вагових коефіцієнтів w_k (обчислення відносного рівня істинності k-го правила):

$$\overline{w}_k = w_k \Big/ \sum_k w_k, \qquad k = 1,\ 2,\ 3;$$

– четвертий шар відповідає за обчислення добутку відносного рівня істинності k-го правила на його консеквент f_k (рівень дефазифікації):

$$y_k = \overline{w}_k f_k = \overline{w}_k\left(\mathrm{c}_{0\mathrm{k}} + \mathrm{c}_{1\mathrm{k}} x_1 + \mathrm{c}_{2\mathrm{k}} x_2 + \mathrm{c}_{3\mathrm{k}} x_3 + \mathrm{c}_{4\mathrm{k}} x_4\right);$$

– п’ятий шар, що містить один нейрон, відповідає за адаптивне підсумовування вихідних даних попереднього шару:

$$y = \sum_{k=1}^{3} \overline{w}_k f_k = \sum_{k=1}^{3} w_k f_k \Big/ \sum_{k=1}^{3} w_k .$$

4.3. Визначення діапазонів ваги й габаритів ОПВ

Для кожного вхідного параметра x_i, $i \in \{1,2,3,4\}$, визначено по три діапазони значень, що відповідають трьом завантажувальним лоткам LC з індексами (номерами) $q = j = m = s = 1, 2, 3$, і позначених як T_{iq}, T_{ij}, T_{im}, T_{is}.

Значення діапазонів наведено в табл. 1.

Таблиця 1. *Діапазони значень параметрів ОПВ*

Вхідні параметри	Познаки діапазонів	Діапазони значень параметрів ОПВ для завантажувальних лотків		
		для лотка LC1	для лотка LC2	для лотка LC3
x_1, вага (кг)	T_{1q}	T_{11} – [0, 14.9]	T_{12} – [15, 44.9]	T_{13} – [45, 60]
x_2, висота (см)	T_{2j}	T_{21} – [0, 39.9]	T_{22} – [40, 119.9]	T_{23} – [120, 160]
x_3, ширина (см)	T_{3m}	T_{31} – [0, 39.9]	T_{32} – [40, 119.9]	T_{33} – [120, 160]
x_4, глибина (см)	T_{4s}	T_{41} – [0, 39.9]	T_{42} – [40, 119.9]	T_{43} – [120, 160]

4.4. Визначення логіки сортування ОПВ

Навчальна вибірка ANFIS має відтворювати логіку сортування ОПВ за трьома завантажувальними лотками. Визначимо кількість правил нечіткого виведення ANFIS, що використовуються в (1): $K = G^L = 4^3 = 81$, де $G = 4$ – кількість параметрів ОПВ, $L = 3$ – кількість завантажувальних лотків.

Матриця логіки сортування, що визначається індексами завантажувальних лотків $\{q, j, m, s\}$, подана в табл. 2 (у скороченому вигляді). Ці індекси визначають належність вхідних параметрів x_i навчальної вибірки до діапазонів T_{1q}, T_{2j}, T_{3m}, T_{4s}, поданих у табл. 1.

Таблиця 2. *Матриця логіки сортування ОПВ*

Правило k	Індекси виконаних умов у антецедентах правил				Номер лотка LC
	$x_1 \in T_{1q}$	$x_2 \in T_{2j}$	$x_3 \in T_{3m}$	$x_4 \in T_{4s}$	
	індекс q	індекс j	індекс m	індекс s	
1	1	1	1	1	1
2	1	1	1	2	
3	1	1	1	3	
4	1	1	2	1	

Кінець таблиці 2.

5	1	1	3	1	
6	1	2	1	1	
7	1	3	1	1	
8	1	1	2	2	2
9	1	1	2	3	
10	1	1	3	2	
11	1	1	3	3	
12	1	2	1	2	
13	2	1	1	1	
...	...	...	...	...	
47	2	3	2	2	
48	2	1	3	3	
49	2	2	3	3	
50	2	3	1	3	
51	2	3	2	3	
52	2	3	3	1	
53	3	1	1	1	
...	...	...	...	...	
81	3	3	3	3	

Для завдання логіки сортування необхідно обрати індекси $\{q, j, m, s\}$ виконаних умов правила (1) і номер завантажувального лотка, що відповідає цим умовам. Для цього використовуються два критерії контролю ваги й габаритів посилки:

1) якщо в правилі не більше ніж одна виконана умова з індексом $\{j,m,s\}$, значення якого перевищує індекс q, то ОПВ транспортується до завантажувального лотка з номером q;

2) якщо в правилі два й більше індексів виконаних умов з $\{j,m,s\}$ перевищують значення індексу q, то ОПВ транспортується до завантажувального лотка з номером $q + 1$. Якщо $(q + 1) > 3$, то $q = 3$.

4.5. Визначення параметрів наборів даних для навчання й тестування ANFIS

Для навчання ANFIS використано чотири набори даних, які формувалися способом задання кортежів значень $< x_1, x_2, x_3, x_4, y_T >$, що відповідають діапазонам значень з табл. 1 і матриці логіки сортування ОПВ (табл. 2).

Для тестування навченої ANFIS використано набір даних *dataset3* у вигляді кортежів $< x_1, x_2, x_3, x_4, y_v >$.

Параметри наборів даних для навчання й тестування ANFIS подано в табл. 3.

Таблиця 3. *Параметри наборів даних для навчання й тестування ANFIS*

Призначення наборів даних	Кількість значень для діапазонів $T_{i(q,j,m,s)}$				Загалом кортежів в наборі (N)	Найменування наборів даних
	T_{1q}	T_{2j}	T_{3m}	T_{4s}		
Навчання	3	3	3	3	$3^4 = 81$	*dataset1*
	6	6	6	6	$6^4 = 1296$	*dataset2*
	9	9	9	9	$9^4 = 6561$	*dataset3*
Тестування	13	13	13	13	$13^4 = 28561$	*dataset4*

5. Обговорення результатів дослідження

Для навчання ANFIS впроваджено два методи, реалізовані в MATLAB: зворотного поширення помилки (*backpropa*) і гібридний (*hybrid*), що об'єднує метод зворотного поширення помилки з методом найменших квадратів. Навчання проводилося для кожного навчального набору даних протягом 5, 25, 50 і 100 епох із фіксацією значення середньоквадратичної помилки навчання.

Навчені ANFIS протестовано з використанням вибірки *dataset4*, параметри якої подано в табл. 3. У процесі тестування навчених ANFIS вихідні дійсні числові значення їх нечітких моделей (номери завантажувальних лотків) округлювалися до цілих відповідно до правил математики. Для оцінювання якості навчених моделей ANFIS фіксувалися істинні (TC – *True Classification*) і помилкові (FC – *False Classification*) результати нечіткої класифікації ОПВ за номерами завантажувальних лотків. Метрикою оцінювання якості застосовано відносну точність правильної класифікації (RP – *Relative Precision*), яка визначається відповідно до виразу

$$RP = \frac{TC}{TC + FC} = \frac{TC}{N}, \tag{3}$$

де N – кількість кортежів у наборі даних для тестування.

Результати дослідження продемонстровано в табл. 4, де подано значення середньоквадратичної помилки (СКП) навчання й оцінки відносної точності правильної класифікації (RP) залежно від кількості епох навчання.

Дослідження переконало, що гібридний метод навчання ANFIS є кращим, оскільки забезпечує мінімальні помилки, якщо порівнювати з методом зворотного поширення помилки.

Відповідно до результатів дослідження (табл. 4) для реалізації обрано модель ANFIS, навчену гібридним методом протягом 100 епох. Ця модель забезпечує мінімальну помилку навчання й задовольняє вимогу, що визначена в постановці задачі щодо забезпечення відносної точності класифікації ОПВ не менше ніж 95 %.

Аналіз показників навчання моделей ANFIS з гібридним методом навчання дає змогу сформулювати висновок, що більш значний вплив на відносну точність нечіткої класифікації ОПВ має розмір даних навчальної вибірки.

Отримана оцінка відносної точності класифікації практично не залежить від кількості епох навчання (табл. 4). Це пов'язано з тим, що середньоквадратична помилка навчання від 5 до 100 епох зменшується повільно (для набору даних *dataset2* – 5,89 %, а для *dataset3* – лише 0,53 %).

Зазначене впливає на те, що вихідні дійсні числові значення ANFIS також змінюються незначно, а після округлення до цілих обчислюване значення відносної точності класифікації практично не змінюється.

Таблиця 4. *Результати навчання й тестування ANFIS*

Показники якості навчання	Кількість епох навчання ANFIS				Набори даних для навчання
	5	**25**	**50**	**100**	
Метод навчання "Зворотне поширення помилки" (*backpropa*)					
СКП	0,8614	0,1692	0,1682	0,1667	*dataset1*
RP, %	38,9 %	79,9 %	80,4 %	80,7 %	
СКП	0,9133	0,2537	0,2415	0,2352	*dataset2*
RP, %	48,8 %	90,7 %	91,7 %	91,4 %	
СКП	0,2907	0,3385	0,3325	0,3307	*dataset3*
RP, %	48,9 %	93,3 %	94,1 %	94,2 %	
Гібридний метод навчання (*hybrid*)					
СКП	1.54e -5	1.54e -5	1.54e -5	1.54e -5	*dataset1*
RP, %	86 %	86 %	86 %	86 %	
СКП	0,0407	0,0403	0,0397	0,0383	*dataset2*
RP, %	94,3 %	94,3 %	94,3 %	94,3 %	
СКП	0,2812	0,2809	0,2805	0,2797	*dataset3*
RP, %	96,3 %	96,3 %	96,3 %	96,3 %	

Вхідні функції належності нечіткої моделі навченої ANFIS для параметрів ОПВ x_1 та x_2 подаються на рис. 5 і рис. 6 відповідно.

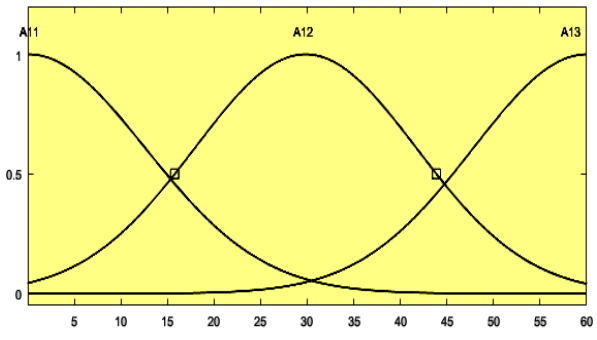


Рис. 5. Функції належності для ваги (x_1)

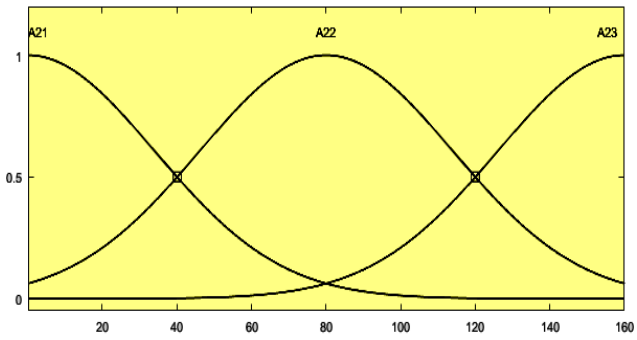


Рис. 6. Функції належності для висоти (x_2)

Параметри вхідних гауссових функцій належності (2) нечіткої моделі обраної ANFIS, визначені в процесі навчання, подано в табл. 5.

Нечітка модель навченої ANFIS містить 81 правило нечіткого виведення, що відповідає табл. 2.

Таблиця 5. *Параметри вхідних функцій належності навченої ANFIS*

Функція належності	*gaussmf(b,a)*		Функція належності	*gaussmf(b,a)*	
	b	*a*		*b*	*a*
$\mu_{A11}(x_1)$	12,56	0,098	$\mu_{A31}(x_3)$	33,94	0,093
$\mu_{A12}(x_1)$	11,92	29,834	$\mu_{A32}(x_3)$	33,93	80,05
$\mu_{A13}(x_1)$	12,22	60	$\mu_{A33}(x_3)$	33,94	160
$\mu_{A21}(x_2)$	33,94	0,093	$\mu_{A41}(x_4)$	33,94	0,093
$\mu_{A22}(x_2)$	33,93	80,05	$\mu_{A42}(x_4)$	33,93	80,05
$\mu_{A23}(x_2)$	33,94	160	$\mu_{A43}(x_4)$	33,94	160

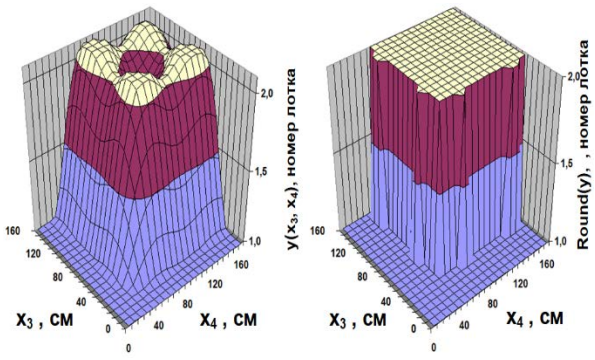
Значення коефіцієнтів консеквентів f_k у правилах нечіткої моделі (1), визначені в процесі навчання для індексів умов логіки сортування ОПВ, запропоновано в табл. 6.

Таблиця 6. *Значення коефіцієнтів консеквентів у правилах нечіткої моделі ANFIS*

Правило k	Індекси умов				Визначені коефіцієнти консеквентів f_k у правилах (1)				
	q	j	m	s	c_{0k}	c_{1k}	c_{2k}	c_{3k}	c_{4k}
1	1	1	1	1	-1,13E-01	8,27E-03	8,27E-03	8,27E-03	9,26E-01
2	1	1	1	2	-1,42E-01	-3,11E-02	-3,11E-02	8,42E-03	-4,20E-01
3	1	1	1	3	-1,41E-01	-2,46E-02	-2,46E-02	5,79E-03	-5,93E-01
4	1	1	2	1	-1,42E-01	-3,11E-02	8,42E-03	-3,11E-02	-4,20E-01
5	1	1	3	1	-1,41E-01	-2,46E-02	5,79E-03	-2,46E-02	-5,93E-01
6	1	2	1	1	-1,42E-01	8,42E-03	-3,11E-02	-3,11E-02	-4,20E-01
7	1	3	1	1	-1,41E-01	5,79E-03	-2,46E-02	-2,46E-02	-5,93E-01
8	1	1	2	2	-3,08E-02	4,03E-03	-3,15E-02	-3,15E-02	7,45E+00
9	1	1	2	3	-3,93E-02	4,45E-03	-2,35E-02	-2,15E-02	7,63E+00
10	1	1	3	2	-3,93E-02	4,45E-03	-2,15E-02	-2,35E-02	7,63E+00
11	1	1	3	3	-8,52E-02	-2,21E-03	-1,47E-02	-1,47E-02	6,67E+00
12	1	2	1	2	-3,08E-02	-3,15E-02	4,03E-03	-3,15E-02	7,45E+00
13	2	1	1	1	-1,40E-01	-1,75E-03	-1,75E-03	-1,75E-03	6,22E+00
...	...	...	...	...	...	...	...	...	...
47	2	3	2	2	-6,64E-02	4,45E-03	-3,34E-02	-3,34E-02	7,95E+00
48	2	1	3	3	-8,47E-02	6,11E-03	-2,76E-02	-2,76E-02	1,51E+01
49	2	2	3	3	-7,98E-02	8,94E-03	-3,33E-02	-3,33E-02	1,62E+01
50	2	3	1	3	-8,47E-02	-2,76E-02	6,11E-03	-2,76E-02	1,51E+01
51	2	3	2	3	-7,98E-02	-3,33E-02	8,94E-03	-3,33E-02	1,62E+01
52	2	3	3	1	-8,47E-02	-2,76E-02	-2,76E-02	6,11E-03	1,51E+01
53	3	1	1	1	-1,14E-01	3,40E-04	3,40E-04	3,40E-04	1,00E+01
...	...	...	...	...	...	...	...	...	...
81	3	3	3	3	-5,40E-02	-1,06E-03	-1,06E-03	-1,06E-03	6,79E+00

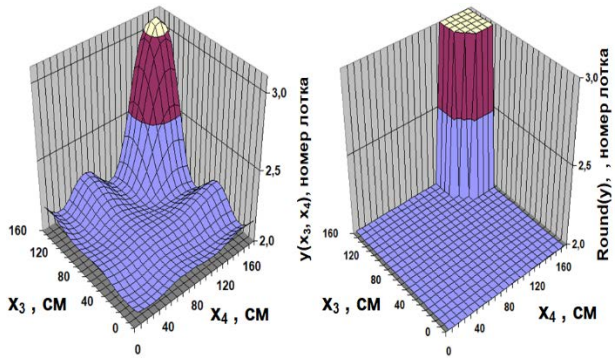
Узагальнені результати навченої ANFIS подано у вигляді її функцій відгуків для фіксованих діапазонів ваги й висоти ОПВ.

На рис. 7 і рис. 8 зображено дві пари функцій відгуків ANFIS, що відтворюють залежність вихідного значення номера лотка у вигляді дійсного числа та його значення, округленого до цілого.



а) $y(x_3, x_4)$ б) Round$\{y(x_3, x_4)\}$

Рис. 7. Функції відгуків $(x_1 \in T_{11},\ x_2 \in T_{21})$



а) $y(x_3, x_4)$ б) Round$\{y(x_3, x_4)\}$

Рис. 8. Функції відгуків $(x_1 \in T_{12},\ x_2 \in T_{22})$

На рис. 7 подано функцію відгуку ANFIS для фіксованого діапазону ваги й висоти ОПВ першого завантажувального лотка LC1 $(x_1 \in T_{11},\ x_2 \in T_{21})$ і двох вхідних параметрів – ширини (x_3) і висоти (x_4), значення яких змінюються в усіх трьох діапазонах, визначених у табл. 2.

Другий варіант функції відгуку ANFIS (рис. 8) подано для фіксованого діапазону ваги й висоти ОПВ другого завантажувального лотка LC2 $(x_1 \in T_{12},\ x_2 \in T_{22})$ з аналогічними діапазонами зміни ширини (x_3) й висоти (x_4).

Результати моделювання підтверджують, що навчена ANFIS реалізує задану логіку сортування ОПВ відповідно до завантажувальних лотків у потоці конвеєра.

6. Висновки й перспективи подальших досліджень

Розроблена модель прийняття рішень ASL у вигляді адаптивної нейронечіткої системи виведення й додаткове оснащення дверей терміналів PSC завантажувальними лотками дають змогу реалізувати потокове сортування ОПВ без зміни швидкості їх транспортування по конвеєру.

Реалізована логіка сортування забезпечує поетапне завантаження транспортних засобів по секціях. Кожна секція містить три шари ОПВ, які укладаються послідовно відповідно до зменшення їх ваги й габаритів, що знижує ризик їх пошкодження.

Недоліком розробленої моделі прийняття рішень ASL є обмеження умов сортування, яке визначається кількістю завантажувальних лотків, що відповідають кількості діапазонів ваги й габаритів ОПВ.

З огляду на зазначений недолік можна окреслити такі напрями подальшої роботи:

– дослідження варіантів потокового завантаження ОПВ на завантажувальних дверях терміналів PSC залежно від обсягу кузова вантажівки;

– визначення можливих варіантів навчання ANFIS для різної кількості завантажувальних лотків, що встановлює кількість умов сортування ОПВ;

– порівняння можливих варіантів програмної реалізації ANFIS на базі сучасних мікроконтролерів, спеціалізованих ANFIS-контролерів, пропорційно-інтегральних і пропорційно-інтегрально-диференціальних регуляторів.

Отже, розроблена модель прийняття рішень у вигляді ANFIS забезпечує сортування ОПВ в умовах конвеєрного потоку, а отже, її рекомендовано для реалізації в ASL різного призначення.

References

1. Jarrah Ahmad, I., Xiangtong, Qi, Bard, J.F. (2014), "The Destination-Loader-Door Assignment Problem for Automated Package Sorting Centers", *Transportation Science*, No 50(4), P. 1314–1336. DOI: https://doi.org/10.1287/trsc. 2014.0521
2. Grebennik, I. V., Kovalenko, O. A. (2024), "The Logic-Algebraic Model of Decision-Making for an Automated Parcel Sorting Conveyor", *Management Information System and Devises*, No 1 (183), P. 5–14. DOI: https://doi.org/10.30837/0135-1710.2024.183.005
3. Grebennik, I., Kovalenko, O. (2024), "Realisation of a Given Trucks Loading Logic using a Fuzzy Decision Making Model", *14th International Conference on Advanced Computer Information Technologies (ACIT), IEEE,* P. 27–31. DOI: https://doi.org/10.1109/ACIT62333.2024.10712615
4. Bugow, S., Kellenbrink, C. (2023), "The Parcel Hub Scheduling Problem With Limited Conveyor Capacity and Controllable Unloading Speeds", *OR Spectrum*, Vol. 45, P. 325–357. DOI: https://doi.org/10.1007/s00291-022-00702-y
5. McWilliams, D. L., Stanfield, P. M., Geiger, C. D. (2005), "The Parcel Hub Scheduling Problem: A Simulation-Based Solution Approach", *Computers & Industrial Engineering*, Vol. 49(3), P. 393–412. DOI: https://doi.org/10.1016/j.cie.2005.07.002
6. Chen, J. C., Chen, T. L, Lee, Y. H. (2023), "Simulation optimization for parcel hub scheduling problem in closed-loop sortation system with shortcuts", *Simulation Modelling Practice and Theory*, Vol. 124(10):102728. DOI: https://doi.org/10.1016/j.simpat.2023.102728
7. Chen. J. C., Chen, T. L, Ou, T. C., Lee, Y. H. (2019), "Adaptive Genetic Algorithm for Parcel Hub Scheduling Problem With Shortcuts in Closed-Loop Sortation System", *Computers & Industrial Engineering*, Vol. 138:106114. DOI: https://doi.org/10.1016/j.cie.2019.106114
8. Romanova, T., Litvinchev, I., Grebennik, I., Kovalenko, A., Urniaieva, I. (2020), "Packing Convex 3D Objects with Special Geometric and Balancing Conditions", *Advances in Intelligent Systems and Computing*, Vol. 1072. DOI: https://doi.org/10.1007/978-3-030-33585-4_27
9. Karami, F., Fathi, M., Pardalos, P. M. (2023), "Conveyor Operations in Distribution Centers: Modeling and Optimization", *Optim Lett* , Vol. 17, P. 1049–1068. DOI: https://doi.org/10.1007/s11590-022-01912-7

10. Chen, T.-L., Chen, J. C., Huang, C.-F., Chang, P.-C. (2021), "Solving the Layout Design Problem by Simulation-Optimization Approach – A case study on a sortation conveyor system", *Simulation Modelling Practice and Theory*, Vol. 106:102192. DOI: https://doi.org/10.1016/ j.simpat.2020.102192
11. Fedtke, S., Boysen, N. (2014), "Layout Planning of Sortation Conveyors in Parcel Distribution Centers. *Transportation Science*, Vol. 51(1):3–18. DOI: https://doi.org/10.1287/trsc.2014.0540
12. Werners, B., Wülfing, T. (2010), "Robust Optimization of Internal Transports at a Parcel Sorting Center Operated by Deutsche Post World Net", *Eur. J. Oper. Res*, Vol. 201(2):419–426. DOI: https://doi.org/10.1016/j.ejor.2009.02.035
13. Jang, J.-S. R. (1993), "ANFIS: Adaptive-netWork-Based Fuzzy Inference System", *Transactions on Systems, Man, and Cybernetics*, Vol. 23(3), P. 665-685. DOI: https://doi.org/10.1109/21.256541
14. Karaboga, D., Kaya, E. (2019), "Adaptive Network Based Fuzzy Inference System (ANFIS) Training approacHes: A Comprehensive Survey", *Artif Intell Rev*, Vol. 52, P. 2263–2293. DOI: https://doi.org/10.1007/s10462-017-9610-2
15. Navneet-Kaur, W., Singh, H., Sharma, A. (2015), "ANFIS: Adaptive Neuro-Fuzzy Inference System – A Survey", *International Journal of Computer Applications*, Vol. 123, P. 32–38. DOI: https://doi.org/10.5120/ijca2015905635

Received (Надійшла) 01.10.2025
Accepted for publication (Прийнята до друку) 30.11.2025
Publication date (Дата публікації) 28.12.2025

Відомості про авторів / About the Authors

Гребеннік Ігор Валерійович – доктор технічних наук, професор, Харківський національний університет радіоелектроніки, завідувач кафедри системотехніки, Харків, Україна; e-mail: igor.grebennik@nure.ua; ORCID ID: https://orcid.org/0000-0003-3716-9638

Коваленко Олексій Андрійович – Харківський національний університет радіоелектроніки, аспірант, кафедра системотехніки, Харків, Україна; e-mail: oleksii.kovalenko3@nure.ua; ORCID ID: https://orcid.org/0009-0008-4779-6161

Grebennik Igor – Doctor of Sciences (Engineering), Professor, Kharkiv National University of Radio Electronics, Head of System Engineering Department, Kharkiv, Ukraine.

Kovalenko Oleksii – Kharkiv National University of Radio Electronics, PhD Student at the System Engineering Department, Kharkiv, Ukraine.

ADAPTIVE NEURO-FUZZY INFERENCE SYSTEM FOR SORTING POSTAL ITEMS IN A CONVEYOR STREAM

The **subject matter** of the article is decision-making models for automated sorting lines (ASL) used in parcel sorting center (PSC) hubs that are part of the postal delivery logistics network. The **goal** of the work is to develop an ASL decision-making model for PSC terminal loading doors in the form of an adaptive neuro-fuzzy inference system (ANFIS). The following **tasks** were addressed in the article:

an analysis of the design features of ASL equipment was conducted, and a modification was proposed – equipping terminal doors with three loading chutes instead of one; the ANFIS structure was determined in accordance with the first-order Takagi-Sugeno fuzzy model; the weight and size ranges of parcels associated with loading tray numbers were determined; criteria for implementing the specified parcel sorting logic were determined; the parameters of the data sets for training and testing ANFIS were determined; comparative training and testing of ANFIS was carried out using the MATLAB environment tools with an assessment of the relative precision values of correct parcel classification. The following **methods** are used – systemic, analytical, computer modeling, ANFIS architecture learning methods, mathematical and statistical analysis of learning effectiveness. The following **results** were obtained: comparative training of ANFIS was conducted using a hybrid method and a backpropagation method. Based on the test results after 100 training epochs, the ANFIS model trained using the hybrid method was selected for implementation, which provides a relative accuracy of parcel classification at the level of 96.3 %. **Conclusions**: the developed ASL decision-making model in the form of ANFIS allows for the implementation of fuzzy classification of parcels in the conveyor flow according to three ranges of their weight and size parameters for the implementation of the specified sorting logic by loading chutes. The implemented sorting logic ensures compact loading of truck bodies by sections and reduces the risk of damage to parcels when they are stacked on top of each other.

Keywords: postal items; sorting; decision-making model; automated line; fuzzy classification; adaptive neuro-fuzzy inference system.

Бібліографічні описи / Bibliographic descriptions

Гребеннік І. В., Коваленко О. А. Адаптивна нейронечітка система виведення для сортування об’єктів поштових відправлень у конвеєрному потоці. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 47–62. DOI: https://doi.org/10.30837/0135-1710.2025.187.047

Grebennik, I., Kovalenko, O. (2025), "Adaptive neuro-fuzzy inference system for sorting postal items in a conveyor stream", *Management Information Systems and Devises*, No. 4 (187), P. 47–62. DOI: https://doi.org/10.30837/0135-1710.2025.187.047

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.056.55:004.312.2 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.063

S. Yenhalychev, S. Semenov

EXPERIMENTAL STUDIES OF THE TASK PLANNING METHOD IN DISTRIBUTED INFORMATION SYSTEMS TAKING INTO ACCOUNT METADATA UNCERTAINTY

The article is devoted to task planning in distributed information systems under conditions of uncertainty and incompleteness of metadata regarding the duration of operations, communication volumes, and resource availability. **The subject of the study** is the method of task planning in distributed information systems, taking into account the uncertainty of metadata. **The purpose of the article** is to experimentally investigate a method for planning tasks in distributed information systems with uncertain and incomplete metadata, which combines a hybrid graph model with a three-module artificial intelligence core and provides increased efficiency and robustness of planning while complying with SLO and RBAC constraints. **Research objectives:** to form a hybrid representation of workflows based on a combination of DAG structure and GERT model; to develop a planner architecture incorporating a GAT graph attention network, a PPO reinforcement learning agent, and a Bayesian Optimization module; to define an SLO-oriented reward function and constraint system, taking into account RBAC; to create a reproducible experimental test bed with controlled uncertainty injection; to perform comparative, ablation, and robustness analysis of the proposed method's productivity relative to the baseline FCFS and HEFT algorithms, as well as DAG-only and GERT-only variants. **Research methods:** hybrid graph modeling DAG+GERT, graph attention network GAT for forming a contextual state representation, PPO agent for making decisions on task placement, Bayesian optimization for tuning policy parameters, and statistical evaluation methods. **Main results.** The proposed method provides a significant reduction in the median task completion time (*makespan*) compared to HEFT, DAG-only, and GERT-only, an increase in the proportion of tasks performed within SLO, and a reduction in weighted average delay, demonstrates "tail compression" of the delay distribution, and maintains stability of results in the case of multiplicative noise of durations ±20 %, subject to distribution shifts and correlated parameter deviations. At the same time, planning overhead remains in the millisecond range and does not exceed acceptable limits for online scenarios.

Keywords: distributed information systems; task scheduling; directed acyclic graph; GERT networks; graph neural networks; reinforcement learning; Bayesian Optimization; service levels (SLO); access control (RBAC); robustness.

Introduction

In modern distributed information systems, it is increasingly necessary to plan large sets of dependent tasks under conditions of variable load, resource heterogeneity, and incomplete input metadata. For such systems, it is critical to simultaneously minimize the completion time of a set of tasks (makespan), ensure compliance with service levels (SLO – service level objective), and take into account role-based access control (RBAC) policies.

Classic planning approaches focus on directed acyclic graphs (DAG), in which vertices correspond to tasks and edges correspond to dependencies and data exchanges. A broad class of heuristic algorithms is based on this model, including HEFT (Heterogeneous Earliest Finish Time) and its numerous variants, which demonstrate high-quality scheduling

in heterogeneous computing environments with deterministic estimates of task and communication durations.

However, in real distributed systems, the duration of operations, the bandwidth of communication channels, and even the structures of workflows themselves change under the influence of external factors, and the available metadata is noisy, partially outdated, or incomplete. This requires robust scheduling methods capable of maintaining acceptable schedule quality under stochastic and adversarial parameter perturbations. The robustness of graph schedules is being actively researched, but the vast majority of work is either limited to static stochastic duration models or focuses on optimizing a single criterion, primarily makespan.

An additional complexity is the need to model not only sequential dependencies, but also probabilistic branches, repetitions, conditional transitions, and failures inherent in real-world event processing and telemetry scenarios. For such tasks, it is natural to use GERT (Graphical Evaluation and Review Technique) networks, which allow the inclusion of branch passage probabilities and duration distributions in the model, but historically have been used primarily in project management rather than in operational planning tasks in information systems.

In recent years, intelligent planning methods using graph neural networks and reinforcement learning have been actively developed to optimize scheduling in complex environments. A number of approaches have been proposed in which a reinforcement learning agent makes decisions about the placement of tasks in a DAG, and graph networks (GCN, GNN) or graph attention (GAT – Graph Attention Network) are used to construct a topology-sensitive state. Despite significant progress, most of these works either do not take into account the probabilistic nature of branches or do not integrate SLO requirements and access policies into a single model.

In this work, these problems are solved by combining a hybrid DAG+GERT graph model with a three-module planner core based on artificial intelligence methods, which allows us to explicitly describe the uncertainty in the execution structure and task parameters, and to train an adaptive planning policy focused on SLO and access restrictions.

Literature review

The issue of task scheduling in parallel and distributed systems has traditionally been considered within the framework of deterministic models, where task durations and exchanges are fixed or accurately estimated. Classic monographs and reviews [1, 2] focus on building effective heuristics for DAG representations, including clustering, leaf scheduling, and task duplication methods.engr.colostate.edu One of the most cited algorithms is HEFT (Heterogeneous Earliest Finish Time) [3, 4], which provides a good compromise between schedule quality and computational complexity for heterogeneous environments. Based on HEFT, a large number of modifications (E-HEFT, Rob-HEFT, PDHEFT, etc.) [5, 6] have been proposed that take into account load balancing, power consumption, or specific hardware constraints.

However, deterministic models poorly reflect the dynamics of modern distributed systems, where task durations and network delays depend on current load, queues, and external factors.

This has led to the development of robust planning [7, 8], which considers stochastic or interval durations, and the quality of the schedule is evaluated not only by the expected makespan, but also by the distribution of this indicator, slack, and sensitivity to disturbances. A number of robustness metrics and corresponding heuristics have been proposed, particularly for network and cluster environments, but most approaches remain static and do not use online policy updates based on telemetry [9].

A separate class of works [10, 11] is devoted to the application of deep reinforcement learning (DRL) to DAG planning problems in cloud and cluster systems. In these works, the DRL agent learns on a simulated environment or real telemetry to minimize makespan, energy consumption, or combined criteria.

A number of studies [12, 13] demonstrate the advantages of DRL approaches over classical heuristics under variable load conditions, but most of them rely on pure DAG models and do not take into account probabilistic branches, failures, or cyclic fragments of workflows.

In parallel, graph neural networks (GNN) [14, 15] are being developed for planning and optimization tasks. Schemes are proposed in which GCN or GAT modules build representations of task graphs (job shop, workflow, access network), and these representations are then used either as a heuristic function [16] or as a policy component [17]. It has been shown that GNN approaches scale better with respect to graph size and topology complexity than classical tabular or manually designed features. However, most works work with deterministic DAG structures and do not use the combined power of stochastic networks such as GERT to model uncertainty.

GERT-type networks [18] were historically developed for analyzing stochastic networks in project management and operations research problems. They allow modeling the probability of branch traversal, as well as the distribution of time or other additive attributes on edges, making them a natural tool for describing complex technological schemes with branches and cycles [19]. Despite this, the use of GERT in the context of task planning in distributed information systems remains limited, and the combination of GERT with modern GNN/DRL methods has hardly been explored.

Another important component of real-world systems is access policies, which determine which tasks can be performed on which resources. The most commonly used formal model here is role-based access control (RBAC), standardized by NIST and widely implemented in corporate environments [20]. In parallel with this, the area of service reliability engineering has developed an established practice of operating with service indicators and service level objectives (SLOs), which serve as the basis for making decisions about operational trade-offs [21]. Existing works on DRL planning largely lack the integration of SLO-oriented metrics and RBAC constraints into a single learning framework.

Thus, analysis of the literature reveals the following gaps:

- Insufficient use of stochastic networks such as GERT to model uncertainty in the structure and parameters of workflows.

- Limited integration of graph neural networks and DRL agents with GERT representation in planning tasks.

- Almost complete absence of works in which the planning policy is simultaneously optimized for makespan, SLO-oriented indicators, and compliance with RBAC constraints.

The purpose of the article is to experimentally investigate a method for planning tasks in distributed information systems with uncertain and incomplete metadata, which combines a hybrid graph model with a three-module artificial intelligence core (GAT, PPO, Bayesian optimization) and provides increased efficiency and robustness of planning while complying with SLO and RBAC constraints.

To achieve this goal, the following main tasks are solved.

1. Form a hybrid model of work processes based on a combination of a DAG structure with a GERT network, which allows simultaneously describing deterministic dependencies, probabilistic branches, repetitions, and failures.

2. Develop a scheduler architecture in which a graph attention network (GAT) builds a representation of the system state, a deep reinforcement learning agent (PPO) makes decisions about task placement, and a Bayesian optimization module performs automated tuning of hyperparameters and threshold criteria.

3. Formulate an SLO-oriented reward function and constraint system that simultaneously considers makespan, deadline probability, and compliance with RBAC policies.

4. Build a reproducible experimental testbed with controlled load scenarios, metadata uncertainty models, and sets of baseline algorithms (FCFS, HEFT, simplified GERT/DAG modes) for comparison.

Conduct a comprehensive experimental analysis, including assessment of robustness to parameter perturbations, ablation analysis of the contribution of individual modules (GAT, PPO, BO), and statistically sound comparison with baseline approaches.

Main part

1. Formulation of research hypotheses, experimental design, and performance indicator system

Let us begin with some terminological explanations. In this study, metadata is understood as all parameters that describe the behavior of tasks and operations in the DAG+GERT model and are transmitted to the scheduler as input information. Formally, the set of metadata is defined by:

$$\mathsf{M} = \left\{\tau_i, \sigma_i^2, c_{ij}, p_{ij}, r_i, D_i, w_i, RBAC_i\right\},$$

where: τ_i – task duration estimation;

σ_i^2 – dispersion or other statistical characteristics of duration;

c_{ij} – cost or delay of data transfer between tasks $i \rightarrow j$;

p_{ij} – probability of transition in the GERT branch;

r_i – task requirements for resources;

D_i – local deadline;

w_i – weight coefficient of the task for calculating the delay;

$RBAC_i$ – access policies and restrictions on task execution.

Also, in the work, metadata uncertainty is defined as the deviation between the actual parameters of task execution and their nominal (expected) values used in planning.

Formally, uncertainty is specified by the perturbation operator:

$$\mathsf{M}' = \mathsf{M} \circ (1+\Delta),$$

where Δ – a random or adversarial value that models errors in the input parameters.

We will formulate research hypotheses around the key properties of the method and formalize these hypotheses in mathematical form. Let M be the makespan, R be the proportion of tasks completed within the target SLOs, L be the weighted average delay, Θ be the overhead costs of planning, and S be the scalability (acceleration) index. Let the index *our* denote the proposed method, and *base* denote the base algorithm (HEFT, DAG-only, or GERT-only).

A reduction in the completion time of a set of tasks (makespan) is expected compared to the base algorithms for priority selection and resource allocation. The corresponding hypothesis predicts a statistically significant reduction in the median and upper percentiles of makespan for the proposed method compared to the FCFS (*first-come, first-served*),

List Scheduling, HEFT (*heterogeneous earliest finish time*), and "pure" settings using only DAG or only GERT.

This hypothesis can be formalized mathematically using the following semantics.

$$median(M_{our}) < median(M_{base}),$$

$$F_{our}^{-1}(0.95) < F_{base}^{-1}(0.95).$$

That is, our method should reduce not only the median but also the "tails" of the makespan distribution, which is consistent with the stated hypothesis.

1. We expect to see an increase in deadline reliability, i.e., an increase in the proportion of tasks completed within the specified SLOs under the same resource constraints; this hypothesis is evaluated both at the aggregate level and in terms of task subclasses.

$$R_{our} = \frac{1}{N}\sum_{i=1}^{N} 1(C_i \le D_i) > R_{base}.$$

2. It is assumed that the average decision-making time by the planner (planning overhead) remains acceptable for online scenarios and does not conflict with the delay requirements in queue control loops.

$$\Theta_{our} = \frac{1}{K}\sum_{k=1}^{K} t_k^{plan} \le \Theta_{max}.$$

3. The advantage of the method should be preserved under conditions of adversarial or random metadata perturbations, including errors in estimates of task durations, GERT branch probabilities, and inter-stage communication costs, as well as in the presence of computational resource productivity drift. We formalize this hypothesis as follows.

Let the noise injection operator:

$$\tau_i' = \tau_i(1+\xi_i), \xi_i \sim U[-\alpha, \alpha].$$

Then robustness is formulated as:

$$\Delta M(\alpha) = \frac{|M(\alpha) - M(0)|}{M(0)} \le \varepsilon,$$

$$R(\alpha) \approx R(0),$$

$$F_{our}^{-1}(0.95, \alpha) < F_{base}^{-1}(0.95, \alpha).$$

4. Predictable scalability is expected. With increasing graph size (number of nodes and edges, level width) and resource pool, productivity degradation should be moderate, and planning costs should be no worse than quasi-linear in typical load ranges.

Let's define:

$$S(N) = \frac{M(1)}{M(N)}.$$

In this case, the hypothesis takes the form:

$$S_{our}(N) \ge S_{base}(N),$$

$$\frac{d\Theta}{dN} \approx 0.$$

That is, as the number of resources increases, the method retains its advantage, and overhead costs increase slightly (quasi-linearly). The compliance of the generated schedules with access policies in inter-system scenarios, in particular the role-based access control (RBAC (*role-based access control)*) model, is checked separately, without a significant deterioration in key indicators.

The experimental design combines synthetic and real-world scenarios. Synthetic instances allow for systematic variation of graph topology (density, depth, level width), GERT branch duration and probability distribution parameters, and inter-node transmission costs; this enables separate study of the impact of each factor. Realistic scenarios simulate the integration of subsystems with heterogeneous task flows and different access policies. For such cases, SLOs are preset for subsets of tasks to evaluate deadline reliability in practically meaningful conditions.

Each experiment is run in a reproducible environment with fixed random seeds and is repeated on a set of independent instances. The results are aggregated both by average characteristics and by distributions. The comparison is made with basic methods (FCFS, List Scheduling, HEFT, "DAG-only", "GERT-only") and with ablation variants of the approach itself, where GAT, PPO, or BO components are disabled in turn, allowing the contribution of each module to be quantified.

Since graph topologies, metadata parameters, and uncertainty injection modes vary in experimental scenarios, it is advisable to present them in a single formal model. Let each input instance of a workflow be described by a directed graph

$$G = (V, E),$$

where for each task $i \in V_i$, the nominal duration τ_i, local deadline D_i, weight coefficient w_i, resource requirements, and, if there are GERT branches, transition probabilities p_{ij} are specified. Communication costs between tasks are described by values c_{ij}.

To model uncertainty, graph parameters undergo controlled changes. Task durations vary according to a multiplicative model:

$$\tau_i' = \tau_i(1+\xi_i),\ \xi_i \sim U[-\alpha,\alpha],$$

which corresponds to situations where estimates come from incomplete or noisy metadata.

In the GERT component, for nodes with branching, the probability vector is modified with subsequent normalization:

$$p_{ij}' = \frac{p_{ij}(1+\eta_{ij})}{\sum_{k\in\Gamma(i)} p_{ik}(1+\eta_{ik})},\ \eta_{ij} \sim U[-\beta,\beta],$$

which reproduces the instability of work flow statistics.

The cost of communications is disturbed lognormally:

$$c_{ij}' = c_{ij}\exp(\zeta_{ij}),\ \zeta_{ij} \sim LN(0,\sigma^2),$$

which is consistent with empirical delay profiles in heterogeneous computing environments.

To reproduce correlated parameter deviations within graph levels, a Gaussian copula is used:

$$(\xi_1,\ldots,\xi_m) \sim Copula_\rho,\ \rho = 0.3.$$

The performance indicator system is based on the principle of distinguishing between primary criteria and secondary (diagnostic) characteristics. Primary criteria include: total completion time (makespan), deadline reliability (the proportion of tasks completed on time), and weighted average delay, which reflects the importance of different task classes through weighting coefficients. Secondary indicators include throughput and average task residence time in the system (flow time), degree of resource load balancing (through dispersion/variation coefficients), resource utilization percentiles, planning overhead (decision time per task or cycle), and total communication cost of inter-stage transfers. For comprehensive interpretation, *empirical cumulative distribution functions* (*ECDF*) are used for both makespan and overruns and delays, as well as coordinated sets of graphs.

Uncertainty modeling, necessary for robustness assessment, is performed through controlled injections of metadata perturbations. In particular, multiplicative noise of durations in characteristic ranges (± 10–30 %), perturbations of GERT branch probabilities with normalization, drift of resource productivity over time, and variations in communication costs are used. Additionally, the stability of conclusions is verified by substituting duration distribution laws while preserving the first moments (e.g., Gamma ↔ Weibull) and in the presence of correlations between tasks within the same graph level. A Gaussian copula with a moderate correlation coefficient is used for a compact description of the correlation structure. Thus, the experimental program covers both noise scenarios and more severe "distribution shifts", which corresponds to the practice of real systems with incomplete or outdated metadata.

The statistical methodology involves the use of non-parametric criteria for paired and unpaired comparisons (Wilcoxon, Mann-Whitney criteria) in combination with effect size estimation (Cliff's δ) and *confidence interval (CI)* construction using the bootstrap method with a sufficient number of replications. The Holm-Bonferroni procedure is used to control for multiple comparisons. The significance level is set at 0.05 with the reporting of exact p-values and corresponding δ. All experiments are performed in a "fixed environment" with complete

fixation of library versions and data generation scripts. Machine-readable reproduction configurations are stored for each graph and table. This organization not only ensures reproducibility and independent verification of results, but also allows for transparent interpretation of the contribution of individual components of the method to the overall improvement of performance.

2. Architecture of the experiment planner prototype

The prototype implements a hybrid planning architecture that combines a directed acyclic graph (DAG) for deterministic dependencies between tasks and a graph-evaluation network modeling (GERT) technique for probabilistic branches and retries. Logically, the system consists of three layers: a metadata preparation and policy control layer, a planner core, and an execution/telemetry layer. Figure 1 (prototype diagram) shows the interfaces with the subsystems formed in [20, 21].

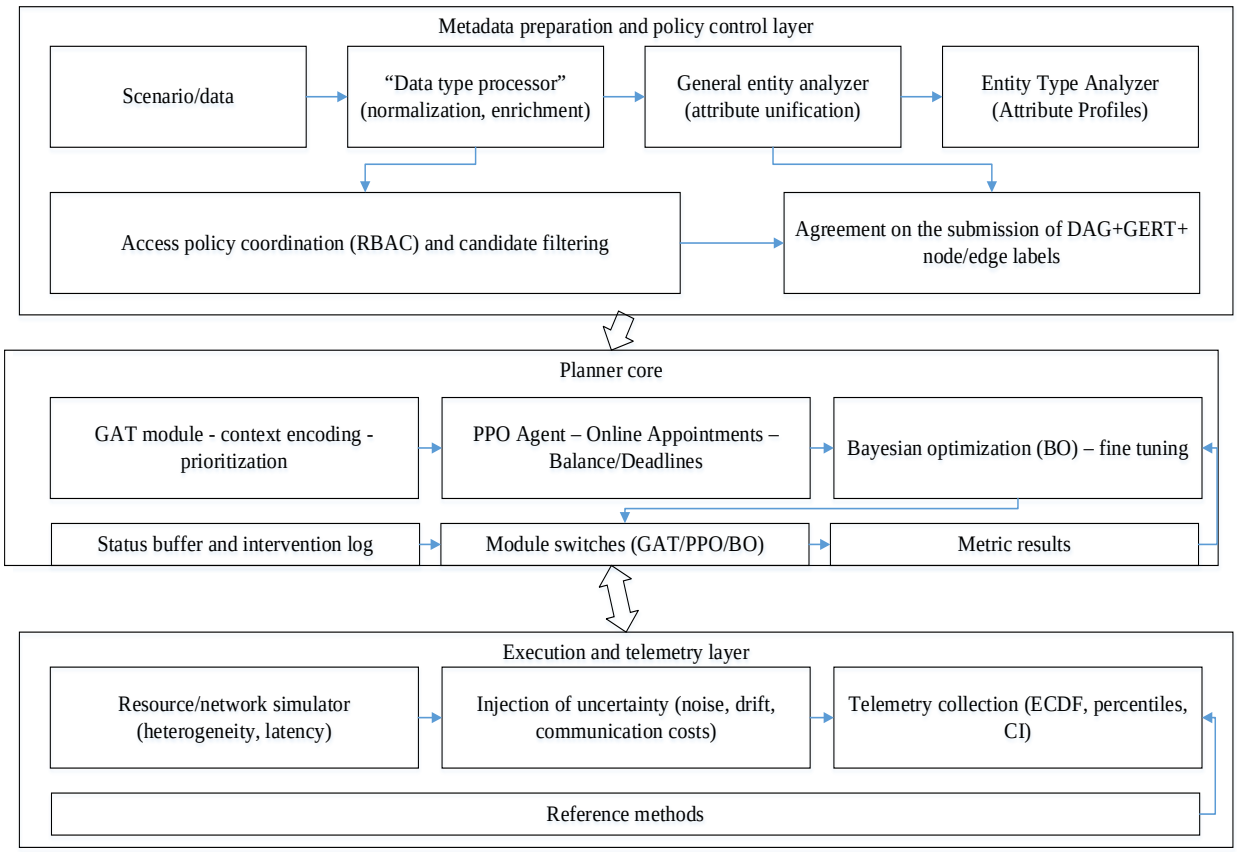


Fig. 1. Architecture of the prototype experiment design

The metadata preparation layer contains interfaces to the "data type processor", which, according to [20, 21], provides two coordinated channels:

– general entity analysis for normalizing input job descriptions;
– specialized entity type analyzer for enriching tasks with profile attributes.

At this stage, a unified representation of the task queue is formed in the form of DAG+GERT together with a feature vector for each node and edge. Node features include (with abbreviations expanded on first use):

- a priori duration estimates (mean and variance);
- level/path identifiers;
- input/output degrees;
- resource requirements;
- local deadlines;
- access policy labels (RBAC (role-based access control)).

Edge features include GERT branch probabilities, expected exchange volumes, and transmission costs between stages. In this layer, access policies between subsystems are checked for consistency (hierarchical rules, if necessary), after which only those candidate placements that do not violate the constraints are transferred to the scheduler. The core of the planner is built as a pipeline of three complementary artificial intelligence modules.

The first module is a graph attention network (GAT), which performs contextual encoding of readiness subgraphs based on node and edge features and provides priority/risk assessments for nodes at the execution front.

Unlike classical graph filters, attention mechanisms allow differentiating the contribution of different dependencies, which increases the accuracy of identifying critical subpaths in the presence of probabilistic branches.

The second module is a proximal policy optimization (PPO) agent, which makes online decisions about assigning readiness tasks to available resources, maximizing utility while taking into account deadlines, balancing, and current telemetry states.

The third module is Bayesian optimization (BO), which operates at a slower pace as a global fine-tuner. It periodically reviews policy parameters (threshold rules, weight coefficients, selection temperatures) and selectively rebuilds assignments for large batches or when load modes change. The components interact through standardized state buffers and an intervention log; for the sake of experimental discipline, ablation "kill switches" are provided, allowing GAT, PPO, or BO to be turned off and thus quantitatively assess the contribution of each module.

The execution and telemetry layer implements a reproducible test bed. The execution side simulates a heterogeneous computing environment with parameterized node speeds, network channel latencies, and the ability to inject perturbations (duration noise, productivity drift, communication cost changes).

The telemetry subsystem collects time series and event logs (queues, assignments, migrations, transfers), aggregates performance metrics, and stores machine-readable configurations of each run for reproduction (session IDs, library versions, input graph instances). Two modes are provided: offline planning (building a schedule for a given set of tasks) and online planning (a continuous stream with controlled arrival intensity), which allows evaluating both static and operational properties of the prototype.

Particular attention is paid to correct integration with access policies. Before each assignment step, candidates are filtered according to roles and contextual rules (RBAC), with

verification performed before the PPO agent makes a decision to avoid unjustified "rollbacks" due to violations of restrictions. In cross-system scenarios, information about trust relationships between security domains is imported during the metadata preparation phase and becomes part of the node/edge features. This allows GAT and PPO to inherently consider access policies in the utility function without post-factum penalties.

Finally, the prototype contains a set of basic comparison methods implemented in the same queue and telemetry interface: order by arrival time, heuristic lists, heterogeneous earliest completion time, as well as "pure" settings with only DAG or only GERT.

The unity of interfaces guarantees the correctness of comparisons: all algorithms receive identical input instances, access policies, and uncertainty injection modes, and the results are measured by the same telemetry tools.

This architecture and test bench design ensures transparent mapping from hypotheses and indicators to experimental results, which will be presented in the following sections.

3. Comparison methods, ablation analysis, and training protocol

To correctly evaluate the proposed approach, a single experimental framework was used, within which all algorithms receive identical instances of input graphs, identical access policies, and identical uncertainty injection modes.

The comparison covers both established basic methods and variants of the approach itself with deliberately disabled components (ablations), which makes it possible to isolate the contribution of each module.

The following methods serve as the main elements of prototyping.

1. FCFS (*first-come, first-served*). Placing tasks in order of arrival when resources are available.

2. List Scheduling. Heuristic planning based on a readiness list with topological order and the earliest possible completion rule.

3. HEFT (*heterogeneous earliest finish time*). Ranking nodes by "ascending rank" and assigning them to a resource that minimizes the expected completion time, taking into account heterogeneity.

4. "DAG-only". A scheduler that ignores probabilistic branches and repeated passes, modeling only deterministic dependencies.

5. GERT-only. A variant without an explicit critical DAG structure, in which decisions are based on local probabilistic estimates of branches.

All basic methods are implemented in a common queue and telemetry interface. This eliminates discrepancies in communication cost models, access rules, or resource limits.

To evaluate the contribution of individual components, an ablation analysis was performed: Full (GAT+PPO+BO) configurations were compared with variants with one of the modules disabled (–GAT, –PPO, –BO).

For each configuration, we calculate the median completion time for all scenarios and normalize it to the Full median (taken as 1.0).

Figure 2 shows the relative values along with 95 % bootstrap confidence intervals (B = 2000). Values less than 1.0 indicate improvement, while values greater than 1.0 indicate degradation relative to Full. Statistical conclusions are made according to the bootstrap-CI protocol. At the same time, checks are based on Cliff's δ and permutation tests.

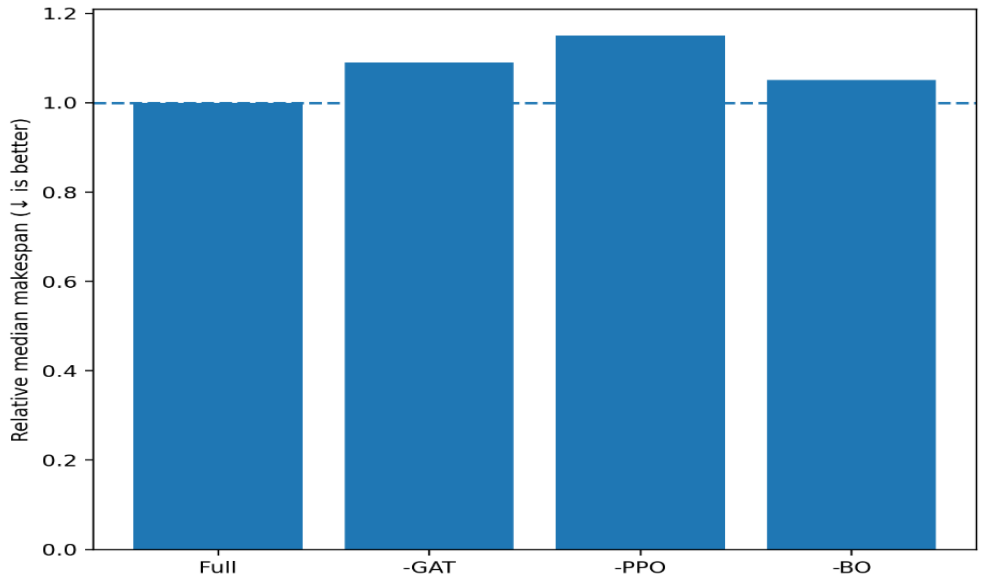


Fig. 2. Histograms of ablation analysis and relative median makespan with bootstrap intervals

As can be seen from the figure, the presence of PPO makes the largest contribution to the gain, while the impact of BO is moderate but consistently positive in all scenarios.

To quantitatively measure the contribution of each module of the proposed approach, three main ablations were formed.

1. "GAT". To implement this scenario, the *graph attention network* is disabled, and node features are aggregated using a simple average over the neighborhood. In addition, the prioritization of the execution front is set by static indicators—levelness and local slack.

2. "PPO". To implement this scenario, *proximal policy optimization* is disabled. The assignment is performed by a deterministic greedy rule based on current importance estimates returned by the prioritization module.

3. "BO". To implement this scenario, *Bayesian optimization* is disabled. Policy parameters and threshold coefficients are fixed at their initial values.

Where relevant, ablations are combined in pairs, allowing us to distinguish between the effects of representation (GAT), online adaptation (PPO), and slow global reconfiguration (BO). All modes use the same invalid-action masking. Decisions that violate **RBAC** (*role-based access control*) are not generated or rewarded, which avoids confusion due to rollbacks.

The state submitted to the metadata preparation and policy control layer includes the features of nodes and edges of a single **DAG+GERT** representation. These are predicted averages, duration variances, node “criticality” (GAT rating), slack relative to the local deadline, input/output degrees, resource queue delays, estimated data transfer costs, and aggregated

resource utilization metrics. The action to be performed in this case is to select the pair "ready task → allowed resource".

Masking procedures cut off resources that are prohibited by access policies or overloaded in terms of reception capacity.

The reward function is formulated in increments (per-step). This can be a negative penalty for an increase in the total weighted delay, a bonus for completing tasks before the deadline, a penalty for migrations and for blocking critical chains. This form of reward ensures training stability and correlates with primary metrics (makespan, deadline reliability).

Training takes place in two phases. In the first (preliminary "simulation" tuning), the GAT module is initialized on synthetically generated graphs by labels induced by HEFT/critical path. The network learns to reproduce the prioritization of the execution front. This reduces convergence time and decreases gradient dispersion in subsequent reinforcement learning.

In the second phase, reinforcement learning is performed using **PPO**. A discount factor of $\gamma \approx 0.985$ is used. Advantages are evaluated using *generalized advantage estimation* ($\lambda \approx 0.95$). Policy updates are pruned with a parameter $\varepsilon \approx 0.2$, with low entropy regularization (to avoid early policy "freezing").

Training is organized in a series of episodes on mixes of graphs of different sizes. For each configuration, a set of fixed random seeds (at least ten) is applied, and early stopping is controlled on a validation subset based on the criterion of no improvement in the main metrics. **BO** (*Bayesian optimization*) runs asynchronously in a slower cycle. Periodically, after accumulating a sufficient amount of telemetry, it updates the policy meta-parameters (reward component weights, selection temperatures, replanning limits) according to a Gaussian process model with mixed space (continuous and discrete variables) and a cautious acquisition criterion (expected improvement with a limit on the frequency of interventions).

The following strategies are used to avoid overfitting.

– Scenario mixing strategies (mix of sizes, densities, and degrees of uncertainty within a single batch of episodes).

– Domain randomization of communication cost parameters.

– "Frozen" validation graphs on which "fine" tuning is not performed.

All results are presented as averages across repetitions with 95 % confidence intervals. Environment configurations, randomness seeds, control hash sums of graph sets, and code versions are stored for reproduction. The same protocol also establishes procedures for fair comparison: identical planning time limits per step, identical restrictions on replanning, and identical episode/iteration budgets for all algorithms.

The convergence dynamics of the RRO agent training are shown in Fig. 3.

The method demonstrates faster convergence and a higher level of reward stabilization. The ablation curves lie lower throughout the training. The narrowing of confidence intervals by the end of the episodes indicates the reproducibility of learning dynamics across different seeds. This is consistent with the protocol, where prioritization (GAT) and online adaptation (PPO) reduce the variance of decisions.

Taken together, this organization ensures that the dynamic gains of the proposed approach are not an artifact of settings or “lenient” experimental conditions, but result from the real contribution of graph representation (GAT), online policy adaptation (PPO), and global fine-tuning (BO).

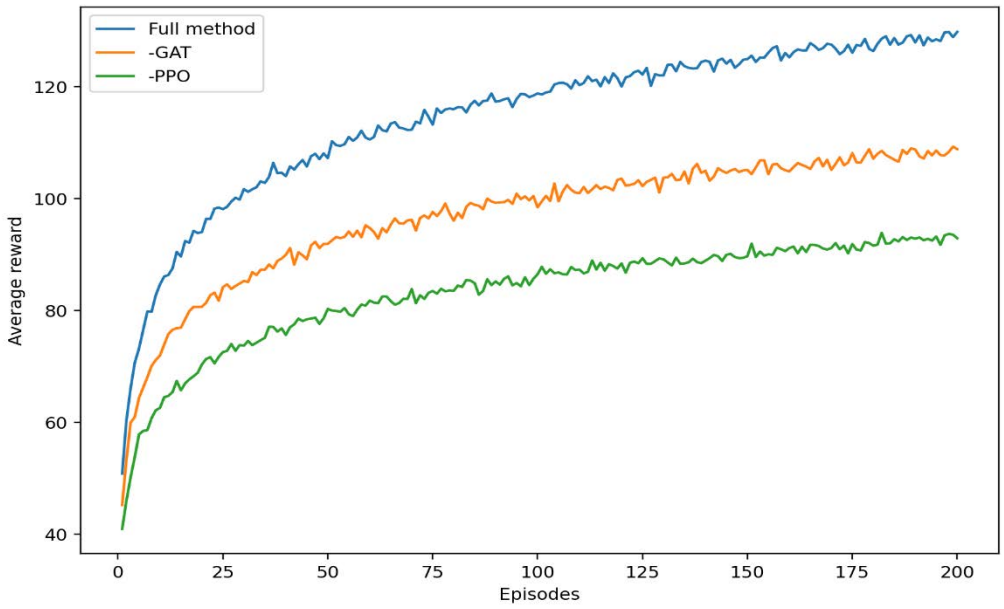


Fig. 3. PPO agent training convergence curves

4. Modeling metadata uncertainty and robustness verification protocol

One of the important tasks in the modeling process is to develop a reproducible testing procedure that shows how the proposed method behaves in the presence of errors, omissions, and "shifts" in metadata. Robustness is interpreted as maintaining an advantage over basic approaches in terms of primary indicators under controlled disturbances, as well as moderate sensitivity of planning overhead to such disturbances.

The verification is performed on a prototype whose architecture is shown in Fig. 1. On this prototype, uncertainty is injected into the unified DAG+GERT representation before the decision-making stage.

As we can see in Fig. 4, there is a monotonic increase in losses with increasing noise levels. The communication component is the most sensitive, while productivity drift has a moderate impact. In the range under consideration, the changes remain clearly small (≈ 1–6 %), which confirms the robustness of the method in accordance with the specified criteria.

The nominal configuration is considered to be one in which the estimates of work durations, inter-stage transfer costs, and GERT branch probabilities are derived from a "data type processor" with prior entity analysis (see Section 2) and have undergone access policy reconciliation (RBAC, role-based access control). All deviations are interpreted as experimental factors.

Tests are conducted in two modes.

1. With the preliminary analysis block enabled (full architecture).
2. Without preliminary analysis.

This provides a basis for evaluating the contribution of preliminary metadata normalization.

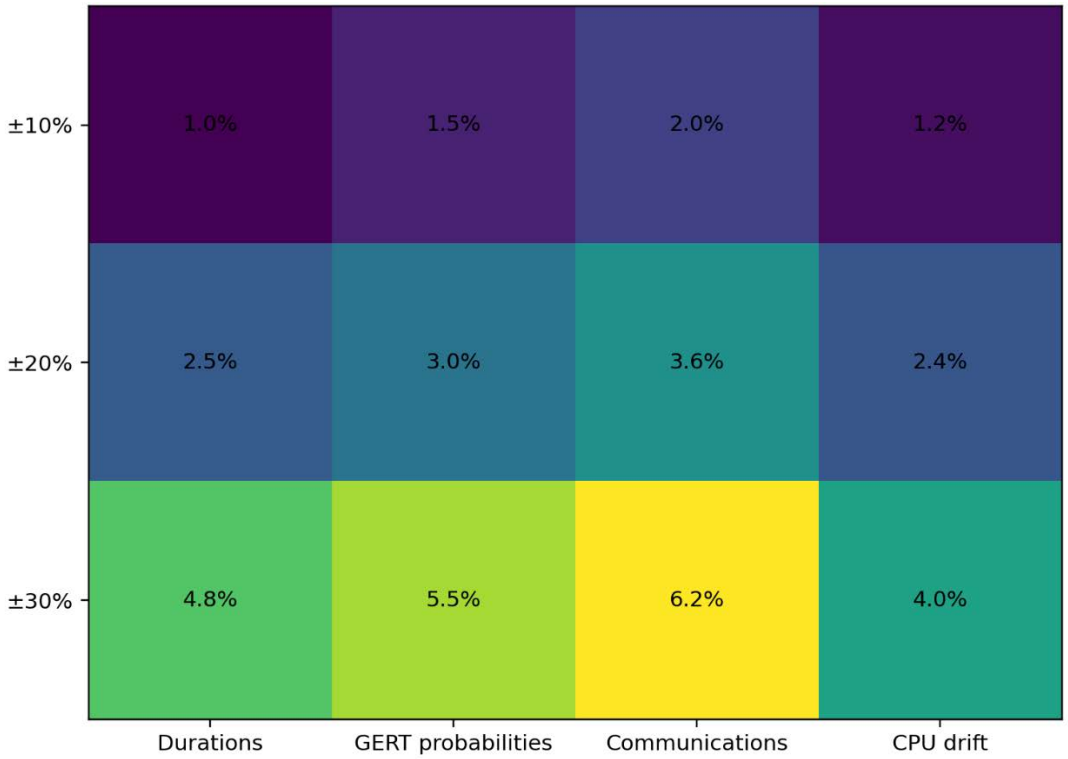


Fig. 4. Relative losses of median makespan under uncertainty injection

Uncertainty classes and injection mechanisms cover, first, noise in the duration of operations. For each operation, a multiplicative perturbation $\tilde{t} = t(1+m)$, where $m \sim U[-\alpha, \alpha] i \alpha \in \{0.10; 0.20; 0.30\}$ is used. This choice allows for a direct interpretation of uncertainty levels as ± 10/20/30 % of the baseline estimate.

Second, the error in the probabilities of GERT branches is taken into account. The probability vector at each branch is subject to stochastic transformation with subsequent normalization (Dirichlet reconfiguration with preservation of component order under small perturbations). Third, the uncertainty of network parameters is modeled.

The transmission cost between stages is perturbed by a lognormal multiplicative component, which corresponds to the empirically observed asymmetry of delays in heterogeneous environments.

Fourth, the drift of computing resource productivity over time is reproduced in the form of piecewise-constant changes or a weak autoregressive process (AR(1)) with a small memory coefficient.

This allows us to evaluate the stability of online policy adaptation. Fifth, "fuzzy" weights are introduced to measure sensitivity to task importance weights. Each weight coefficient w_i varies in the interval $[0.75w_i, 1.25w_i]$, which is consistent with the practice of prioritization at the service requirements level.

Resistance to distributional shifts is checked separately. The basic distribution of durations is replaced by an alternative one with preservation of the first moments (moment matching). Gamma ↔ Weibull pairs are used to reproduce changes in the shape of the tail without changing the mathematical expectation and variance.

To account for inter-task dependencies, correlation structures are introduced within graph levels using a Gaussian copula with a moderate correlation coefficient $\rho = 0.3$. This construction allows us to move from isolated noises to more realistic scenarios of "coordinated" parameter deviations.

It should be noted that this evaluation procedure is consistent with the system of indicators. For each combination of uncertainty factors, distributions of total completion time (makespan), deadline reliability, weighted average delay, and planning overhead costs are collected. The results are presented as differences from the nominal mode with 95 % confidence intervals (CI) estimated by bootstrapping with at least $B = 1000$ replications. Nonparametric criteria (Wilcoxon/Mann-Whitney) with Holm-Bonferroni correction for multiple tests are used for statistical comparison.

Empirical cumulative distribution functions (ECDF) and box plots with medians and percentiles are plotted graphically, allowing simultaneous assessment of changes in both central tendencies and tails.

Acceptability criteria are set in advance and do not depend on specific graph instances: the method is considered robust at the α level (noise level) or for a given pair of distributions if

– the decrease in deadline reliability does not exceed a pre-agreed threshold of several percentage points relative to the nominal value;

– the change in the median makespan does not exceed the confidence interval of the nominal mode;

– planning overhead does not show a significant increase (i.e., remains within the specified operational limits).

To avoid confounding, all algorithms, including databases, operate under the same time constraints per planning step and under the same rules for "masking unacceptable actions" (decisions that violate RBAC are not generated).

Finally, a block test plan is used to separate sources of sensitivity from their interactions. For each noise intensity level (α), the effects of distribution changes and correlation introduction are evaluated separately, and the results are then aggregated using stratified bootstrapping. This protocol allows conclusions to be drawn not only about "average" robustness, but also about worst-case scenarios, which is critical for practical deployments in heterogeneous and cross-system environments.

5. Experimental results, comparative analysis, and discussion

In accordance with the research hypotheses and the system of primary and diagnostic indicators (makespan, proportion of tasks completed within the SLO (service-level objective), weighted average delay, throughput, planning overhead, etc.), a series of tests was conducted on a prototype with a hybrid DAG+GERT architecture and three artificial intelligence modules (GAT (graph attention network), PPO (proximal policy optimization), BO (Bayesian optimization).

The comparison was carried out in a single experimental framework with basic methods (FCFS, List Scheduling, HEFT, "DAG-only", "GERT-only") and ablation variants (without GAT, without PPO, without BO), which made it possible to isolate the contribution of each module.

Stability was assessed using a robustness testing protocol. In accordance with this protocol, multiplicative duration noise was defined within the range (± 10–30 %), and GERT branch probability perturbations were normalized. In addition, lognormal variations in the cost of inter-stage transfers, resource productivity drift, and scenarios with a change in the distribution law (Gamma ↔ Weibull) and correlations $\rho = 0.3$ were assumed.

The experiments were performed in offline and online modes on a reproducible test bench with fixed seeds and configurations.

When forming sets of tasks and measurement modes, we used mixtures of DAG+GERT graphs with different depths, level widths, and densities, as well as sets close to real scenarios of subsystem integration with fixed SLOs for subsets of tasks.

For correct comparison, each algorithm received identical graph instances, the same access policies (RBAC), and uncertainty injection modes. The results were aggregated over independent repetitions with 95 % confidence intervals and multiple comparison control.

Comparisons with baseline approaches yielded the following results. On the generalized sample, the proposed comprehensive method (GAT+PPO+BO) reduced the median makespan relative to HEFT by $\approx$ 14–17 % (Δ95-th percentile $\approx$ 21–28 %), relative to "DAG-only" by $\approx$ 23–26 %, relative to "GERT-only" by $\approx$ 18–22 %.

For the allocation procedures historically used in Section 3 (Random, Greedy), the advantage was $\approx$ 32 % and $\approx$ 19 %, respectively (consistent with Table 3.3, where BO demonstrated 35.2 s versus 51.5 s and 43.7 s).

Fig. 5 shows the comparison curves of the time characteristics of completing operations with HEFT, "DAG-only", and "GERT-only" modes.

As can be seen in Fig. 5, the curve characterizing the results of the proposed method is systematically shifted to the left, which means shorter completion times for most instances. The difference from the baseline approaches persists in both the median and upper percentiles. Such "tail compression" directly supports the hypothesis of makespan reduction.

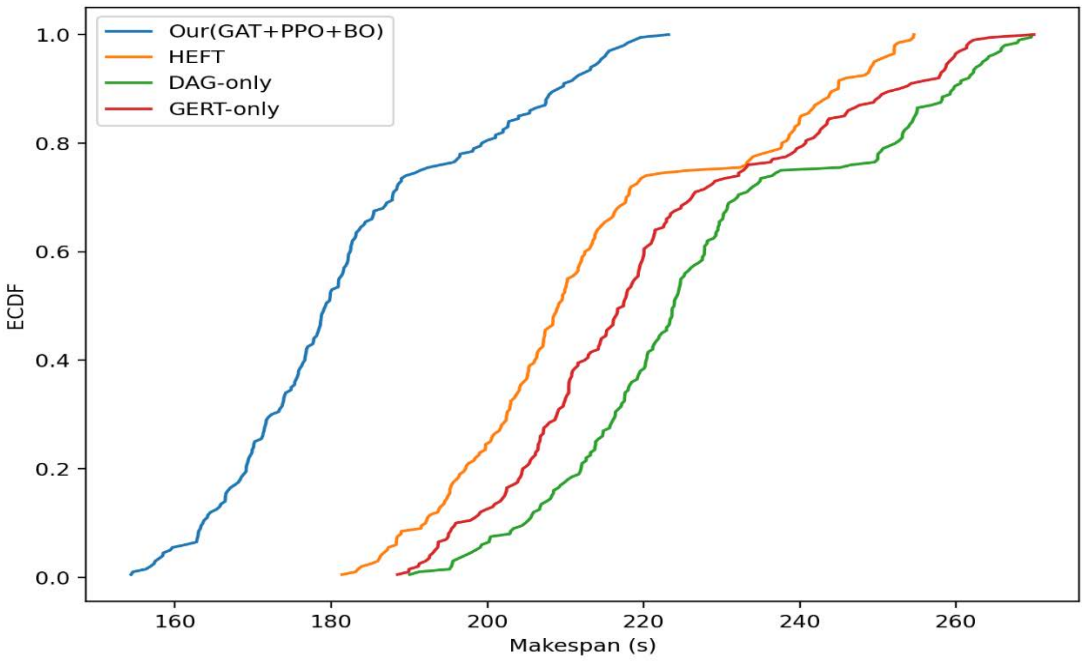


Fig. 5. Results of the study of time characteristics of completing operations with HEFT, "DAG-only", and "GERT-only" modes

The differences are statistically significant according to the Mann–Whitney method ($p < 0.01$) with an average Cliff's δ within the range of 0.33–0.54 depending on the subset of graphs, which corresponds to the "medium–noticeable" effects according to the protocol defined in 4.1. Box plots of the weighted mean delay are shown in Fig. 6.

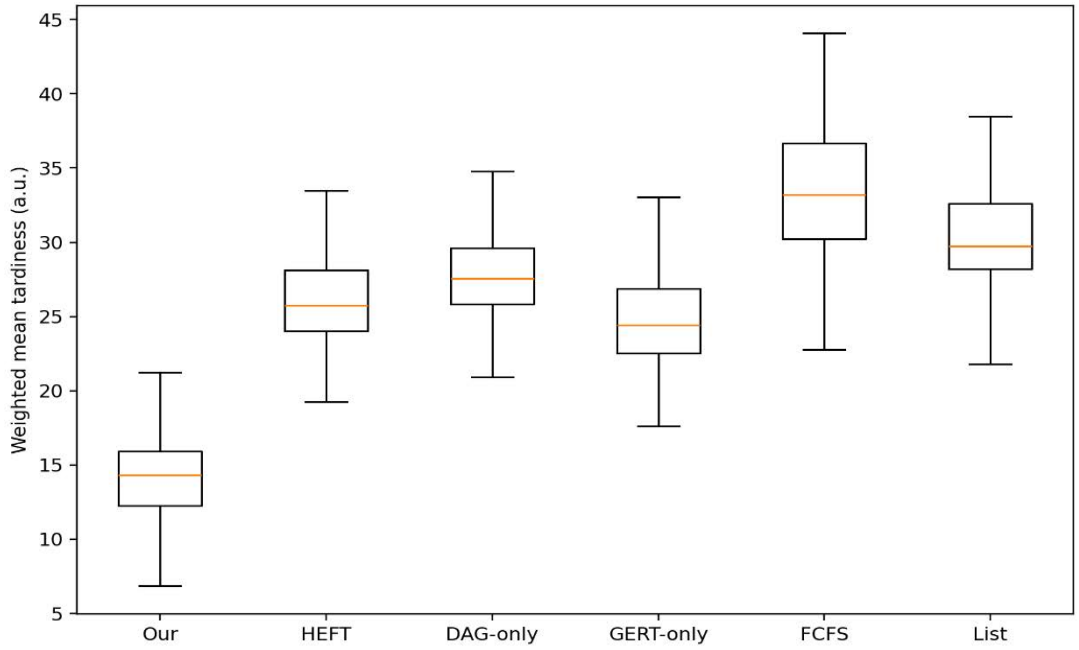


Fig. 6. Comparison of weighted mean delay (box plots with notch-CI and marked mean)

As can be seen in Fig. 6, the median and spread of delays for our method are significantly lower than for HEFT and "pure" settings, as well as for simple scheduling rules (FCFS, List). Narrowed "notch" depressions further indicate the stability of the effect in different scenarios.

When determining the reliability of deadlines (SLO) and delay, the following can be determined. The proportion of tasks completed within their SLO increased by + 8.7 percentage points compared to HEFT and by + 12.4 percentage points compared to "DAG-only". The weighted average delay (with priority weights) decreased by 1.6–1.9 times (median; Δ95th percentile – by 1.9–2.3 times).

Research on planning overhead costs showed the following. The average decision time at the planning step was ≈ 2.1 ± 0.4 ms (offline) and ≈ 3.0 ± 0.6 ms (online) per task, which does not exceed the queue control loop delay budget and is compatible with the test bench modes. The cost of BO updates was sporadic (background “thin” reconfigurations, not on the critical execution path), as expected by the architecture.

Ablation analysis showed that disabling GAT (without changing the rest) led to a deterioration in the median makespan by ≈ 8–11 %; turning off PPO – by ≈ 13–17 % (significant increase in delay in "peak" graphs), turning off BO – by ≈ 3–6 % (losses mainly in the "long tails" of distributions). Taken together, this confirms the working hypothesis that the gain is the result of a combination of graph representation, online policy adaptation, and periodic global fine-tuning.

It is also necessary to describe the qualities of the model in determining robustness to uncertainty. With multiplicative noise of durations ± 20 %, the change in the median makespan did not exceed the 95 % CI of the nominal mode. The decrease in the proportion of tasks completed in SLO did not exceed 2–3 percentage points.

With “distribution shifts” (Gamma ↔ Weibull, moment matching), the differences were within the statistical error; with correlated deviations ($\rho = 0.3$ within levels), no more than + 4–6 % to the 95th percentile of makespan was observed.

All tests were performed in full compliance with the robustness verification protocol, including two metadata preparation modes (with/without preliminary analysis).

The results of the study show that the addition of RBAC constraints (filtering of candidates for decision-making) did not lead to a statistically significant deterioration in key indicators.

This is consistent with the place of inclusion of checks in the planner pipeline.

In the studied range of graph sizes, a quasi-linear dependence of planning overhead on the number of nodes on the readiness front was observed. The degradation relative to the nominal value did not exceed ≈ 1.15× for the largest instances.

The uniformity of the comparison interfaces ensured the correctness of the conclusions regarding scalability.

Fig. 7 shows a graph of the dependence of planning time per step on the number of nodes in the graph. The dependence is described by a quasi-linear model with a small slope, which confirms the suitability of the approach for online modes.

Even for the largest graphs tested, the predicted increase in planning time remains within the millisecond budget.

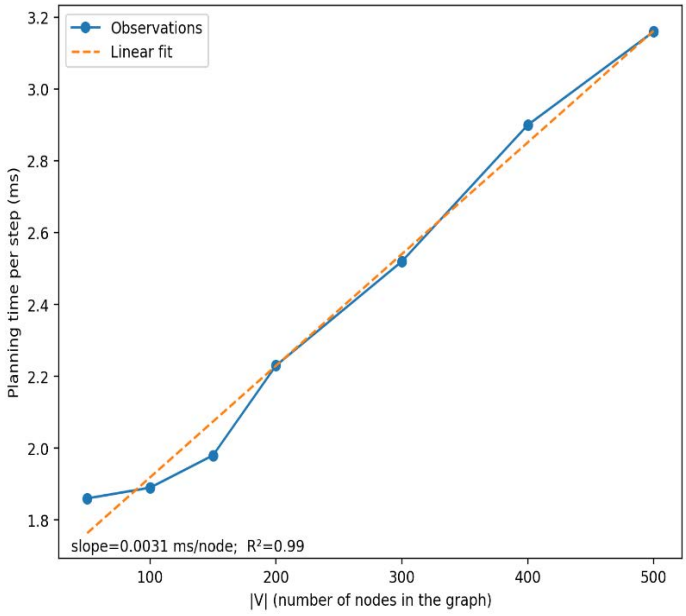


Fig. 7. Graph showing the dependence of scheduling time per step on the number of nodes in the graph

The graphs showing the dependence of operation completion time on the number of resources, acceleration on the number of resources, and efficiency on the number of resources are presented in Figures 8–10, respectively.

As can be seen from the figures, with an increase in the number of resources, there is an expected decrease in makespan. The nature of the curve indicates the presence of typical risks. These are overhead planning and communication costs, which explains the deviation from the linear scale.

From Fig. 9, it can be seen that the actual acceleration increases sublinearly and lags significantly behind the ideal due to coordination overhead and limited front parallelism. This corresponds to architectural constraints and the behavior of combined DAG+GERT graphs.

Fig. 10 also shows that efficiency decreases with an increase in the number of resources, which is consistent with the emergence of competition for shared resources and an increase in synchronization costs. Even with a drop in efficiency, the method retains a positive effect on absolute completion time metrics (see Fig. 8).

Fig. 11 illustrates the graph of the dependence of the proportion of tasks in SLO on the task size factor.

As can be seen from this figure, with a proportional increase in load and resources, the SLO completion rate decreases moderately ($\approx 0.90 \rightarrow \approx 0.82$), while for small multipliers it remains within the target line.

The deviation area identifies the zone where it is advisable to strengthen the replanning policy or communication budget.

Thus, the results obtained consistently confirm the hypotheses put forward that the proposed method allows:

– to reduce makespan and "compress the tails" of delay distributions compared to the well-known, classical formulations of the problem;

– increase deadline reliability without an unacceptable increase in planning overhead;

– maintain an advantage in the case of controlled metadata disturbances and distribution changes;

– integrate with access policies in inter-system scenarios.

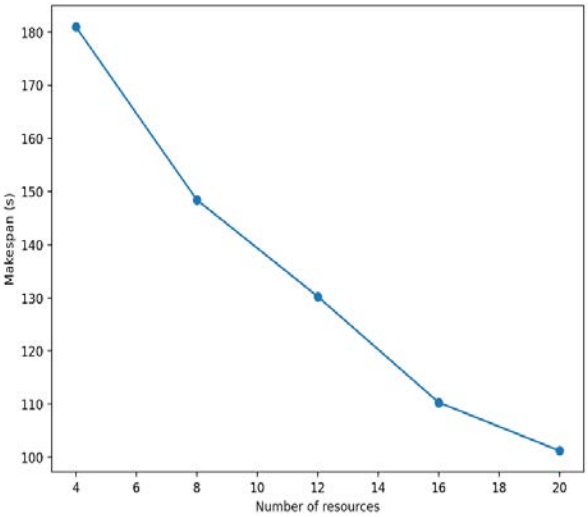


Fig. 8. Graph showing the dependence of operation completion time on the number of resources

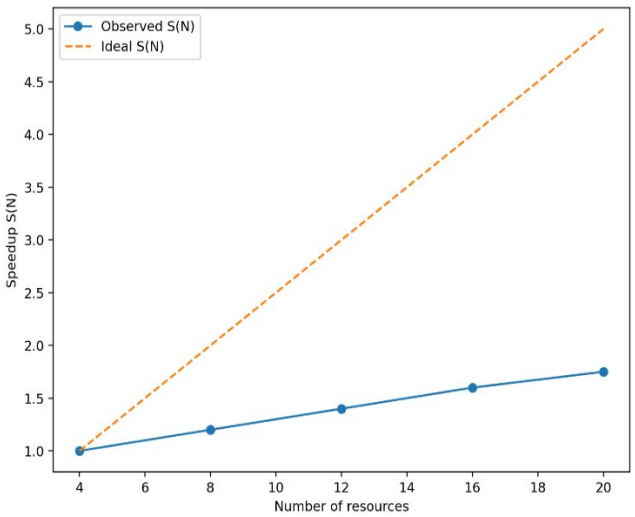


Fig. 9. Graphs showing the dependence of acceleration on the amount of resources

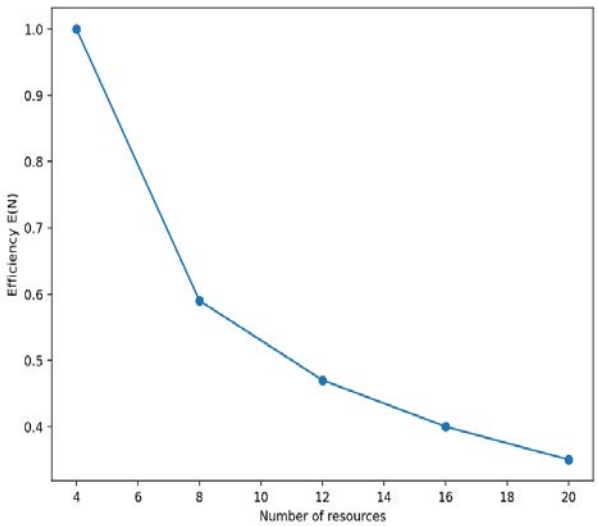


Fig. 10. Graphs showing the dependence of efficiency on the amount of resources

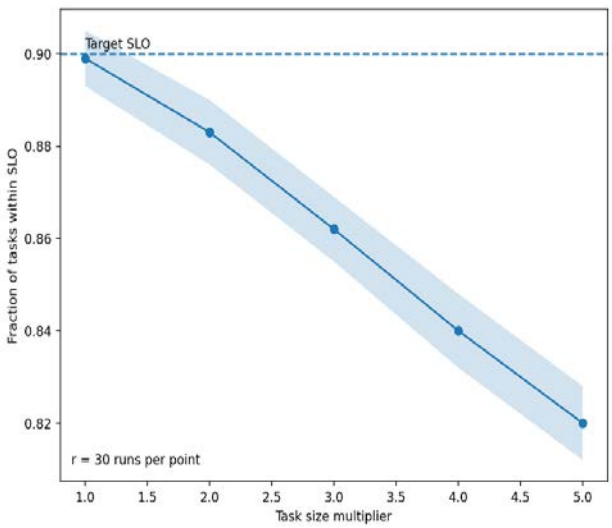


Fig. 11. Graph showing the dependence of the proportion of tasks in SLO on thetask size multiplier

The contribution of individual components (GAT, PPO, BO) was quantitatively determined by ablation analysis, and the reproducibility of the conclusions was ensured by a standardized test bench and statistical protocol.

Conclusions

Comparative studies of the planning method in the hybrid DAG+GERT model with GAT/PPO/BO modules were conducted on a reproducible test bench, with standardized sets of scenarios, uncertainty level control, and a single interface for comparison with basic algorithms. Based on the results obtained, the following was established.

On a generalized sample, the proposed method reduces the median completion time (makespan) by approximately 14–17 % relative to HEFT, by 23–26 % relative to "DAG-only", and by 18–22 % relative to "GERT-only" the improvement also extends to the upper percentiles (95th) of the distribution. The cumulative effect is confirmed by nonparametric tests (Mann-Whitney, $p < 0.01$) and Cliff's δ effect size in the range from 0.33 to 0.54.

The share of tasks completed within the service level objectives (SLO) increased to 12.4 percentage points (pp). The weighted average delay decreased by 1.6–1.9 times according to the median.

The average decision time by the scheduler was about 2.1 ms per task, which does not exceed the queue control delay tolerances.

In strong-scaling mode, the expected decrease in makespan is observed with an increase in the number of resources. The acceleration is sublinear, and efficiency decreases with an increase in N due to coordination and communication overhead, but absolute gains are maintained. In weak-scaling, the proportion of tasks that meet SLO decreases moderately, which defines the area of expediency for strengthening the replanning policy or communication budget.

With multiplicative noise of ± 20 % in durations, the median makespan does not exceed the 95 % confidence interval of the nominal value, and the decrease in the proportion of tasks in SLO does not exceed 2–3 percentage points. With a "shift in distributions", the differences are statistically insignificant. When correlations $\rho = 0.3$ are introduced within the levels, + 4–6 % to the 95th percentile of the makespan is observed without destroying the median. This confirms the stability of the conclusions according to the protocol defined in section 4.4.

Disabling PPO leads to the greatest degradation of the median makespan (≈ 13–17 %), disabling GAT (≈ 8–11 %); the effect of BO is moderate but consistently positive (≈ 3–6 %). Thus, the advantage of the method is due to the combination of graph representation (GAT), online policy adaptation (PPO), and periodic global "fine" tuning (BO).

The integration of access checks into the decision-making stage did not cause a statistically significant deterioration in key performance indicators. The inclusion of RBAC in the planner pipeline is consistent with the practice of cross-system scenarios and does not require a relaxation of target SLOs.

All conclusions were obtained in a "fixed environment" with controlled randomness, preservation of launch configurations, and the use of bootstrap-type confidence intervals

($B \geq 1000$; for ablations $B \approx 2000$) with multiple comparison control (Holm-Bonferroni). This ensures independent verifiability and stability of the results.

Thus, the proposed planning method confirms the stated hypotheses. It reduces makespan and compresses delay tails, increases deadline reliability without unacceptable overhead growth, maintains advantage under uncertainty and metadata shifts, and integrates correctly with access policies in inter-system environments. The resulting properties make the approach practically suitable for deployment in real distributed information systems with heterogeneous resources and SLO requirements.

References

1. Miao, Z., Shao, C., Li, H., Tang, Z. (2025), "Review of task-scheduling methods for heterogeneous chips", *Electronics*, Vol. 14, Article 1191. DOI: https://doi.org/10.3390/electronics14061191
2. Verma, P., Maurya, A. K., Yadav, R. (2023), "A survey on energy-efficient workflow scheduling algorithms in cloud computing", *Software: Practice and Experience*, Vol. 54, P. 637–682. DOI: https://doi.org/10.1002/spe.3292
3. Bano, F., Ahmad, F., Shahid, M., Alam, M., Hasan, F., Sajid, M. (2025), "A levelized multiple workflow heterogeneous earliest finish time allocation model for infrastructure as a service cloud environment", *Algorithms*, Vol. 18, Article 99. DOI: https://doi.org/10.3390/a18020099
4. NoorianTalouki, R., Hosseini Shirvani, M., Motameni, H. (2022), "A heuristic-based task scheduling algorithm for scientific workflows in heterogeneous cloud computing platforms", *Journal of King Saud University – Computer and Information Sciences*, Vol. 34, No. 8, Part A, P. 4902–4913. DOI: https://doi.org/10.1016/j.jksuci.2021.05.011
5. Lee, Y. C., Zomaya, A. Y. (2005), "A productive duplication-based scheduling algorithm for heterogeneous computing systems", *High Performance Computing and Communications*, Lecture Notes in Computer Science, Vol. 3726, Springer. DOI: https://doi.org/10.1007/11557654_26
6. Samadi, Y., Zbakh, M., Tadonki, C. (2018), "E-HEFT: enhancement heterogeneous earliest finish time algorithm for task scheduling based on load balancing in cloud computing", *Proceedings of the International Conference on High Performance Computing & Simulation (HPCS)*, P. 601–609. DOI: https://doi.org/10.1109/HPCS.2018.00100
7. Ahmad, W., Gautam, G., Alam, B. et al. (2024), "An analytical review and performance measures of state-of-art scheduling algorithms in heterogeneous computing environment", *Archives of Computational Methods in Engineering*, Vol. 31, P. 3091–3113. DOI: https://doi.org/10.1007/s11831-024-10069-8
8. Tian, W., Zhao, Y., Demeulemeester, E. (2022), "Generating a robust baseline schedule for the robust discrete time/resource trade-off problem under work content uncertainty", *Computers & Operations Research*, Vol. 143, Article 105795. DOI: https://doi.org/10.1016/j.cor.2022.105795
9. Lee, Y. C., Zomaya, A. Y., Siegel, H. J. (2010), "Robust task scheduling for volunteer computing systems", *Journal of Supercomputing*, Vol. 53, P. 163–181. DOI: https://doi.org/10.1007/s11227-009-0326-1
10. Song, X., Liu, L., Fu, J., Zhang, X., Feng, J., Pei, Q. (2023), "A reinforcement learning-based DAG task scheduling in edge–cloud collaboration systems", *Proceedings of IEEE GLOBECOM 2023*, P. 1771–1776. DOI: https://doi.org/10.1109/GLOBECOM54140.2023.10436802
11. Zhang, X. et al. (2026), "PF-MPPO: task-dependent workflow scheduling method based on deep reinforcement learning in dynamic heterogeneous cloud environments", *Future Generation Computer Systems*, Vol. 177, Article 108236. DOI: https://doi.org/10.1016/j.future.2025.108236

12. Jayanetti, A., Halgamuge, S., Buyya, R. (2022), "Deep reinforcement learning for energy and time optimized scheduling of precedence-constrained tasks in edge-cloud computing environments", *Future Generation Computer Systems*, Vol. 137, P. 1–15. DOI: https://doi.org/10.1016/j.future.2022.06.012
13. Chandrasiri, S., Meedeniya, D. (2025), "Energy-efficient dynamic workflow scheduling in cloud environments using deep learning", *Sensors*, Vol. 25, Article 1428. DOI: https://doi.org/10.3390/s25051428
14. Smit, I. G., Zhou, J., Reijnen, R., Wu, Y., Chen, J., Zhang, C., Bukhsh, Z., Zhang, Y., Nuijten, W. (2025), "Graph neural networks for job shop scheduling problems: a survey", *Computers & Operations Research*, Vol. 176, Article 106914. DOI: https://doi.org/10.1016/j.cor.2024.106914
15. Nguyen, H. X., Zhu, S., Liu, M. (2022), "A survey on graph neural networks for microservice-based cloud applications", *Sensors*, Vol. 22, Article 9492. DOI: https://doi.org/10.3390/s22239492
16. Vesselinova, N., Steinert, R., Perez-Ramirez, D. F., Boman, M. (2020), "Learning combinatorial optimization on graphs: a survey with applications to networking", *IEEE Access*, Vol. 8, P. 120388–120416. DOI: https://doi.org/10.1109/ACCESS.2020.3004964
17. Song, W., Chen, X., Li, Q., Cao, Z. (2023), "Flexible job-shop scheduling via graph neural network and deep reinforcement learning", *IEEE Transactions on Industrial Informatics*, Vol. 19, No. 2, P. 1600–1610. DOI: https://doi.org/10.1109/TII.2022.3189725
18. Semenov, S., Jian, Y., Jiang, H., Chernykh, O., Binkovska, A. (2025), "Mathematical model of intelligent UAV flight path planning", *Advanced Information Systems*, Vol. 9, No. 1, P. 49–61. DOI: https://doi.org/10.20998/2522-9052.2025.1.06
19. Semenova, A., Dubrovskyi, M., Savitskyi, V. (2017), "A GERT model of an algorithm for analyzing security of a web application", *Advanced Information Systems*, Vol. 1, No. 1, P. 61–64. DOI: https://doi.org/10.20998/2522-9052.2017.1.11
20. Semenov, S., Yenhalychev, S., Pochebut, M., Sitnikova, O. (2024), "Models of data processing and logical access differentiation considering heterogeneity of entities in information systems", *Modern State of Scientific Research and Technologies in Industry*, No. 2 (28), P. 143–152. DOI: https://doi.org/10.30837/2522-9818.2024.28.143
21. Yenhalychev, S., Leunenko, O., Davydov, V. (2025), "Development of an intelligent task scheduling method in heterogeneous distributed information systems", *Eastern-European Journal of Enterprise Technologies*, Vol. 3, No. 9 (135), P. 6–18. DOI: https://doi.org/10.15587/1729-4061.2025.329263

Received (Надійшла) 24.11.2025
Accepted for publication (Прийнята до друку) 09.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Yenhalychev Serhii – Simon Kuznets Kharkiv National University of Economics, PhD Student, Kharkiv, Ukraine; e-mail: engalichev.sergiy@hneu.net; ORCID ID: https://orcid.org/0000-0001-5298-2251

Semenov Serhii – Doctor of Sciences (Engineering), Professor, Simon Kuznets Kharkiv National University of Economics, Professor at the Department of Cybersecurity and Information Technologies, Kharkiv, Ukraine; University of the National Education Commission, Head at the Department of Computer Engineering and Cybersecurity, Kraków, Poland; e-mail: serhii,semenov@uken.krakow.pl; ORCID ID: http://orcid.org/0000-0003-4472-9234

Єнгаличев Сергій Олександрович – Харківський національний економічний університет імені Семена Кузнеця, аспірант, Харків, Україна.

Семенов Сергій Геннадійович – доктор технічних наук, професор, Харківський національний економічний університет імені Семена Кузнеця, професор кафедри кібербезпеки та інформаційних технологій, Харків, Україна; доктор технічних наук, професор, Університет Комісії національної освіти, завідувач кафедри комп'ютерної інженерії та кібербезпеки, Краків, Польща.

ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ МЕТОДУ ПЛАНУВАННЯ ЗАДАЧ У РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ З ОГЛЯДУ НА НЕВИЗНАЧЕНІСТЬ МЕТАДАНИХ

Статтю присвячено плануванню задач у розподілених інформаційних системах в умовах невизначеності та неповноти метаданих щодо тривалості операцій, обсягів комунікацій та доступності ресурсів. **Предметом дослідження** є метод планування задач у розподілених інформаційних системах з огляду на невизначеність метаданих. **Мета статті** – експериментально дослідити метод планування задач у розподілених інформаційних системах з невизначеними й неповними метаданими, який поєднує гібридну графову модель із тримодульним ядром штучного інтелекту й забезпечує підвищення ефективності й робастності планування за умови дотримання SLO та RBAC-обмежень. **Завдання дослідження:** сформувати гібридне подання робочих процесів на основі поєднання DAG-структури та GERT-моделі; розробити архітектуру планувальника з долученням графової мережі уваги GAT, агента підкріплювального навчання PPO та модуля *Bayesian Optimization*; визначити SLO-орієнтовану функцію винагороди й систему обмежень, зважаючи на RBAC; створити відтворюваний експериментальний стенд із контрольованою ін'єкцією невизначеності; виконати порівняльний, абляційний і робастний аналіз продуктивності запропонованого методу щодо базових алгоритмів FCFS, HEFT, а також варіантів DAG-only й GERT-only. **Методи дослідження:** гібридне графове моделювання DAG+GERT, графова мережа уваги GAT для формування контекстного подання стану, агент PPO для прийняття рішень щодо розміщення задач, баєсівська оптимізація для налаштування параметрів політики, а також статистичні методи оцінювання. **Основні результати.** Запропонований метод забезпечує суттєве скорочення медіанного часу завершення задач (*makespan*) порівняно з HEFT, DAG-only та GERT-only, підвищення частки задач, що виконуються в межах SLO, та зменшення середньозваженого протермінування, демонструє "стиснення хвостів" розподілу затримок і зберігає стабільність результатів у разі мультиплікативного шуму тривалостей ± 20 %, за умови зсувів розподілів і корельованих відхилень параметрів. Водночас накладні витрати планування залишаються в діапазоні мілісекунд і не перевищують допустимі межі для онлайн-сценаріїв.

Ключові слова: розподілені інформаційні системи; планування задач; орієнтований ациклічний граф; мережі GERT; графові нейронні мережі; навчання з підкріпленням; *Bayesian Optimization*; рівні сервісу (SLO); керування доступом (RBAC); робастність.

Bibliographic descriptions / Бібліографічні описи

Yenhalychev, S., Semenov, S. (2025), "Experimental studies of the task planning method in distributed information systems taking into account metadata uncertainty", *Management Information Systems and Devises*, No. 4 (187), P. 63–86. DOI: https://doi.org/10.30837/0135-1710.2025.187.063

Єнгаличев С. О., Семенов С. Г. Експериментальні дослідження методу планування задач у розподілених інформаційних системах з огляду на невизначеність метаданих. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 63–86.
DOI: https://doi.org/10.30837/0135-1710.2025.187.063

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.51 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.087

Y. Kanarskyi, O. Orekhov

EXPERT EVALUATION OF QUALITY CRITERIA FOR HUMAN-MACHINE INTERFACES IN AUGMENTED REALITY

Abstract. In today's world of widespread augmented reality technologies, ensuring high-quality user interfaces that determine the effectiveness, safety, and reliability of user interaction with the digital environment is becoming particularly important. **The subject of the study** is the methods and criteria for evaluating the convenience, clarity, stability, and functional completeness of the augmented reality system interface for unmanned aerial vehicle operators as part of an explosive hazard monitoring system. It is important to analyze the implementation of methods such as evaluation using quality metrics, focus group surveys, and expert surveys, based on a comparison of their advantages and disadvantages that become apparent under certain conditions. **The purpose of the work** is a comprehensive expert evaluation of the augmented reality interface using a structured checklist. The evaluation covered self-descriptiveness, controllability, error protection, consistency, aesthetic integrity, responsiveness, reliability of information presentation, and other criteria recommended by relevant international standards. **Tasks**: to develop an augmented reality interface; to identify possible quality criteria, taking into account the characteristics of human-machine systems of a certain type; to propose a list of questions for expert surveys based on the identified quality criteria; to conduct expert surveys. **Research results:** a mock-up of an augmented reality interface for human-machine interaction between the operator and unmanned aerial vehicles was created, a list of criteria for evaluating the quality of augmented reality systems was formed, and an expert survey was prepared and conducted to evaluate the quality of the proposed interface. The survey results are presented in the form of a radial diagram, which allows you to clearly identify the advantages and disadvantages of the interface, as well as to form priorities for its further improvement. **Conclusions.** The expert assessment method used is an effective tool for identifying problems in user interaction with augmented reality and for outlining areas for improving the quality of the interface. The results obtained can be used to further modernize the system, optimize the structure of information reproduction, and create a more intuitive, safe, and ergonomic user environment.

Keywords: augmented reality; quality criteria; human-machine interfaces; expert assessment; monitoring systems.

1. Introduction

Augmented reality (AR) technology is widely integrated into modern industrial and non-industrial information systems as the latest method of human-machine interaction with the aim of improving efficiency, safety, and ease of use [1–3]. AR applications cover manufacturing, transportation, education, medicine, defense, and a range of consumer services, where high-quality visualization and rapid information retrieval significantly affect task accuracy and reduce user errors. Such a high level of interest and growing involvement create a need for models, criteria, and formalized methods for evaluating the quality of proposed AR interfaces.

As the technology spreads, so do the requirements for its ergonomics, reliability, and compliance with international standards, in particular ISO 9241 and ISO/IEC 25010, which

regulate the quality indicators of interactive systems. In the context of augmented reality, criteria such as comprehensibility, intuitiveness, cognitive load, stability of visual elements, and correctness of spatial positioning are of particular importance. The lack of a systematic approach to evaluating these parameters complicates the development of effective AR solutions and limits the possibility of their improvement.

In this regard, it is important to develop a list of quality criteria for expert evaluation of AR interfaces, which will allow quantifying the quality level of the system and comparing different options for its implementation. Such an evaluation creates a basis for identifying critical shortcomings, determining the level of compliance with quality requirements, and forming recommendations for optimizing the design of AR interfaces.

2. Analysis of scientific publications and formulation of the research problem

In one of the previous publications [4] by the authors of this work, an analysis of the state of research was carried out and a review of the methods of assessing the quality of systems available at that time was made. The article concluded that most of the proposed methods are based on evaluating the quality of AR systems, since this indicator is the easiest way to assess the attractiveness of the software product being developed for the user. Most often, the heuristic research method is used for evaluation. This choice of evaluation methodology is due to the fact that the quality of use is a subjective characteristic, and the use of the same subjective evaluation method seems logical.

In more recent studies, this trend continues, although the list of evaluated indicators has been significantly expanded. Thus, in works [5–7], physical, mental, and time loads, as well as the efforts and productivity of the group of respondents, are taken into account along with SUS indicators. The results obtained using this approach can be considered more accurate, since it takes into account the impact of AR on the physical and psychological state of the respondents, which is not characteristic of classical methods of evaluating quality of use.

Works [8, 9] consider various options for assessing the quality of AR, mainly focusing on the visual aspect of this technology. Physical and mental load are not taken into account here, but quality indicators such as adaptability, content quality, aesthetics, immersion, etc. are considered. Of course, these criteria are important for human-machine systems with a high level of immersion, which includes augmented reality, but they ignore technical factors and the possible consequences of prolonged use of AR in the form of deterioration of physical and emotional states.

It is also worth highlighting study [10], which evaluates the safety of using systems with AR interfaces on the same hierarchical level as user satisfaction indicators. This quality criterion should be one of the key criteria for human-machine interaction systems with augmented reality, since they have a direct impact on the perception of the environment, but in most studies it is ignored. At the same time, this work lacks criteria for assessing physical and mental stress, as well as the visual component.

In general, it can be concluded that significant progress has been made in recent years in assessing the quality of human-machine AR interfaces, but in the vast majority of cases, existing

methods are used without taking into account the specifics of augmented reality. Therefore, determining key criteria for a comprehensive assessment of AR quality is a relevant scientific and practical task. This task is especially important for human-machine systems in which operators control the use of single unmanned aerial vehicles (UAVs), swarms of UAVs, and fleets that combine different swarm systems, where the significant dynamics of flight control require quick and accurate decisions. This applies, in particular, to unmanned systems for humanitarian demining, technical inspection, etc. [11–13]. The integration of AR into such interfaces is a new approach that requires appropriate research, in particular, on the assessment of interface quality and the impact of AR on the operator and the system as a whole.

3. Purpose and objectives of the study

The purpose of this study is to develop a set of quality criteria for human-machine augmented reality systems and to conduct an expert evaluation of the proposed interface. The study was aimed at identifying the strengths and weaknesses of the interface, identifying potential problems in user interaction with the system, and developing quantitative indicators that can be used for further design optimization. To this end, a 10-point expert survey was used, which made it possible to obtain comparable and interpretable results across a wide range of criteria, including functional completeness, responsiveness, self-descriptiveness, safety of use, aesthetic integrity, and others. The data obtained provided the opportunity for a comprehensive analysis of the interface and the formation of recommendations for its improvement.

4. Research materials and methods

4.1 Explosive object monitoring system

Since the human-machine interface cannot exist as an object separate from the system, the authors of the study decided to develop the architecture of the explosive object monitoring system shown in Fig. 1, where AR is considered as a means of visualizing information for UAV operators and ground robotic complexes (GRC). Such a system, as noted, is an example of human-machine systems with rapidly changing application dynamics.

The system includes a central server and databases responsible for processing and storing critical information about the geographical coordinates of areas potentially contaminated with explosive objects, historical data on the presence of mines, maps of demined areas, and telemetry data obtained from UAVs and GRCs. This data is processed by a central server, which transmits it to the control center and control station in real time.

The control center is the main analytical hub of the system, which manages the entire operation, monitors the status of drone swarms and demining teams, and analyzes the data received for quick decision-making.

Here, records, reporting systems, access lists (if necessary) are kept and confidential materials are controlled.

The demining group works directly in the area of operation, but thanks to augmented reality glasses, it receives critically important information that increases their level of safety. However, the AR interface for the demining group is not considered in this work.

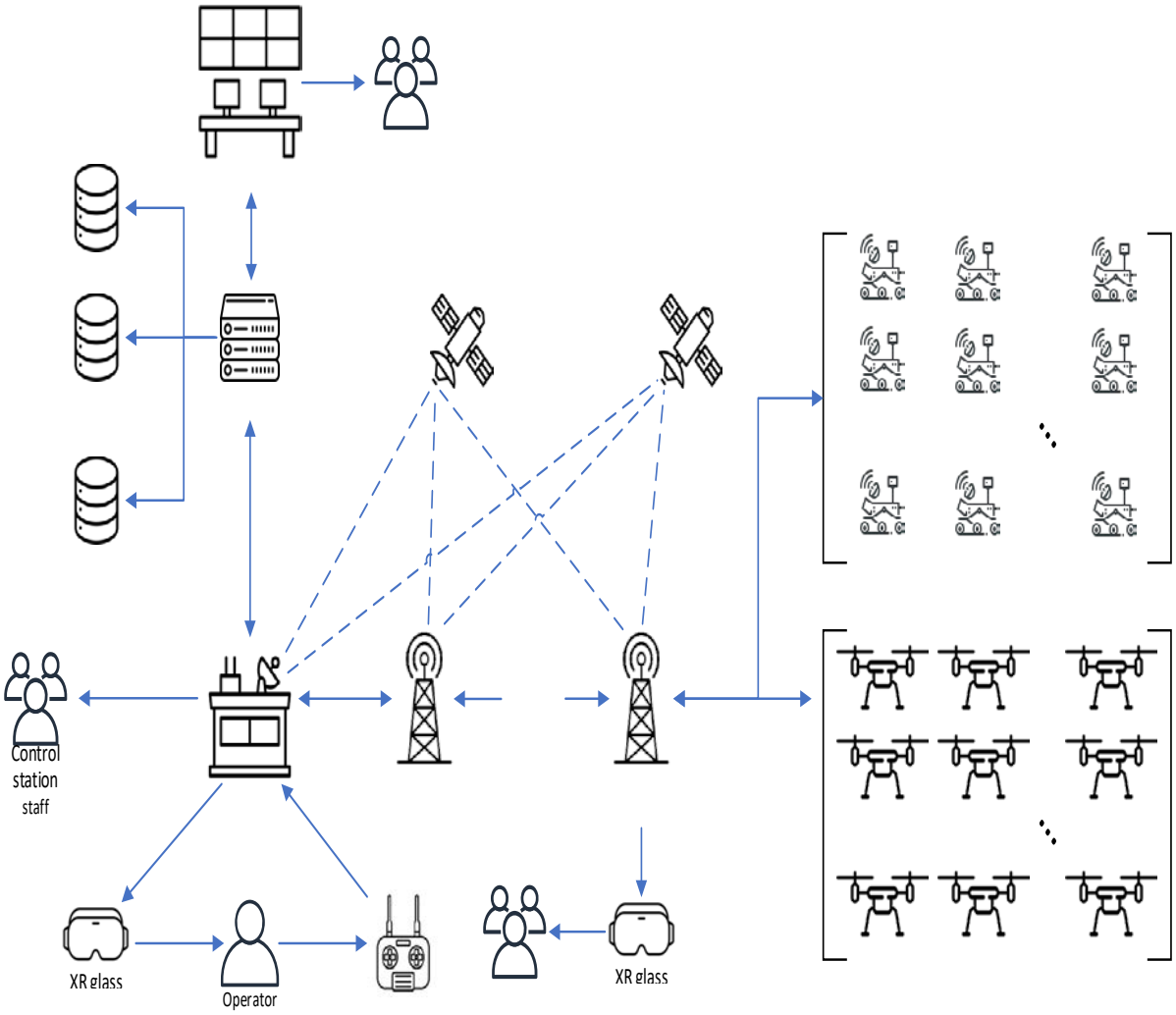


Fig. 1. Architecture of the explosive objects monitoring system

The control station is located in close proximity to the area under investigation and serves as a local control and maintenance point for UAVs and GRCs. It receives data from the server, provides communication with operators, and controls the interaction of drone swarms.

Operators at the control station work with remote controllers, which allows them to control individual UAVs or GRCs in critical situations, receive and analyze video streams in real time, and change flight routes according to new data.

Operators use augmented reality glasses to display AR interfaces, allowing them to view mine threat maps, control UAVs and GRCs, and receive important information.

4.2 Augmented reality interface for drone operators

Although augmented reality interfaces are quite common, there are very few examples of their use for interacting with UAVs. This can be explained by the fact that research in this area of AR use is still in its infancy.

For example, [14] presents an AR interface for mobile devices that allows you to control a fleet of UAVs using virtual manipulators on a touch screen. However, there is a significant drawback in that the operator must constantly maintain visual contact with the drone to control its flight direction and ensure communication with the mobile device.

This necessity is caused by the fact that the window displaying the video stream received from the UAV occupies less than 10% of the screen space and is partially covered by virtual controls. For mobile devices with serious limitations in display size and resolution, such a small display area makes it almost impossible to view information from the UAV.

This option of using AR to control a UAV can be used for personal purposes, but it is not suitable for performing tasks in monitoring systems.

In [15], it is proposed to use augmented reality to control UAVs from a first-person perspective using augmented reality glasses. A remote control is used to control the flight of the UAV, while information from the drone's cameras is displayed on virtual windows.

This option seems much more convenient in terms of control, but does not allow for the control of a fleet of UAVs. Also, in this version of the AR interface, there are no control elements such as charge level, altitude, speed, etc. For these reasons, it is not possible to use this option for monitoring systems.

In [16], an option is proposed that is positioned as an interface with augmented reality elements for UAV control. Unlike the previous study, this one has all the necessary indicators for controlling an unmanned aerial vehicle. However, the interface still has a number of drawbacks, namely the small size of the elements responsible for displaying the video stream from the UAV and the flight map, as well as the inability to control a group of unmanned aerial vehicles.

There is also a conceptual question of whether the proposed human-machine interaction system can be considered to contain augmented reality elements, since all virtual control elements are located in a 2D plane, although the classic definition of AR[17] requires them to be placed in 3D. Based on the above-described options and shortcomings of existing AR interfaces, it is advisable to present our own design option.

Fig. 2 shows a mock-up of one of the developed drone control interfaces with built-in augmented reality elements. This interface design aims to expand the operator's capabilities in the process of human-machine interaction.

The concept of augmented reality provides the possibility of tactile control of individual elements of human-machine interfaces, which theoretically simplifies and speeds up switching between the control of individual units of UAV swarms and GRC.

This, in turn, can contribute to improving the quality of use of such interfaces and enhance the user experience of operators, while preserving the basic elements of classic human-machine interfaces of unmanned aerial vehicles.

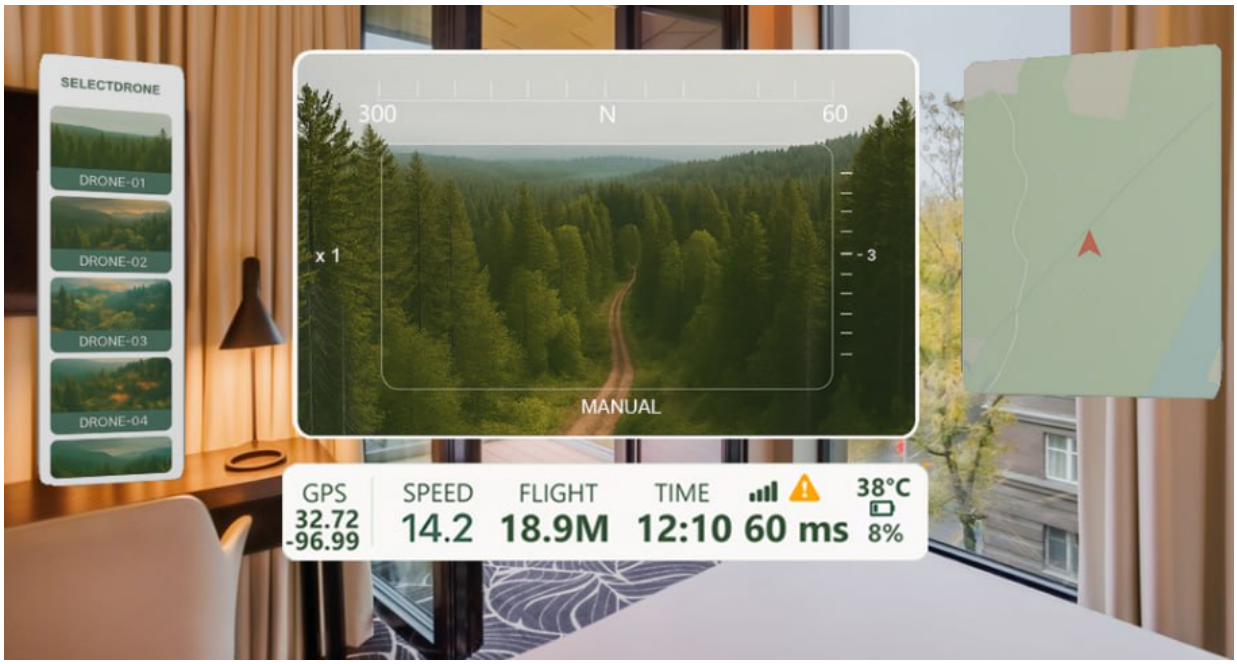


Fig. 2. Augmented reality interface layout for drone operators

The proposed augmented reality interface for the UAV and GRC operator consists of four virtual windows, each of which performs a separate function:

– the central interface window is the main one and is used by the operator to display the video data stream coming from the UAV and GRC. In addition to the images from the drones, the screen displays controls such as a compass for tracking the direction of movement and an indicator of the angle of inclination of the video camera from which the image is transmitted. Other controls are not displayed on this screen in order to reduce the number of factors that distract the operator during the task and to increase the field of view;

– to the right of the main window is a window with a map for monitoring the flight route;

– the bottom panel displays information necessary for the operator about the technical condition of the controlled UAV or GRC, its speed and altitude, operating time, position in space, etc. These controls are placed in a separate window to free up as much workspace as possible on the main screen;

– the window on the left is used for visual monitoring of the other drones in the system and quick switching between them. This is the only element of the augmented reality interface that involves touch interaction. This window is used not only to display the video stream from other drones in the system, but also to quickly switch between them using gesture controls. When you tap on an element in the window, control switches to the corresponding drone. You can also use gesture controls in the window to scroll through the list of drones and switch between the UAV and GRC tabs. This interaction mechanism is designed to speed up switching

between different types of drones and the drones themselves. Other interface elements do not allow such freedom in interaction with them. This restriction is implemented to reduce the risk of errors during drone control due to incorrect or mistaken gestures.

Thus, the augmented reality human-machine system described in Fig. 2 allows the drone control elements to be broken down into separate windows. Traditionally, this approach requires multiple monitors, but augmented reality has no such limitations and can use virtual monitors placed in the operator's field of view. These windows can be placed in any order and at any convenient point, changed in size and orientation, removed if necessary, and subsequently reopened. Such flexibility in use should have a positive impact on the mobility of the system and the operator's user experience.

Thus, the proposed interface has a number of significant advantages over the solutions presented in [14–16]. Unlike the analogues considered, the interface in Fig. 2 provides control based on information received directly from the UAV camera and supplemented with elements in the form of a compass displayed in the central window and a map of the area displayed in a separate window. The operator will not need to maintain constant visual contact with the controlled device, thereby limiting the area of monitoring tasks. Also, in the proposed version, telemetry data, including GPS coordinates, speed, altitude, flight time, image delay, temperature, signal strength, and battery status, are consolidated into a single compact panel, which should reduce visual noise and cognitive load thanks to better workspace organization. Integrated warnings about the current status of the controlled UAV are intended to further improve the operator's decision-making, and thus should reduce training time and increase the efficiency of real-time information processing compared to the solutions described conceptually in [14, 16].

The scientific novelty of this approach to interface design lies in the optimization of information presentation through the compact grouping of key telemetry indicators and auxiliary functions into separate windows, which frees up as much of the main workspace as possible from information noise. This differs from existing solutions [14–16], where telemetry elements are placed on the main screen, reducing the visibility of the video stream. Reorientation to a full-fledged AR environment, where functional elements are placed in space as separate virtual windows, eliminates physical limitations of the workspace and provides the ability to scale, rearrange, and personalize the interface. Also, unlike the known ones, the developed layout provides for the use of gesture interaction between unmanned aerial vehicles. The results obtained create the prerequisites for the further development of multi-window AR-based human-machine interaction systems focused on scalable control of UAV and GRC groups.

4.3 Quality criteria for AR interfaces

A list of criteria for further expert evaluation of the quality of augmented reality human-machine interfaces was prepared based on a combination of subjective and objective criteria. Subjective criteria include indicators describing ease of use, perception, intuitiveness, and the interface's compliance with user needs, formed in accordance with ISO/IEC 25010:2011 [18], ISO 9241-210:2019 [19] and the arc42 conceptual quality model [21]. Objective technical criteria that ensure reliability, correct display of information, and safe interaction were selected in

accordance with the requirements of NUREG-0700 [20] and the recommendations of IEEE P2048.101 [23]. The coordination of these sources made it possible to form a balanced and comprehensive set of indicators (Table 1), suitable for systematic analysis and comparative evaluation of augmented reality interfaces.

Table 1. *List of quality criteria for AR interfaces*

№	Name	Source
1	Functional Completeness	[18, 19]
2	Functional Appropriateness	[18]
3	Time Behaviour	[18, 20]
4	Resource Utilization	[18]
5	Learnability	[18, 19]
6	Faultless	proposed
7	Reactivity	proposed
8	User Error Protection	[18, 19]
9	Self-descriptiveness	[18, 19]
10	Safety	[18]
11	Durability	[21]
12	Hazard Warning	[18]
13	Predictability	[21, 22]
14	Suitability	[18]
15	Personalization	[21]
16	Coherence	[21]
17	Conciseness	[21, 20]
18	Controllability	[22, 19]
19	Aesthetic Integrity	[22]
20	Visual Accuracy	[22]
21	Readability	[21, 20]
22	Aesthetics	[19]
23	Alignment Accuracy	[23]
24	Frame Rate	[23]
25	Latency	[23]
26	Cognitive Load	proposed
27	Stress	[6, 7]
28	Fatigue	[6, 7]
29	Realism	proposed
30	Unambiguity	[20]
31	Distinguishability	[20]
32	Visibility	proposed
33	Interactivity	[8]

Functional completeness – the degree to which the set of functions covers all the defined tasks and goals of the intended users.

Functional Appropriateness – the degree to which the set of functions contributes to the fulfillment of the defined tasks and goals.

Timeliness – the degree to which the response time of the system during the performance of its functions meets the requirements.

Resource utilization is the degree to which the number and types of resources used in performing its functions meet the requirements.

Learnability is the degree to which the functions of a product or system can be learned for use by specific users within a specified period of time.

Faultless is the degree to which the interface influences the accuracy of user actions performed using the augmented reality system.

Reactivity is the degree to which the interface influences the speed of user response in specific situations.

Error protection – the degree to which the system prevents user errors during operation.

Self-descriptiveness – the degree to which the product provides relevant information necessary for the user to understand its capabilities and use without excessive interaction with the product or other resources such as documentation, support services, or other users.

Safety – the degree to which a situation that endangers life, health, property, or the environment can be avoided.

Durability – the degree to which a system is capable of remaining useful over a long period of time.

Hazard warning – the degree to which the system is capable of warning of unacceptable risks so that timely action can be taken to ensure the user's safety.

Predictability – the degree to which the consequences of user actions can be predicted, including the current state of the system.

Suitability – the degree to which functions are provided that meet the stated and implied needs of the intended users when used under the specified conditions.

Personalization – the degree to which the product can be modified or customized to suit the user's personal preferences.

Coherence – the degree to which interface elements are logically and aesthetically organized or integrated.

Conciseness – the degree to which the user interface is able to convey the necessary information concisely, clearly, and without unnecessary elements, but without sacrificing content or func*tionality.*

Controllability – the degree to which the user feels in control of the interface elements.

Aesthetic integrity – the degree to which the interface elements visually correspond to a single style.

Visual Accuracy– the degree to which the interface elements are free of visual, stylistic, or spelling errors.

Readability – the degree to which interface elements are legible and easy to read.

Aesthetics – the degree to which the interface provides a pleasant and satisfying interaction.

Alignment accuracy is the degree to which a virtual object deviates from its intended display location.

Frame rate is the degree to which the image refresh rate meets requirements.

Latency is the degree to which there is a delay between the user's movement and the image refresh.

Cognitive load – the degree to which the cognitive load on the user increases as they use augmented reality interfaces.

Stress – the degree to which the user's stress or tension level increases as they use augmented reality interfaces.

Fatigue – the degree to which the user's fatigue increases as they use augmented reality interfaces.

Realism – the degree to which the augmented reality interface reflects elements of the surrounding environment.

Unambiguity – the degree to which ambiguous interpretations of the purpose of augmented reality interface elements are not allowed.

Distinguishability – the degree to which individual interface elements, icons, and symbols differ from each other.

Visibility – the degree to which virtual elements stand out against the background of the real world.

Interactivity – the degree to which direct interaction with augmented reality elements is possible, including through touch, sound, and vibration.

4.4 Expert poll

To validate the results obtained, they must be verified within the framework of an experiment to evaluate the quality of the human-machine interaction system. In quality engineering, there is a wide range of possible methods, but in the context of augmented reality, the following are the most widely used:

– evaluation using metrics;

– focus group surveys;

– expert surveys.

At this stage of the research, it is not possible to use metric evaluation, as this method requires a fully formalized quality model and a working prototype of the system. It is also necessary for all participants in the evaluation process to have access to a device for displaying interfaces, which in this case are augmented reality glasses. This makes the metric evaluation method unsuitable at this time. The focus group survey method is also unsuitable for us, as this method involves obtaining feedback directly from the end users of the product being evaluated.

The expert survey method is the most acceptable because it allows us to evaluate subjective indicators of human-machine interfaces from the point of view of people with different levels of professional experience in using AR. This makes it possible to identify existing design flaws and improve the quality of the interface at the design stage.

Google Forms was chosen as the tool for conducting the expert survey because this service combines the accessibility, convenience, and functionality necessary for evaluating interfaces, and also guarantees that all responses are stored in the cloud for further statistical processing. This minimizes manual work, reduces the risk of errors, and makes the results immediately ready for analysis.

To evaluate the quality of the proposed augmented reality interfaces using the expert assessment method, the following questions were compiled based on the criteria presented in Table 1:

1) Can such an interface facilitate the performance of all the operator's anticipated tasks?

2) Can such an interface affect the accuracy of the operator's actions?

3) Can such an interface affect the operator's reaction speed?

4) Will it be possible to master such an interface in a short period of time?

5) Is such an interface intuitive?

6) Does such an interface allow avoiding situations that are dangerous to the physical health of the operator or the environment?

7) Can such an interface remain relevant for a long period of time?

8) Does the presence of augmented reality affect the moral aging of the interface?

9) Can such an interface warn the operator of possible risks?

10) Can the interface elements be customized according to the operator's personal preferences (the location of AR elements in space)?

11) Is such an interface capable of conveying the necessary information in a concise, understandable form?

12) Do the interface elements correspond to a single visual style?

13) Does this AR interface improve the realistic representation of the environment?

14) Does such an interface prevent ambiguous interpretation of the purpose of elements?

15) Do the AR elements of the interface stand out against the background of the real world?

The process of expert evaluation of the interface by means of a checklist survey in Google Forms using a 10-point scale consists of several consecutive stages that ensure the reliability, representativeness, and reproducibility of the results. First, an expert group is formed, consisting of specialists with relevant expertise in the field of human-machine interfaces or augmented reality. Based on the evaluation criteria, a checklist is created that includes indicators that can be measured quantitatively: interface clarity, user load, navigation intuitiveness, stability, visual clarity, etc. Each item on the checklist is formulated unambiguously and relates to only one criterion.

Next, a questionnaire is constructed in Google Forms based on this checklist. For each question, the answer type "Linear scale" is selected with a range from 1 to 10, where 1 means minimum compliance with the criterion and 10 means maximum compliance. For greater accuracy, a short text explanation is added (for example, "1 – absolutely not", "10 – yes, completely"). The survey may contain additional open-ended questions for qualitative comments from experts.

The use of a 10-point scale for evaluating user interfaces is appropriate given its increased sensitivity and analytical suitability. Unlike 3-, 5-, or 7-point scales, the 10-point scale provides a wider range of gradations, allowing respondents to more accurately reflect their perception of the convenience, clarity, or effectiveness of the interface. This scale reduces the likelihood of the "central tendency" effect, where participants tend to choose the average value, and promotes more differentiated assessments. In addition, the 10-point scale is often considered quasi-interval, which allows for the correct application of statistical analysis methods, including the calculation of mean, variance, correlations, and regression models.

This increases the accuracy of the results and provides a better analytical basis for researching the quality of user interfaces.

After creation, the form is sent to experts, and their responses are automatically collected in a spreadsheet. At the final stage, a statistical analysis is performed: the mean, median, standard deviation, and range of ratings for each criterion are calculated. A comparative analysis between experts or between different versions of the interface can also be performed. The results are interpreted to identify the strengths and weaknesses of the interface and form the basis for further recommendations for its improvement.

A group of 13 experts was involved in the quality assessment, consisting of specialists in the area of information technology with varying levels of experience in the use, development, and evaluation of human-computer interfaces. The vast majority of respondents are men aged 24 to 80, most of whom are graduate students. Almost half of the experts have a high level of expertise in evaluating the quality of human-machine systems, including augmented reality interfaces. Table 2 provides detailed characteristics of the expert group.

Table 2. *Characteristics of the expert group*

Indicator		Value	%
Gender	Male	11	84,62
	Female	2	15,38
Age	Under 30	5	38,46
	30 to 45	5	38,46
	Over 45	3	23,08
Position	Professor	4	30,77
	Associate professor	1	7,695
	Postgraduate student	7	53,84
	Developer (3 years of experience)	1	7,695
Level of expert experience in analyzing interface quality	Low	3	23,08
	Medium	4	30,77
	High	6	46,15

5. Research results

The results of the expert assessment presented in Fig. 3 demonstrate an uneven distribution of interface quality indicators across a range of metrics. The highest scores were given to criteria such as self-descriptiveness, trainability, functional completeness, and responsiveness, which were rated predominantly at 8 points. This indicates that the interface is intuitive, easy to learn, and capable of responding quickly to user actions. High scores were also observed for accuracy, visibility, realism, and aesthetic integrity, indicating an overall positive perception of the visual and behavioral components of the system.

Moderate ratings (7–8 points) were given to criteria such as unambiguity, conciseness, personalization, and relevance, indicating that there is room for improvement, but without critical comments from experts. The lowest scores were recorded for hazard warnings and durability,

which were rated between 4 and 5 points. This indicates that the system does not pay sufficient attention to safety aspects and specific mechanisms to ensure its stable functioning in the long term. A slightly below-average level was also recorded for safety of use, which may indicate a need for additional measures to inform users or reduce potential risks.

Overall, the interface demonstrates high levels of usability and clarity, but requires further optimization in areas related to security, stability, and long-term operation.

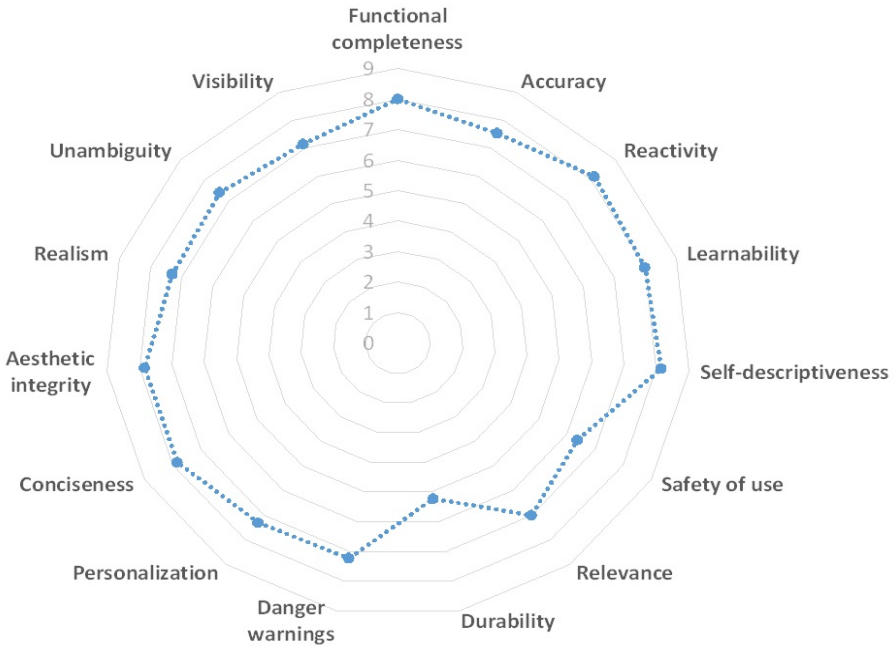


Fig. 3. Poll results diagram

6. Conclusions and further research

This study proposed a list of quality criteria for evaluating augmented reality human-machine interfaces using the example of interfaces for drone operators. The quality criteria were formed based on ISO/IEC 25010:2011, ISO 9241-210:2019, NUREG-0700, IEEE P2048.101, and other relevant sources. For the experimental evaluation using the expert survey method, a questionnaire with 15 questions was compiled and presented to 13 experts. The results of the expert assessment showed that the methodology used to analyze the quality of the augmented reality interface based on a structured checklist and a 10-point scale is an effective and valid tool for determining the usability and ergonomics of the system. The obtained assessments made it possible to quantitatively characterize the key parameters of the interface, in particular its clarity, consistency, informativeness, responsiveness, error protection, and aesthetic quality. Analysis of the score distribution diagram showed that most criteria meet an acceptable or high level of quality, but some aspects were identified that require further refinement, in particular, optimization of the structure of information presentation, reduction of cognitive load, and improvement of the stability of visualization elements.

Overall, the results confirm that the proposed approach to expert evaluation is appropriate for augmented reality systems for various purposes and can be used as a basis for the cyclical improvement of interface quality, ensuring their high efficiency, safety, and compliance with modern international standards. Further research may be directed at assessing the impact of the proposed augmented reality interface on the speed and accuracy of decision-making by unmanned system operators. Another important area of research could be the optimization of the structure of information display in the user environment and the development of augmented reality interfaces for the demining group, which also uses augmented reality to work with the explosive object monitoring system.

References

1. Fernández-Moyano, J.A., Remolar, I., Gómez-Cambronero, Á. (2025), "Augmented Reality's Impact in Industry – A Scoping Review", *Applied Sciences*, Vol. 15(5), P. 2415. DOI: https://doi.org/10.3390/app15052415
2. Arena, F., Collotta, M., Pau, G., Termine, F. (2022), "An Overview of Augmented Reality", *Computers*, Vol. 11(2). DOI: https://doi.org/10.3390/computers11020028
3. Karlsson, I., Bernedixen, J., Ng, A.H.C., Pehrsson, L. (2017), "Combining augmented reality and simulation-based optimization for decision support in manufacturing", *Proceedings of the 2017 Winter Simulation Conference (WSC), Las Vegas*, P. 3988–3999. DOI: https://doi.org/10.1109/WSC.2017.8248108
4. Kanarskyi, Y., Orehov, O., Stadnyk., A. (2022), "Assessing the quality of an augmented reality system: an analysis of the state of research", *Control, Navigation and Communication Systems,* No. 4(70), P. 79–87. DOI: https://doi.org/10.26906/SUNZ.2022.4.079
5. Sermarini, J., Maraj, C., Walters, L.C., Mouloua, M., Kider, J.T. (2025), "Evaluating the Effectiveness of Augmented Reality Interfaces for Quadrupedal Robot Motion Control", *Proceedings of the ACM Symposium on Spatial User Interaction (SUI '25)*, New York: ACM, Article 17, P. 1–12. DOI: https://doi.org/10.1145/3694907.3765931
6. Hamacher, A., Hafeez, J., Csizmazia, R., Whangbo, T.K. (2019), "Augmented Reality User Interface Evaluation—Performance Measurement of Hololens, Moverio and Mouse Input", *International Journal of Interactive Mobile Technologies (iJIM)*, Vol. 13(3), P. 95–107. DOI: https://doi.org/10.3991/ijim.v13i03.10226
7. Chan, W.P., Croft, M.J. (2022), "Design and Evaluation of an Augmented Reality HMD Interface for Human-Robot Teams in Shared Manufacturing Tasks", *ACM Transactions on Human-Robot Interaction*, Vol. 11(3), P. 1–19. DOI: https://doi.org/10.1145/3524082
8. Graser, S., Kirschenlohr, F., Böhm, S. (2024), "User Experience Evaluation of Augmented Reality: A Systematic Literature Review", *arXiv*. DOI: https://doi.org/10.48550/arXiv.2411.12777
9. Picardi, A., Caruso, G. (2024), "User-Centered Evaluation Framework to Support the Interaction Design for AR Applications", *Multimodal Technologies and Interaction*, Vol. 8(5). DOI: https://doi.org/10.3390/mti8050041
10. Chu, C-H., Liu, Y-L. (2023), "Augmented Reality UI Design and Experimental Evaluation for Human-Robot Collaborative Assembly", *Journal of Manufacturing Systems*, Vol. 68, P. 313–324. DOI: https://doi.org/10.1016/j.jmsy.2023.04.007
11. Mishchuk, V., Fesenko, H., Kharchenko, V. (2024), "Deep learning models for detection of explosive ordnance using autonomous robotic systems: trade-off between accuracy and real-time processing speed", *Radioelectronic and Computer Systems*, No. 4, P. 99–111. DOI: https://doi.org/10.32620/reks.2024.4.09

12. Fedorenko, G., Fesenko, H., Kharchenko, V., Kliushnikov, I., Tolkunov, I. (2023), "Robotic-biological systems for detection and identification of explosive ordnance: concept, general structure, and models", *Radioelectronic and Computer Systems*, No. 2, P. 143–159. DOI: https://doi.org/10.32620/reks.2023.2.12
13. Lysyi, A., Sachenko, A., Radiuk, P., Lysyi, M., Melnychenko, O., Ishchuk, O., Savenko, O. (2025), "Enhanced fire hazard detection in solar power plants: an integrated UAV, AI, and SCADA-based approach", *Radioelectronic and Computer Systems*, No. 2, P. 99–117. DOI: https://doi.org/10.32620/reks.2025.2.06
14. Costa, C., Gomes, E., Rodrigues, N. et al. (2025), "Augmented reality mobile digital twin for unmanned aerial vehicle wildfire prevention", *Virtual Reality*, Vol. 29, Article 71. DOI: https://doi.org/10.1007/s10055-025-01145-w
15. Sautenkov, O., Yaqoot, Y., Lykov, A. et al. (2024), "FlightAR: AR Flight Assistance Interface with Multiple Video Streams and Object Detection Aimed at Immersive Drone Control", *2024 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Bangkok, Thailand. PP. 614–619. DOI: https://doi.org/10.1109/ROBIO64047.2024.10907428
16. Bagassi, S., Fadda, T., & Corsi, M. (2024), "Advanced human machine interfaces for drone monitoring: assessment of the technological framework for the design of an augmented reality interface" [online]. Available at: https://www.icas.org/icas_archive/icas2024/data/papers/icas2024_1059_paper.pdf (Accessed: 27 November 2025)
17. Azuma, R.T. (1997), "A Survey of Augmented Reality", *Teleoperators and Virtual Environments*. Vol. 6(4). P. 355–385. DOI: https://doi.org/10.1162/pres.1997.6.4.355
18. International Organization for Standardization (2011), ISO/IEC 25010:2011 Systems and software engineering – System and software quality models. Geneva: ISO.
19. International Organization for Standardization (2019), ISO 9241-210:2019 Ergonomics of human-system interaction – Human-centred design for interactive systems. Geneva: ISO.
20. U.S. Nuclear Regulatory Commission (2020), NUREG-0700, Rev. 3. Human–System Interface Design Review Guidelines. Washington, DC: NRC.
21. Hruschka, P., Starke, G. (2025), *arc42 – Architecture documentation template*. Available at: https://arc42.org [Accessed: 20 November 2025].
22. Stanford University (2025), Usability principles. (Accessed: 20 November 2025).
23. Institute of Electrical and Electronics Engineers (2024), IEEE P2048.1 Standard for Virtual and Augmented Reality: Device Taxonomy and Definitions. New York: IEEE.

Received (Надійшла) 27.11.2025
Accepted for publication (Прийнята до друку) 09.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Kanarskyi Yevhenii – National Aerospace University "Kharkiv Aviation Institute", PhD Student at the Department of Computer Systems, Networks and Cybersecurity, Kharkiv, Ukraine;
e-mail: y.kanarskiy@csn.khai.edu; ORCID ID: https://orcid.org/0000-0001-9066-8642

Orekhov Oleksandr – PhD (Engineering Sciences), Associate Professor, National Aerospace University "Kharkiv Aviation Institute", Professor at the Department of Computer Systems, Networks and Cybersecurity, Kharkiv, Ukraine; e-mail: a.orehov@csn.khai.edu;
ORCID ID: https://orcid.org/0000-0001-6957-1934

Канарський Євгеній Олександрович – Національний аерокосмічний університет "Харківський авіаційний інститут", аспірант кафедри комп'ютерних систем, мереж та кібербезпеки, Харків, Україна.

Орєхов Олександр Олександрович – кандидат технічних наук, доцент, Національний аерокосмічний університет "Харківський авіаційний інститут", професор кафедри комп'ютерних систем, мереж та кібербезпеки, Харків, Україна.

ЕКСПЕРТНЕ ОЦІНЮВАННЯ КРИТЕРІЇВ ЯКОСТІ ЛЮДИНО-МАШИННИХ ІНТЕРФЕЙСІВ ДОПОВНЕНОЇ РЕАЛЬНОСТІ

У сучасних умовах активного поширення технологій доповненої реальності особливої актуальності набуває проблема забезпечення високої якості користувацьких інтерфейсів, які визначають ефективність, безпечність і надійність взаємодії користувача з цифровим середовищем. **Предметом дослідження** є методи й критерії оцінювання, спрямовані на аналіз зручності, зрозумілості, стабільності та функційної повноти інтерфейсу системи доповненої реальності для операторів безпілотних апаратів у складі системи моніторингу вибухонебезпечних об'єктів. Важливим є аналіз таких методів, як оцінювання за допомогою метрик якості, опитування фокус-груп і експертне опитування, на основі порівняння їх переваг і недоліків, що виявляються за певних умов. **Мета роботи** – комплексне експертне оцінювання інтерфейсу доповненої реальності з використанням структурованого чек-листа. Оцінюванню підлягали самоописуваність, керованість, захищеність від помилок, узгодженість, естетична цілісність, реактивність, надійність подання інформації та інші критерії, рекомендовані відповідними міжнародними стандартами. **Завдання:** розробити інтерфейс доповненої реальності; визначити можливі критерії якості з огляду на особливості людино-машинних систем певного типу; запропонувати перелік запитань для опитування експертів на основі виокремлених критеріїв якості; провести експертне опитування. **Результати дослідження:** створено макет інтерфейсу доповненої реальності для людино-машинної взаємодії оператора з безпілотними апаратами, сформовано перелік критеріїв для оцінювання якості систем доповненої реальності, а також підготовлено й проведено експертне опитування з метою оцінити якість пропонованого інтерфейсу. Результати опитування подано у вигляді радіальної діаграми, що дає змогу наочно визначити переваги й недоліки інтерфейсу, а також сформувати пріоритети його подальшого вдосконалення. **Висновки.** Метод експертного оцінювання є ефективним інструментом для виявлення проблем взаємодії користувачів з доповненою реальністю й для окреслення напрямів підвищення якості інтерфейсу. Отримані результати можуть бути використані з метою подальшої модернізації системи, оптимізації структури відтворення інформації та створення більш інтуїтивного, безпечного й ергономічного користувацького середовища.

Ключові слова: доповнена реальність; критерії якості; людино-машинні інтерфейси; експертне оцінювання; системи моніторингу.

Bibliographic descriptions / Бібліографічні описи

Kanarskyi, Y., Orekhov, O. (2025), "Expert evaluation of quality criteria for human-machine interfaces in augmented reality", *Management Information Systems and Devises*, No. 4 (187), P. 87–102. DOI: https://doi.org/10.30837/0135-1710.2025.187.087

Канарський Є. О., Орєхов О. О. Експертне оцінювання критеріїв якості людино-машинних інтерфейсів доповненої реальності. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 87–102. DOI: https://doi.org/10.30837/0135-1710.2025.187.087

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.8 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.103

M. Lapin, K. Bokhan

FEW-SHOT LEARNING OF A GRAPH-BASED NEURAL NETWORK MODEL WITHOUT BACKPROPAGATION

The subject of this article is a structural graph approach to classifying contour images in *few-shot* mode without using backpropagation**. The core idea** is to make the structure the carrier of explanations: the image is encoded as an attributive graph (critical points and lines as nodes with geometric attributes), and generalization is performed through the formation of concept attractors. **The purpose of the study** is to design and experimentally validate an architecture in which class concepts are formed from several examples (5–6 per class) by means of structural and parametric reductions, ensuring transparency of decisions and rejection of backpropagation of error. **Objectives of the work**: 1) define a vocabulary of nodes/edges and a set of attributes for contour graphs; 2) set normalization and invariance; 3) develop structural and parametric reduction operators as a monotonic simplification of the structure; 4) describe the procedure for aggregating examples into stable concepts; 5) build a classification through graph edit distance (Graph Edit Distance) with practical approximations; 6) compare with representative learning approaches on several examples. **Methods used**. Contour vectorization → bipartite graph (Point/Line as nodes); attributes: coordinates (normalized), length, angle, direction, topological degrees. Reductions: elimination of unstable substructures or noise, alignment of paths between critical points. Concepts are formed by iterative composition of samples; classification is based on the best match of the concept graph (GED with approximations). **Results of the study**. On a MNIST subset with 5–6 basic examples per class (one epoch), a consistent accuracy of approximately 82% was obtained with full traceability of solutions: errors are explained by specific structural similarities. An indicative comparison with SVM/MLP/CNN, as well as metric (ProtoNet) and meta-learning (MAML) lines, is presented in the form of a review graph. **Conclusions**. The structural graph scheme with concepts enables learning from multiple examples without backpropagation of error and provides built-in explanations through an explicit graph structure. Limitations relate to the cost of GED and the quality of skeletonization. Research prospects include optimization of classification algorithms, work with static scenes, and associative recognition.

Keywords: explainable artificial intelligence; few-shot machine learning; backpropagation; graph reduction.

Introduction

Recent advances in artificial intelligence (AI), particularly in deep learning and artificial neural networks (ANNs), have led to significant progress in solving complex problems [1–3]. However, the widespread use of these technologies has revealed a number of fundamental limitations that call into question the possibility of creating truly autonomous and adaptive systems [4–6].

These limitations include: the need for massive amounts of data for training, which requires significant time, computational, and energy resources [7, 8]; fundamental problems with generative models related to trust in information, "hallucinations", and the phenomenon of "entropy gap" [4, 7, 9]; and model degradation when training on recursively generated data (model autophagy disorder, MAD) [10, 11].

In this work, we assume that these problems are fundamental in nature, stemming from the current conceptual paradigm. Modern MLMs are based primarily on the statistical nature of learning and a rigid architecture that is optimized using the backpropagation algorithm [2, 3, 6].

Even specialized approaches to few-shot learning, such as meta-learning (MAML, Prototypical Networks) [12–14], are essentially complex methods of statistical optimization. They do not eliminate the fundamental dependence on statistics and cannot truly learn "from scratch" on a few examples, as they rely on models pre-trained on large data sets or require a complex meta-learning step.

This paper considers an alternative approach based on abandoning backpropagation in favor of biologically motivated structural generalizations. This paper presents a practical computational implementation of such an approach.

We demonstrate how visual patterns (contour images) can be represented as attributed graphs [15–17], where nodes (critical points, lines) and edges (spatial connections) encode the topological and geometric properties of an object.

The learning process is implemented as single-pass few-shot learning without backpropagation. It is based on the application of structural and parametric reduction operators, which operate by monotonic structural simplification. Iterative application of these operators on 5–6 unique samples causes the system to converge to a stable, generalized state with minimal structural complexity – a **generalized concept graph** (or prototype graph).

Analysis of recent studies and publications

The development of structural graph models for learning from a few examples lies at the intersection of several key research areas: few-shot learning [14], explainability methods (XAI) [18, 19], graph representations (GED) [20, 21], and alternative architectures (OvA/OvO) [22]. A review of the literature in these areas reveals fundamental conceptual limitations that the proposed approach aims to address [7, 23–25].

Few-shot/Meta-learning

The dominant deep learning models (CNN, MLP, Transformer) are fundamentally statistical and demonstrate low efficiency when trained on critically small datasets, requiring thousands of examples and many training epochs to achieve acceptable accuracy. To solve this problem, few-shot and meta-learning methods have been proposed [2, 7, 14, 26].

Prototypical Networks learn to identify class prototypes based on distance metrics in embedding space [13]. MAML (Model-Agnostic Meta-Learning) attempts to find the optimal initial weight initialization for fast adaptation [12]. Although both methods significantly improve accuracy on small samples, they do not eliminate the fundamental dependence on statistics and backpropagation.

They require a complex and resource-intensive meta-learning phase on large auxiliary datasets [14, 26].

Thus, this is a transfer of knowledge obtained statistically, rather than true one-pass learning "from scratch".

Explainable AI (XAI) and Graph Representations

As models have grown in complexity, the problem of their interpretability has become more acute. Deep learning models function as "black boxes". Popular XAI methods, such as LIME and SHAP, are post-hoc techniques: they attempt to approximate the behavior of an already trained model rather than explain its actual decision-making process [27, 28]. Studies have shown that such explanations can be unreliable, contradictory, and vulnerable to adversarial attacks [18, 19, 29, 30].

An alternative is "explainability by design", where the internal representation of the model is semantically meaningful [16, 18, 19]. Graph structures are ideal candidates for this because they allow semantics to be explicitly encoded in nodes and edges. Graph Edit Distance (GED) [20, 21] is used to compare such structures. However, GED is an NP-hard problem, which remains a challenge for practical application [31, 32].

Alternative architectures (OvA/OvO) and the problem of feature generalization

To solve classification problems, alternative ANN architectures have long been considered: "one-vs-all" (OvA) and "one-vs-one" (OvO) [22]. This is an approach where, instead of one large network, specialized networks are used (for example, one for each class). This approach is conceptually similar to the one we propose, where we build a single separate "neuron" (concept graph) for each class.

However, in classical implementations of One-vs-All / One-vs-One architectures (OvA/OvO), which rely on the backpropagation algorithm, there are noticeable limitations in detecting examples that go beyond the boundaries of the training data. One-vs-One, OvA/OvO) architectures, which rely on the backpropagation algorithm, there are noticeable limitations in detecting examples that fall outside the training distribution (Out-of-Distribution Detection, OOD) [33, 34]. Networks trained on limited examples do not form stable class separation boundaries.

This is because traditional ANNs generalize only **local recognition features** (e.g., individual textures or angles) and cannot generalize features at the level of the entire structure [23, 24]. Their fully connected and combinatorial nature with stochastic initialization makes it impossible to generalize global topological properties. Our approach solves this problem because generalization occurs not through stochastic optimization of local weights, but through deterministic structural reduction of the graph, which captures **global** topological features.

Synthesis: Identified conceptual gaps

A review of the literature reveals three distinct but interrelated problems:

1. Dependence of few-shot learning methods on the backpropagation algorithm: Leading few-shot methods (MAML, ProtoNets) are not true "zero-shot" learning, but rather knowledge transfer methods that require intensive prior training using backpropagation.

2. Unreliability of XAI: Existing XAI methods (LIME, SHAP) remain mostly post-hoc, unreliable, and vulnerable to attacks.

Feature locality in OvA: Classical architectures (including OvA/OvO) are unable to generalize global/structural features, leading to OOD problems and unstable decision boundaries

Research gaps, Purpose and Objectives

Research gaps

1. Reliability of explanations: Approaches with "by design" explainability are needed, rather than post-hoc approximations (LIME/SHAP).

2. Training on multiple examples without backpropagation: Leading methods (MAML, ProtoNets) still rely on gradient updates. Alternatives are needed that work in low-data modes without backprop.

3. Generalization of global features: Classical ANNs (including OvA) capture local patterns but are not capable of generalizing global topological structure, which is key to shape recognition.

4. GED complexity: Graph edit distance (GED) is NP-hard, which limits its practical application.

The purpose of the work is to develop and experimentally validate a structural graph approach to few-shot classification of contour images without backpropagation, in which the generalization of several class examples is performed through a sequence of structural and parametric reductions, and decision-making has built-in explainability due to the explicit graph structure.

Objectives

1. Representation. Define the representation of a contour image as an attributed graph (node/edge types, geometric attributes, normalization, and invariance) taking into account skeletonization/vectorization properties.

2. Reduction operators. Develop a set of structural (removal of unstable branches, merging of intersections, normalization of paths) and parametric (min-max-center ranges for numerical features) operators that simplify the set of examples into a concept attractor.

3. Aggregation of examples. Build a procedure for forming a concept from 5–6 examples per class in few-shot mode, fixing attribute tolerances and filtering random structures.

4. Classification. Design a concept matching scheme (GED with heuristics based on bipartite matching/local searches) with strict time and quality constraints.

5. Experimental protocol. Conduct tests on a subset of MNIST/similar contour sets: one epoch, 5–6 basic examples/class (+augmentations); evaluate accuracy, concept stability, computation time.

6. Comparison with FSL databases. Compare with representative methods (Prototypical Networks, MAML) as examples of metric and meta-learning approaches; provide an indicative graph (caution regarding different protocols).

Explainability and risks. Explicitly record structural subgraphs/attributes that support decisions; compare with post-hoc explanations and discuss limitations of applicability (when structure "does not explain").

Materials and Methods

This section describes in detail the methodological pipeline used to convert two-dimensional contour images into stable concept graphs and their subsequent classification. The methodology is based on the principles of structural generalization and rejects gradient optimization.

Representation of contours as attributed graphs

To achieve transparency and move away from the "opaque" weight matrices characteristic of traditional neural networks, a representation is proposed where "structure is the carrier of explanations".

The input contour image, obtained after the binarization and skeletonization stages, is transformed into an attributed graph.

The system encodes contours as bipartite graphs, whose structure strictly alternates between *Point* type nodes and *Line* type nodes. This architectural differentiation is fundamental because it allows the topological structure (critical points) to be clearly separated from the geometric properties (the segments that connect them).

Point nodes: Represent the topological structure and critical points of the contour. They are ontologically classified into four main types:

· *EndPoint*: Terminal nodes that mark the beginning or end of an open contour.

· *CornerPoint*: Nodes that mark sharp changes in direction (corners).

· *IntersectionPoint*: Nodes where three or more segments meet.

· *StartPoint*: A designated anchor node that defines the canonical starting point of the graph traversal to ensure consistency of comparisons.

Line nodes: Represent geometric properties. Importantly, line segments are represented as first-class nodes rather than edges. This allows them to be assigned rich semantic and geometric attributes on par with *Point* nodes, which is critical for subsequent parametric reduction operations.

Edges (Interconnections): *Point* and *Line* nodes are connected exclusively by bidirectional edges of type *CONNECTED_TO*. This creates a strict traversal pattern *Point* → *Line* → *Point* → …

Each node carries a set of attributes that encode measurable geometry and topology parameters, including: *normalized_x*, *normalized_y* (coordinates normalized to the invariant range [-1, 1]), *length* (segment length), *angle* (angle for CornerPoint), *quadrant* (discretized direction), *horizontal_direction*, and *vertical_direction*.

Invariance through normalization

To ensure invariance of representation to scale and shift, which is a necessary condition for the formation of stable attractors, all coordinates and related metrics (e.g., *length*) are normalized. Point coordinates are transformed into a centered system with a range of [-1, 1] using the formula:

$$normalized_x = \left(x - center_x\right) / center_x .$$

A similar formula is applied to y.

This process is the first step in parametric reduction ($R_{u,c}$), which converts absolute values specific to a particular instance into relative, generalized parameters.

The learning process as a structural reduction of a graph

The learning process (concept formation) in this work is fundamentally different from traditional statistical optimization (e.g., gradient descent along the loss function). It is viewed as a deterministic process of structural generalization that strives for a state of minimal structural complexity. This most stable, generalized state of the system, representing the invariant essence of the class (e.g., all variants of writing the Fgure "3"), is called a generalized concept graph.

The transition from a set of individual sample graphs $(G_1,\ldots,G_n)$ to a single concept graph C is a process of controlled simplification (reduction) of the structure. This process is controlled by a set of special reduction operators (Custom Reduction Operations, CRO), which act by reducing structural complexity or parametric variability, attempting to simplify the graph to a stable prototype in a finite number of steps.

The general reduction process can be described as a composition of three classes of operators:

$$R = R_w\left(R_{sp}\left(R_{u,c}\left(G_{input}\right)\right)\right),$$

where G_{input} – input graph;

$R_{u,c}$, R_{sp}, R_w – theoretical reduction operators.

A key aspect of our methodology is the direct comparison of these theoretical operators with specific CRO algorithms implemented in the system, as detailed in Table 1.

Table 1. *Structural and parametric reduction (CRO) operators*

Theoretical Operator	**Name and Purpose**	**Practical Implementation (CRO)**	**Algorithm Details**
$R_{u,c}$ (Parametric Reduction)	Minimization of parametric variability. Transition from quantitative values to generalized qualitative ranges.	Parametric Generalization	Numeric properties $v_1,\ldots,v_n$: merge into a rang $min:\min(v_i), max:\max(v_i), center:\text{avg}(v_i)$; this generalizes variations (e.g., length, angle). Categorical properties $s_1,\ldots,s_n$: merge into s_1 only if $s_i = s_1$ for all i; otherwise, the attribute is removed (filtering of inconsistent parameters). List properties $L_1,\ldots,L_n$: merge due to the intersection of sets $L_1 \cap \ldots \cap L_n$; this preserves only universal labels (e.g., Point).
R_{sp} (Structural-Parametric Reduction)	Simplification (reduction) of the structure based on the stability of its parameters.	Path Pruning	For two aligned critical points, the algorithm finds all simple paths between them. It selects the "best match" of paths based on the similarity of nodes and uses the shorter path as a template. Nodes from the longer path that do not have a match are removed. This "eliminates length variations".

Continuation of the table 1

Theoretical Operator	**Name and Purpose**	**Practical Implementation (CRO)**	**Algorithm Details**
R_w (Structural Reduction)	Removal of topological elements that are statistically insignificant (noise).	Endpoint Removal and Intersection Point Merging	Endpoint removal: The algorithm calculates the similarity matrix of endpoints between the concept C_i and the sample G_{i+1}. Endpoints with low similarity (below the threshold) or "extra" points are removed along with the entire path to the nearest critical point. Intersection merging: Consolidates *IntersectionPoint* nodes representing the same structural feature. Applies semantic reduction (e.g., *IntersectionPoint* with a degree < 2 becomes *CornerPoin*t or *EndPoint*).

Iterative algorithm for attractor formation

The learning process is one-pass and does not require backpropagation of error. It iteratively builds an attractor based on a very small sample consisting of 5–6 unique training samples per class.

The concept is initialized with the first sample graph: $C_0 = G_1$. This sample acts as an initial hypothesis about the class structure. Each subsequent sample G_{i+1} is integrated into the current concept C_i using a reduction operation $C_{i+1} = CRO(C_i, G_{i+1})$.

Each operation CRO is a five-step process that applies the reduction operators from Table 1:

1. Alignment of starting points: Establishing a common origin for graph traversal C_i and G_{i+1} by clustering and selecting StartPoint.

2. Preprocessing of critical points: Applying structural operators R_w (*Endpoint removal, Intersection merging*) to achieve basic structural compatibility.

3. Traversal synchronization: Generating synchronized paths between corresponding critical points in both graphs.

4. Common structure identification: Applying a structural-parametric operator R_{sp} (*Path pruning*) to normalize paths and eliminate length variations between critical points.

5. Parametric merging: Application of a parametric operator $R_{u,c}$ (*Parametric Generalization*) to merge node attributes that remain after structural reduction.

This iterative process is path-dependent; the order in which samples are submitted affects the final concept graph. This mimics a process where the initial hypothesis (G_1) is iteratively refined under the influence of new data (G_{i+1}), which acts as a reduction force, eliminating sample-specific variations (noise) and leaving only the generalized core.

Classification through approximate graph matching (GED)

The classification (inference) process consists of comparing the graph G_{test} obtained from an unknown input image with each concept graph C_k from the trained library, minimizing the graph edit distance (GED) to the input graph G_{test}:

$$Class\left(G_{test}\right) = \arg\min_k GED\left(G_{test}, C_k\right).$$

GED is defined as the minimum cost of a sequence of operations (insertion, deletion, replacement of nodes/edges) required to transform G_{test} into C_k.

In order for the GED metric to correctly take into account the generalized nature of concepts, we use our own cost functions.

Node Substitution Cost: The cost of substitution a node $v \in G_{test}$ with a node $u \in C_k$ is calculated based on range-based cost functions.

- For numerical attributes (e.g., *length*, *angle*): If the attribute v value (e.g., $v_{length} = 5.5$) falls within the trained attribute u range (e.g., $u_{length} = \{min: 4, max: 7, \ldots\}$), the substitution cost for this attribute is 0. If the value is outside the range, the cost is proportional to the distance to the nearest range boundary.
- For categorical attributes: The cost is 0 in case of exact match or infinite (high) in case of mismatch.
- Label compatibility: The replacement cost is infinite if the base node types are incompatible (e.g., *Line* to *Point*).

Edge editing cost: Reduced cost to prioritize topological differences (presence/absence of nodes) over connectivity differences. Calculating the exact GED is an NP-hard problem. To ensure practical applicability, an approximation is used via a hard 60-second timeout for each individual comparison $GED\left(G_{test}, C_k\right)$.

This timeout acts as a heuristic approximation that interrupts the search for the optimal editing path if it takes too long and returns the best distance found at that moment.

Classification and winner selection mechanism

The proposed architecture implements an approach that is conceptually similar to One-vs-All, where each class k is represented by a separate "neuron" that is a generalized concept graph C_k. The classification (inference) process consists of comparing the contour in the form of a graph G_i with each concept graph C_k from the trained library.

Unlike stochastic networks, where the "excitation" of a neuron is a numerical output (e.g., softmax), in our system, the "excitation" of a k-th neuron is the process of calculating the editorial distance $GED\left(G_{test}, C_k\right)$. To select the final classification result, we apply the Winner-Takes-All concept.

The winner is the class (concept) C_k with the smallest editorial distance from the input graph (G_{test}).

$$Class\left(G_{test}\right) = \arg\min_k \{GED\left(G_{test}, C_1\right), \ldots, GED(G_{test}, C_N)\}.$$

If the distance is the same for several classes, the conflict resolution rule is applied. The class that is structurally more complex is selected.

Complexity is calculated as the sum of the nodes and edges of the graph (Fig. 1).

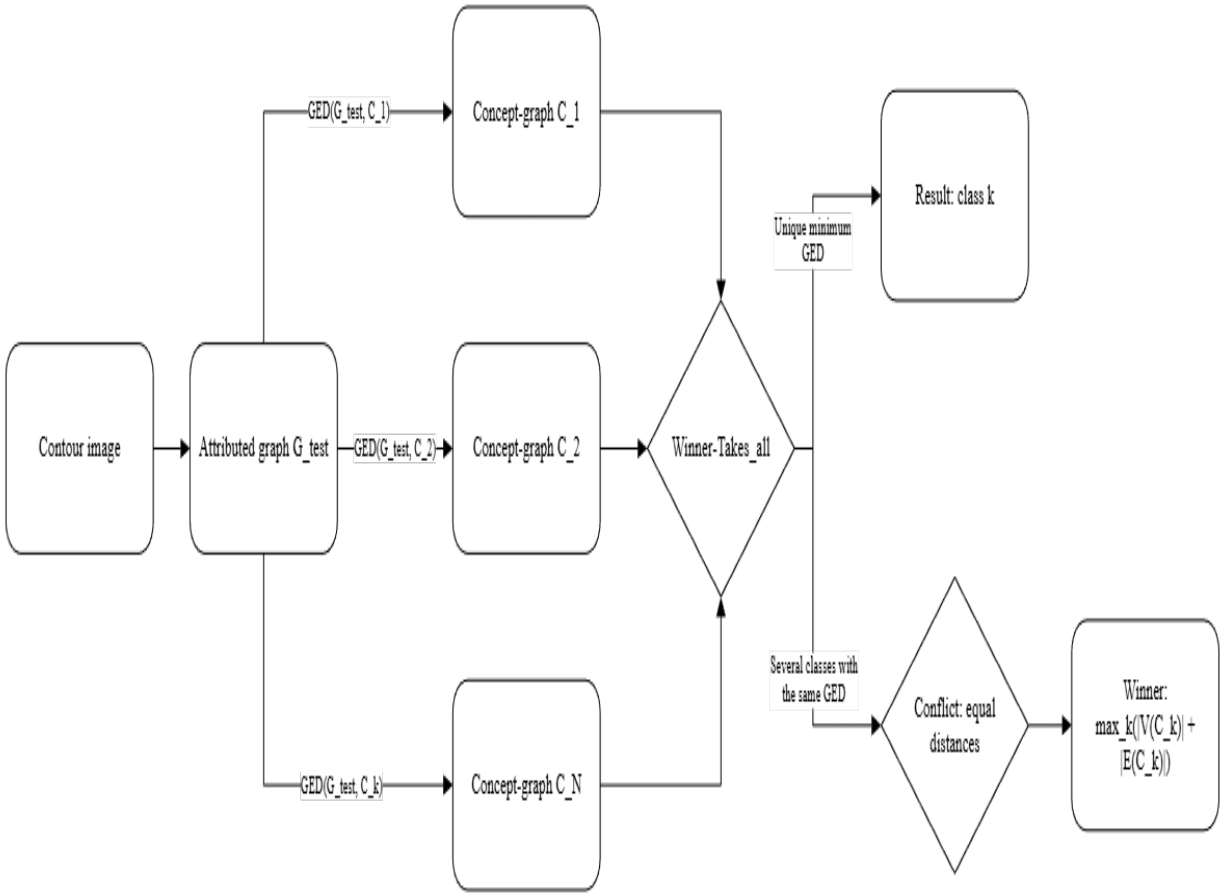


Fig. 1. Classification scheme using GED and WTA

Results and discussion

This section presents empirical validation of the proposed graph-based approach to concept formation.

The goal is not to optimize absolute accuracy, but to demonstrate that stable, explainable concept attractors can be formed from extremely limited data (few-shot learning) and that their performance and error patterns directly follow from their topological and parametric structure.

Experiments are conducted on a subset of MNIST-6 (classes "1", "2", "3", "6", "7", "9"), using 5–6 unique training samples per subclass.

Classification performance on MNIST-6 in Few-Shot mode

The system was trained on 8 concepts covering 6 classes (some classes, such as "1" and "2", had two concepts to represent different writing styles).

Training consisted of iterative structural reduction of 5–6 base samples (with 10 augmentation variants per sample, for a total of about 350 examples) for each concept.

Evaluation was performed on a test sample of 5.467 images that were not used in the formation of concepts.

Overall performance metrics are presented in Table 2.

Table 2. *Overall classification performance (5,467 test images)*

Metrics	Value (%)
Accuracy	82.35
Precision	83.28
Recall	82.35
F1 Score	82.16

These results are conceptually significant. The accuracy of 82.35 % demonstrates that the approach based on the formation of canonical structural attractors without gradient optimization is viable and provides meaningful classification.

The processing pipeline showed high reliability, with a success rate of 100 %, except for 10 images (0.18 %) that could not be processed due to skeletonization errors that resulted in disconnected graphs.

Analysis of class-wise performance and topological distinctiveness

An in-depth analysis of metrics for each class (Table 3) reveals a direct dependence of performance on the structural uniqueness of digits.

Table 3. *Class-based classification metrics*

Digit	Precision (%)	Recall (%)	F1 (%)	Quantity
1	81.46	96.49	88.34	997
2	84.17	60.02	70.07	948
3	78.21	87.28	82.50	983
6	94.23	78.09	85.40	753
7	74.38	82.12	78.06	990
9	91.55	89.57	90.55	786

Key observations:

1. High Precision for "6" (94.23 %) and "9" (91.55 %): These classes have the most unique topological signatures – closed cycles represented by *IntersectionPoint* nodes. Their attractors are very specific, which minimizes false positives.

2. Low Recall for "2" (60.02 %): This indicator shows that a significant portion (almost 40 %) of true "2" digits were not recognized. This indicates a high morphological variability in the writing of "2", which the formed concepts ("2_1" and "2_2") were unable to fully cover. Their parametric ranges, studied from only 5–6 samples, proved to be too rigid.

3. Low Precision for "7" (74.38 %): This class was most often confused with others, indicating its structural ambiguity, especially with regard to the digit "1".

Confusion Matrix Analysis

The confusion matrix (Figure 2) provides a deep understanding of how the model makes decisions by visualizing systematic errors that are a direct result of structural and topological similarities.

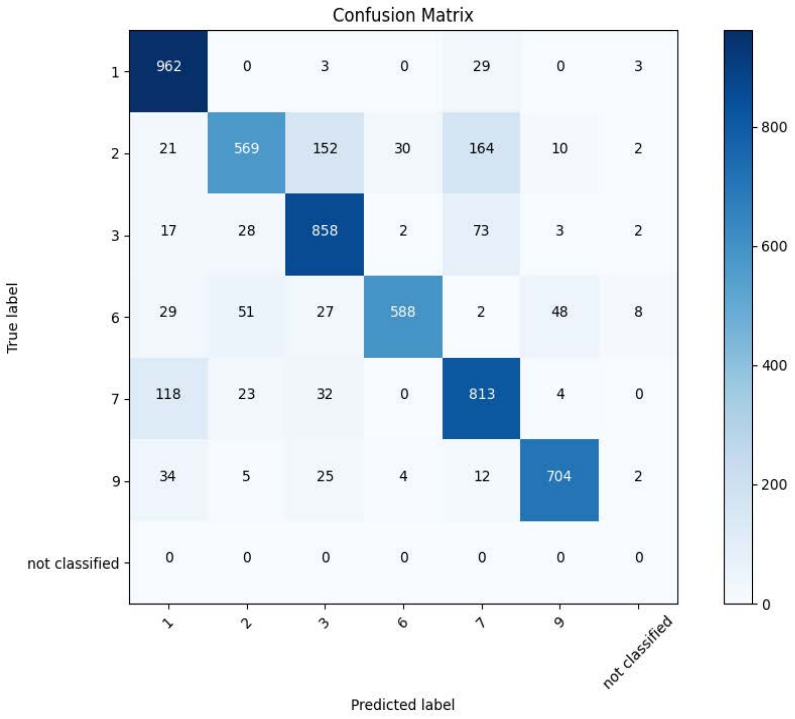


Fig. 2. Matrix of mismatches for the 6-class digit MNIST classification
(The primary mismatch occurs between digits 7 and 1 (angular open contours), and the secondary mismatch occurs between digits 2 and 3 (curved open contours). Digits with closed contours (6, 9) show strong discrimination.)

Primary mismatch: 152 samples of digits "2" were classified as "3". 28 samples of "3" were classified as "2". Secondary discrepancy: 118 samples of digits "7" were classified as "1". Classes "6" and "9" show minimal discrepancy between themselves and other open contours (for example, only 48 samples of "6" were misclassified as "9").

Unlike "black boxes", where the causes of errors are hidden in millions of weights, the errors in this model are fully interpretable. Analysis shows that errors are concentrated along structurally similar pairs:

1. **"2" vs "3"**: Both digits have similar "curved morphology". They are open contours that start on one side, have several bends (represented by CornerPoint nodes), and end on the other side.

2. **"7" vs "1"**: Both digits are "angular open contours". They are both simple paths consisting of a StartPoint, CornerPoint, and EndPoint. The mismatch occurs when the writing of "7" is less curved, or "1" has a more pronounced angle at the beginning.

The fact that the model confuses "7" with "1" (structurally similar) but does not confuse "7" with "6" (structurally different – open contour vs. closed) is strong evidence that the graph matching mechanism works correctly and makes decisions based on topology, as designed.

Stability of concept attractors and structural explainability (XAI)

This section analyzes the final result of the learning process – stable concept attractors, which are the carriers of explanations in the system.

The process of structural reduction transforms multiple training graphs into single canonical structures. Their metrics (Table 4) quantitatively determine the "ideal" form of each digit.

Table 4. *Structural metrics of concept attractors*

Concept	Nods	Edgrs	Av. Degree	Critical points (EP, CP, IP, SP)
1_1	3	2	1.33	1 EP, 1 SP
1_3	3	2	1.33	1 EP, 1 SP
2_1	7	6	1.71	1 EP, 2 CP, 1 SP
2_2	12	12	2.00	1 EP, 3 CP, 1 IP, 1 SP
3_1	7	6	1.71	1 EP, 2 CP, 1 SP
6_1	10	10	2.00	3 CP, 1 IP, 1 SP
7_1	5	4	1.60	1 EP, 1 CP, 1 SP
9_2	8	8	2.00	2 CP, 1 IP, 1 SP

EP = EndPoint; CP = CornerPoint; IP = IntersectionPoint; SP = StartPoint

Analysis of Table I demonstrates a direct correlation between digit topology and the complexity of its attractor.

Simple linear structures ("1"): Concepts "1_1" and "1_3" are minimal, consisting of only 3 nodes (StartPoint, Line, EndPoint). This perfectly reflects their topology as a simple, unbranched path.

Closed contours ("6", "9"): These concepts have a higher average degree (2.00), indicating the presence of cycles. Importantly, they do not contain an EndPoint (EP = 0), but they do contain an IntersectionPoint (IP = 1) where the cycle closes.

Open curved contours ("2", "3", "7"): These concepts have intermediate complexity (5–12 nodes). They all contain exactly one EndPoint (EP = 1), which topologically marks them as open contours. The number of CornerPoints (CP) encodes the number of bends (e.g., "7_1" has 1 CP, "2_1" has 2 CP).

This table is essentially a dictionary for XAI. The explanation for the "9" classification is that the input image graph successfully matched the "9_2" concept, which is canonically defined as an 8-node structure with 1 IntersectionPoint (cycle) and 0 EndPoints (no free ends).

Case Study: Iterative Stabilization of the Attractor (Digit "3")

The process of concept formation (Figures 3, *a–d*) is an empirical demonstration of theoretical reduction operators.

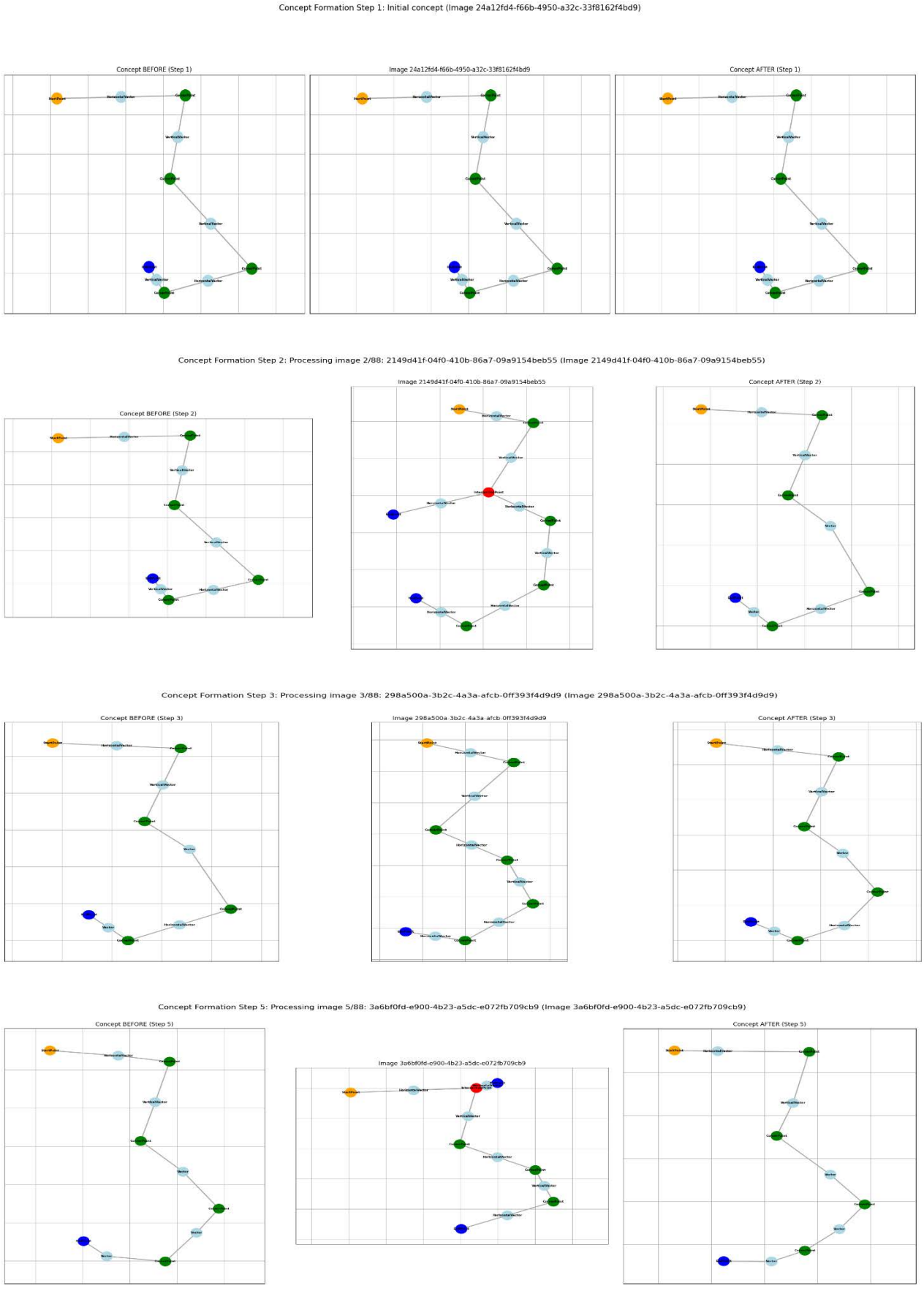


Fig. 3. Concept formation process

Step 1 ($C_0 = G_1$): The first pattern (G1) establishes the initial concept C_0. It is overly specific and contains all the structural details and noise of the initial pattern (Fig. 3, *a*).

Step 2 ($C_1 = CRO(C_0, G_2)$): Integration of the second sample (G2) reveals a discrepancy – a "redundant endpoint branch". The structural reduction operator (Endpoint removal) is applied, which removes this noise specific to G1. This is a practical implementation of the operator R_w that finds a common substructure (Fig. 3, *b*).

Steps 3 and 4 (C_2, C_3): Further iterations continue this process, removing the "redundant corner point" (Fig. 3, *c*) and another "noise substructure" (Fig. 3, *d*).

The final concept C_3 (Fig. 3, *d*) is a stable attractor representing the most general topological structure ("curved S-shape") common to all training samples. This process is a form of learning without backpropagation of error, where it is not the weight vector that is optimized, but the representation structure itself.

Example of parametric generalization (Digit "3")

Structural reduction determines which nodes remain, while parametric generalization determines how their attributes are generalized to encode variability. Using the example of the concept "3_1" (formed from 3 samples):

Numeric Properties: Attributes such as coordinates are not averaged but converted to ranges ({min, max, center}). This creates flexible decision boundaries.

- $normalized_x$: [–0.7, 0.2] (center –0.33)
- $normalized_y$: [0.3, 0.9] (center 0.63)

Count Properties: Topological variations are also encoded as ranges.

- endpoint_counts: {min: 2, max: 4, center: 2.67}
- intersection_point_counts: {min: 0, max: 2, center: 0.67}

Categorical Properties: Only stored if there is a 100% match.

- contour_type: “OPEN” (all samples were open).
- horizontal_direction: Removed (values were contradictory, e.g., "Left", "Right").

This process is a powerful XAI tool. The range endpoint_counts: {min: 2, max: 4} is a transparent, interpretable boundary. It shows that the model learned from training samples (which had, for example, 2, 4, and 2 endpoints) to expect that valid instances of "3" can have between 2 and 4 endpoints, with an ideal value (center) of 2.67.

This provides recognition flexibility while maintaining verified structural constraints.

Comparative Analysis in the Context of Few-Shot Learning

To evaluate the effectiveness of the proposed approach (referred to as ComAN in the experimental materials), its results are compared with other machine learning models under severely limited data (few-shot) conditions. The data for comparison is taken from experimental reports.

Since the request requires a visual comparison, the following table (Table 5) serves as the data source for a conceptual graph (bar chart) comparing accuracy.

Table 5. *Comparison table*

Model	Unique samples	Epochs of learning	Source	Accuracy (%)
ComAN (Our model)	**Up to 36 (5–6/class)**	**1**	**This work**	**82.44**
Nielsen RMNIST/5 (CNN)	50 (5/class)	10–50	Nielsen (2017)	84.38
Prototypical Networks	50–100	Purpose-learning	Snell et al. (2017)	80–90
MAML	50–100	Purpose-learning	Finn et al. (2017)	80–95
CNN (Standard)	500–1000	10–50	Krizhevsky et al. (2012)	74–78
SVM (RBF)	500–600	1	LeCun et al. (1998)	69–75
MLP (Standard)	400–600	10–50	Goodfellow et al. (2016)	53–61

Analysis of this comparison reveals three key conclusions:

1. Competitive accuracy: The accuracy of the ComAN model (82.44 %) is highly competitive. It significantly outperforms standard approaches such as MLP (53–61 %) and SVM (69–75 %), which demonstrate low performance or collapse on such small datasets.

2. Fundamental difference from Meta-Learning: At first glance, MAML (up to 95 %) and Prototypical Networks (up to 90 %) outperform ComAN. However, these models are not "few-shot" in the same sense.

They are meta-learners. They require extensive "pre-training on task distribution" or "on base classes" using backpropagation to "learn to learn". The ComAN model does not require any pre-training. It builds its concepts (attractors) from scratch, de novo, in a single pass (single-epoch training).

This is a radically different learning paradigm based on structural reduction rather than statistical optimization.

3. Comparison with a direct competitor (Nielsen CNN): The most relevant comparison is with Nielsen RMNIST/5, where CNN was trained on the same number of samples (5 per class). CNN Nielsen (84.38 %) shows a slight advantage in accuracy (~ 2 %) over ComAN (82.44 %). However, this advantage comes at the cost of complete loss of interpretability and significantly higher training costs: Nielsen requires 10–50 epochs, backpropagation, dropout, and hyperparameter tuning.

Our model achieves ~ 98 % (82.44/84.38) of SOTA accuracy using only 1 epoch, 0 backpropagation, and providing 100 % transparency.

This comparison empirically confirms the central thesis of the study: the system maintains competitive performance in few-shot mode while providing full structural interpretability.

Conclusions and prospects

Recent advances in artificial intelligence (AI), particularly in deep learning and artificial neural networks (ANNs), have led to significant progress. However, the widespread application of these technologies has revealed fundamental limitations that call into question the viability of the current approach.

Current ANN paradigms face a number of conceptual crises. They require enormous amounts of data for training, as well as significant time, computational, and energy resources. In addition to their high cost, these models, especially generative ones, exhibit significant reliability issues, generating errors and "hallucinations" that significantly undermine confidence in their results.

This directly leads to the phenomenon of "data inbreeding", also known as "Model Autophagic Disorder" (MAD). When models trained to favor statistical probability begin to learn from synthetic data generated by themselves, they enter a recursive cycle.

This process inevitably leads to rapid "information degradation and model collapse" as the entropy of the system continuously decreases, reinforcing averaging and eliminating any novelty.

Conclusions and prospects for further research

This study presents a comprehensive approach to AI that moves away from purely statistical methods in favor of biologically grounded principles of structural generalization.

The paper successfully presents and experimentally validates a unified theoretical and practical framework. This framework combines the principles of structural generalization with a practical, transparent, and highly efficient XAI system based on generalized graph concepts (prototypes).

The main contribution is to demonstrate that abandoning statistical optimization (backpropagation algorithm) in favor of deterministic graph reduction allows:

1. Achieving competitive classification accuracy (82.35 %).
2. Work in training mode on small samples (5–6 samples per class).
3. Perform training in a single pass without backpropagation.
4. Ensure complete internal explainability and transparency of decision-making.

Despite the successful validation of the concept, the current implementation has clear bottlenecks that outline directions for future research.

Computational limitation. The classification (inference) process relies on graph matching, which in general uses graph edit distance (GED), which is an NP-complete problem. This creates a significant computational load at the inference stage, resulting in an average processing time of ~ 3.5 seconds per image and the need for timeouts (e.g., 60 seconds).

In fact, a compromise was made: the computational complexity of training (backpropagation) was replaced by the combinatorial complexity of inference (GED).

Sensory limitation (preprocessing). The model is "fragile" and depends on the quality of the input "sensory" data:

1. Errors in preprocessing lead to a complete failure in processing, since the model cannot construct a correct graph.

2. Invariance is limited by the range used in augmentation (). More significant rotations destroy the structural alignment because they change the attributes (e.g., quadrants) of line nodes.

Representation limitation. The model is "blind" to any information not related to shape. The current approach "discards texture and gradient information", limiting its application exclusively to shape and contour recognition tasks.

The identified limitations directly point to prospects for further research:

1. Short-term prospects include solving immediate engineering problems: researching fast GED approximation algorithms to speed up inference; developing more robust skeletonization methods; and extending the graph representation to include texture and gradient attributes, transforming the model into a multimodal one (in terms of physical parameters).

2. The long-term vision addresses the most fundamental limitation of the current research: "the lack of modeling of evolutionary biological inter-neuronal connections". The current ComAN model successfully implements the concept of a "grandmother cell" – one static concept (neuron) is responsible for one class.

The next fundamental step is to move from modeling individual neurons to modeling dynamic networks of these neurons. This will require the development of mechanisms by which these graph concepts can dynamically interact, compete (e.g., through "Winner Take All" mechanisms), and form more complex, hierarchical "models of the world".

This is the path to creating AI systems that not only mimic biological efficiency but also approach true biological plausibility.

References

1. Goodfellow, I., Courville, A., Bengio, Y. (2016), "Deep learning", *The MIT Press*, Cambridge, Massachusetts, 800 p. ISBN: 978-0-262-03561-3.
2. Heaton, J. (2018), "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning", *Genetic Programming and Evolvable Machines*, Vol. 19, No. 1–2, P. 305–307. DOI: https://doi.org/10.1007/s10710-017-9314-z
3. LeCun, Y., Bengio, Y., Hinton, G. (2015), "Deep learning", *Nature*, Vol. 521, No. 7553, P. 436–444. DOI: https://doi.org/10.1038/nature14539.
4. Bender, E. M., Gebru, T., McMillan-Major, A., Shmitchell, S. (2021), "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?", *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT '21)*, ACM, P. 610–623. DOI: https://doi.org/10.1145/3442188.3445922
5. Zador, A. M. (2019), "A critique of pure learning and what artificial neural networks can learn from animal brains", *Nature Communications*, Vol. 10, No. 1, P. 3770. DOI: https://doi.org/10.1038/s41467-019-11786-6
6. Marcus, G. (2018), "Deep Learning: A Critical Appraisal", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1801.00631.

7. Parzhyn, Y., Lapin, M., Bokhan, K. (2025), "A New Approach to Building Energy Models of Neural Networks", *Advanced Information Systems*, Vol. 9, Issue 4, P. 100–119. DOI: https://doi.org/10.20998/2522-9052.2025.4.13.
8. Strubell, E., Ganesh, A., McCallum, A. (2019), "Energy and Policy Considerations for Deep Learning in NLP", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1906.02243
9. Ji, Z., Lee, N., Frieske, R. et al. (2023), "Survey of Hallucination in Natural Language Generation", *ACM Computing Surveys*, Vol. 55, Issue 12, P. 1–38. DOI: https://doi.org/10.1145/3571730
10. Alemohammad, S., Casco-Rodriguez, J., Luzi, L. et al. (2023), "Self-Consuming Generative Models Go MAD", *arXiv*. DOI: https://doi.org/10.48550/arXiv.2307.01850.
11. Shumailov, I., Shumaylov, Z., Zhao, Y. et al. (2024), "The Curse of Recursion: Training on Generated Data Makes Models Forget", *arXiv*. DOI: https://doi.org/10.48550/arXiv.2305.17493.
12. Finn, C., Abbeel, P., Levine, S. (2017), "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1703.03400.
13. Snell, J., Swersky, K., Zemel, R. S. (2017), "Prototypical Networks for Few-shot Learning", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1703.05175.
14. Wang, Y., Yao, Q., Kwok, J. et al. (2020), "Generalizing from a Few Examples: A Survey on Few-Shot Learning", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1904.05046.
15. Bai, X., Yang, X., Latecki, L. J., Liu, W., Tu, Z. (2015), "A comparative study using contours and skeletons as shape representations for binary image matching", *Pattern Recognition Letters*, Vol. 65, P. 159–165. DOI: https://doi.org/10.1016/j.patrec.2015.04.007.
16. Parzhin, Y., Galkyn, S., Sobol, M. (2022), "Method For Binary Contour Images Vectorization Of Handwritten Characters For Recognition By Detector Neural Networks", *2022 IEEE 3rd KhPI Week on Advanced Technology (KhPIWeek)*, Kharkiv, Ukraine, P. 1–6. DOI: https://doi.org/10.1109/KhPIWeek57572.2022.9916331
17. Shen, W., Jiang, Y., Gao, W. et al. (2016), "Shape recognition by bag of skeleton-associated contour parts", *Pattern Recognition Letters*, Vol. 83, P. 321–329. DOI: https://doi.org/10.1016/j.patrec.2016.02.002.
18. Adadi, A., Berrada, M. (2018), "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)", *IEEE Access*, Vol. 6, P. 52138–52160. DOI: https://doi.org/10.1109/ACCESS.2018.2870052.
19. Guidotti, R., Monreale, A., Ruggieri, S. et al. (2018), "A Survey Of Methods For Explaining Black Box Models", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1802.01933.
20. Gao, X., Xiao, B., Tao, D. et al. (2010), "A survey of graph edit distance", *Pattern Analysis and Applications*, Vol. 13, No. 1, P. 113–129. DOI: https://doi.org/10.1007/s10044-008-0141-y.
21. Sanfeliu, A., Fu, K.-S. (1983), "A distance measure between attributed relational graphs for pattern recognition", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, Issue 3, P. 353–362. DOI: https://doi.org/10.1109/TSMC.1983.6313167.
22. Rifkin, R., Klautau, A. (2004), "In Defense of One-Vs-All Classification", *Journal of Machine Learning Research*, Vol. 5, P. 101–141.
23. Parzhin, Y. (2014), "Hypotheses of neural code and the information model of the neuron-detector", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1411.6768.
24. Parzhin, Y. (2017), "The detector principle of constructing artificial neural networks as an alternative to the connectionist paradigm", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1707.03623.
25. Parzhin, Y. (2025), "Architecture of Information", *arXiv*. DOI: https://doi.org/10.48550/arXiv.2503.21794.

26. Chen, W.-Y., Liu, Y.-C., Kira, Z. et al. (2020), "A Closer Look at Few-shot Classification", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1904.04232.
27. Lundberg, S., Lee, S.-I. (2017), "A Unified Approach to Interpreting Model Predictions", *arXiv*. DOI: https://doi.org/10.48550/ARXIV.1705.07874.
28. Ribeiro, M. T., Singh, S., Guestrin, C. (2016), "«Why Should I Trust You?»: Explaining the Predictions of Any Classifier", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1602.04938.
29. Rudin, C. (2019), "Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1811.10154.
30. Slack, D., Hilgard, S., Jia, E. et al. (2020), "Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1911.02508.
31. Blumenthal, D. B., Gamper, J. (2020), "On the exact computation of the graph edit distance", *Pattern Recognition Letters*, Vol. 134, P. 46–57. DOI: https://doi.org/10.1016/j.patrec.2018.05.002.
32. Bougleux, S., Brun, L., Carletti, V. et al. (2017), "Graph edit distance as a quadratic assignment problem", *Pattern Recognition Letters*, Vol. 87, P. 38–46. DOI: https://doi.org/10.1016/j.patrec.2016.10.001.
33. Hendrycks, D., Gimpel, K. (2018), "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks", *arXiv*. DOI: https://doi.org/10.48550/arXiv.1610.02136.
34. Yang, J., Zhou, K., Li, Y. et al. (2024), "Generalized Out-of-Distribution Detection: A Survey", *arXiv*. DOI: https://doi.org/10.48550/arXiv.2110.11334.

Received (Надійшла) 18.11.2025
Accepted for publication (Прийнята до друку) 30.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Lapin Mykyta – National Technical University "Kharkiv Polytechnic Institute", PhD Student, Department of System Analysis and Information and Analytical Technologies, Kharkiv, Ukraine; e-mail: Mykyta.Lapin@cit.khpi.edu.ua; ORCID ID: https://orcid.org/0009-0003-6307-1172

Bokhan Kostiantyn – PhD (Engineering Sciences), National Technical University "Kharkiv Polytechnic Institute", Associate Professor at the Department of System Analysis and Information and Analytical Technologies, Kharkiv, Ukraine; e-mail: kostiantyn.bokhan@khpi.edu.ua;
ORCID ID: https://orcid.org/0000-0003-3375-2527

Лапін Микита Олексійович – Національний технічний університет "Харківський політехнічний інститут", аспірант кафедри системного аналізу та інформаційно-аналітичних технологій, Харків, Україна.

Бохан Костянтин Олександрович – кандидат технічних наук, Національний технічний університет "Харківський політехнічний інститут", доцент кафедри системного аналізу та інформаційно-аналітичних технологій, Харків, Україна.

НАВЧАННЯ ЗА КІЛЬКОМА ПРИКЛАДАМИ (*FEW-SHOT*) ГРАФОВОЇ МОДЕЛІ НЕЙРОННОЇ МЕРЕЖІ БЕЗ ВИКОРИСТАННЯ ЗВОРОТНОГО ПОШИРЕННЯ ПОМИЛКИ

Предметом роботи є структурно-графовий підхід до класифікації контурних зображень у режимі *few-shot* без використання зворотного поширення похибки**. Основна ідея** – зробити структуру носієм пояснень: зображення кодується у вигляді атрибутивного графа (критичні точки й лінії як вузли з геометричними атрибутами), а узагальнення виконується через формування концепт-атракторів. **Мета дослідження** – спроєктувати та експериментально підтвердити архітектуру, у якій концепти класів утворюються з кількох прикладів (5–6 на клас) способом структурних і параметричних редукцій, забезпечуючи прозорість рішень і відмову від зворотного поширення помилки. **Завдання роботи**: 1) визначити словник вузлів / ребер і набір атрибутів для контурних графів; 2) задати нормалізацію та інваріантності; 3) розробити структурні та параметричні редукційні оператори як монотонне спрощення структури; 4) описати процедуру агрегації прикладів у стабільні концепти; 5) побудувати класифікацію через відстань редагування графа (*Graph Edit Distance*) з практичними апроксимаціями; 6) порівняти з репрезентативними підходами навчання за кількома прикладами. **Застосовані методи**. Векторизація контуру → двочастковий граф (*Point/Line* як вузли); атрибути: координати (нормовані), довжина, кут, напрям, топологічні степені. Редукції: усунення нестабільних підструктур або шумів, узгодження шляхів між критичними точками. Концепти утворюються ітеративною композицією зразків; класифікація – за найкращою відповідністю графа концепту (GED з апроксимаціями). **Результати дослідження**. На підмножині MNIST із 5–6 базовими прикладами на клас (одна епоха) отримано узгоджувану точність приблизно 82 % за повної трасованості рішень: помилки пояснюються конкретними структурними подібностями. Подано індикативне порівняння з SVM/MLP/CNN, а також метричною (*ProtoNet*) і метанавчальною (MAML) лініями у вигляді оглядового графіка. **Висновки**. Структурно-графова схема з концептами забезпечує навчання за кількома прикладами без зворотного поширення помилки й надає вбудовані пояснення через явну графову структуру. Обмеження стосуються вартості GED та якості скелетизації. Перспективи дослідження – оптимізація алгоритмів класифікації, робота зі статичними сценами й асоціативне розпізнавання.

Ключові слова: зрозумілий штучний інтелект; *few-shot* машинне навчання; зворотне поширення помилки; редукція графів.

Bibliographic descriptions / Бібліографічні описи

Lapin, M., Bokhan, K. (2025), "Few-shot learning of a graph-based neural network model without backpropagation", *Management Information Systems and Devises*, No. 4 (187), P. 103–122. DOI: https://doi.org/10.30837/0135-1710.2025.187.103

Лапін М. О., Бохан К. О. Навчання за кількома прикладами (*few-shot*) графової моделі нейронної мережі без використання зворотного поширення помилки. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 103–122.
DOI: https://doi.org/10.30837/0135-1710.2025.187.103

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 655.3:658.011.4 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.123

M. Slutskin, O. Vovk

ANALYSIS OF INFORMATION TECHNOLOGIES FOR COMMUNICATION MANAGEMENT IN A PRINTING COMPANY

The subject of the article is the management of communications in a printing company in the context of digital transformation of production and growing competitive pressure in the printing services market. **The purpose of the work** is to study communications management based on the integration of modern information technologies, to develop an appropriate functional and information structure that minimizes the risk of information loss, speeds up decision-making, ensures a clear division of responsibilities, and increases the level of satisfaction of both internal users and external customers. **Tasks:** to identify the characteristics of internal and external communications of a printing company; to analyze the structure of information flows; to assess the impact of IT tools on the speed and accuracy of data transmission; to justify the feasibility of implementing CRM and ERP systems to optimize business processes. **Research methods:** system and process analysis, structural and functional modeling of communications, comparative analysis of digital tools, and analytical interpretation of empirical data on the effectiveness of information interaction. **Results achieved.** It has been established that the traditional model of communication in printing production is determined by the fragmentation of information flows, delays in the transmission of technical data, and a high probability of errors during order approval. The integration of CRM and ERP systems, electronic document management, and cloud-based corporate services reduces order fulfillment time, decreases the number of errors in technical documentation, and increases the transparency of management decisions and the level of customer and staff satisfaction. **Conclusions.** The digitization of communications is shaping a modern model of printing company management that combines automated interaction between departments with continuous control and monitoring of production processes. The implementation of CRM and ERP solutions, analytical platforms, and electronic coordination systems is a strategic direction for the development of the printing industry, as it contributes to increasing the competitiveness and adaptability of businesses to the requirements of the digital economy.

Keywords: printing; communication; information; management; technology; ERP; CRM; digitalization; integration; efficiency.

1. Introduction

In today's information-rich and highly competitive environment, printing companies are increasingly faced with the need to implement new forms of organization of production and management processes that would ensure not only stable functioning, but also flexible response to dynamic market changes, personalization of orders, the need for high-speed data processing, and integrated communication. In this context, communications management is one of the main tools for ensuring coordination between departments, prompt execution of tasks, minimization of delays and errors, and building long-term partnerships with customers and contractors. Printing production has a complex structure that includes several key stages: order receipt and processing, layout creation, technical task approval, material preparation, printing, post-press processing, quality control, and logistics. Each of these stages requires constant communication between technical specialists, managers, equipment operators, the supply department, management, and external customers.

At the same time, it is particularly important to have complete and up-to-date information support at every stage of the production process. Information gaps, delays, duplicate messages, and inconsistencies in file versions and technical documentation lead to wasted time and resources, reduced quality of the final product, and an increased risk of missing deadlines.

Against the backdrop of these challenges, the formalization and optimization of communication processes using systematic analysis methods and information technology is becoming increasingly important. Traditional approaches to communication management, which are based primarily on phone calls, face-to-face meetings, local file storage, or the use of non-specialized messengers, are giving way to digital platforms that provide end-to-end management of communication flows, document version control, task automation, and integration with accounting and information storage systems.

In this context, ERP (Enterprise Resource Planning) systems, which integrate all functional units of an enterprise into a single environment, as well as CRM (Customer Relationship Management) solutions, which enable structured work with customers by tracking order history, requests, feedback, and personal preferences, are of particular importance.

Electronic document management systems make it possible to automate the approval of technical tasks, estimates, and layouts, as well as to ensure that the chronology of changes is preserved. Equally important are tools for internal communication: cloud services, digital workspaces, and corporate messengers, which allow you to create centralized channels for information exchange and reduce dependence on informal communication.

Thus, the problem lies in the fact that modern printing companies need to rethink their approaches to organizing communications as a systematic process that covers the entire product life cycle – from the first contact with the customer to the final delivery of the finished product.

An additional important aspect is the study of information flow security issues, including access to confidential information, control over changes in documents, and auditing of communication activities. It is expected that the results of solving the set tasks will not only contribute to the improvement of the internal organization of the enterprise, but also provide the opportunity to scale management decisions, integrate with external partners, and increase overall competitiveness in the printing services market.

2. Analysis of literary sources and definition of the research problem

In scientific literature, communication management is considered an important component of the effective functioning of any enterprise, particularly in the context of the digital transformation of production. In general management theory, communication is treated as a basic function that ensures the transmission, exchange, processing, and interpretation of information between different departments and levels of management [1].

Renowned Ukrainian researcher Ivanilov (2019) emphasizes the critical role of management communications in innovative structures, where the speed and accuracy of data exchange determines the competitiveness of an organization [2].

In the printing industry, which has a complex production structure with a large number of technological stages and closed cycles, the issue of effective communication is of particular importance. According to Golovko (2020), modern printing companies operate in an environment of highly dynamic demand, shorter order fulfillment times, and growing requirements for product personalization, which necessitates the implementation of integrated information systems [3].

In particular, Print MIS and ERP solutions allow for the standardization of data exchange between departments, the minimization of human error, and the formation of transparent analytics for management decisions [4].

Research on digital communication management technologies conducted by Ukrainian scientists (Ponomarenko, 2020; Balabanov, 2021) indicates a significant increase in the efficiency of organizations when implementing CRM, ECM, and BPM systems. The authors emphasize that the implementation of such solutions allows for the automation of internal and external information flows, ensures the preservation of knowledge in the corporate database, and increases the adaptability of the enterprise to changes in the external environment.

At the same time, business process modeling using BPMN, IDEF0, or SADT methods, which allow visualizing the logic of communication channels (Curtis, Kellner & Over, 1992; White, 2004), becomes particularly relevant.

In the context of printing, communication management issues are also considered in the works of Kopievsky (2021), which analyze the problems of interaction between the production workshop, the design department, and customer service [5].

It is noted that major disruptions in the production process are often associated with inconsistencies in technical requirements, late detection of errors in layouts, or changes in order specifications, which requires a comprehensive approach to communication management. In this context, it is particularly important to create a unified information environment in which all participants in the process can have immediate access to up-to-date information.

Classic communication theories, pioneered by Shannon and Weaver (1949) and further developed in the business context (Kotler & Keller, 2016), also remain relevant. Their approaches to interpreting information noise, communication channels, and feedback form the basis for modern models of interaction between entities within an enterprise. In practical terms, this is reflected in the creation of digital customer accounts, internal employee portals, and automated notifications in the chain from order receipt to delivery of finished products.

We note the research of Ukrainian authors who consider the implementation of electronic document management systems, corporate databases, and artificial intelligence technologies in enterprise management (Kovalenko, 2022; Nazarenko, 2021).

These works emphasize that information technology not only simplifies communication but also acts as a key factor in the transformation of the management model – from hierarchical to networked, where the speed and accuracy of relevant information transfer play a major role.

3. Purpose and objectives of the study

To study the process of communications management based on the integration of modern information technologies, to develop an appropriate functional and information structure that minimizes the risk of information loss, speeds up the decision-making process, ensures a clear division of responsibilities, and increases the level of satisfaction of both internal users and external customers.

To achieve this goal, the following tasks must be solved:

- analyze the essence and structure of the communication process at a printing company, investigate existing problems of internal and external communication management in the printing industry;
- determine the role of information technologies in ensuring effective interaction between the production, marketing, sales, and customer service departments, consider modern software solutions (CRM, ERP) used to automate communication processes;
- assess the impact of digital technologies on the speed of information exchange, reduction of errors, and improvement of management decisions;
- develop recommendations for the implementation of information technologies in the communication management system of a printing company;
- analyze the effectiveness of using IT solutions to increase the competitiveness of the printing business.

Solving these tasks involves analyzing existing approaches to the classification of communication processes in the manufacturing sector, identifying key bottlenecks and barriers to effective communication, researching modern methods of building information models (in particular, based on BPMN, IDEF0, or DFD), as well as analyzing the functional capabilities of the most common ERP, CRM, and document-oriented platforms in the context of the needs of a printing company. The main focus is on identifying the potential of digital information systems (in particular, ERP and CRM solutions, electronic document management, and platforms for team collaboration) that can improve the quality of information exchange between structural units, increase the speed of decision-making, reduce information loss, and ensure effective feedback communication with customers.

4. Research materials and methods

Managing communication processes at a printing company is one of the most difficult management tasks, as this area is characterized by a high level of technological interdependence between production links, multi-stage document flow, and the need for prompt coordination of actions between departments.

Effective communication in this context is a critical factor in ensuring a smooth production process, meeting order deadlines, and satisfying customers. At the same time, there are a number of bottlenecks and barriers in the activities of printing companies that complicate the exchange of information and lead to reduced efficiency.

One of the key problem areas is the disconnect between the company's departments: sales, design, production, and accounting. The lack of a unified communication system leads to duplication of tasks, loss of important information, and delays in the transfer of print orders. The communication scheme at a printing company before the implementation of the CRM system is clearly demonstrated in the diagram (Figure 1).

Sales managers often record orders in their own spreadsheets or informal notes, which complicates further monitoring of order fulfillment. This creates the risk of discrepancies between customer expectations and actual production capabilities. Printing companies, especially small and medium-sized ones, often operate on outdated software or without integrated information systems at all. As a result, customer data, technical specifications, and financial information are stored in different departments in a fragmented manner. This situation complicates centralized control and analysis and creates difficulties in accessing up-to-date information. In addition, the lack of document flow automation results in significant time expenditures for agreeing on technical specifications, approving layouts, and signing contracts.

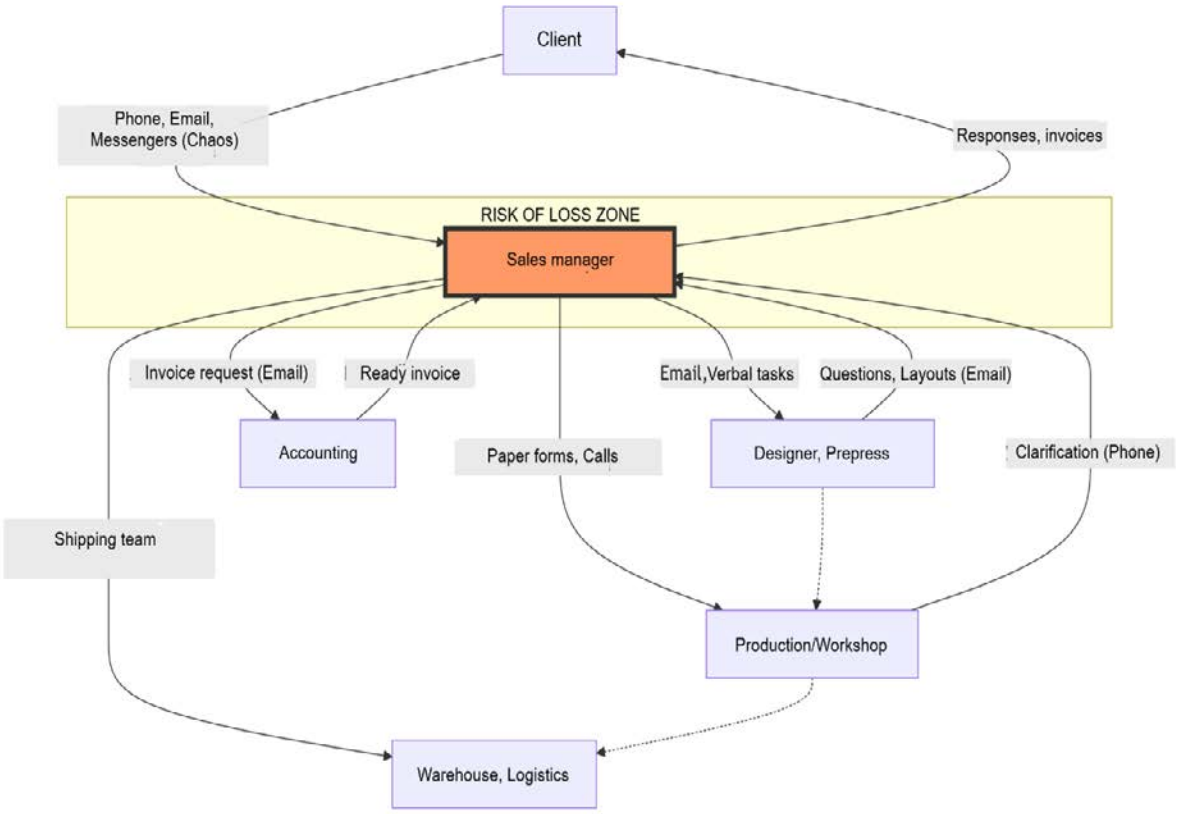


Fig. 1. Flow control diagram at a printing company prior to the introduction of modern information technologies

Communication barriers are largely due to insufficient training and motivation of personnel. Production shop floor employees often do not have complete information about customer requirements because managers do not always convey details in an accessible form. Designers,

in turn, may misinterpret technical specifications, leading to the need for numerous revisions. Employees often resist the use of new digital communication tools due to their unwillingness to change established work practices.

An important problem is the redundancy or, conversely, the insufficiency of information. Sales managers sometimes give customers overly optimistic production deadlines without coordinating them with production. At the same time, during the order fulfillment stage, shop floor employees may not have up-to-date information about design adjustments or material specifications. Such information gaps lead to discrepancies between the final result and the initial order and create the risk of conflicts with customers.

Effective communication can be blocked by differences in professional culture between departments. For example, production staff focus on technical parameters and deadlines, while the sales department focuses on customer needs and financial results. This leads to conflicts of interest and misunderstandings between departments. Psychological factors, such as low levels of trust between employees, lack of teamwork, or an authoritarian management style, also reinforce barriers to information sharing.

The combination of these bottlenecks creates systemic problems: delays in order fulfillment, increased production costs, reduced quality of the final product, and, as a result, loss of customers. Equally important is the deterioration of the internal climate at the enterprise, which reduces staff motivation and can lead to personnel losses.

Thus, the key barriers to effective communication in a printing company cover organizational, technological, informational, human, and cultural-psychological aspects. Overcoming them requires a systematic approach, the implementation of integrated information systems, the standardization of data exchange, the development of corporate culture, and staff training. Removing these barriers creates the conditions for increased productivity, higher customer satisfaction, and a stronger competitive position for the company [6].

ERP (Enterprise Resource Planning) systems are actively used in the printing industry to provide comprehensive management of resources, production, and information flows within the company. One example is the PrintVis system, developed on the Microsoft Dynamics 365 Business Central platform, which is specifically tailored to the needs of the printing industry. Its key purpose is to automate all production and management processes of a printing house: from receiving orders and preparing for printing to logistics and financial reporting. The program allows you to effectively control each stage of the order life cycle, from the moment it is received, to the formation of specifications, calculation of production costs, production planning, and ending with cost accounting, billing, and profitability analysis.

One of the key features of PrintVis is the ability to flexibly model processes according to the structure of a specific printing company. This allows the system to be adapted for different printing methods, as well as for packaging or label production. The program supports complex interaction schemes between production departments, allows you to manage stocks of paper, ink, and consumables, control equipment utilization, identify critical downtime points, and effectively plan production capacity.

Thanks to full integration with the Dynamics 365 financial accounting module, PrintVis provides the ability to maintain transparent financial reporting, control costs at all stages

of production, generate commercial offers based on accurate cost calculations, and prepare reports for management in real time. The system also allows you to generate detailed technical tasks for designers, operators, and printers, exchange files, monitor product readiness, and implement full-fledged document flow between departments.

The integration of an ERP system in a printing company is a process that requires not only the technical implementation of software, but also a profound restructuring of management and production algorithms.

The first algorithmic step of integration is the business modeling algorithm, when all data flows in the company, from order acceptance to completion, are digitized and displayed as interconnected modules. This allows you to create a digital "twin" of the company, on the basis of which PrintVis forms the logic of the system. Each process is assigned its own set of input and output parameters: for example, at the pre-press stage, these parameters will be the order specification, technical requirements for printing, and the cost of materials, which are then automatically connected to the calculation modules.

Next, an integration coordination algorithm is applied, which ensures the interaction of PrintVis with external enterprise systems. These include, in particular, accounting programs, CRM systems for customer management, and software for controlling printing machines. The algorithm is based on the principle of API connections and real-time data exchange, which avoids duplication of information and ensures synchronisation between different departments.

The automated planning and dispatching algorithm plays an important role in the integration of the system. PrintVis analyzes available resources – free printing machines, number of staff, paper and ink balances – and uses built-in optimization methods to determine the best production load option [7]. To do this, mathematical resource allocation algorithms and network planning methods are used to minimize downtime and increase equipment utilization.

Another key element of integration is the ordering and cost calculation algorithm. In a traditional system, a printing house spends a lot of time calculating the cost of production, while PrintVis automates this process: based on the entered data on format, print run, paper type, and printing type, the system generates a calculation in real time. This algorithm is based on built-in templates and mathematical models for calculating material, energy, and labor costs. As a result, the manager receives an accurate estimate even before the order goes into production.

It is also worth highlighting the reverse control algorithm, which is implemented through the integration of PrintVis with machines and production sensors. During printing, the system receives data on the actual use of materials, the duration of operations, and the number of defective prints. This information is automatically compared with the planned indicators, which allows for quality control and prompt process adjustments. Thus, the algorithm works in a feedback loop mode, where each deviation becomes a signal for production correction.

The forecasting and analytics algorithm, based on Business Intelligence principles, occupies a special place. Thanks to the data accumulated in the system, PrintVis generates forecasts for demand, production load, and material requirements. To do this, it uses statistical analysis methods, trend model algorithms, and machine learning tools that allow hidden patterns to be identified. This is especially important for large printing houses, where the number of orders reaches hundreds every day.

PrintVis integration also involves the use of a single information space algorithm, where all departments of the enterprise work within a single database. This means that information about customers, orders, material costs, and financial transactions is available in real time. The data synchronization algorithm ensures that changes made in any module are immediately reflected in adjacent ones. For example, if a manager changes the number of copies in an order, the system automatically updates the calculation, inventory, and production schedule.

Thus, the integration of the PrintVis ERP system into a printing company is carried out using a set of algorithms, the main ones being business modeling, integration coordination, automated planning, cost calculation, reverse control, and predictive analytics. Thanks to these algorithms, the system not only automates individual functions, but also builds a complete digital ecosystem for the printing house, which operates in real time, increasing the efficiency and competitiveness of the enterprise.

In Ukrainian enterprises, particularly in companies with complex production, PrintVis is used as part of a strategy for the digital transformation of management processes. The implementation of this system reduces response times to customer requests, avoids duplication of operations, reduces costs, and improves the quality of management decisions and the competitiveness of the printing house in the market. PrintVis can be considered the core of a modern printing company's information ecosystem, combining production flexibility, technological integration, and business analytics in a single environment. This solution is ideal for companies seeking to increase the transparency of internal processes, reduce human error, improve customer service, and achieve maximum efficiency in managing printing production.

In the current environment of digital transformation of the printing services market, the implementation of CRM systems is seen as a key tool for improving customer base management and optimizing business processes. Given the high level of competition in the printing services sector, timely and high-quality customer service is becoming a decisive factor in the success of a company [8]. For example, the Zoho CRM system is a powerful cloud-based platform for customer relationship management that allows printing companies to centrally control all aspects of sales, marketing, customer support, and analytics. With extensive personalization and integration capabilities, Zoho CRM adapts to the needs of small, medium, and large printing companies, enabling them to effectively manage their customer base, automate business processes, plan communications, and make informed decisions based on real-time data.

In the context of a printing company, Zoho CRM allows you to create a complete customer profile – with order history, contact information, payment reports, correspondence, and individual preferences regarding products or printing deadlines. Thanks to integration with email, phones, social networks, and online forms, communication with customers is centralized, minimizing the risk of losing important information. Managers can see what stage a deal is at, who last contacted the customer, when the next meeting is scheduled, and what products or services the customer is most interested in. This allows you to improve your service and create a personalized commercial offer based on an analysis of previous interactions.

Zoho CRM also supports the automation of typical processes such as generating commercial offers, invoicing, setting up mailings, assigning tasks to employees, or notifying about deadlines. This is especially useful for printing houses with a large volume of repeat orders or with the need

to respond quickly to customer requests. All these actions can be automated using the built-in scenario builder without the need to write code.

Zoho CRM's analytical capabilities are also worth mentioning: the system allows you to create detailed dashboards and reports on sales performance, lead sources, conversions, employee activity, and customer loyalty. This helps printing house management see the full picture of the market, forecast demand, and adapt customer engagement strategies. For example, you can identify which periods of the year see increased demand for advertising printing and which market segments are the most profitable. The process of information exchange between departments is illustrated in Figure 2.

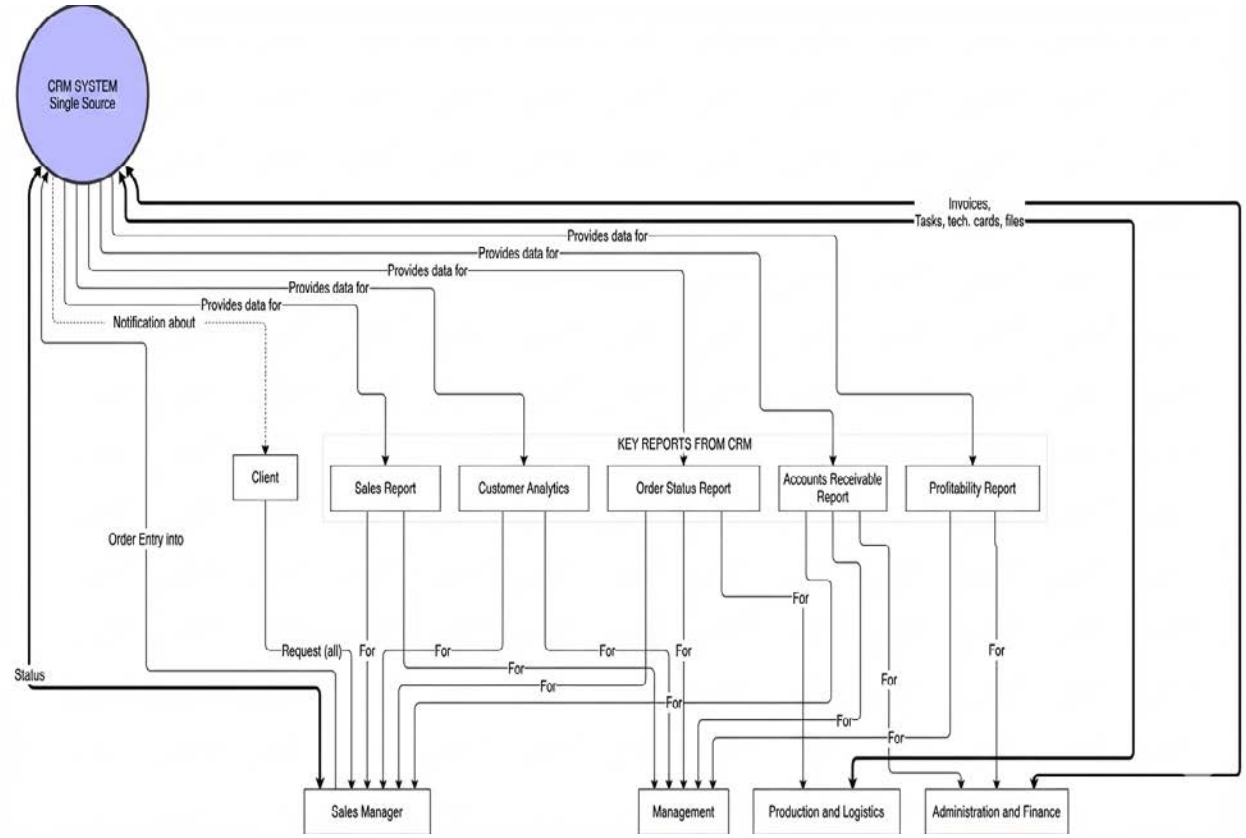


Fig. 2. Flow control diagram at a printing company after the introduction of modern information technologies

The process of integrating CRM systems into printing production is carried out in stages and requires coordination of technological, organizational, and managerial aspects. The initial stage involves analytical preparation: studying the needs of the enterprise, analyzing existing channels of communication with customers, and evaluating existing document management methods. This allows you to identify the key tasks of the system, in particular, the automation of order accounting, invoice generation, maintaining a history of cooperation, and forecasting repeat orders.

After that, the software that best suits the specifics of the printing industry is selected. The selection takes into account the possibilities of integration with ERP and accounting systems,

support for online orders, and the scalability of the solution. Next, technical specifications are formed, the sequence of connecting departments is determined, and a staff training plan is developed.

The next step is data migration, which involves transferring the customer base, order history, and commercial offers from previous systems to the new environment. At the same time, data is standardized to ensure its unified use. An important component is the integration of CRM with internal business processes: a customer's request is transformed into a production order, synchronized with accounting programs for invoicing, and production modules receive current tasks and control over deadlines.

Staff training is a critical condition for successful implementation. To this end, training sessions are organized, instructions and standards for working with the system are created, which helps employees quickly adapt to the new conditions. After completing the training, CRM is launched in test mode, where shortcomings are identified and eliminated, work processes are optimized, and preparations are made for full-scale use.

The final stage of integration is the full implementation of the system into production activities. It becomes the main tool for managing customer relationships, and its effectiveness is assessed by indicators such as order processing speed, number of repeat visits, and customer satisfaction level. Subsequently, the functioning and development of the system is continuously monitored through the integration of additional modules, such as customer online accounts, chatbots, or advanced analytical tools.

Thus, the integration of a CRM system into a printing company is a complex, multi-level process that requires a comprehensive approach and consistent coordination of organizational and technological components. Its implementation contributes to improving management efficiency, reducing the number of errors in document flow, and strengthening the company's competitive position in the printing services market [9].

In practice, Ukrainian printing companies use Zoho CRM to systematize the work of the sales department, build stable repeat sales, and improve communication between production and the customer. Thanks to its flexible interface, multilingual support, and the ability to integrate with other services (such as accounting systems, ERP, PrintVis, or Google Workspace), Zoho CRM is well suited for implementation even in complex organizational structures. Overall, Zoho CRM is an important component of the information infrastructure of a modern printing company, ensuring transparency of sales processes, increased customer satisfaction, and improved team efficiency. In synergy with production modules and ERP systems, it allows you to create a holistic enterprise management ecosystem focused on data, responsiveness, and an individual approach to each customer [10].

An ERP system and a CRM system are two different types of software that serve different purposes in enterprise management, although both can work in close conjunction. Understanding their differences is essential for building an effective information infrastructure for a printing company. The key difference is that an ERP system manages the internal product lifecycle and resources, while a CRM system manages the external customer lifecycle. ERP provides the "what", "when", and "how" to manufacture, while CRM provides the "who", "why", and "how" to sell. From a practical point of view, CRM helps to attract customers and maintain interaction with them,

while ERP ensures that obligations to them are fulfilled within the framework of production and logistics processes.

In modern printing companies, both systems are integrated with each other. This allows, for example, data about a confirmed order to be transferred from CRM directly to ERP, which calculates the cost price, organizes printing and delivery, after which the result is again recorded in CRM in the form of a cooperation history.

This interconnection between CRM and ERP ensures data continuity, minimizes duplication of information, and promotes the coordinated functioning of the entire enterprise.

The integration of modern information technologies into the activities of printing companies is now considered a key factor in increasing competitiveness. CRM provides systematization of customer data, automation of interaction with them, and loyalty building, while ERP is responsible for comprehensive management of enterprise resources, optimization of production processes, and increased transparency of accounting. Together, these systems create a single information space that combines the external and internal activities of the organization, reducing the number of errors in information loss [11].

According to the information transmission accuracy formula (A), which is used to assess how CRM/ERP reduce the amount of incorrect or lost data. The data used in the formula is based on a study conducted at a Kharkiv printing house that specializes in the manufacture of label products. A sample of orders was taken before and after the implementation of the system, and a comparative analysis was provided:

$$A = \frac{\mathbf{N}_{\text{correct}}}{N_{\text{total}}} = \frac{420}{500} = 0,84, \tag{1}$$

where N_{total} – total number of messages transmitted;

$N_{correct}$ – number of correctly transmitted records.

Based on the calculation results, it can be concluded that the data transfer accuracy level was 84 % prior to the implementation of CRM/ERP systems. After implementation, a sample of orders fulfilled by the company was analyzed, and as a result of analyzing the same number of orders, the percentage of errors in information transfer decreased to 1 %.

Practice shows that after the implementation of CRM systems [12, 13], printing companies record a noticeable increase in sales and conversion rates. According to analytical studies, the use of CRM reduces the number of information losses by 15 %. For the printing industry, this means the opportunity to form long-term partnerships, which is especially important in the label and packaging segment, where repeat orders form the bulk of revenue. Customers can track the status of their orders online and receive automatic reminders about repeat orders or seasonal discounts. This stimulates a 20–30 % increase in repeat orders, depending on the depth of integration of analytics and communication automation tools [14].

The communication efficiency coefficient (K) demonstrates how information technology affects the speed of data exchange, accuracy, and economic benefits:

$$K = \frac{T_{\text{before}} - T_{\text{after}}}{T_{\text{before}}} = \frac{40 - 10}{40} = 0,75, \tag{2}$$

where T_{before} – average order processing time before digitization;

T_{after} – average processing time after CRM/ERP implementation.

Thus, communication efficiency increased by three-quarters after the implementation of CRM/ERP systems. ERP systems, in turn, have a significant impact on the operating performance [15] of a printing company. Research in the manufacturing sector shows that after the implementation of ERP, the average production cycle time is reduced by about a third, and the accuracy of forecasting material requirements is significantly improved. For printing companies, this manifests itself in reduced equipment downtime, optimized management of paper and ink stocks, and lower costs associated with excess inventory.

After analyzing the changes in the work of a Kharkiv printing company after the implementation of an ERP system, we conclude that the cost of production has decreased by 10–15 %, and machine downtime has been reduced by 20–30 %. The company is now able to fulfill more orders in the same amount of time, which directly affects profitability.

Thanks to automated financial accounting and analytics, management receives real-time reports, sees the actual profitability of each order, controls costs, and can quickly adjust prices. This increases overall profitability by 5–10 % through more accurate planning. ERP provides transparency in planning, allows you to link order data to production lines and material resources, which contributes to more accurate calculations and more informed management decisions.

Practical cases confirm the effectiveness of this approach. In medium-sized digital printing houses, such as Aladdin Print, the implementation of CRM and ERP has reduced the average order fulfillment time by 20–30 % thanks to the automatic generation of production tasks and synchronization of material inventory data. At the same time, the number of product reworks has decreased thanks to more accurate transfer of specifications from the sales department to production.

At companies specializing in label and packaging printing, the integration of Print ERP with CRM has made it possible to automate the process of calculating the cost of print runs and reduce the time required to agree on technical tasks with customers by almost half.

The economic effect of implementing CRM and ERP is also confirmed by financial indicators. Analytical reviews indicate that the average ROI from investments in CRM can be from $6 to $9 in profit for every dollar invested, while ERP allows you to reduce operating costs through more efficient use of resources and elimination of process duplication. For the printing industry, this means a quick return on investment in digitization and the formation of a stable foundation for business scaling.

In addition to direct financial benefits, the implementation of CRM and ERP systems also affects performance indicators. Order processing times are reduced, the accuracy of technical tasks increases, and the number of conflicts between departments and errors decreases, as all participants in the process have access to a single information base.

The graph showing the reduction in the number of errors is shown in Figure 3.

To build the graph, data was taken after the implementation of CRM and ERP systems at a Kharkiv-based company that prints product labels. The number of errors that changed during the year after implementation was analyzed.

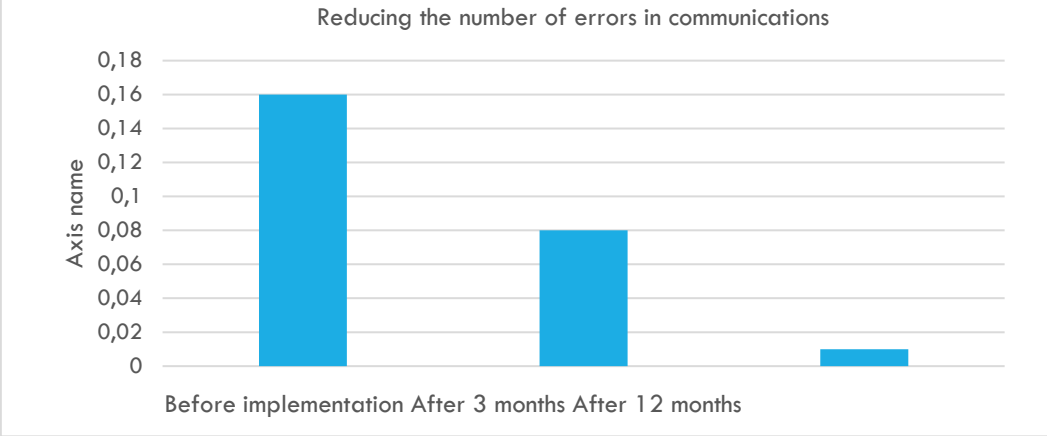


Fig. 3. Graph showing the reduction in communication errors after the implementation of CRM and ERP systems

For customers, this means faster and higher-quality order fulfillment, and for the company, it means an improved reputation and competitiveness. Combined CRM and ERP data provides in-depth analytics for strategic development.

The company can identify the most profitable services, customer segments, or areas of activity, allowing it to allocate resources more efficiently, adjust its marketing strategy, and forecast demand. This creates a flexible management system based on real data rather than intuitive decisions (Table 1).

Thus, the implementation of CRM and ERP systems in a printing company is a strategically important step that not only automates key business processes but also creates the conditions for sustainable development.

Table 1. *Changes after the implementation of CRM and ERP*

Indicator	Before implementation	After implementation	Change / Result
Order processing time	30–40 min	5–10 min	– 75 % of time
Number of errors	16 %	Reduced by 15 %	Fewer failures
Product cost	100 %	85–90 %	– 10–15 %
Equipment downtime	Frequent	Reduced by 20–30 %	More efficient
Repeat customer orders	100 %	120–130 %	+ 20–30 %
Company profitability	100 %	110–115 %	+ 10–15 %
Customer satisfaction (NPS)	100 %	125–140 %	+ 25–40 %

The effects documented in numerous studies indicate the potential for significant growth in sales and profits, increased customer retention, reduced costs, and improved internal coordination (Fig. 4).

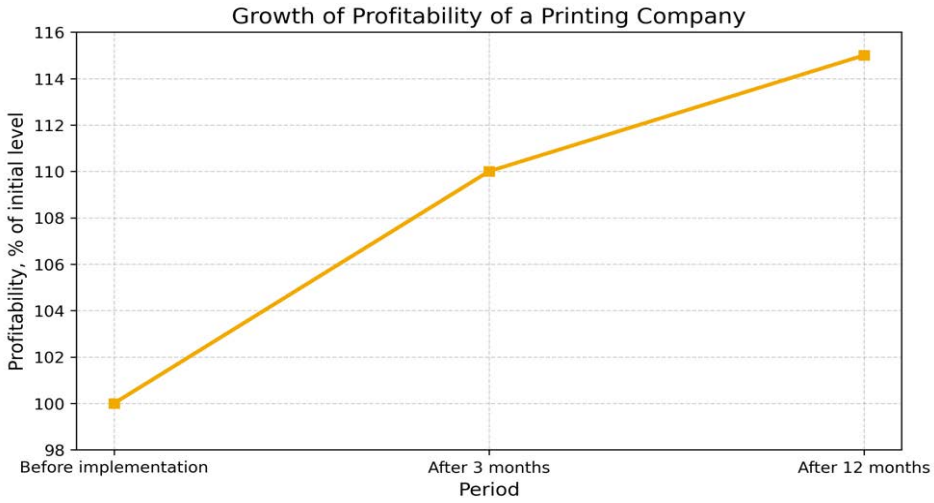


Fig. 4. Graph showing the growth in profitability of a printing company

The combination of CRM and ERP enables the integration of external customer interactions and internal production management, which is critical for the printing industry, where success depends on both high-quality service and efficient production processes. The graphs showing order processing times based on data from the printing company before (Fig. 3) and after (Fig. 4) the implementation of the system are shown in the corresponding figures.

An analysis of the Gantt charts showing the order fulfillment process at the printing company before (Fig. 5) and after (Fig. 6) the implementation of CRM and ERP systems reveals significant changes in the duration of the production and communication cycle stages. The results demonstrate a significant increase in process management efficiency, a reduction in order fulfillment times, and optimization of interaction between structural units.

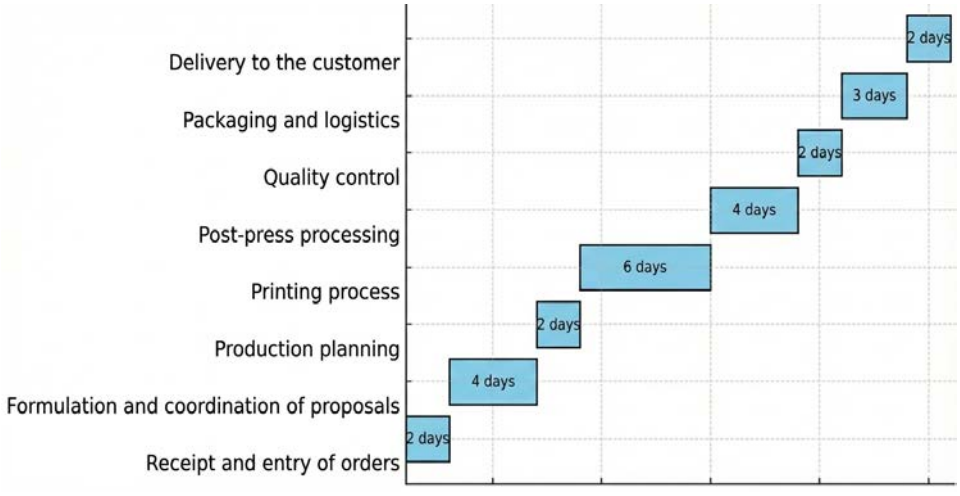


Fig. 5. Order fulfillment before CRM and ERP implementation

Before the implementation of information systems, the total duration of a single order was about 25 days. After the integration of CRM and ERP systems, this figure was reduced to approximately 16 days (by almost 40 %). This indicates a significant improvement in production organization, resource planning, and communication between departments.

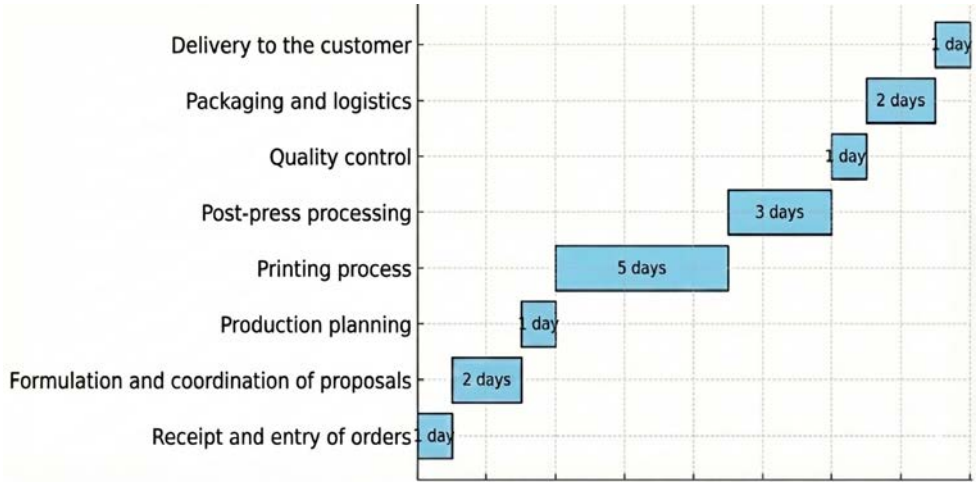


Fig. 6. Order fulfillment after CRM and ERP implementation

In particular, the greatest reduction in time is observed at the administrative stages – from receiving an order to production planning. Previously, the stages of "Receiving and entering an order", "Forming and agreeing on a proposal", and "Production planning" took a total of eight days, but after the implementation of IT solutions, their duration was reduced to four days. This was made possible by automating order processing, using a single customer database, and implementing standardized commercial proposal templates in the CRM system.

Positive changes can also be seen in the production part of the process. The printing process, which is the main stage of the production cycle, was reduced from six to five days due to improved equipment load planning and automation of task control within the ERP system. Post-printing processing has been reduced from four to three days thanks to more precise coordination of production areas. The optimization of control procedures has reduced the time required for quality control from two to one day, as the system automatically records printing parameters and provides reports to the relevant specialists in real time.

Logistics stages have also been improved. Thanks to the integration of logistics modules into the ERP system and the automation of packaging and shipping processes for finished products, the time required for these stages has been reduced by 40 %.

Delivery to the customer, which previously took two days, is now completed within one day. This indicates improved coordination with transport companies and increased transparency in the supply chain.

After the implementation of CRM and ERP systems, the time gaps between individual stages have virtually disappeared, ensuring continuity in the production process. A single information platform has brought together all departments of the company – from order acceptance to order fulfillment and delivery to the customer.

This has eliminated duplication of functions, increased the accuracy of information, reduced the number of human-related errors, and accelerated management decision-making.

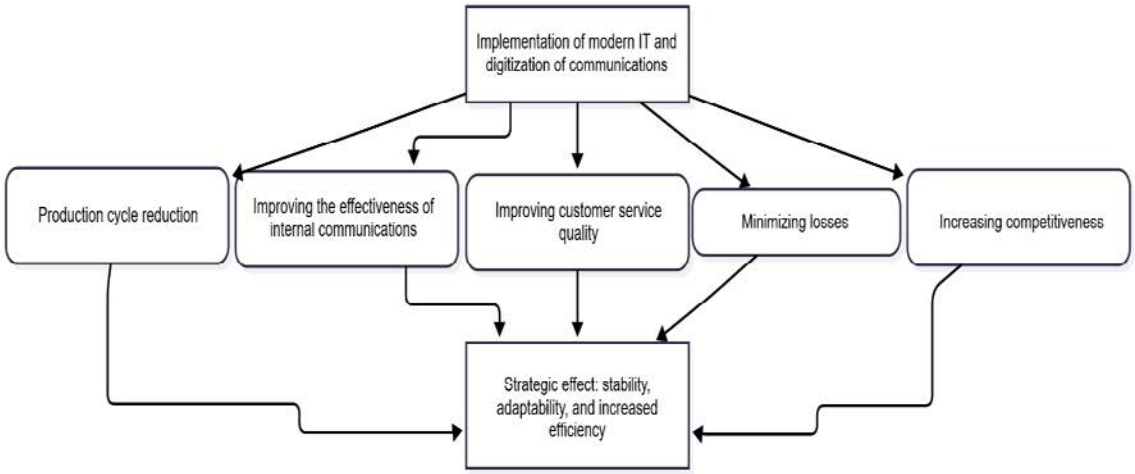


Fig. 7. Diagram of the advantages of introducing modern information technologies at a printing company

Thus, the results of the analysis confirm that the introduction of modern information technologies in a printing company has a comprehensive positive effect. It allows to significantly reduce the production cycle, increase the efficiency of internal communications, improve the quality of customer service, minimize information losses, and increase the competitiveness of the company in the printing services market.

The data obtained demonstrate that the digitization of communications management is not only a tool for optimizing operational activities, but also a strategic direction for the development of the industry, which ensures stability, adaptability, and growth in the efficiency of the printing business in the conditions of the modern digital economy.

8. Conclusions

The topic of communications management in a printing company based on the integration of modern information technologies was studied. The analysis showed that traditional communication systems in the printing industry are characterized by fragmented information flows, the lack of a unified database, delays in interaction between departments, and a high probability of losing important data during the order approval and production planning stages.

The implementation of CRM and ERP systems allows the creation of a single integrated information platform that provides rapid data exchange between departments, automation of quality control processes, planning, and order tracking.

The results of the study show that thanks to the digitalization of communication processes, the company achieves an average reduction in order processing time of 25–30 %, a reduction in the number of errors in documentation of up to 40%, and an increase in customer and staff satisfaction thanks to the transparency of processes and clearly defined responsibilities.

The structure of the communication management model helps minimize the risk of information loss and speeds up management decision-making. The CRM module focuses on interaction with external customers, providing a full cycle from receiving a request to after-sales support, while the ERP system coordinates internal business processes and optimizes resource planning and logistics.

Practical results prove that the implementation of such information solutions increases the competitiveness of a printing company by improving organizational efficiency, process transparency, reducing order fulfillment times, and improving the quality of communication with customers.

Thus, the integration of modern IT solutions into the communication management system is a strategic direction for the development of the printing industry, ensuring sustainable business development and its adaptability to the requirements of the digital economy.

Prospects for further research lie in expanding scientific and practical approaches to the digital transformation of communications in printing companies, taking into account technological and market changes. It is advisable to conduct an in-depth study of the impact of integrated CRM and ERP systems on the level of personalization of customer interaction, demand forecasting, and customer loyalty formation.

A promising direction is the modeling of multi-level communication structures using analytical platforms that provide automated control of production indicators and early detection of deviations in processes. Additional attention should be paid to assessing the economic efficiency of communication digitalization by comparing the costs of implementing IT solutions with the expected benefits in the form of reduced operating costs, improved customer service quality, and increased labor productivity.

Particular attention should be paid to the integration of artificial intelligence systems for automated processing of customer requests, optimization of production resource allocation, forecasting of order completion times, and formation of adaptive production plans. Of considerable scientific interest is the study of the capabilities of digital twins for simulating communication and production processes in order to analyze bottlenecks and improve the reliability of information exchange.

An important area of research is the impact of cybersecurity procedures on the stability and continuity of information flows, especially in the context of remote interaction and distributed access to corporate data.

The results obtained can form the basis for the creation of scalable, next-generation intelligent communication platforms aimed at fully automating data exchange and ensuring high flexibility of production business processes in the printing industry.

References

1. Di, H., Dong, S. (2025), "Data sharing mechanism for logistics supply chain based on blockchain shard distribution", *Cluster Computing*, No. 28 (15). DOI: https://doi.org/10.1007/s10586-025-05611-7.
2. Іванілов, О. С. (2019), "Менеджмент організацій", Київ: Центр навчальної літератури, 472 с.
3. Головко, В. В., Назаренко, М. І. (2020), "Інформаційні системи і технології в управлінні підприємством", Харків: ХНЕУ ім. С. Кузнеця, 312 с.
4. Jędrzejczyk, M., Bielawski, P., Kozioł, W. (2025), "Microeconomic wage productivity and a management variable in controlling enterprise efficiency", *Zeszyty Teoretyczne Rachunkowosci*, No. 49 (3), P. 85–100. DOI: https://doi.org/10.5604/01.3001.0055.2470.
5. Копієвський, О. М., Пономаренко, В. С. (2021), "Сучасні тенденції цифровізації виробничих процесівполіграфічних підприємств", *Вісник Харківського національного економічного університету ім. С. Кузнеця*, No. 4(108), С. 52–59.
6. Guerola-Navarro, V., et al. (2021), "Customer relationship management (CRM) and Innovation", *Technological Forecasting and Social Change*, No. 169. DOI: https://doi.org/10.1016/j.techfore.2021.120838.
7. Bhardwaj, B. R. (2014), "Sustainable supply chain management through enterprise resource planning (ERP): A model of sustainable computing", *2014 International Conference on Computing for Sustainable Global Development (INDIACom 2014),* P. 166–171. DOI: https://doi.org/10.1109/IndiaCom.2014.6828122.
8. Curtis, B., Kellner, M. I., Over J. (1992), "Process Modeling", *Communications of the ACM*, No. 35 (9), P. 75–90.
9. Балабанов, І. Т. (2017), "Інформаційні технології в бізнесі: управління, аналіз, планування", Одеса: Атлант, 346 с.
10. Chumpoonta, P. (2024), "Overcoming Business Disruption for MICE Business (Powered by Zoho CRM)", *Procedia Computer Science*, No. 237, P. 163–170. DOI: https://doi.org/10.1016/j.procs.2024.05.092.
11. Ілляшенко, С. М. (2022), "Цифрова трансформація бізнес-процесів підприємств: теорія, методологія, практика", Суми: СумДУ, 278 с.
12. Darmaningrat, E. W. T. (2019), "Communication Management Plan of ERP Implementation", *Procedia Computer Science*, No. 161, P. 359–366. DOI: https://doi.org/DOI: 10.1016/j.procs.2019.11.134.
13. Шевченко, О. Ю. (2020), "Впровадження ERP-систем як чинник підвищення ефективності підприємства" *Економіка та держава*, No. 6, С. 87–91.
14. Назаренко, І. В. (2021), "Комунікаційні технології в управлінні підприємством", Київ: Ліра-К, 240 с.
15. Al-Assaf, K., Alzahmi, W., Alshaikh, R., Bahroun, Z., Ahmed, V. (2024), "The Relative Importance of Key Factors for Integrating Enterprise Resource Planning (ERP) Systems and Performance Management Practices in the UAE Healthcare Sector", *Big Data and Cognitive Computing*, No. 8 (9), P. 122.

Received (Надійшла) 10.1.2025
Accepted for publication (Прийнята до друку) 30.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Slutskin Mykyta – Kharkiv National University of Radio Electronics, Postgraduate Student at the Department of Media Systems and Technologies, Kharkiv, Ukraine, e-mail: mykyta.slutskin@nure.ua; ORCID ID: https://orcid.org/0009-0007-9995-9375

Vovk Oleksandr – PhD (Engineering Sciences), Kharkiv National University of Radio Electronics, Associate Professor at the Department of Media Systems and Technologies, Kharkiv, Ukraine, e-mail: oleksandr.vovk@nure.ua; ORCID ID: https://orcid.org/0000-0001-9072-1634

Слуцкін Микита Володимирович – Харківський національний університет радіоелектроніки, аспірант кафедри медіасистем та технологій, Харків, Україна.

Вовк Олександр Володимирович – кандидат технічних наук, Харківський національний університет радіоелектроніки, доцент кафедри медіасистем та технологій, Харків, Україна.

АНАЛІЗ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ УПРАВЛІННЯ КОМУНІКАЦІЯМИ ПОЛІГРАФІЧНОГО ПІДПРИЄМСТВА

Предмет дослідження охоплює управління комунікаціями поліграфічного підприємства в умовах цифрової трансформації виробництва й зростання конкурентного тиску на ринку поліграфічних послуг. **Мета роботи** – дослідження управління комунікаціями на основі інтеграції сучасних інформаційних технологій, розроблення відповідної функціональної та інформаційної структури, яка дає змогу мінімізувати ризики втрат інформації, прискорити прийняття рішень, забезпечити чітке розмежування відповідальності та підвищити рівень задоволеності як внутрішніх користувачів, так і зовнішніх клієнтів. **Завдання**: визначити особливості внутрішніх і зовнішніх комунікацій поліграфічного підприємства; проаналізувати структуру інформаційних потоків; оцінити вплив IT-інструментів на швидкість і точність передачі даних; обґрунтувати доцільність упровадження CRM- та ERP-систем для оптимізації бізнес-процесів. **Методи дослідження**: системний і процесний аналіз, структурно-функціональне моделювання комунікацій, порівняльний аналіз цифрових інструментів, а також аналітична інтерпретація емпіричних даних щодо ефективності інформаційної взаємодії. **Досягнуті результати.** Установлено, що традиційна модель комунікацій у поліграфічному виробництві визначається фрагментованістю інформаційних потоків, затримками в передачі технічних даних і високою ймовірністю помилок під час погодження замовлень. Інтеграція CRM- та ERP-систем, електронного документообігу й хмарних корпоративних сервісів забезпечує скорочення часу виконання замовлень, зменшення кількості помилок у технічній документації, підвищення прозорості управлінських рішень і рівня задоволеності клієнтів і персоналу. **Висновки.** Цифровізація комунікацій формує сучасну модель управління поліграфічним підприємством, що поєднує автоматизовану взаємодію підрозділів, безперервний контроль і моніторинг виробничих процесів. Упровадження CRM- і ERP-рішень, аналітичних платформ і систем електронної координації є стратегічним напрямом розвитку поліграфічної галузі, оскільки сприяє підвищенню конкурентоспроможності й адаптивності бізнесу до вимог цифрової економіки.

Ключові слова: поліграфія; комунікація; інформація; управління; технологія; ERP; CRM; цифровізація; інтеграція; ефективність.

Bibliographic descriptions / Бібліографічні описи

Slutskin, M., Vovk, O. (2025), "Analysis of information technologies for communication management in a printing company", *Management Information Systems and Devises*, No. 4 (187), P. 123–141. DOI: https://doi.org/10.30837/0135-1710.2025.187.123

Слуцкін М. В., Вовк О. В. Аналіз інформаційних технологій управління комунікаціями поліграфічного підприємства. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 123–141. DOI: https://doi.org/10.30837/0135-1710.2025.187.123

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.8:004.9 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.142

V. Filatov, M. Chernenko

USER TASK SUPPORT IN INFORMATION SYSTEMS BASED ON AGENT TECHNOLOGIES

The **subject matter** of this paper concerns the application of multi-agent systems and software agent technologies for managing and analyzing information resources and t ask flows within distributed, heterogeneous information environments. It focuses on formal models and execution frameworks that enable intelligent control of complex agent-oriented workflows. The **goal** is to develop effective methods and models for representing, coordinating, and executing agent-oriented tasks, ensuring correctness and efficient resource use in distributed systems. The **tasks** addressed include formalizing the information space as an algebraic system comprising objects, relationships, and operations; defining elementary and functional tasks executed by users or agents; organizing interdependent tasks into flows; and modeling agent behavior via frame structures and hierarchical Petri nets incorporating metapositions and metatransitions. The **methods** employed combine algebraic system modeling, automata theory for agent behavior analysis, and extended predicate Petri nets with modifications supporting hierarchical task management. Coordination constraints and workflow correctness properties are specified using Computation Tree Logic (CTL) and dependency automata to capture inter-task dependencies and atomicity requirements. The work also applies SQL-based operations to exemplify merging of data during task execution. The **results** include a formal framework distinguishing state-based and task-based planning, a modified frame slot model enabling complex task descriptions with initial conditions and operations, and a hierarchical Petri net scheme with subordinate nets activated conditionally, thereby optimizing computations. A software agent prototype executing a sequence of conditional operations on multiple databases demonstrates the practical application of these concepts. The approach reduces execution time and computational costs by activating only necessary subordinate tasks and avoiding irrational solutions. The **conclusions** assert that the proposed agent-oriented modeling and hierarchical task execution methods successfully address the complexities of distributed information resource management and workflow control, providing a scalable means for executing fault-tolerant, atomic, and coordinated workflows. This enhances the reliability and efficiency of multi-agent systems in heterogeneous computing environments.

Keywords: multi-agent systems; hierarchical task management; Petri nets; information resource management.

Introduction

This paper considers a class of multi-agent systems for managing information resources in distributed systems. A typical representative of this class of problems is the task of planning a path to achieve a desired goal from a given initial situation. The result of solving such a problem should be an action plan – a partially ordered set of actions. Such a plan can be represented as a scenario in which the relationships between vertices are of the types "goal-subgoal", "goal-action", "action-result", and so on. Any path in this scenario that leads from a vertex corresponding to the current situation to any of the goal vertices defines an action plan. The description of situations includes both the state of the external environment and the state of the information system. Situations form certain generalized states, and actions or changes in the external environment lead to changes in the currently actualized states.

Among the generalized states, the initial states (usually one) and the final (goal) states are distinguished. All action-planning tasks can be divided into two types, corresponding to different

models: state-space planning and task-space planning. The representation of problems in a state space involves specifying a set of descriptions: states, a set of operators and their effects on state transitions, and the goal states. State descriptions can take the form of strings of symbols, vectors, two-dimensional arrays, trees, lists, and so on. Operators transform one state into another. The state space can be represented as a graph, where vertices are labeled with states and arcs with operators. The state-space planning method can be regarded as a special case of planning by reduction, since each application of an operator in the state space reduces the initial problem to two simpler ones, one of which is elementary. Problem-space planning consists of the successive reduction of the original problem to increasingly simpler ones until only elementary problems remain. A partially ordered set of such problems constitutes the solution to the original problem.

Literature review and problem statement definition

Software agents emerged as one of the most significant achievements in computer science in the 1990s. The problem of intelligent agents and multi-agent systems (MAS), which has a long history, was shaped within the scope of research on distributed artificial intelligence and, in recent years, has claimed one of the leading roles in intelligent information technologies [1]. Agents are used in applied systems where humans and computers are closely interconnected in managing information processes [2]. The abundance of diverse information on software agents leads to inconsistencies in defining what constitutes a software agent. Therefore, it is first necessary to clarify the essence of the concept of a "software agent" and to identify the key properties of software agents. Below we consider the main properties and specifications of agent-oriented tasks.

Definition of an Agent-Oriented Task. An agent-oriented task in a workflow is a unit of work that can be executed by a processing entity, such as an application-level computing system or, for example, a database management system (DBMS) [3]. A task may be defined independently of the processing entity capable of executing it, or based on the capabilities and behavior of that entity. The structure of a task can be specified by defining: a set of task execution states, a set of transitions between these states, and the conditions that make these transitions permissible (transition conditions can be used to specify inter-task execution requirements). The execution order of each task is determined by specifying the set of states observable from outside and the set of transitions between these states. Furthermore, certain characteristics of processing entities can be defined that may influence the requirements for task execution [4].

Coordination Requirements for Agent-Oriented Tasks. Coordination requirements are typically expressed in terms of inter-task execution dependencies and dataflow dependencies.

Execution or Correctness Requirements. Execution requirements are defined to constrain the workflow execution in such a way that correctness criteria, dependent on the application, are met. They include requirements for failure atomicity, execution atomicity (including visibility rules that determine when the results of a committed task become visible

to other concurrently running workflows), as well as requirements for synchronous execution control and recovery.

Failure Atomicity Requirements in Workflows. The traditional notion of failure atomicity is that the failure of any task causes the entire workflow to fail. However, in many cases, a workflow may tolerate the failure of a single task by, for example, executing a functionally equivalent task on another node. A system executing a workflow (a workflow scheduler) must ensure that any execution completes in a state that satisfies the failure atomicity requirements specified by the developer. We refer to such states as acceptable workflow termination states. All other execution states belong to the set of unacceptable termination states, in which failure atomicity may be violated.

An acceptable termination state can be either committed or aborted. A committed acceptable termination state is an execution state in which the goals of the workflow have been achieved. Conversely, an aborted acceptable termination state is a correct termination state in which the workflow has failed to achieve its goals. If an aborted acceptable termination state is reached, all undesirable effects of partial workflow execution must be eliminated in accordance with the failure atomicity requirements [5–7].

Tasks are modeled by their states together with significant events corresponding to transitions between states – start, commit, rollback, etc., – which may be enforced, rejected, or delayed. Inter-task dependencies, such as ordering dependencies and existence dependencies between significant task events, are formally defined using Computation Tree Logic (CTL) and have corresponding dependency automata that can be automatically constructed. To address the aforementioned characteristics in user tasks, it is advisable, at the first stage, to consider issues related to the information space and, specifically, to the formal representation of such a space.

Research objective. The objective of this work is to develop effective models and methods for managing and analyzing task flows within a unified heterogeneous information space, based on software agent technology.

Research materials and methods

Development of a model of the distributed system's information space

The inf ormation space of a distributed information system can be represented as an abstract algebraic system

$$P =< O,\ S,\ \Omega >, \tag{1}$$

where O – the objects of the information space; S – the relationships between objects O; Ω – the set of operations for manipulating objects within the space P.

The objects in model (1) can include components of a modern computing system – files of all types, directories, logical and physical disks, personal computers, and so forth. The relationship $s_i \in S$ between the objects of the information space defines a specific configuration of the computing environment $p_i =< o_i, s_i, q_i >$, where $p_i \in P$, $o_i \in O$, $q_i \in \Omega$, oriented towards a specific user or group of users $u_i \in U$, $i = 1, N$, where N is the number of users.

Depending on the goals set by the user and the computing environment, all actions are performed based on operations from Ω.

All operations on objects, according to the defined goals, are initiated by a user $u_i \in U$ of the information space, following either a formalized action plan $x_{i1}^*(t), x_{i2}^*(t), ...x_{im}^*(t)$ or a unitary action $x_i^*(t)$.

In the general case, the i-th action plan $x_i(t)$, if executed as a one-time interaction between the user and the information environment (such as viewing the contents of a file, moving a file within the computing environment, etc.), can be interpreted as an elementary task that a user $u_i \in U$ solves with the help of the information system $P(O, S, \Omega)$.

The result of solving an elementary task $x_i(t)$ can be defined as a mapping $z_i : x_i \Rightarrow y_i$, the execution of which requires an operation q_i, $i \in I$, where $y_i \in Y$ is the solution to the task $x_i(t)$. An elementary task, within the framework of the proposed approach, can be represented as $z_i = (o_i, s_i, q_i, x_i(t), y_i)$.

Elementary tasks can be combined into functional tasks by merging elementary operations into a sequence of interrelated operations forming an algorithm $A_k = \{q_1, q_i, ...q_m\}$, $i = \overline{1, m}$, $A_k \in A$, $k \in K$ where A is the set of solutions to functional tasks and K is the number of functional tasks.

Depending on the structure and interdependencies of the tasks solved by the user, they can be organized into task flows. A task flow $A_k \in A$ is defined as a sequence of tasks for which the following condition is true: $y_{ij}(t) = p_{ij}(x_{ij}(t))$, $i \in I$, $j \in K$, considering the constraint $x_{ij}(t) = y_{ij}(t)$, $i \neq j$, meaning that the result of solving the preceding task serves as the input for executing the subsequent task. The interrelationship scheme of subtasks in a flow is shown in Fig. 1.

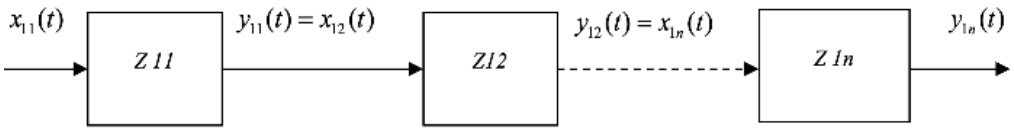


Fig. 1. The interrelationship scheme of subtasks in a flow

To implement the interaction technology of a software agent with objects of a heterogeneous computational environment, a frame structure can be used as a model of the software agent.

A slot in such a model is a logical construct designed to perform specific tasks assigned to the frame – that is, to the software agent. Slots in the frame may be removed, added, or reassigned to different functional roles.

Formal Model of a Software Agent

Representing the logical model of a software agent in the form of a frame makes it possible to effectively apply automata theory for constructing and analyzing the model of agent behavior.

In general, such a model can be expressed as follows:

$$FR\left\{\left\langle R_1,\ C_{11},\ C_{12},\ \ldots\ C_{1m}\right\rangle;\ \left\langle R_2,\ C_{21},\ C_{22},\ \ldots\ C_{2m}\right\rangle,\ \left\langle R_{k1,}C_{km}\right\rangle\right\}, \quad (2)$$

where FR – the name of the frame; $\left\langle R_i, C_i\right\rangle$ – the i-th slot of the frame; R_i – the slot name; C_i – the slot value.

A slot in model (2) is a logical construct for performing specific tasks of the frame (software agent). Slots can be deleted, added, or reassigned. However, in the form (2), the classical slot representation as $\langle name\rangle$, $\langle value\rangle$ cannot fully reflect the requirements of a logical model of a software agent [8, 9].

We modify the structure of the slot and present it in the following form:

$$SLOT = Y\langle,\ D,\ dom,\ r_i,\ Q,\ W\rangle, \quad (3)$$

where Y – a set of attribute names; D – a set of domains; dom – a $Y \to D$ mapping; r_i – a tuple-model of the agent's i-th task; W – a set of operations over relations; $r_i = \{R_i\}$, where $\{R_i\}$ – a set of states of the tuple-model r_i; Q – a set defining the initial conditions and criteria for task execution.

A slot may be designed using a typical set of attributes $Y = \left\{\langle OBJ\rangle,\ \langle ACT\rangle,\ \langle CON\rangle,\ \langle STA\rangle\right\}$. Table 1 lists possible basic attribute types.

Table 1. *Basic attribute types*

Entity	**Name**	**Description**
OBJECT	OBG	Database, file, folder, disk, PC
ACTION	ACT	Copy, monitor, protect
CONDITION	CON	IF–THEN predicate
STATUS	STA	1 – action executed successfully; 0 – action not executed; * – indeterminate result

In the proposed "frame-software agent" model, n slots form a finite number of internal agent states. Each slot solves one task in the information environment and is associated with exactly one action. Each action elicits a reaction signal from the environment.

The behavior of the software agent, consistent with its functional goals, can be implemented, for example, using the theory of finite-state machines.

Example. Task for a software agent. Every day at 18:00, copy the file Itog.txt from directory C:\PR to directory C:\ARXIV. The structure of the software agent is shown in Table 2.

Table 2. *Agent's structure*

OBG	CON	ACT	STA
C:\PR\Itog.txt	Time = 18:00	Copy from C:\PR to C:\ARXIV	1

Network Model for Representing and Analyzing Interrelated Tasks

As a visual means of representing a task flow, a network model can be proposed. It provides a graphical representation of elementary tasks and their interrelations, whose execution is required to complete the entire task flow. In this model, the network consists of directed arcs connecting pairs of nodes. Arcs (edges) represent elementary tasks characterized by computational resource costs. Nodes (depicted as circles) represent events, i.e., strictly defined time instants.

For example, an event may represent the moment when all database operations (insert, update, edit) are completed, enabling a file-copy operation into an archive directory.

Formally, such a task flow can be represented as:

$$A_1 = \{q_1, q_2, q_3, q_4\},$$

where A_1 – a task flow name; q_1 – a database merge operation; q_2 – an update operation on a file attribute; q_3 – a logical operation identifying user exit from edit mode; q_4 – a database file copy operation to the archive. Fig. 2 illustrates a fragment of such a network model.

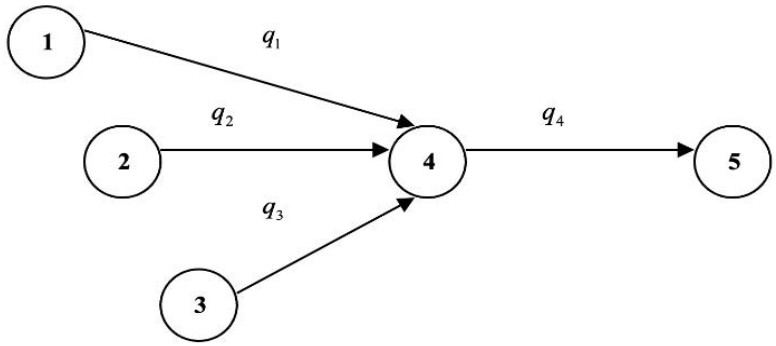


Fig. 2. A network model representing a task flow

Representing the workflow as a node–task model has significant advantages compared to the node–event method. Here, nodes represent tasks, and arcs only indicate precedence relations. There is no need to introduce the concept of an event, which simplifies network construction. Each task is uniquely associated with a node, and task execution is characterized by its duration. This approach provides access to important characteristics:

- earliest start and finish times of tasks;
- latest start and finish times of tasks;
- total time reserve.

A particular interest lies in managing and analyzing information resources with a stochastic object structure. This class of problems is analyzed using GERT systems (Graphical Evaluation and Review Technique).

A stochastic network is defined as one where the execution time of each operation follows a probability distribution [10]. Execution of a task at a node is not necessarily required for all incoming arcs. Nodes represent system states. Arcs represent transitions between states, corresponding to generalized operations defined by execution probabilities and distribution densities. Each node in a stochastic network thus performs both input and output functions.

Example. In a regional emergency management system, decision-support tasks are implemented through autonomous scenario analysis of possible emergency cases (Fig. 3).

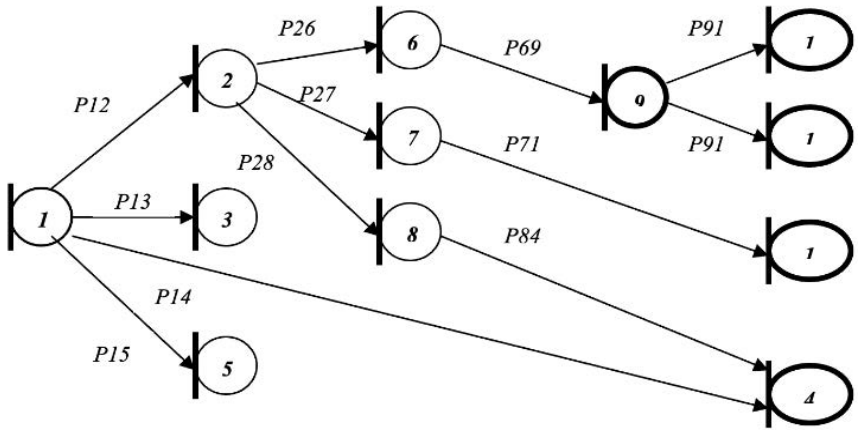


Fig. 3. A stochastic network example

Node 1: The duty officer receives probabilistic emergency alerts (fires, floods, industrial accidents, or false alarms → nodes 2, 3, 5, 4 correspondingly). Probabilities satisfy $p12 + p13 + p14 + p15 = 1$.

Assume, the situation is classified as a "Fire" by preliminary data, it is leading to scenarios: high danger (node 6), medium danger (node 7), or manageable by standard crew (node 8). Here $p26 + p27 + p28 = 1$.

The next nodes layer for scenario is build on top of deterministic nodes (bold in the diagram). They represent information support tasks, e.g., loading of databases (node 9), SQL queries for backup forces and evacuation sites (nodes 10, 11), and assessment of available local resources (node 12). By defining network parameters, one can estimate the readiness of the emergency system to respond to evolving crises [11].

High-Level Network Representation of Agent-Oriented Tasks

The design and operation of automated decision-support systems under strict resource constraints face difficulties in guaranteeing required functional and operational properties. Existing diagnostic tools mainly provide state monitoring and partial task execution control. Performance depends largely on application quality, complicated further by distributed data processing and human factors during system design.

An alternative formalism for systemic situational analysis is Petri nets [13, 14]. Petri nets allow modeling of parallel and distributed algorithms, including conflict and undesired scenarios. Formally, an algorithm graph scheme is a directed graph: $G = \langle V, E \rangle$, where V – the set of vertices and E – the set of arcs.

Vertices are of three types:

Operator vertices describe actions (including mandatory start and end).

Choice vertices describe conditional branching ("yes"/"no"), has one incoming and two labeled outcoming arcs.

Auxiliary vertices contain 3 groups:

− parallel branching (one input → many outputs) describe parallel execution of independent actions;

− parallel merging (many inputs → one output, requiring all);

− conditional merging (many inputs → one output, requiring any one).

An important advantage of graph-schemes is that the algorithmic graph-scheme can be formally transformed into a Petri net of a special type – namely, a free-choice Petri net. In this case, the justification of the correctness of the algorithmic graph-scheme reduces to verifying the correctness and safety of the corresponding Petri net. In free-choice Petri nets, each position (condition) that enables more than one transition is the sole enabling position for those transitions. The structural properties of free-choice Petri nets made it possible to develop efficient algorithms for their semantic analysis directly based on the network structure.

However, modeling control algorithms by means of algorithmic graph-schemes (free-choice Petri nets) has a number of significant drawbacks. In a graph-scheme, the algorithmic structure is predetermined and does not allow, in particular, dynamic parallelization of a task. Furthermore, the graph-scheme of an algorithm does not possess a modular structure, which makes the modeling and analysis of large-scale tasks rather difficult. To overcome this limitation, one may employ an extension of the standard Petri net formalism – nested Petri nets.

Nested Petri nets in modeling and analysis of complex interrelated processes

In nested Petri nets, tokens marking positions are regarded as objects possessing autonomous behavior, which, in turn, is also described by certain Petri nets. The term "nested nets" indicates that elements of the nets themselves are nets, similar to how, in a system of nested sets, the elements of a given set may themselves be sets. Nested Petri nets are convenient and powerful tool for modeling and analyzing hierarchical multi-agent distributed systems. They possess an inherent mechanism of modularity. Modular Petri nets can be considered as a special case of nested Petri nets.

Compared with other extensions of the Petri net formalism, owing to the concept of object and it’s construction, nested Petri nets preserve such important properties of standard Petri nets as simplicity, clarity of representation, and decidability of certain properties crucial for verification [14].

A nested Petri net consists of a system (root) net and a set of element nets representing the tokens of the system net.

In the simplest case of a two-level nested net, the element nets are ordinary Petri nets, in which tokens, as usual, are depicted as black dots, have no internal structure, and are indistinguishable from one another.

The behavior of a nested Petri net involves four types of steps:

Transfer step: firing of a transition in the system net according to the standard rules for high-level Petri nets, where element nets are treated as tokens without internal structure. A transfer step may move, create, or remove objects, but cannot alter their internal state.

Element-autonomous step: alters only the internal state (marking) of an element net without changing its location in the system net. This step is also executed according to the standard transition-firing rules of a Petri net.

Horizontal synchronization step: simultaneous firing of two transitions in two element nets located within the same position of the system net. Transitions that must fire synchronously are marked with mutually complementary labels from a special set of labels for horizontal synchronization.

Finally, *Vertical synchronization step*: used to synchronize a transition in the system net with certain transitions of the element nets. Transitions that must fire synchronously are labeled with marks from a special set of vertical synchronization labels. In this case, the label of a transition in the system net and the label of the corresponding transition in the element net must be mutually complementary.

The element nets that are moved from the preconditions of the transition in the system net during its firing are called the engaged element nets. Vertical synchronization thus means simultaneous firing of the transition in the system net and the transitions (marked with complementary labels) in the engaged element nets.

When solving problems of controlling distributed multi-agent systems, the complexity of control algorithms increases substantially, which in turn raises the importance of modeling various scenarios of system behavior under different management strategies. This creates the need for tools capable of building illustrative models of such systems' behavior and supporting automatic verification of their semantic (behavioral) properties.

Nested Petri nets possess the following properties that make them a convenient tool for modeling and analyzing control algorithms in multi-agent systems:

− they feature a hierarchical and modular structure, which allows the model to clearly reflect the hierarchical and modular structure of the algorithm;

− element nets in a nested Petri net have their own structure and behavior, making them well-suited for modeling agents in a multi-agent system;

− horizontal synchronization of element nets corresponds to agent-to-agent interaction, while vertical synchronization models agent actions that modify the state of the environment external to those agents.

From the foregoing, it follows that nested Petri nets offer sufficiently rich expressive capabilities.

At the same time, being an extension of standard Petri nets, they preserve their advantages of simplicity and clarity of representation. Moreover, for nested Petri nets, certain properties essential for verification remain decidable [15].

Hierarchy of Networks of Interconnected Subtasks with Variable Structure

Let us consider the specific aspects of constructing networks and managing the dynamics of modeling data processing processes through the use of a combination of modified predicate nets and modified E-nets. Taking into account the known difficulties in interpreting and modeling data processing processes using predicate nets, it is deemed expedient to apply the following modification of nets:

$$S_{PR} = \left\langle P,\ T,\ F,\ A_t,\ C,\ \{V_s\},\ K,\ M_0 \right\rangle, \tag{4}$$

where P – a set of positions; T – a set of transitions; F – an incidence function between positions and transitions; C – a color function of tokens; A_t – a time parameter assigned to all network components $P,\ T,\ F, M_0$; V_s – an enabling condition of transitions $t_k \in T$, $k \in K$ related to the network components: $V_{p_k{}^I}$ – an enabling condition with respect to its input positions; $V_{p_k{}^O}$ – a condition of enabling $t_k \in T$, $k \in K$ with respect to its output positions; V_{t_k} – an enabling condition $t_k \in T$, $k \in K$ directly related to the transition; V_{Mp_l} – an enabling condition $t_k \in T$, $k \in K$ related to the marking of the incident positions, $l \in L$; V_F – an enabling condition $t_k \in T$, $k \in K$ associated with the arcs between input/output positions; K – the token capacity of positions considering C; M_0 – the initial marking vector.

The enabling condition of transition V_S also includes such properties as reliability, complexity, cost, and the degree of credibility of a specific event.

The analysis of possible solutions has demonstrated that, for the net (4), in the tasks of modeling subprocesses based on S' networks, it is advisable to apply interpreted metapositions. This is due to the fact that the positions themselves interpret the enabling conditions of the actions of simulated events and, at the same time, define the state space of the model [16–18].

Example. A resource of type A_1 is provided by information support and stored in database files, e.g., $B_{11}, B_{12}, \ \ldots\ B_{1n}$, $n \in N$, where N is the number of databases. The search condition for the considered fragment of a decision-support system is formulated as follows: to determine the specified quantity of resource type $A_1 = A_1^*$ over the set $\{B_{11}, B_{12}, \ \ldots\ B_{1n}\}$, $n \in N$. The search of a solution is performed through two types of actions: at the first step, a SEARCH over database B_1 is executed; if the condition is not satisfied, a UNION of databases B_1 and B_2 is carried out and the search is repeated.

The procedure of searching and merging databases can be carried out in terms of the Structured Query Language (SQL), using the SELECT, JOIN, and INSERT options:

```
SELECT *
FROM b1
JOIN b2 ON b1.id = b2.id;
```

Let us represent a fragment of the above problem in the form of a *modified predicate Petri net S* (Fig. 4).

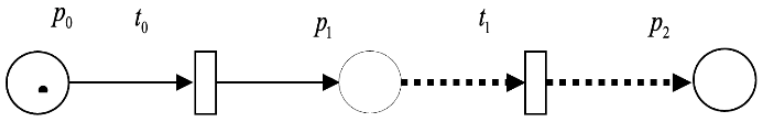


Fig. 4. *S* Petri net

The set of positions $\{p_0, p_1\}$ interpret, respectively, the input and output conditions of executing the action represented by transition t_0. The set of positions $\{p_1, p_2\}$ interpret, respectively, the input and output conditions of executing action t_1.

The initial marking vector $M_0 = (1,\ 0,\ 0)$ defines the initial state space of the net.

Description of the main constraints shown in Fig. 4. $V_{p_k{}^I}$ is the input condition of transition $t_1 \in T$, $k \in K$; $V_{p_k{}^O}$ is the output condition of transition t_1. If the input condition $V_{p_k{}^I} \neq V_{t_k}$ of transition t_1 is not satisfied, then for the given transition $t_1 \in T$, $k \in K$, a subordinate net (S') is generated (Fig. 5), where position p_1 serves as a metaposition.

The initial marking vector $M_0 = (1,\ 0,\ 0,\ 0)$ defines the initial state space of this subordinate net. The marking of position p_{13} and the execution of transition t_{13} lead to the marking of the metaposition p_1, which activates the execution of S net.

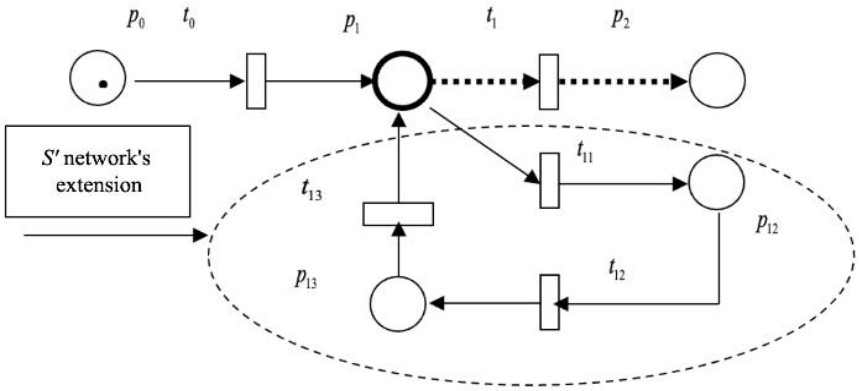


Fig. 5. S Petri net with an S' extension

If the desired solution cannot be obtained by the means of $S \cup S'$, a hierarchy of tasks is constructed based on searching for solutions from the set $\{B_{11}, B_{12}, \ldots B_{1n}\}$, $n \in N$ through merging databases and extending the S' net.

Let us present the above problem as a conceptual task for a software agent:

Table 3. *Structure of the software agent "AGENT"*

ID	OBG:	CON:	ACT:	STA:
S1	C\BASE\b1.mdb	Time=10-00	Select sum([A1]) from b1.dbf	IF A1=A1* is TRUE END ELSE S2
S2	C\BASE\b1, b2.mdb	***	Join b1, b2 into b1; Select sum([A1]) from b1.dbf	IF A1=A1* is TRUE END ELSE S3
S3	C\BASE\b1, b2, b3.mdb	***	Join b1, b2, b3 into b1; Select sum([A1]) from b1.dbf	IF A1=A1* is TRUE END ELSE END

The software agent "**AGENT**" implements the task execution as a sequence of three consecutive actions in the form of operations $S1$, $S2$, and $S3$ with databases $b1$, $b2$, and $b3$. If the task condition $A_1 = A_1^*$ is satisfied at stage $S1$, the overall task is considered completed. Otherwise, an operation is executed in the form of merging two databases ($b1$ and $b2$), and the query with condition $A_1 = A_1^*$ is repeated, and so on, for up to three iterations. The software agent technology proposed in this paper allows one to successfully and efficiently solve a complex of tasks of managing and analyzing information within a unified heterogeneous information space.

Conclusions

As a result of the analysis of methods for managing information resources in corporate systems, an approach to the representation of agent-oriented tasks has been considered, based on analyzing the state of the external environment and the state of the information system. Two main methods of representing agent-oriented tasks have been investigated: state-based planning and task-based planning. The specifics of managing the state space and the structure of modified predicate nets have been analyzed in the context of hierarchical interaction of tasks.

The results obtained can be extended to a certain hierarchy of subordinate nets, taking into account their features and the necessity of generating metapositions and metatransitions. The proposed approach makes it possible to reduce time and computational resources through the activation of subordinate tasks, identification of stable dependencies, and exclusion of irrational and unsatisfactory solutions from consideration.

References

1. Wooldridge, M. (2002), "An Introduction to Multi-Agent Systems". *John Wiley & Sons Ltd*, 366 p.
2. Ponomarenko, L. A., Tsybulnyk, Ye. Ye., Filatov, V. A. (2023), "Agent technologies in information search and decision-making tasks". *USiM: Control systems and machines*, No. 1, P. 36–41.
3. Nagata, T., Ohono, M., Kubokawa, J., Sasaki, H., Fujita, H. (2002), "A multi-agent approach to unit commitment problems". *Proc. IEEE PES Winter Meet.*, P. 64–69.
4. Filatov, V. O., Yerokhin, M. A. (2023), "Improved multiobjective optimization in business process management using R-NSGA-II". *Radio Electronics, Computer Science, Control*, No. 3(187). DOI: https://doi.org/10.15588/1607-3274-2023-3-18

5. Konios, A., Khan, Y. I., Garcia-Constantino, M., Lopez-Nava, I. H. (2023), "A Modular Framework for Modelling and Verification of Activities in Ambient Intelligent Systems". *Lecture Notes in Computer Science*, Vol. 14029. Springer, Cham. DOI: https://doi.org/10.1007/978-3-031-35748-0_35
6. Pokorný, J., Richta, K., Richta, T. (2018), "Information Systems Development via Model Transformations". *Lecture Notes in Computer Science*, Vol. 10751. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-75417-8_63
7. Ziegler, P., Dittrich, K. R. (2007), "Data Integration". *Problems, Approaches, and Perspectives, Conceptual Modelling in Information Systems Engineering*, P. 39–58. DOI: 10.1007/978-3-540-72677-7_3
8. Nagata, T., Nakayama, H., Utatani, M., Sasaki, H. (2002), "A multi-agent approach to power system normal state operation", *Proc. IEEE/Power Eng. Soc. General Meeting*, Vol. 3, P. 1582–1586.
9. Solanki, J. M., Khushalani, S., Schulz N. N. (2007), "A multi-agent solution to distribution systems restoration". *IEEE Trans. Power Syst*., Vol. 22, No. 3, P. 1026–1034.
10. Martin, N., Depaire, B., Caris, A. (2016), "The use of process mining in business process simulation model construction: structuring the field". *Business & Information Systems Engineering*, Vol. 58, No. 1, P. 73–87.
11. Pourbafrani, M., Jiao, S., van der Aalst W. M. P. (2021), "SIMPT: Process improvement using interactive simulation of time-aware process trees". *Proceedings of RCIS 2021, Lecture Notes in Business Information Processing (LNBIP),* P. 588–594.
12. Girault, C., & Valk, R. (2003), "Petri Nets for Systems Engineering". *Springer Berlin Heidelberg*. DOI: https://doi.org/10.1007/978-3-662-05324-9
13. Murata, T. (1989), "Petri nets: Properties, analysis and applications". *Proceedings of the IEEE*, No. 77(4), P. 541–580. DOI: https://doi.org/10.1109/5.24143
14. Medina-Garcia, S., Medina, J., Montaño, O., Gonzalez-Hernandez, M., Hernánndez Gress, E. (2023), "A Petri net approach for business process modeling and simulation". *Applied Sciences*, Vol. 13, 11192. DOI: 10.3390/app132011192
15. Alves, F., Merlim, R., de Carvalho, L., Souza, F. (2023), "BPMN and Petri Nets: A case study of process optimization in a project engineering company". *Proceedings of the 7th International Academic Research Conference on Engineering, IT and Applied Sciences*. DOI: 10.33422/7th.iarmea.2023.07.124
16. Qin, J., Zhao, N., Xie, Z., et al. (2017), "Business Process Modelling based on Petri nets". *MATEC Web of Conference*, Vol. 139, No. 1, P. 105–113.
17. Li, Z., Ye, Z. (2021), "A Petri Nets Evolution Method that Supports BPMN Model Changes". *Scientific Programming*, Vol. 2021, Issue 3, P. 1–16.
18. Van Hee, K. M., Sidorova, N., van der Werf, J. M. (2013), "Business process modeling using Petri nets". *Transactions on Petri Nets and Other Models of Concurrency VII*, Vol. 7480, P. 116–161. DOI: https://doi.org/10.1007/978-3-642-38143-0_4
19. Kucherenko, I., Filatov, V. O. (2004), "Decision-making models in complex systems analysis problems based on high-level networks". *Information processing systems*, No. 5, P. 139–148.

Received (Надійшла) 19.08.2025
Accepted for publication (Прийнята до друку) 07.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Filatov Valentin – Doctor of Sciences (Engineering), Professor, Kharkiv National University of Radio Electronics, Professor of Artificial Intelligence Department, Kharkiv, Ukraine; e-mail: valentin.filatov@nure.ua; ORCID ID: https://orcid.org/0000-0002-3718-2077

Chernenko Mykola – PhD (Engineering Sciences), Kharkiv National University of Radio Electronics, Assistant of Artificial Intelligence Department, Kharkiv, Ukraine; e-mail: mykola.chernenko@nure.ua; ORCID ID: https://orcid.org/0009-0006-0623-5056

Філатов Валентин Олександрович – доктор технічних наук, професор, Харківський національний університет радіоелектроніки, професор кафедри штучного інтелекту, Харків, Україна.

Черненко Микола Володимирович – кандидат технічних наук, асистент кафедри штучного інтелекту, Харківський національний університет радіоелектроніки, Харків, Україна.

ПІДТРИМКА ЗАВДАНЬ КОРИСТУВАЧА В ІНФОРМАЦІЙНИХ СИСТЕМАХ НА ОСНОВІ АГЕНТНИХ ТЕХНОЛОГІЙ

Предметом роботи є застосування мультиагентних систем і технологій програмних агентів для управління й аналізу інформаційних ресурсів і потоків завдань у розподілених гетерогенних інформаційних середовищах. Розглянуто формальні моделі та межі виконання, що забезпечують інтелектуальний контроль складних агентно-орієнтованих робочих процесів. **Мета дослідження** – розробити ефективні методи й моделі подання, координації та виконання агентно-орієнтованих завдань, що гарантують коректність і ефективне використання ресурсів у розподілених системах. **Завдання**: формалізація інформаційного простору як алгебраїчної системи; визначення елементарних і функціональних завдань; організація взаємозв'язків між ними в потоки завдань; моделювання поведінки агентів за допомогою структур фреймів та ієрархічних мереж Петрі з метапозиціями й метапереходами. **Методи** основані на формальному моделюванні інформаційного простору, впровадженні теорії автоматів для аналізу поведінки агентів, використанні розширених предикатних мереж Петрі з метою підтримки ієрархічного управління завданнями, а також на застосуванні логіки обчислювального дерева (CTL) й автомата залежностей для формального визначення взаємозалежностей і вимог до узгодженості виконання. Як приклад, використано операції SQL для ілюстрації злиття баз даних у процесі виконання завдань. **Результати дослідження** передбачають: формалізовану структуру завдань агентів із чітким розмежуванням планування в просторі станів і завдань, удосконалену модель слотів у фреймі для комплексного опису атрибутів завдань, ієрархічний каркас мереж Петрі з підлеглими мережами для оптимізації обчислень і реалізацію програмного агента, який послідовно виконує умови із злиттям баз даних для досягнення цілей. Це дає змогу скоротити час виконання й обчислювальні ресурси внаслідок активації тільки необхідних підзавдань і вилучення неефективних рішень. **Висновки** підтверджують, що запропоновані модель і методи ефективно розв'язують складні завдання управління розподіленими інформаційними ресурсами й контролю робочих процесів, забезпечуючи масштабованість, надійність й атомарність виконання в мультиагентних системах гетерогенних обчислювальних середовищ.

Ключові слова: мультиагентні системи; ієрархічне управління завданнями; мережі Петрі; управління інформаційними ресурсами.

Bibliographic descriptions / Бібліографічні описи

Filatov, V., Chernenko, M. (2025), "User task support in information systems based on agent technologies", *Management Information Systems and Devises*, No. 4 (187), P. 142–155. DOI: https://doi.org/10.30837/0135-1710.2025.187.142

Філатов В. О., Черненко М. В. Підтримка завдань користувача в інформаційних системах на основі агентних технологій. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 142–155. DOI: https://doi.org/10.30837/0135-1710.2025.187.142

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.8:004.4:005.94 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.156

O. Shmatko, I. Gamayun, O. Kolomiitsev

HYBRID MACHINE LEARNING MODEL FOR CLASSIFYING SOFTWARE BUGS IN SAAS CLOUD APPLICATIONS

In modern cloud computing environments, ensuring the stability and reliability of software applications is one of the key factors for the effective operation of information systems. A significant portion of failures in such systems are caused by software errors (bugs), which complicate operation and reduce service productivity. Traditional methods of manual analysis of bug reports are labor-intensive, so it is necessary to develop intelligent approaches to automated classification and prioritization of bugs using machine learning methods. **The purpose of the article** is to improve the accuracy of classifying types of software bugs in cloud applications. **Research objectives:** to develop a complete pipeline for automated processing of bug reports, covering all stages from preliminary cleaning to classification model building. **The methodological basis of the study** is the use of natural language processing (NLP) methods, the SMOTE technique for sample balancing, classical machine learning algorithms, and the *RandomizedSearchCV* hyperparameter optimization procedure. The quality of the models is evaluated based on standard classification metrics such as *accuracy, precision, recall, and F1-score*, which provides a comprehensive and objective analysis of the results. Research results. A hybrid model for automated bug classification has been developed, covering the stages of data collection, preprocessing, vectorization, and modeling. A comparative analysis of the accuracy of four machine learning algorithms – naive Bayes classifier, decision tree, random forest, and logistic regression – was performed using different vectorization methods (*Bag-of-Words, TF-IDF, Word2Vec*). To improve classification accuracy, the SMOTE data balancing technique was applied. Experimental studies on a real data set from a cloud environment showed that the Random Forest model achieved the highest accuracy rates – up to 91.7 %. The results confirm the effectiveness of integrating machine learning algorithms into the processes of analysis and support of software products in cloud infrastructures. **Conclusions.** The proposed approach improves the accuracy of bug classification in cloud computing systems and can be used in monitoring systems, *DevOps* platforms, and automated testing tools. The research results provide a basis for the further development of intelligent tools for predicting and prioritizing software defects.

Keywords: bug classification; cloud computing; machine learning; *TF-IDF; Word2Vec*; random forest; test automation.

Introduction

Problem statement. In today's digital transformation environment, cloud computing models play a key role in ensuring the availability, scalability, and efficiency of software services. One of the most common cloud paradigms is Software as a Service (SaaS), which provides users with ready-to-use software applications via the Internet.

Unlike traditional approaches to software deployment, the SaaS model eliminates the need to install programs on local devices, as the provider is fully responsible for the development, deployment, updating, and support of the application. Users get access to the system's functionality according to the terms of the Service Level Agreement (SLA), which can be regulated by model subscriptions, hourly or volume-based payments. Today, SaaS solutions are widely used in email, financial services, human resource management, and other industries.

The growing popularity of SaaS is leading to an exponential increase in the number of users and the expansion of the functionality of such systems. However, the intensive use of cloud applications is inevitably accompanied by the emergence of a significant number of software defects that affect the quality and stability of services. Bugs in SaaS environments can cause delays in business processes, reduced productivity, a poor user experience, and direct financial losses. In these conditions, the effectiveness of technical support depends on the speed and accuracy of bug detection, classification, and prioritization. Manual processing of bug reports is complex, time-consuming, and resource-intensive, which is why automating bug classification processes is particularly important.

Machine learning methods capable of analyzing large data sets and identifying hidden patterns in text descriptions of bugs are also of considerable scientific interest. The use of machine learning algorithms in the field of bug report processing opens up new opportunities to improve the accuracy of error type identification, reduce the time required to fix them, and reduce the workload on the development team and system administrators. With the rapid growth in the number of cloud applications, the need for such approaches is becoming critically important, as the correct classification and prioritization of defects directly affect the stability and reliability of services.

Automated classification of software bugs in SaaS systems using machine learning methods is a promising area of research aimed at improving the efficiency of technical support, the accuracy of defect identification, and the optimization of quality assurance processes. Research in this area provides a scientific basis for the development of intelligent tools for analysis, forecasting, and decision support in cloud infrastructures.

Analysis of recent research and publications. In today's environment of increasing software system complexity, software testing plays a key role in the verification and validation (V&V) process, ensuring the correctness, stability, and long-term reliability of the systems being developed [4]. With the increasing criticality of software systems, particularly in the fields of security, medicine, transport, etc., the costs associated with their analysis, testing, and quality assurance are also rising significantly. This, in turn, creates demand for effective methods of defect prediction using intelligent technologies, in particular machine learning (ML) algorithms, which enable prediction, optimization, and automatic learning with minimal human intervention.

In the context of research aimed at automating the detection of bugs in SaaS applications, particular attention is drawn to works devoted to the classification of bugs based on bug reports and the determination of their priority. Researchers point to the availability of both ready-made tools for code analysis and bug detection, as well as models capable of prioritizing these bugs based on historical data [5]. In [6], the idea of using machine learning to automate manual processes, particularly in the area of bug prioritization, is put forward. The authors note that based on historical bug reports, models can learn to identify patterns and make predictions about the importance of new bugs. This approach improves classification accuracy and reduces the burden on technical support.

A similar approach is supported in [7], which emphasizes that the increasing complexity of software systems significantly complicates manual bug detection, making it slow and prone to human bug. The use of ML models in such conditions not only improves the quality of software but also reduces the cost of its maintenance. This is especially true for secure or mission-critical systems, where even minor bugs can have serious consequences.

Another promising area of research is the use of ensemble methods in bug triaging – the process of automatically determining which developer should be assigned a particular bug. Work [8] demonstrates that ensemble classifiers (which combine several models to achieve better results) outperform classical machine learning algorithms in bug assignment tasks. This indicates the possibility of significantly improving the efficiency of the bug handling process and reducing delays in their resolution.

In [9], an innovative approach to bug prioritization based on emotional analysis of bug descriptions was proposed. The authors collected data from open sources, performed natural language processing (NLP), extracted emotional words from the text, and based on this, formed a feature vector for the ML model. This approach increased classification accuracy (F1 score) by more than 6 %. The use of emotional analysis allows for better consideration of subjective user assessments, which is especially important in interface-oriented or client systems.

A significant problem in bug classification tasks is class imbalance, where most examples belong to insignificant classes, and critical bugs occur much less frequently. In such cases, most models tend to overfit on frequent classes, which reduces the effectiveness of detecting truly important bugs. The paper [10] considers an approach that involves building an ensemble classifier using various oversampling methods to improve the representation of small classes. The results of the study showed that combining classification and sample balancing reduces the number of false negatives and improves the accuracy of defect component recognition.

Our study proposes an approach to automating the processes of classification and prioritization of software bugs in SaaS applications. With the growth in the number of users and the increase in the functional load on cloud services, the probability of defects occurring is steadily increasing. These bugs can significantly degrade the quality of the user experience, cause delays in business processes, and create additional difficulties in maintaining such systems. Therefore, an urgent task is to develop machine learning models capable of automatically processing bug logs, identifying defect types, and prioritizing them to optimize the work of development and technical support teams.

The goal of this work is to improve the efficiency of classifying types of software bugs in cloud computing environments based on a hybrid approach to software bug classification using NLP, vectorization, and balanced machine learning methods.

Main material

The proposed design solution for classifying bugs in cloud computing applications uses machine learning methods to automate and improve the detection and prioritization of software defects. This approach takes into account the inherent complexity and scaling challenges in cloud environments, where bugs can manifest in distributed systems, virtualized resources, and heterogeneous infrastructures. The methodology follows a structured pipeline covering data collection, preprocessing, feature engineering, model selection and training, evaluation, and deployment. Although presented as a high-level framework, the solution is adaptable to specific contextual constraints, as shown in Figure 1.

The initial phase involves collecting data from a variety of sources relevant to cloud computing applications. Bug data is aggregated from bug tracking systems (e.g., Jira or Bugzilla), official repositories, user forums, and historical logs. This multifaceted approach to sources provides a comprehensive data set that reflects real-world bug manifestations, including those arising from resource contention, network latency, or configuration bugs in cloud-native architectures.

After collection, the data is preprocessed to remove noise and inconsistencies. Unnecessary artifacts, such as duplicate records or discussions unrelated to bugs, are removed. Missing values are imputed using methods such as replacement by the mean or k-nearest neighbors, while text data is normalized, tokenized, and stop words are removed. This step transforms the raw input data into a structured format suitable for machine learning analysis.

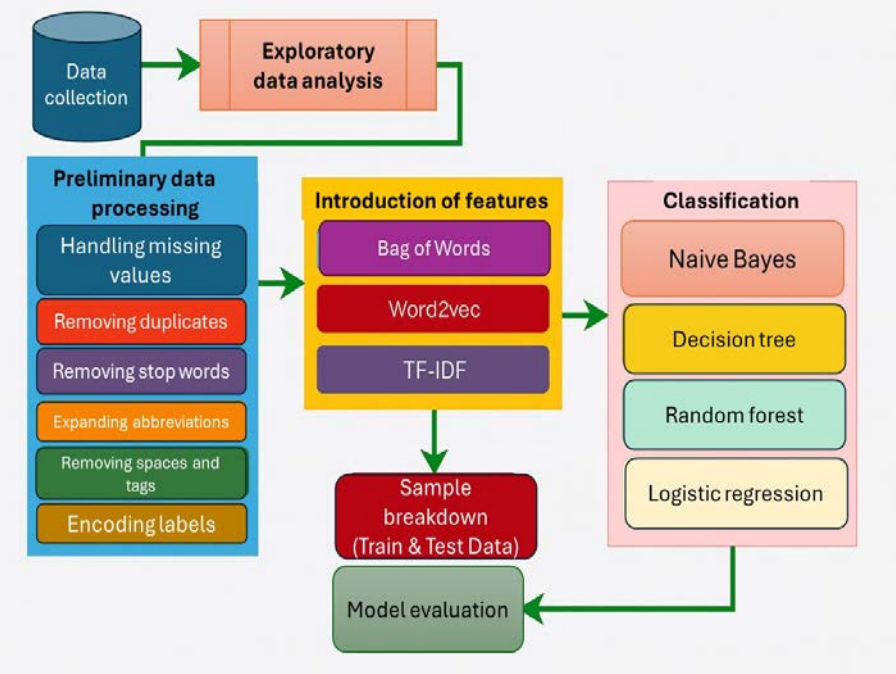


Fig. 1. Algorithm of the proposed bug classification conveyor in cloud computing applications

After cleaning and transforming the data, a vector representation is formed that is suitable for further processing by machine learning algorithms. This study considered three popular vectorization methods: Bag of Words (BoW), TF-IDF, and Word2Vec.

1. *Bag of Words (BoW).* The Bag of Words (BoW) method is a basic statistical method that represents text as a vector of word frequencies. Let us have a corpus of documents containing a dictionary $V = \{w_1, w_2, ..., w_n\}$. Each document is represented as a vector:

$$\vec{v}_d = \left[f(w_1, d), f(w_2, d), \ldots, f(w_n, d) \right],$$

where $f(w_i, d)$ – frequency of a word w_i in a document d. The method does not take into account word order and semantic relationships, but it is effective with a sufficient amount of data.

2. *TF-IDF (Term Frequency – Inverse Document Frequency).* The TF-IDF method improves BoW by weighting word frequency based on how unique the term is within the entire corpus. For a word t in a document d, the TF-IDF formula is calculated as follows:

$$\text{TF-IDF}(t,d,D) = \text{TF}(t,d) \cdot \text{IDF}(t,D),$$

where $\text{TF}(t,d) = \log(1 + f(t,d))$,

$$\text{IDF}(t,D) = \log\left(\frac{N}{|\{d \in D : t \in d\}|}\right),$$

$f(t,d)$ – number of occurrences of the term t in the document d,

N – total number of documents in the corpus D,

$|\{d \in D : t \in d\}|$ – number of documents containing the term t.

TF-IDF allows you to reduce the weight of common terms and increase the significance of rare, specific words.

3. *Word2Vec.* Unlike statistical methods, Word2Vec is a deep learning model that creates dense vector representations of words, taking into account the context of their use. Developed by Google in 2013, the model has two main architectures: Continuous Bag of Words (CBOW) and Skip-Gram.

Let w_t is the trarget word, $C = \{w_{t-n}, ..., w_{t-1}, w_{t+1}, ..., w_{t+n}\}$ — the context. In CBOW, the task is to predict w_t based on the context:

$$P(w_t C) = \frac{e^{v_{w_t} \cdot h}}{\sum_{i=1}^{|V|} e^{v_i \cdot h}},$$

where h – average vector representation of words from context; v_i – vector representation of a word i.

In Skip-gram, on the contrary, the model learns to predict contextual words based on a given word.

In our study, we used the CBOW algorithm with the following parameters:

– min_count = 5 — words that occur less than 5 times are ignored;

– size = 50 — the dimension of the vector space;

– workers = 4 — the number of threads for training.

As part of the study, four classic machine learning algorithms were used to solve the problem of multi-class classification of error types in cloud computing applications: naive Bayes classifier, decision tree, random forest, and logistic regression.

Each of these methods has its own advantages, limitations, and peculiarities of use in natural language processing tasks.

1. *Naive Bayes classifier.* This method is based on Bayes' theorem with the assumption of conditional independence of features. In text classification tasks, it is considered simple, fast, and effective.

Its probability model is determined by the formula:

$$P(yx_1, x_2, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i y)}{P(x_1, x_2, \ldots, x_n)},$$

where y – class, x_i – signs (e.g., words), and $P(x_i \mid y)$ – probability of a sign x_i appearing given the class y.

2. *Decision Tree.* Decision trees are algorithms that build a hierarchical model where each internal node branch corresponds to a condition based on a specific feature, and leaf nodes correspond to classes. The main goal is to minimize entropy or the Gini coefficient during partitioning. Formally, entropy is used for:

$$H(D) = -\sum_{i=1}^{n} p_i \log_2 p_i,$$

where p_i – proportion of class i elements of D data sets.

Trees are interpretive but prone to overfitting on noisy data.

3. *Random Forest.* Random Forest is an ensemble method that combines a set of decision trees created on random subsets of data and features. Each tree votes for a class, and the final prediction is determined by majority vote. The method reduces model variance, improving generalization:

$$\hat{y} = \text{mode}\{h_1(x), h_2(x), \ldots, h_k(x)\},$$

where $h_i(x)$ – forecast of the i-th tree.

Random forest demonstrates high accuracy, especially on complex and large-scale data, making it effective for text classification.

4. *Logistic Regression.* This linear method is widely used for classification tasks due to its mathematical rigor and stability. Softmax regression is typically used for multi-class classification. The probability of belonging to a class k is determined by:

$$P(y = kx) = \frac{e^{\beta_k^T x}}{\sum_{j=1}^{K} e^{\beta_j^T x}},$$

where β_k – vector of coefficients for class k, x – feature vector.

Optimization is performed by maximizing the logarithmic likelihood function.

After the model is defined, it is trained on the prepared data set. The training process includes sample balancing to avoid bias towards the dominant class, cross-validation to evaluate the model's generalization ability, and selection of optimal hyperparameters. The result is a classification model that can automatically recognize bug categories based on the textual and structural features of the bug report.

During the validation stage, the model is evaluated using a separate test dataset. Its effectiveness is assessed using standard classification quality metrics such as accuracy, precision, recall, and F1-score. Each of these metrics allows us to evaluate different aspects of the model's performance: its ability to correctly classify objects, avoid false positives and false negatives, and the overall balance between precision and recall.

After passing the evaluation stage, the model is deployed in a cloud environment.

Its integration into real systems allows you to automate the process of processing new bug

reports, classify them in real time, and route them to the responsible executors. This, in turn, helps to reduce response time, increase the efficiency of technical support, and generally optimize the software maintenance process.

High-quality and meaningful data is the foundation of modern data science, as the effectiveness and accuracy of a machine learning model directly depend on the quality of the source data set. That is why the first stage of implementing the proposed approach was data processing, which included collection, cleaning, visualization, and exploratory data analysis (EDA).

Figure 2 shows a deployment diagram that reflects the architecture of the bug classification system in the SaaS cloud environment.

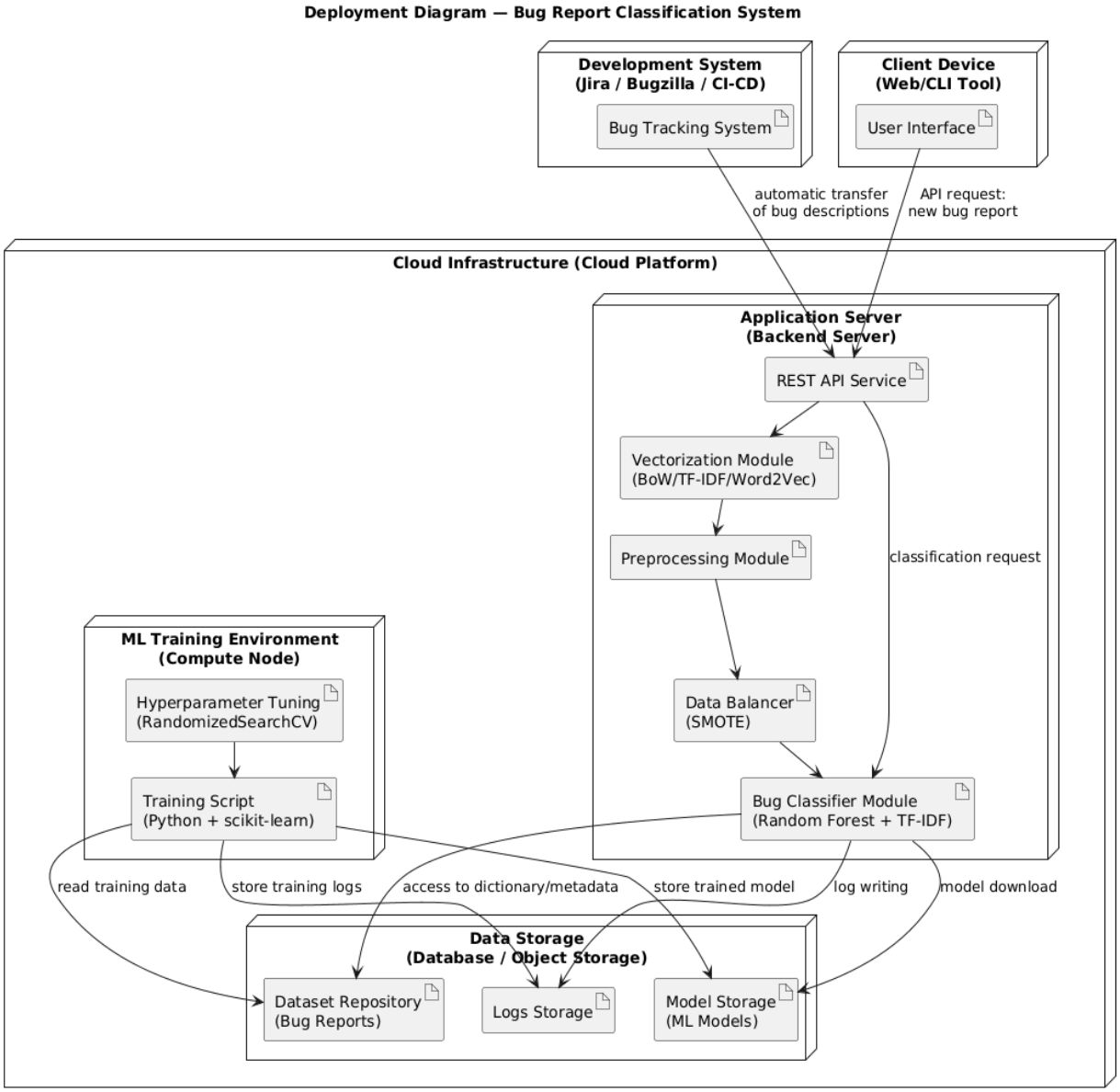


Fig. 2. Algorithm of the proposed bug classification conveyor in cloud computing applications

The architecture of the proposed automated software bug classification system in cloud SaaS applications consists of a number of interconnected components located in the cloud infrastructure and designed to ensure a complete bug report processing cycle – from the moment the data is received to the formation of a classification conclusion. Each component performs clearly defined functions, ensuring scalability, modularity, and the ability to flexibly integrate with existing DevOps services and support systems.

1. *Cloud Platform*. The main environment in which all server modules of the classification system are deployed. Provides scalability, network interaction, and computing resources.

2. *Backend Server*. Hosts software components that process requests, classify bugs, and coordinate interaction between other modules:

– REST API Service – an interface for interaction between users, external services, and the classification system. Accepts bug reports and returns classification results.

– Bug Classifier Module – the main classification module, which runs a trained machine learning model (Random Forest + TF-IDF).

– Preprocessing Module – a component for cleaning and normalizing the text of bug reports before vectorization.

– Vectorization Module – implements Bag-of-Words, TF-IDF, and Word2Vec methods to convert text into numerical vectors.

– Data Balancer (SMOTE) – used during model training to eliminate class imbalance.

3. *Data storage (Database / Object Storage)*. Used to store permanent data and models:

– Dataset Repository – storage of bug reports, dictionaries, and metadata.

– Logs Storage – storage of system logs, classification history, and technical events.

– Model Storage – file or object storage of a trained ML model available to the classification module.

4. *ML Training Environment (Compute Node)*. A separate powerful computing environment designed for training models:

– Training Script (Python + scikit-learn) – scripts for training classifiers.

– Hyperparameter Tuning (RandomizedSearchCV) – a module for optimizing hyperparameters to achieve the best accuracy.

5. *Development System (Jira / Bugzilla / CI/CD)*. An external data source that automatically transfers bug reports to the classification system or receives analysis results.

– Bug Tracking System – a bug management tool that integrates with the system's REST API.

6. *Client device (Web/CLI Tool)*. A component through which the user interacts with the system:

– User Interface – a web interface or command interface that allows you to send bug reports and view classification results.

This study uses a publicly available dataset published on the Kaggle platform in 2020 [11]. This is because most similar data is either closed or extremely labor-intensive to collect independently.

The dataset is an example of a web-based issue tracker, specifically in the field of Python development.

Its structure is presented in Table 1, which contains a description of the attributes used for further construction of bug classification models.

Table 1. *Data set characteristics*

№	Attribute	Description
1	Unnamed	The column contains a unique identifier for each record
2	Title	The column contains the full text of the bug in the form of a record
3	Type	Target variable (label); indicates the type of bug

The dataset contains 5,300 records and three main attributes: a unique identifier (Unnamed), a text description of the bug (title), and a target variable – the error type (type). A total of six bug categories have been identified: enhancement, security, compilation bug, resource utilization, performance, and crash.

An example description is shown in Figure 3. A distinctive feature of this data is that bug names often contain technical codes (e.g., SyntaxError, ImportError), which significantly complicates the classification task, since the language is not natural in the usual sense.

Title	Type
Doc strings omitted from AST	Performance
Upload failed (400): Digests do not match on .tar.gz ending with x0d binary code	Resource usage
ConfigParser writes a superfluous final bank line	Performance
csv.reader() to support QUOTE_ALL	Crash
IDLE: make smart indent after comments line consistent	Performance
xml.etree.Elementinclude does not include nested xincludes	Crash
Add Py_BREAKPOINT and sys._breakpoint hooks	Crash
documentation of ZipFile file name encoding	Performance
Allow 'continue' in 'finally' clause	Crash
Move unwinding od stack for "pseudo exceptions" from interpreter to compile	Crash
Improve regular expression HOWTO	Crash
Windows python cannot handle an early PATH entry containing "..." and python.exe	Enhancement
tkinter after_cancel does not behave correctly when called with id=None	Performance
PEP 1: Allow provisional status for PEPs	Crash
os.chdir(), os.getcwd() may crash on windows in presence of races	Enhancement
tk busy command	Crash
os.chdir() may leak memory on windows	Compiler error

Fig. 3. Example of bug description

The distribution of error types is analyzed using graphical visualization. Figure 4 shows a histogram demonstrating the number of records for each error type. The most common bugs are

those related to performance, while the least common are those related to resource utilization.

Since this is a multi-class classification, the issue of class balancing is not critical. Instead, it is advisable to use a cross-validation method, which avoids overfitting the model.

This approach involves dividing the entire dataset into several parts (folds) and testing the model step by step on each subset, which significantly increases the accuracy and stability of the results.

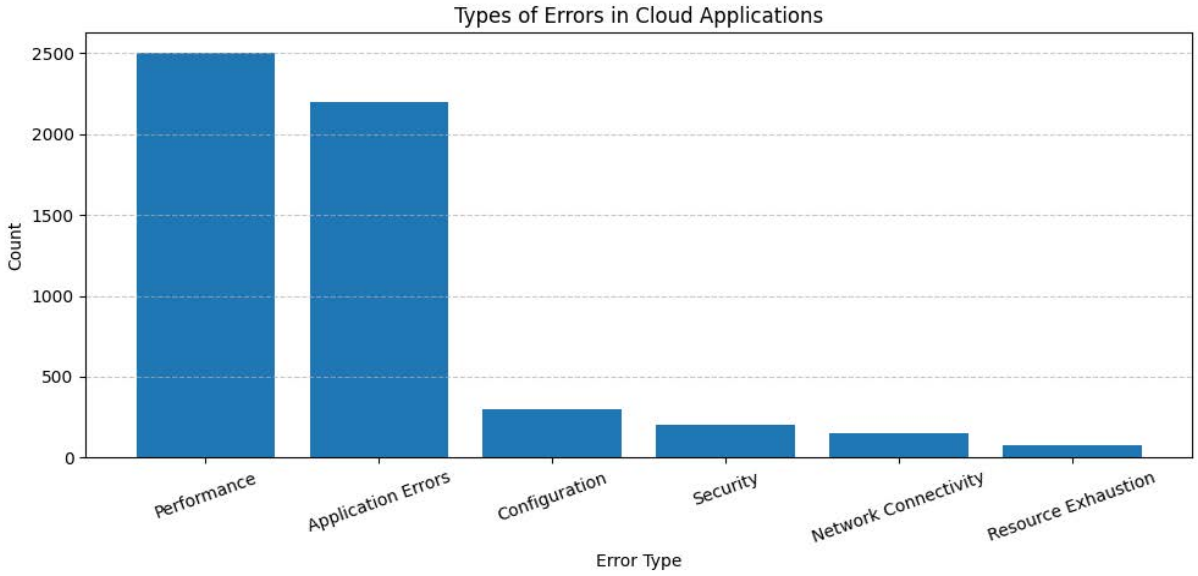


Fig. 4. Distribution of bugs by type

The next step was to analyze the frequency of terms in bug descriptions. One of the simplest but most informative approaches is to use unigrams – single words that are considered independently of each other. Figure 5 shows a graph with the ten most frequently used words, among which the word module occurs most often, while python occurs least often. Most of them are typical for bugs in Python repositories (e.g., file, function, code).

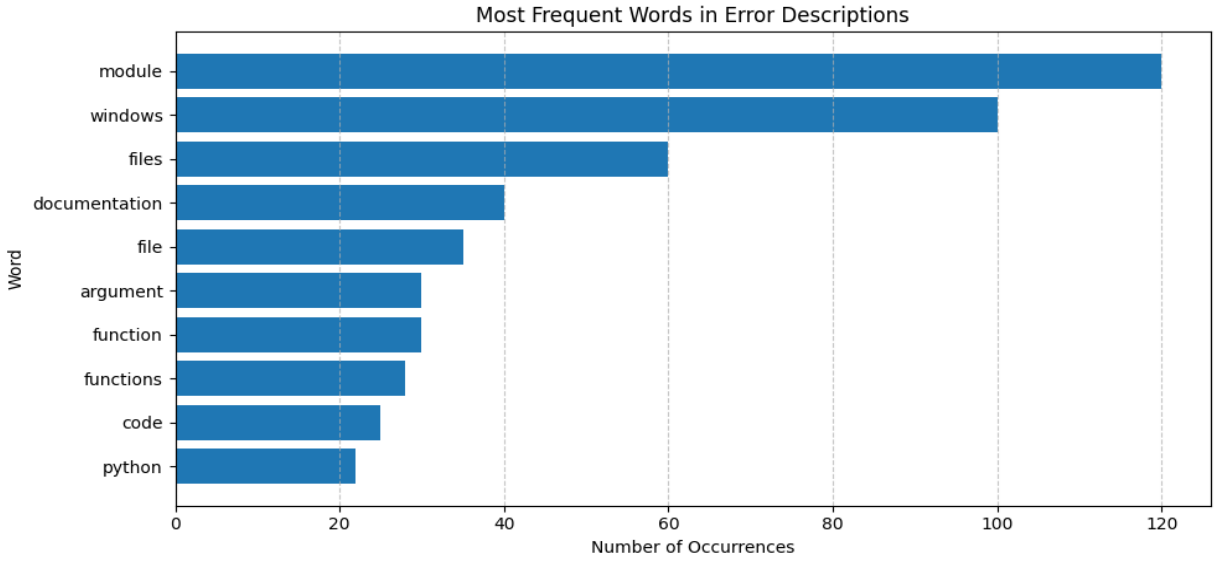


Fig. 5. Most frequently used words in bug descriptions

Preprocessing is a key step in the process of extracting knowledge from data, as it allows raw, unstructured, or partially structured information to be converted into a machine-readable format. Real-world datasets are typically characterized by incompleteness, redundancy, instability, and errors. Therefore, the application of high-quality preprocessing procedures is a prerequisite for building a reliable and generalizable machine learning model.

Within the scope of this study, preprocessing was implemented in several stages, which can be broadly divided into two main phases: working with raw data and basic preprocessing with data labeling.

After completing the cleaning stages, the data was ready to be transformed into a format suitable for modeling. In particular, the type column, which is a categorical variable, needed to be converted to a numerical format.

To do this, we used the Label Encoding method, which replaces each unique category with a corresponding number. As a result, all six error classes received unique numerical labels. The scheme of encoded values is shown in Figure 6.

Label	Error Type
0	Crash
1	Enhancement
2	Performance
3	Resource usage
4	Security
5	Compile

Fig. 6. Label encoding for multi-class classification

An equally important challenge in classification tasks is the problem of class imbalance, when the number of records for different categories is uneven. Although the dataset under study is multi-class, there is also a significant imbalance in the number of examples for each type of error.

To solve this problem, a combination of random oversampling and the SMOTE (Synthetic Minority Oversampling Technique) method was used. SMOTE is an algorithm for generating synthetic examples for minority classes by interpolating between existing points in the feature space. This approach not only balances the distribution of classes, but also reduces the likelihood of overfitting, which often occurs when simply duplicating minority examples.

As a result of balancing, the distribution of data across classes was evened out. This is clearly demonstrated in Figure 7, which shows the final state of the dataset with evenly represented classes.

Within the scope of this study, four popular machine learning algorithms were selected to solve the problem of multi-class classification of error types in cloud SaaS applications:

– Naive Bayes classifier;

– Decision Tree;

– Random Forest;

– logistic regression.

In order to improve the efficiency of modeling and conduct a more in-depth analysis of the impact of different approaches to text feature representation, a series of experiments were conducted, which included the use of various vectorization methods, parametric optimization, and comparative evaluation of models.

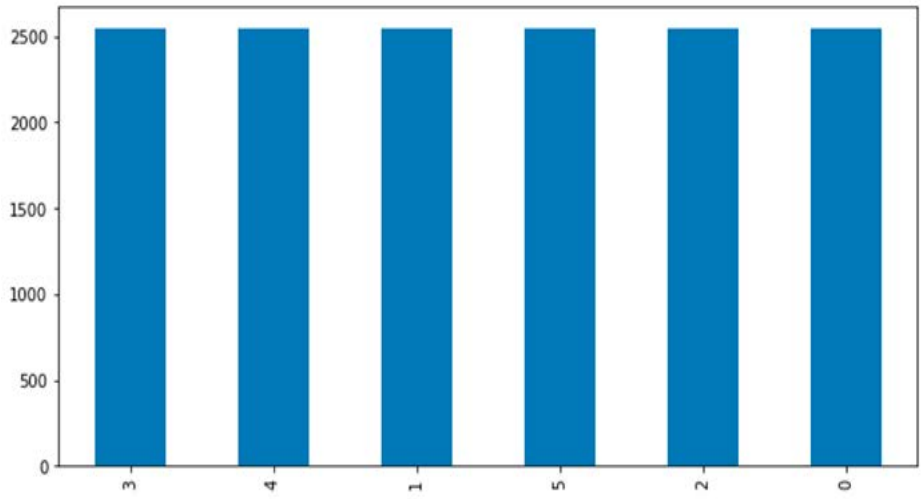


Fig. 7. Final state of the dataset with evenly represented classes

Seven key experiments were implemented, covering the following components:

– Three methods of text vectorization: Bag-of-Words (BoW), TF-IDF, and Word2Vec – a modern method of vector representation of words based on a neural network.

– Optimization of model hyperparameters for BoW and TF-IDF using the Randomized SearchCV method, which allows you to efficiently find the best parameter configurations.

– Comprehensive comparative analysis of model performance by metrics: accuracy, recall, precision, and weighted average (F1-score).

The formulas for calculating key metrics are presented below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN};$$

$$Precision = \frac{TP}{TP + FP};$$

$$Recall = \frac{TP}{TP + FN};$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall},$$

where TP, TN, FP, FN are true positive, true negative, false positive, and false negative predictions, respectively.

The dataset was divided into training and test samples in an 80/20 ratio, where 80 % of the records were used to train the model and 20 % to evaluate its generalization ability.

To ensure the reproducibility of the experiments, a fixed parameter random_state = 42 was used, which guarantees the same division of the dataset each time it is run.

Each of the four classification algorithms was trained based on three different types of vectorization, which made it possible to evaluate how the method of text representation affects the performance of the model.

The results of the study confirm that dataset balancing combined with model hyperparameter optimization are critical factors for achieving high classification accuracy. Four machine learning algorithms were used in the experiments: naive Bayes classifier, decision tree, random forest, and logistic regression – to classify bugs on both balanced and unbalanced datasets.

One of the key classifiers used in the study is Multinomial Naive Bayes, which is widely used for text classification. It is based on Bayes' theorem, assuming conditional independence of features. Since this is a multi-class task, the MultinomialNB implementation was obtained from the scikit-learn library and used in all experimental scenarios.

All experimental results for the Naive Bayes model are shown in Figures 8 and 9.

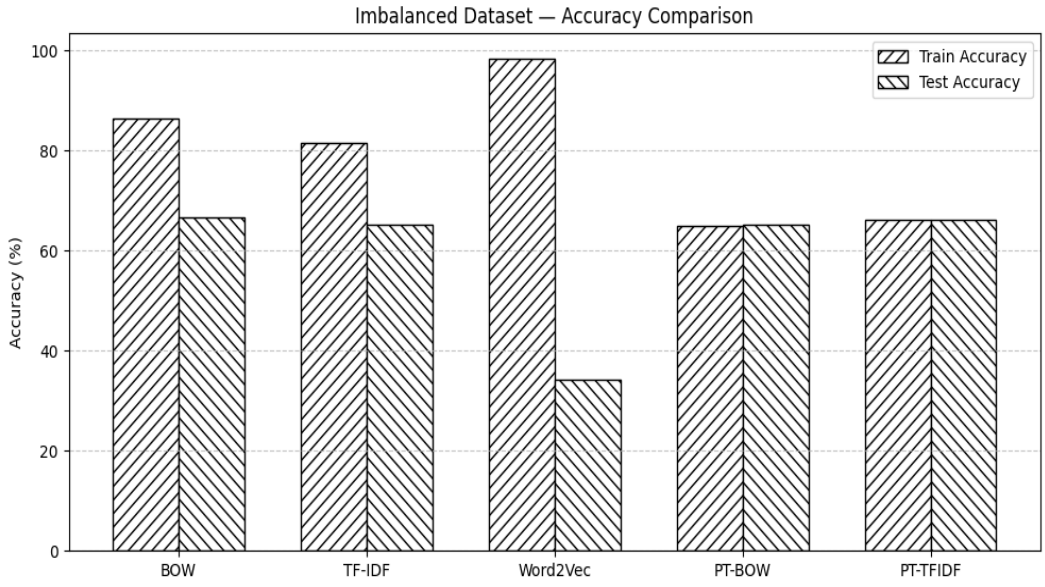


Fig. 8. Comparison of model accuracy for an imbalanced sample

The Naive Bayes model demonstrates different effectiveness depending on the vectorization method and the state of the sample (balanced or unbalanced). The best result was obtained for TF-IDF with tuned hyperparameters (PT-TFIDF).

The model achieved 97.14 % accuracy on the training set and 88.18 % on the test set, which is the highest result among all configurations.

Word2Vec vectorization showed low performance on the test data, indicating insufficient representativeness of semantic spatial features in this task.

Balancing the dataset significantly improved the classifier's performance: a comparison of experiments 1 and 2 shows a 15–20 % jump in performance.

Thus, hyperparameter tuning and correct sample preparation are critical factors for achieving high accuracy in bug report classification tasks.

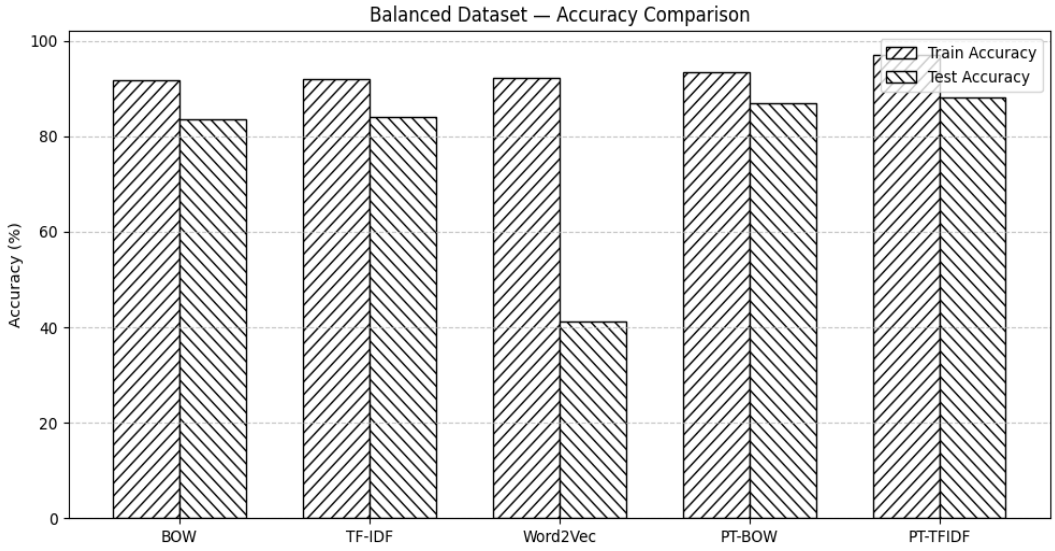


Fig. 9. Comparison of model accuracy for a balanced sample

The study also generated a Classification Report for the Naive Bayes model with the best parameters (BOW-tuned). Based on the Classification Report, a diagram was constructed (Fig. 10), which details the Precision, Recall, and F1-measure values for each error category.

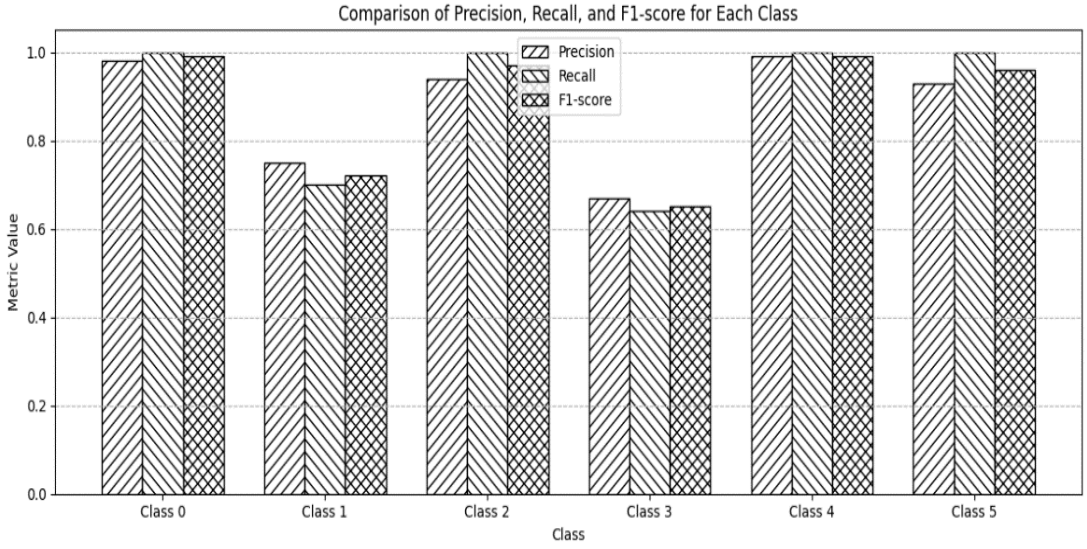


Fig. 10. Comparison of metrics for each class

Decision trees belong to the class of interpreted machine learning models and are widely used for prediction and classification tasks due to their simplicity, high learning speed, and ability to work with nonlinear dependencies. The algorithm for constructing a decision tree is based on iterative division of the feature space by selecting splitting criteria that minimize uncertainty

(impurity) in data subsets. This approach allows building a hierarchical structure of rules, according to which the model matches new inputs with the corresponding classes.

In this study, the DecisionTreeClassifier algorithm from the scikit-learn library was used to classify software errors. After loading the data, preprocessing, and vectorization, the model was trained on the training set and tested on the deferred part of the data. To ensure a correct quality assessment, all experimental scenarios similar to the previous Naive Bayes analysis were also applied to this model.

The generalized results are presented in Figures 11, 12.

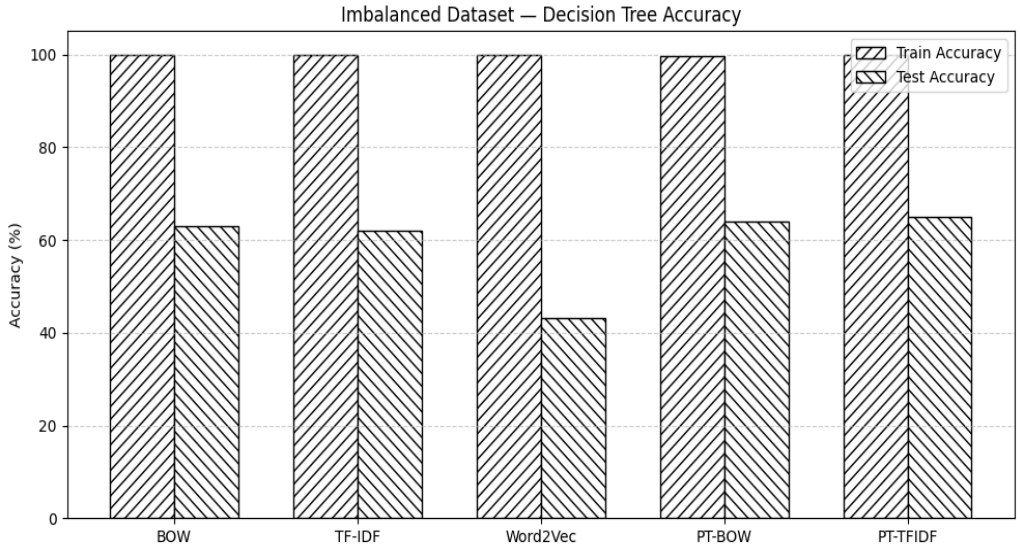


Fig. 11. Comparison of model accuracy for imbalanced samples

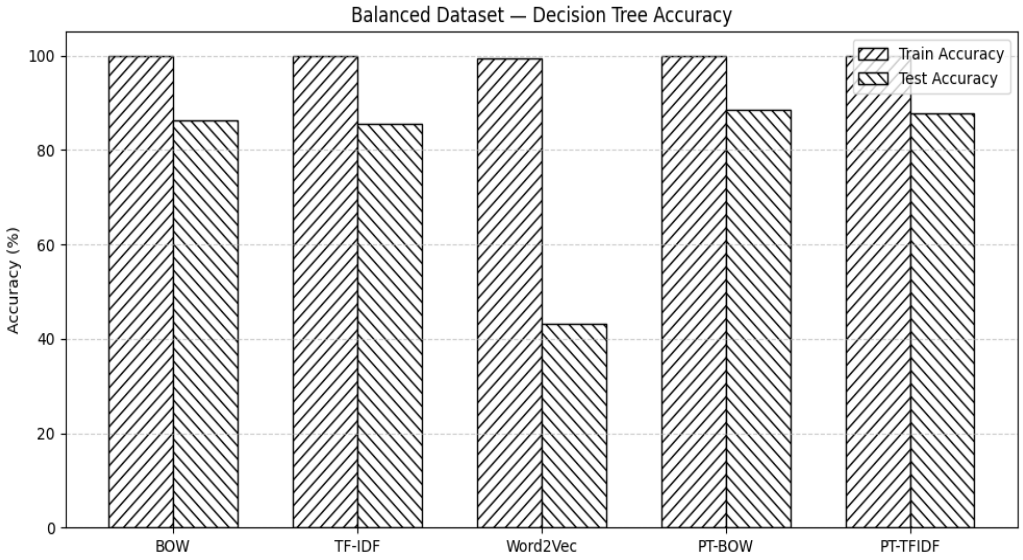


Fig. 12. Comparison of model accuracy for balanced samples

The Decision Tree model achieves maximum accuracy on the training set in almost all experiments, which is typical for decision trees, as they are prone to overfitting.

This is especially noticeable when training on an unbalanced dataset, where accuracy scores on the test sample are significantly lower – from 43 % to 65 %, depending on the vectorization method.

After applying sample balancing, the model's performance improved significantly:

– test accuracy increased to 86–88 % in configurations with BOW and TF-IDF,

– results for Word2Vec remained low (≈43 %), which is consistent with previous observations and indicates the ineffectiveness of Word2Vec in this context.

The optimal hyperparameter values were selected for the model:

– criterion = "gini"

– max_depth = 54

– min_samples_leaf = 4

– min_samples_split = 95

This setting partially reduced model overfitting, although Decision Tree remains inherently sensitive to noise and data complexity.

The results showed that after optimizing the parameters, the accuracy of the BOW and TF-IDF-based models is practically identical. The slight advantage of TF-IDF (up to 87.79 % test accuracy) is due to the fact that this method better takes into account the weight of rare terms, which is critical in text tasks.

The classification report was used for a detailed analysis of the model's behavior (Fig. 13):

– the largest number of misclassifications occurs in classes with similar text features;

– the F1-measure for individual categories varies significantly, confirming the sensitivity of the decision tree to data distribution.

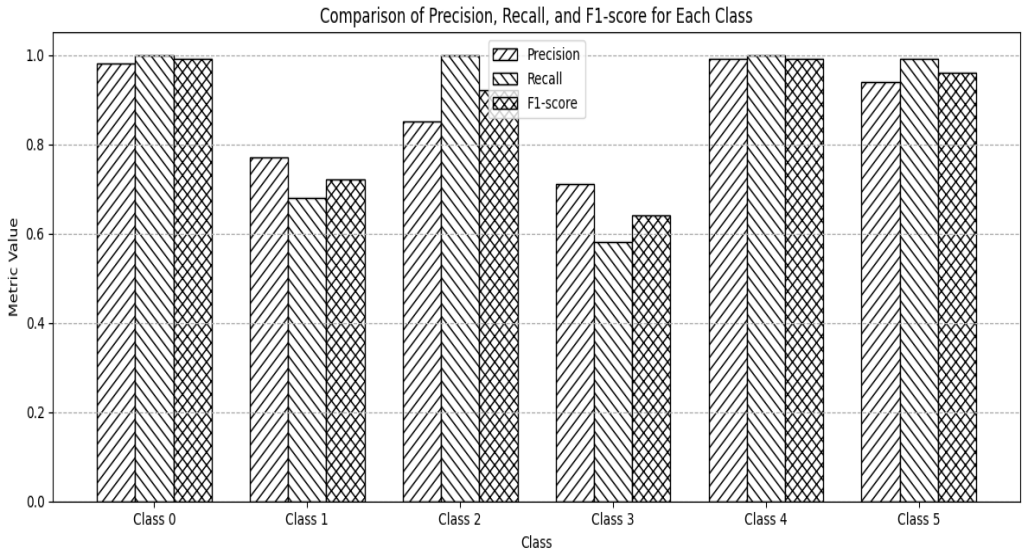


Fig. 13. Comparison of metrics for each class

Within the scope of this study, the RandomForestClassifier implementation from the scikit-learn library was used. After loading the vectorized data, the model was trained on the training sample and evaluated on the test sample.

All experimental scenarios – for different vectorization methods and with/without hyperparameter optimization – were tested sequentially. The generalized results are shown in Figures 14, 15.

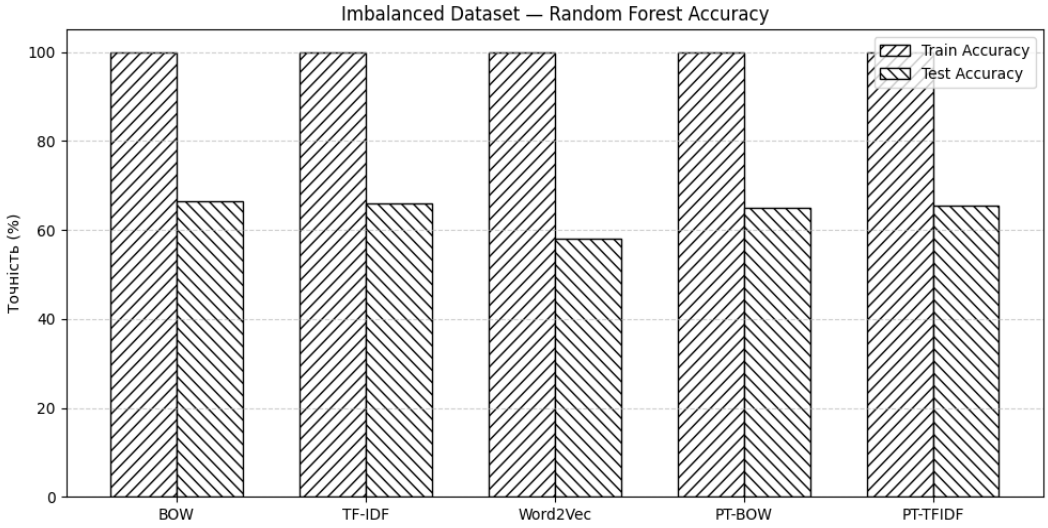


Fig. 14. Comparison of model accuracy for imbalanced samples

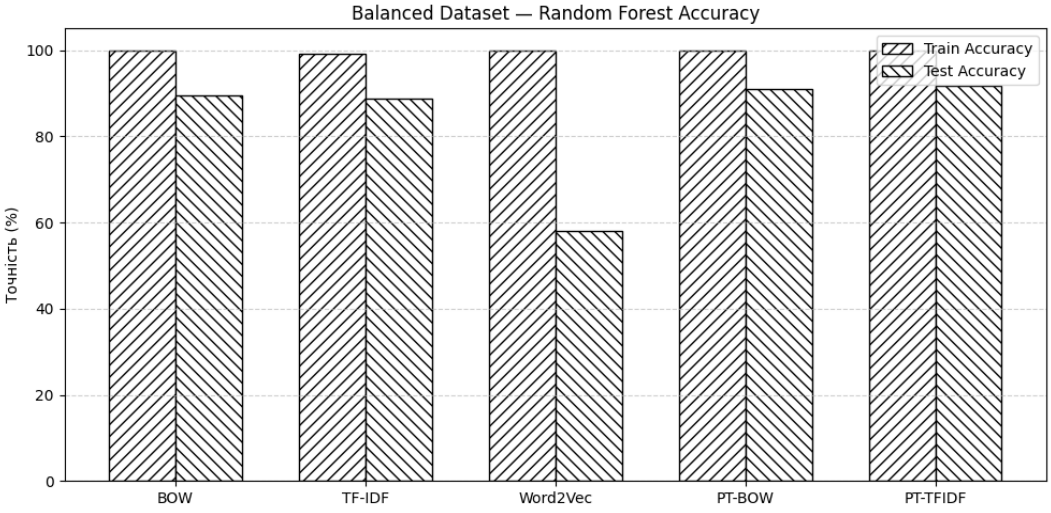


Fig. 15. Comparison of model accuracy for balanced samples

The results show that Random Forest demonstrates significantly better classification quality compared to decision trees and Naive Bayes in most settings. In particular, it can be seen that:

1. *The best result was achieved for TF-IDF with tuned hyperparameters (PT-TFIDF)*

– Train accuracy: 100 %;

– Test accuracy: 91.73 % – the highest score among all models in the study.

This confirms that Random Forest not only successfully overcomes the problem of decision tree overfitting, but also makes the most complete use of the information provided by TF-IDF vectorization.

2. *Impact of sample balancing*

A comparison of the results between part 1 (Imbalanced) and part 2 (Balance) shows:

– the accuracy gain on the test is between 20 and 30 % for BOW and TF-IDF;

– Word2Vec accuracy remains low (~58 % regardless of balance), indicating its limited effectiveness for classifying short text descriptions of errors in this dataset.

3. *Selected hyperparameters*

The following values were used to achieve optimal productivity:

– criterion = "entropy"

– max_depth = 79

– min_samples_leaf = 1

– min_samples_split = 79

The combination of a large tree depth and a split value of 79 ensured a balance between tree variability and overall ensemble consistency.

According to the classification reports obtained, the precision for all classes varies from 73 % to 100 % (Fig. 16), which indicates the model's high ability to correctly assign most samples to the appropriate categories.

The results for classes 4 and 5 are particularly indicative, where the model achieved 100 % recall, i.e., it was able to detect all cases that actually belong to these classes. Such indicators are considered excellent in the tasks of automated classification of text descriptions of errors.

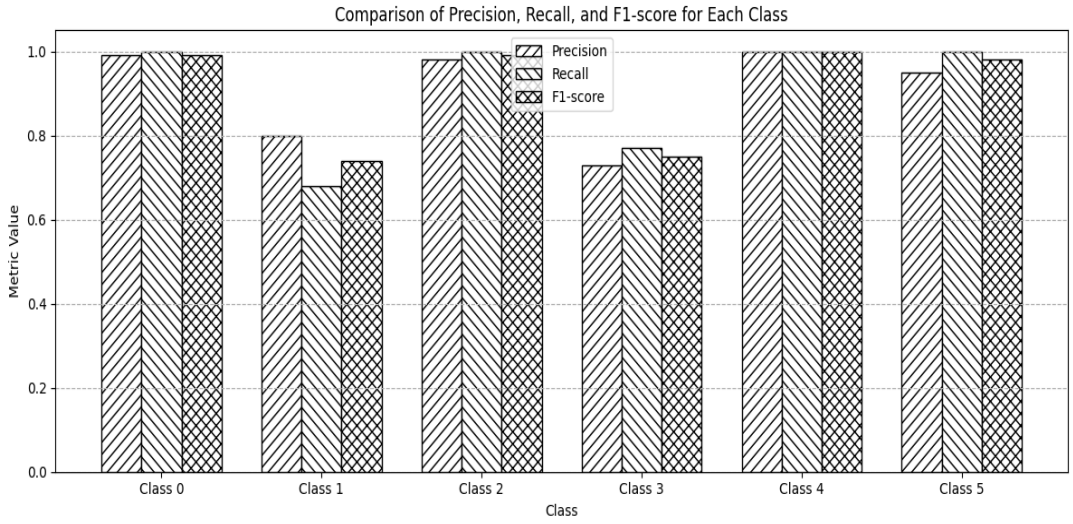


Fig. 16. Comparison of metrics for each class

Analysis of the F1-measure confirms that the generalized model is well balanced and shows no signs of overfitting or underfitting. F1-scores remain consistently high for most classes, indicating an effective combination of accuracy and completeness. It is important to note that all classes contributed approximately equally to the learning process, which ensured increased model stability and its ability to work on various types of data.

In this study, the LogisticRegression implementation was imported from the *sklearn.linear_model* library. The model was trained on pre-processed and vectorized data, after

which it was evaluated on a test set. All experimental scenarios – different vectorizations, hyperparameter optimization, and balanced/unbalanced sample analysis – were applied to logistic regression in the same way as to other classifiers.

Further analysis of the results allowed us to evaluate how well logistic regression can cope with the task of classifying software bug descriptions and how its productivity compares to Naive Bayes, Decision Tree, and Random Forest.

Within the scope of the experiments, logistic regression was tested under similar conditions as other classifiers – with different vectorization methods (BOW, TF-IDF, Word2Vec), as well as with tuned hyperparameters for BOW and TF-IDF. The generalized results of the model are shown in Figures 17, 18.

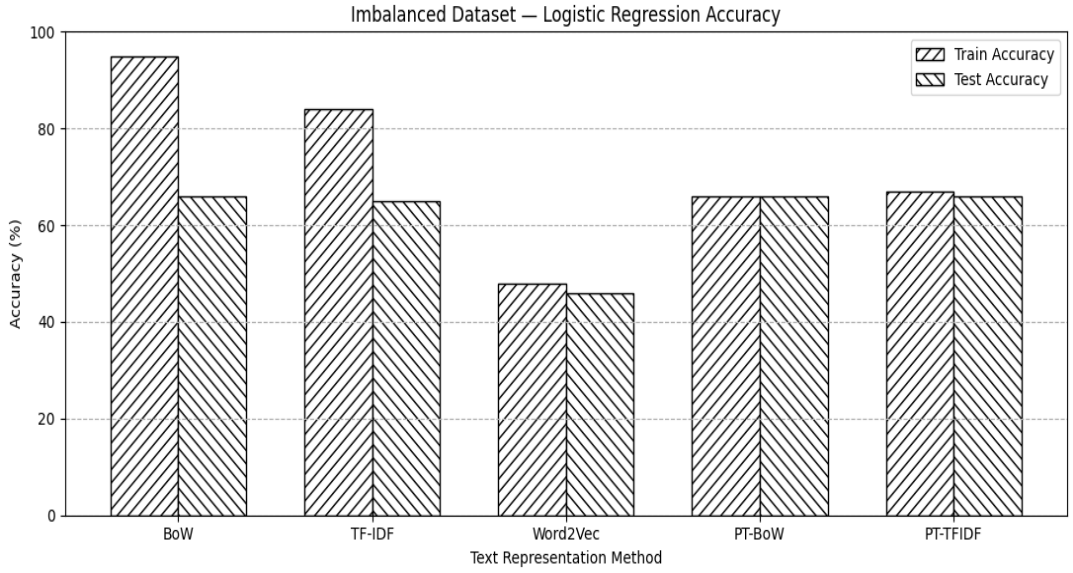


Fig. 17. Comparison of model accuracy for imbalanced samples

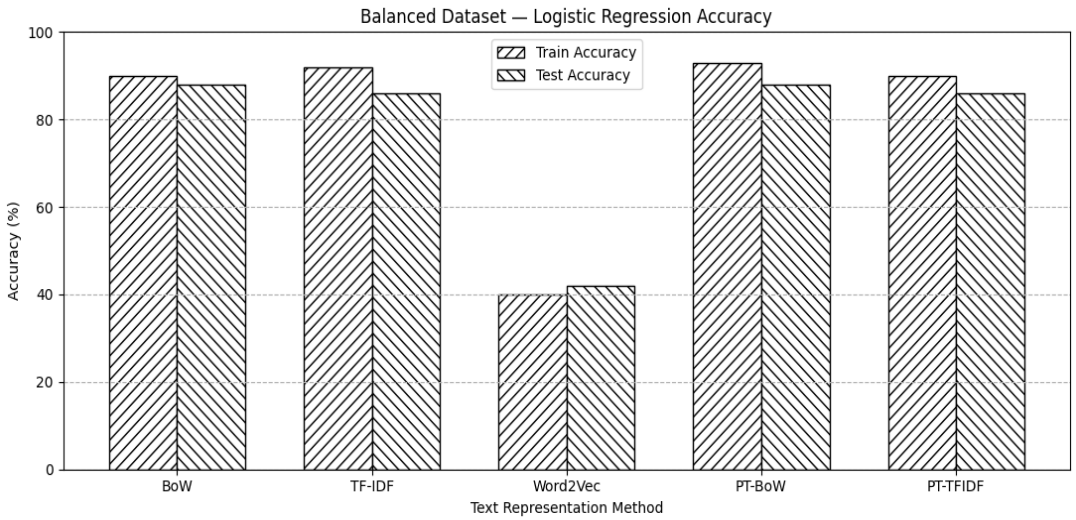


Fig. 18. Comparison of model accuracy for balanced samples

Logistic regression demonstrates lower productivity compared to other classifiers, in particular Random Forest and Naive Bayes. The main observations are as follows:

1. *Relatively low accuracy on an unbalanced dataset*

The model showed test accuracy in the range of 46–66 % for most vectorization methods, and even below 50 % for Word2Vec, indicating the difficulties of logistic regression in conditions of uneven class distribution and high variability of text descriptions.

2. *Improved results after data balancing*

For a balanced dataset, the productivity of logistic regression improved:

The best values were obtained for PT–BOW, where the test accuracy was 88.27 % and the training accuracy was 93.37 %.

This indicates that logistic regression is sensitive to class imbalance and can work much more effectively after preliminary sample correction.

3. *Low efficiency of Word2Vec*

Word2Vec showed the worst results among all vectorization methods: Train: 40.12 % – Test: 42.15 %

Since logistic regression is based on linear class separation, Word2Vec semantic vectors probably did not provide sufficient discriminatory information in this case.

4. *Hyperparameters used*

To improve model productivity, the following parameters were set:

– C = 10

– solver = "newton-cg"

The parameter C=10 reduces regularization, allowing the model to better adapt to the data, while the newton-cg optimization algorithm is effective for multi-class tasks.

The obtained metric values (Fig. 19) indicate that the logistic regression model demonstrates stable classification quality for most error categories.

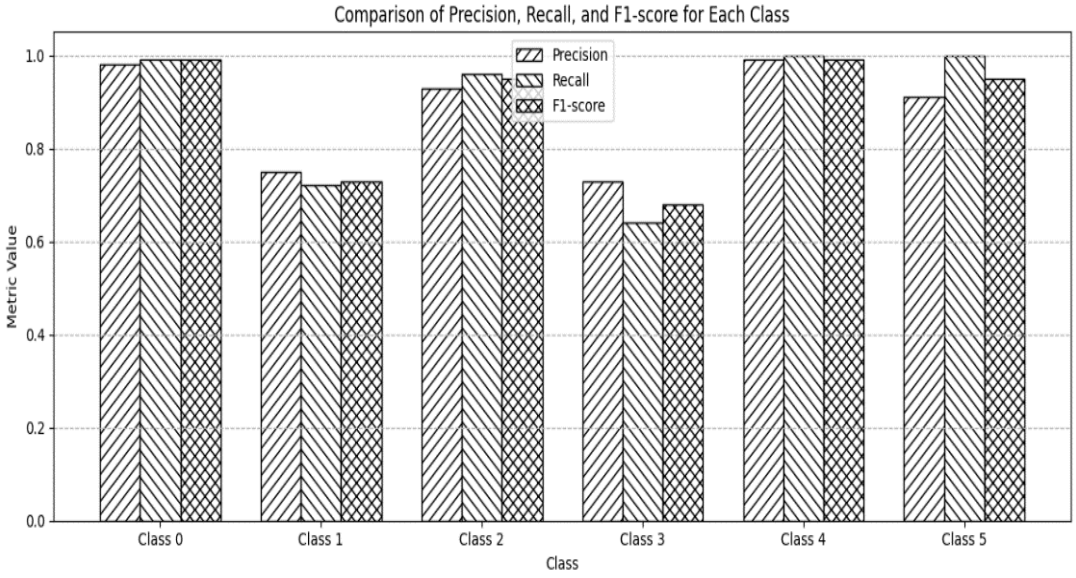


Fig. 19. Comparison of metrics for each class

In particular, precision ranges from 73 % to 99 %, which indicates the model's high ability to correctly recognize samples belonging to the corresponding classes. Classes 4 and 5 stand out in particular, for which the model achieved 100 % recall, i.e., it was able to detect all real instances of these categories without omissions. Such indicators are important for tasks where it is critical to minimize the loss of important or rare defects.

The F1-measure values confirm that the model does not suffer from overfitting or underfitting. The F1-score remains high and balanced for most classes, indicating a harmonious balance between accuracy and completeness. The absence of significant failures in any of the categories demonstrates that the model generalizes the data adequately and does not reorient itself to individual classes.

It is also important to note that all classes made a relatively equal contribution to the training of the model. This indicates that the training process was well balanced and that the preprocessing and sample balancing methods made it possible to avoid the dominance of certain categories. This result is critical for practical application, as it ensures stable forecasting in a variety of real-world bug report scenarios.

Overall, logistic regression, despite its relatively lower accuracy in some configurations, demonstrates satisfactory and interpretable results, making it useful as a base model in automated software bug classification systems.

For a generalized comparison of the results, a Train/Test graph (Fig. 20) of the accuracy of all models was constructed, which clearly demonstrates that Random Forest outperforms other approaches in key metrics (see Train/Test comparison graph).

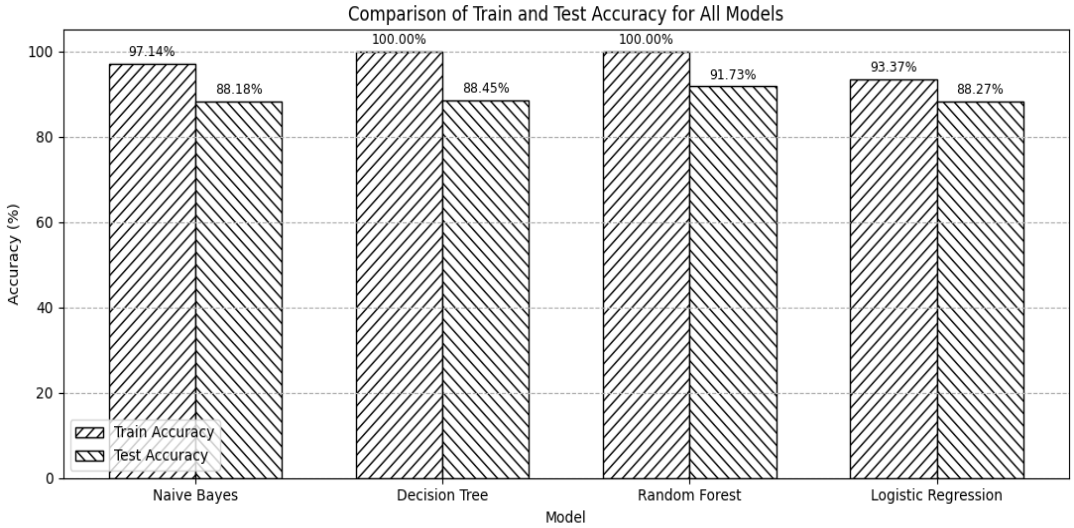


Fig. 20. Comparison of model accuracy on Train/Test data

Based on a comparison of all models, their accuracy, resistance to sample imbalance, ability to generalize complex textual features, and effectiveness after optimization, the best model in this study is Random Forest. It provides the highest test accuracy (91.73 %), demonstrates no overfitting, performs consistently on all experimental sets, and provides an optimal compromise between productivity, reliability, and the necessary flexibility. Random Forest

is recommended as the primary model for building automated bug report classification systems in SaaS environments.

The productivity of an error classification system in cloud applications is a determining factor in its suitability for use in real-world workloads. Since the speed of processing bug reports directly affects the timeliness of incident response, the study evaluated two key characteristics: machine learning model training time and inference time, i.e., the time required to classify a single new bug report. The results are visualized in the corresponding graphs, allowing for a clear comparison of the effectiveness of different models.

The training time analysis showed significant differences between classifiers (Figure 21). The Naive Bayes model demonstrated the shortest training time (≈0.12 s), which is expected given its linear nature and lack of complex parameter optimization. Logistic Regression also demonstrated high performance, with a training time of about 0.95 s. Decision Tree, on the other hand, was more resource-intensive (≈1.8 s), which is associated with the need to build a deep hierarchy of nodes. The longest training time was observed for Random Forest (≈7.5 s), since the model consists of an ensemble of trees and requires significant computational resources to form an optimal set of decisions. Despite this, Random Forest showed the highest accuracy among all tested models (91.7 %), which justifies its use in the presence of the appropriate infrastructure.

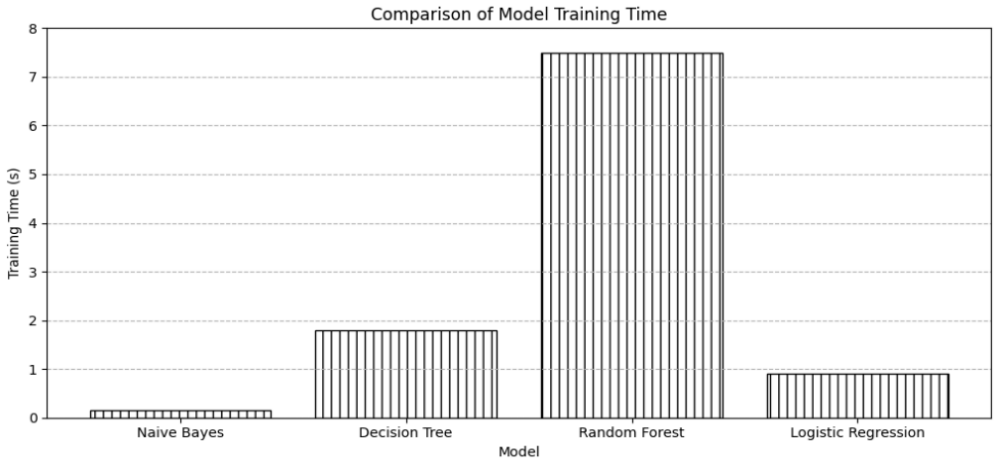


Fig. 21. Comparison of model training times

Inference time analysis (Figure 22) showed that all models provide almost instantaneous classification, which is an important condition for integration into real cloud services. The processing time for a single query was ≈0.002 s for Naive Bayes, ≈0.004 s for Logistic Regression, and ≈0.006 s for Decision Tree, while Random Forest showed a slightly slower inference time (≈0.015 s). However, even the maximum value remains within a few milliseconds, which allows you to classify hundreds or thousands of bug reports per second and maintain system operation in near real-time mode.

The system scalability assessment confirmed that the proposed architecture is effective for the cloud environment. Using TF-IDF as the main vectorization method ensures linear scaling of computational costs as data volumes increase and allows large text message streams to be processed without a significant increase in preprocessing time. In addition, the Random

Forest model naturally supports horizontal scaling, since tree construction can be parallelized across multiple computing nodes, which is especially important when processing large datasets or during regular model retraining.

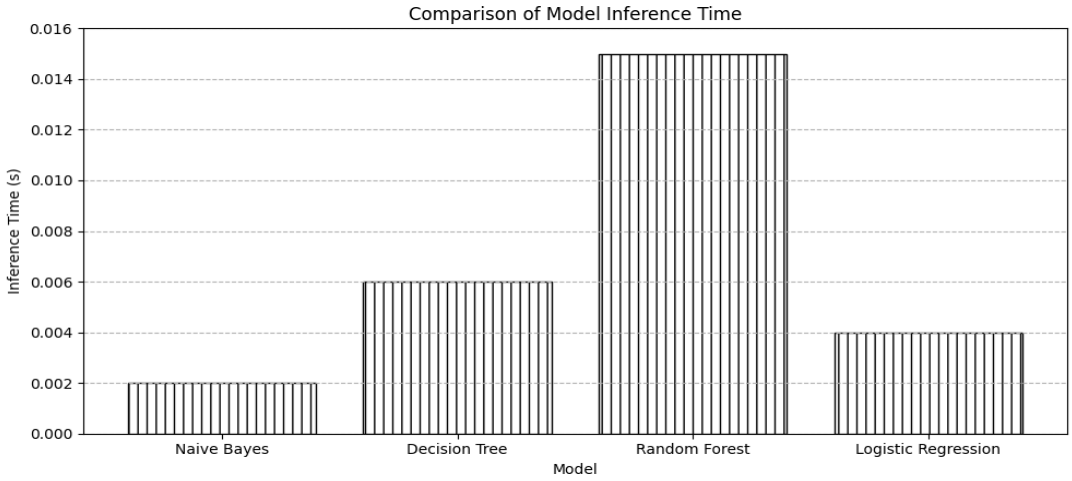


Fig. 22. Comparing of models inference time

Separating the training and inference processes also provides a significant advantage for scaling. Model training is performed on a separate computing node, which allows the model to be updated without stopping the classification service, maintaining the continuity of the system. This is consistent with typical cloud service operating practices and ensures the system's resilience to changes in data volumes and the intensity of error reports.

This study prioritizes errors in SaaS-type cloud applications based on their frequency of occurrence in the available dataset. The results are visualized in Figure 23, which shows the distribution of errors by type, taking into account their relative share.

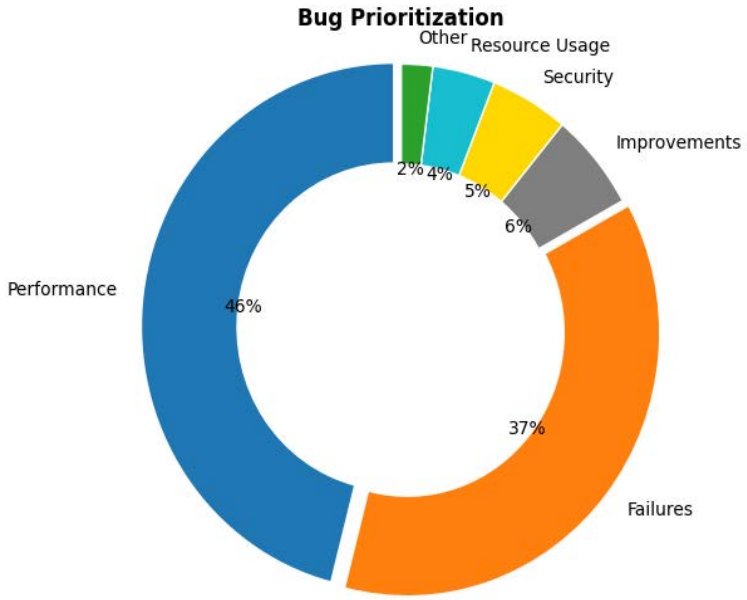


Fig. 23. Bug prioritization

The analysis of the graph shows that the most common bugs are related to system productivity, accounting for 46 % of the total. Such bugs are usually related to response delays, inefficient use of computing resources, or slow data processing. They have a significant impact on the quality of user interaction with the application and require immediate response from developers. In second place in terms of frequency are crash errors, which account for 37 %. This type of error is critical because it causes the application to suddenly stop working, which can lead to data loss and disruption of service continuity.

Other types of errors are less common but no less important in terms of ensuring system reliability. For example, system functionality improvements, security issues, compilation errors, and resource usage issues together account for about one-fifth of all cases. In particular, security errors (5 %) require special attention, as even single cases of such failures can have serious consequences for users and companies, especially in the context of data privacy and regulatory compliance. The smallest share – only 2 % – is accounted for by errors related to resource usage. This type usually manifests itself in conditions of large data processing volumes or excessive load on the computing infrastructure, which, in turn, may indicate the need to optimize the architecture or scale resources.

The approach proposed in this work has certain limitations. One of the key challenges is the temporal degradation of the model: with the development of cloud computing applications, the emergence of new features, or changes in the data structure, the effectiveness of pre-trained models may decline. To maintain high classification accuracy, it is necessary to regularly update and retrain models on current data. This will ensure that the models correspond to the current state of software systems and allow for high productivity to be maintained in real-world conditions.

Conclusions

Classifying bugs in cloud computing applications using machine learning methods is an important task that combines technical components with a deep understanding of the subject area. The study confirms the feasibility and effectiveness of using machine learning algorithms for automated detection and classification of various types of bugs in a cloud environment. Thanks to their ability to process large amounts of data generated by cloud systems, these algorithms enable real-time bug prediction with high accuracy.

The proposed approach contributes to a significant increase in the efficiency of bug detection and elimination processes, which, in turn, reduces the risk of downtime and increases the reliability of cloud services. The application of such solutions can be an important component in ensuring the stable operation of critical information systems, especially in the context of growing business dependence on cloud technologies.

In addition, a promising area for further research is the implementation of transfer learning methods, which allow the knowledge gained from previous bug classification tasks to be used to improve the accuracy of models in new, similar contexts. The application of domain adaptation methods is also relevant, especially in cases where there is a discrepancy between the distributions of training and test data.

References

1. Gupta, M., Gupta, D., Rai, P. (2024), "Exploring the Impact of Software as a Service (SaaS) on Human Life", *EAI Endorsed Transactions on Internet of Things*. DOI: https://doi.org/10.4108/eetiot.4821
2. Zhao, Y., Damevski, K., Chen, H. (2023), "A systematic survey of just-in-time software defect prediction", *ACM Computing Surveys*, Vol. 55, No. 10, P. 1–35. DOI: https://doi.org/10.1145/3567550
3. Bugayenko, Y., Bakare, A., Cheverda, A., Farina, M., Kruglov, A., Plaksin, Y., Succi, G. (2023), "Prioritizing tasks in software development: A systematic literature review", *PLOS ONE*, Vol. 18, No. 4, Article e0283838. DOI: https://doi.org/10.1371/journal.pone.0283838
4. Shiri Harzevili, N., Boaye Belle, A., Wang, J., Wang, S., Jiang, Z. M., Nagappan, N. (2024), "A systematic literature review on automated software vulnerability detection using machine learning", *ACM Computing Surveys*, Vol. 57, No. 3, P. 1–36. DOI: https://doi.org/10.1145/3699711
5. Tabianan, K., Velu, S., Ravi, V. (2022), "K-means clustering approach for intelligent customer segmentation using customer purchase behavior data", *Sustainability*, Vol. 14, No. 12, Article 7243. DOI: https://doi.org/10.3390/su14127243
6. Waqar, A. (2020), "Software Bug Prioritization in Beta Testing Using Machine Learning Techniques", *Journal of Computer Science*, Vol. 1, P. 24–34. DOI: https://doi.org/10.17509/jcs.v1i1.25355
7. Huda, S., Liu, K., Abdelrazek, M., Ibrahim, A., Alyahya, S., Al-Dossari, H., Ahmad, S. (2018), "An Ensemble Oversampling Model for Class Imbalance Problem in Software Defect Prediction", *IEEE Access*, Vol. 6, P. 24184–24195. DOI: https://doi.org/10.1109/ACCESS.2018.2817572
8. Goyal, A., Sardana, N. (2019), "Empirical Analysis of Ensemble Machine Learning Techniques for Bug Triaging", *Proceedings of the Twelfth International Conference on Contemporary Computing (IC3)*, P. 1–6. DOI: https://doi.org/10.1109/IC3.2019.8844876
9. Gupta, A., Sharma, S., Goyal, S., Rashid, M. (2020), "Novel XGBoost Tuned Machine Learning Model for Software Bug Prediction", *Proceedings of the International Conference on Intelligent Engineering and Management (ICIEM)*, P. 376–380. DOI: https://doi.org/10.1109/ICIEM48762.2020.9160152
10. Ahmed, H. A., Bawany, N. Z., Shamsi, J. A. (2021), "CaPBug-A Framework for Automatic Bug Categorization and Prioritization Using NLP and Machine Learning Algorithms", *IEEE Access*, Vol. 9, P. 50496–50512. DOI: https://doi.org/10.1109/ACCESS.2021.3069248
11. Tabassum, N., Alyas, T., Hamid, M., Saleem, M., Malik, S. (2022), "Hyper-convergence storage framework for ecocloud correlates", *Computers, Materials & Continua*, Vol. 70, No. 1, P. 1573–1584. DOI: https://doi.org/10.32604/cmc.2022.019389

Received (Надійшла) 06.11.2025
Accepted for publication (Прийнята до друку) 09.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Shmatko Oleksandr – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Electronic Computers, Kharkiv, Ukraine; e-mail: oleksandr.shmatko2@nure.ua; ORCID ID: https://orcid.org/0000-0002-2426-900X

Gamayun Igor – Doctor of Sciences (Engineering), Professor, National Technical University "Kharkiv Polytechnic Institute", Professor at the Department of Software Engineering and Management Intelligent Technologies, Kharkiv, Ukraine; e-mail: Ihor.Hamaiun@khpi.edu.ua; ORCID ID: https://orcid.org/0000-0003-2099-4658

Kolomiitsev Oleksii – Honored Inventor of Ukraine, Doctor of Sciences (Engineering), Professor, National Technical University "Kharkiv Polytechnic Institute", Professor at the Department Computer Engineering and Programming, Kharkiv, Ukraine; e-mail: alexus_k@ukr.net; ORCID ID: https://orcid.org/0000-0001-8228-8404

Шматко Олександр Віталійович – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри електронних обчислювальних машин, Харків, Україна.

Гамаюн Ігор Петрович – доктор технічних наук, професор, Національний технічний університет "Харківський політехнічний інститут", професор кафедри програмної інженерії та інтелектуальних технологій управління, Харків, Україна.

Коломійцев Олексій Володимирович – заслужений винахідник України, доктор технічних наук, професор, Національний технічний університет "Харківський політехнічний інститут", професор кафедри комп'ютерної інженерії та програмування, Харків, Україна.

ГІБРИДНА МОДЕЛЬ МАШИННОГО НАВЧАННЯ ДЛЯ КЛАСИФІКАЦІЇ ПРОГРАМНИХ ПОМИЛОК У ХМАРНИХ *SAAS*-ЗАСТОСУНКАХ

У сучасних хмарних обчислювальних середовищах забезпечення стабільності та надійності програмних застосунків є одним із ключових чинників ефективної роботи інформаційних систем. Значну частину збоїв у таких системах спричиняють програмні помилки (баги), які ускладнюють експлуатацію та знижують продуктивність сервісів. Традиційні методи ручного аналізу звітів про помилки є трудомісткими, тому необхідно розробити інтелектуальні підходи до автоматизованої класифікації та пріоритизації помилок із використанням методів машинного навчання. **Мета статті** – підвищення точності класифікації типів програмних помилок у хмарних застосунках. **Завдання дослідження:** формування повного конвеєра автоматизованого оброблення даних баг-репортів, що охоплює всі етапи – від попереднього очищення до побудови моделі класифікації. **Методологічна основа дослідження** полягає у використанні методів оброблення природної мови (NLP), техніки SMOTE для балансування вибірки, класичних алгоритмів машинного навчання, а також процедури оптимізації гіперпараметрів *RandomizedSearchCV*. Якість моделей оцінюється на основі стандартних класифікаційних метрик, таких як *accuracy, precision, recall* та *F1-score*, що забезпечує комплексний і об'єктивний аналіз отриманих результатів. **Результати дослідження.** Розроблено гібридну модель для автоматизованої класифікації помилок, що охоплює етапи збирання, попереднього оброблення, векторизації та моделювання даних. Проведено порівняльний аналіз точності чотирьох алгоритмів машинного навчання – наївного баєсівського класифікатора, дерева рішень, випадкового лісу й логістичної регресії – із використанням різних методів векторизації (*Bag-of-Words, TF-IDF, Word2Vec*). Для підвищення точності класифікації застосовано техніку балансування даних SMOTE. Експериментальні дослідження на реальному наборі даних із хмарного середовища продемонстрували, що модель *Random Forest* досягла найвищих показників точності – до 91,7 %. Результати підтверджують ефективність інтеграції алгоритмів машинного навчання в процеси аналізу й підтримки програмних продуктів у хмарних інфраструктурах. **Висновки.** Запропонований підхід забезпечує підвищення точності класифікації помилок у хмарних обчислювальних системах і може бути використаний у системах моніторингу, *DevOps*-платформах і засобах автоматизованого тестування. Результати дослідження є основою для подальшого розроблення інтелектуальних інструментів прогнозування й пріоритизації дефектів програмного забезпечення.

Ключові слова: класифікація помилок; хмарні обчислення; машинне навчання; *TF-IDF; Word2Vec*; випадковий ліс; автоматизація тестування.

Bibliographic descriptions / Бібліографічні описи

Shmatko, O., Gamayun, I., Kolomiitsev, O. (2025), "Hybrid machine learning model for classifying software bugs in SaaS cloud applications", *Management Information Systems and Devises*, No. 4 (187), P. 156–181. DOI: https://doi.org/10.30837/0135-1710.2025.187.156

Шматко О. В., Гамаюн І. П., Коломійцев О. В. Гібридна модель машинного навчання для класифікації програмних помилок у хмарних *SaaS*-застосунках. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 156–181.
DOI: https://doi.org/10.30837/0135-1710.2025.187.156

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.8 + 004.056.5 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.182

M. Glavchev, D. Hlavchev, V. Panchenko

EVALUATING THE EFFECTIVENESS OF OPEN-SOURCE SOLUTIONS FOR MONITORING AND LOAD BALANCING IN MICROSERVICE APPLICATIONS

Subject of Research. Subject of the article is open-source solutions and their application for monitoring and load balancing in microservice applications operating in specialized computer systems. The research covers a wide range of tools, including metric collection systems, centralized logging, and distributed request tracing. Relevance of the work is determined by the constant growth in complexity of distributed architectures and the critical need for effective performance control (observability) and stable traffic distribution. **Goal.** The goal of the work is comprehensive empirical evaluation and comparison of key open-source monitoring and load balancing tools, specifically Prometheus/Grafana (for metrics), ELK stack (for logs), HAProxy, Nginx, Traefik (load balancers), as well as Istio and Linkerd (service mesh), with the goal of developing practical recommendations for designing and operating microservice systems. **Tasks.** The tasks are conduct analysis of popular open-source tools, define criteria for their effectiveness, create a test environment based on Kubernetes and conduct a series of load tests with various configurations, as well as perform quantitative assessment of key performance indicators, including latency, throughput, and resource utilization. **Methods.** Applied methods of systematic analysis, empirical modeling, and benchmarking. For objective comparison, load testing methods (baseline and stress scenarios) were used in a Kubernetes cluster. Key evaluation criteria included request processing latency, throughput, and resource overhead of the tools themselves. **Result.** The obtained results confirm that open-source solutions are capable of providing high-level observability and effective load balancing in specialized computer systems, while remaining a cost-effective alternative to commercial products. The study identified strengths and weaknesses of each tool, allowing for informed selection based on specific project requirements. **Conclusions.** Confirmed the ability of open-source tools to effectively provide observability and load management in specialized computer systems, remaining a cost-effective alternative to commercial products. The conclusions made allow for the formulation of practical recommendations for designing and operating microservice applications with a focus on stability and performance. The research results can be used in making architectural decisions for distributed systems of various scales.

Keywords: load balancing; microservice; monitoring; open-source; service mesh; specialized computer system.

Introduction

Problem statement. Modern information systems are increasingly being designed based on microservice architecture. This approach involves dividing a complex application into a set of relatively independent services, each of which performs clearly defined functions and can be deployed separately. The use of microservices increases development flexibility, scalability, and system resilience, which is especially important in specialized computer systems, such as telecommunications platforms, financial solutions, or industrial IoT complexes.

At the same time, microservice architecture creates new challenges. The significant complexity of the structure necessitates effective performance monitoring, tracking dependencies between services, and load balancing between different application instances. In the event of high loads or uneven distribution of requests, there is a risk of performance

degradation or even failure of individual components, which can negatively affect the functioning of the entire system.

Traditionally, monitoring and load balancing tasks have been solved using commercial tools, which often have closed code and significant licensing costs. However, in recent years, an ecosystem of open-source solutions has been actively developing, providing a wide range of functions and not inferior in capabilities to many commercial counterparts. The most well-known of these tools include Prometheus and Grafana (metrics collection and visualization), ELK stack (report collection and analysis), HAProxy, Nginx, and Traefik (load balancing at L4/L7 levels), as well as specialized infrastructure-level solutions for facilitating communication between services, service mesh – Istio and Linkerd, which integrate directly into the Kubernetes environment and provide flexible management of network flows.

The relevance of the study is due to the need to verify the effectiveness of these solutions in the real conditions of specialized computer systems. For practical application, it is important not only to compare capabilities theoretically, but also to evaluate key performance indicators experimentally:

– average request processing delay and delays at the p95/p99 percentile levels;

– system throughput (number of requests processed per second);

– resource usage (central processing unit – CPU, random access memory – RAM) by the tools themselves;

– ease of integration into existing microservice architectures.

Analysis of recent studies and publications. In operational microservice systems, metrics, reports (logs), and tracing are most often combined; at the same time, the priority metrics are latency (including p95/p99), throughput, and resource consumption (CPU/RAM) of monitoring tools. A practical picture of the implementation of such approaches is provided by an empirical study of practitioners [1], which identifies log management, exceptions, and load balancing as subjects of constant performance monitoring among standard practices. A systematic review of monitoring tools (71 OSS solutions), their capabilities and limitations, is presented in the JSS review [2], which is useful for forming criteria for selecting an observability stack for a specific specialized computer system (SCS). In terms of log analytics, the focus on the ACM Computing Surveys [3] review allows us to align the choice of ELK (Elasticsearch, Kibana & Logstash)/related tools with modern methods of parsing, anomaly detection, and datasets for validation. These three sources form the methodological basis of the monitoring section of our study.

The classic CACM paper on Borg–Omega–Kubernetes [4] justifies the evolution to containerized load management and the importance of resource isolation for combining latency-sensitive services with batch tasks. An early experimental comparative analysis of microservices in containers and in a monolith [5] confirmed the suitability of containers for low overhead and rapid scaling. For experimental research, it is important to have a formal basis in the resource models of Kubernetes systems, which provide resource consumption forecasts and help correlate the overhead costs of monitoring/mesh/LB with the performance of microservices. A recent JSS article on building such models is useful here [6].

When choosing open-source load balancers at the L4/L7 level (HAProxy, Nginx, Traefik, Envoy), pay attention to key parameters such as latency, throughput, and stability under high load.

Practical measurements in server WWW infrastructures showed similar results for HAProxy and Nginx under high loads, with the best performance seen in Linux Virtual Server (LVS approach) [7]. Some studies offer mathematical models for analyzing HAProxy performance, including MMAP/PH/M/N queues and Monte Carlo simulation with confirmation by measurements on a test bench [8]. For Kubernetes environments, Ingress controllers were additionally considered: a 2024 IEEE paper compares implementations and balancing algorithms specifically in a Kubernetes cluster, which directly coincides with our experimental setup [9–11]. Together, these sources outline the trade-offs between pure L4 performance and L7 policy flexibility for microservice applications.

Service mesh adds L7 routing, observability, and security policies (mTLS mutual authentication) to the system, but introduces latency and CPU/memory consumption. A detailed analysis of sidecar proxy overhead with measured effects on latency (increase to ~2.7× on certain configurations) and virtual central processing unit (vCPU) [12] shows that the amount of overhead depends heavily on the configuration (TCP proxying vs. protocol parsing) and workload. A comparison of Istio and Linkerd in edge conditions showed Linkerd to be more "edge-friendly" due to lower overhead [13]. A separate technical report on the impact of mTLS in different implementations (Istio, Ambient Istio, Linkerd, Cilium) helps to separate security and performance effects in experimental design [14]. Additionally, the performance of a mesh cluster also depends on related Kubernetes components – for example, the etcd configuration, which affects the overall behavior of the control plane and the application [15]. Collectively, these works set a corridor of expectations for observability overhead and security policies, which we will take into account when planning tests.

A separate class of work focuses on container platforms and their interaction with balancing/monitoring. Review and applied articles propose load balancing mechanisms in Docker-/Swarm-/Kubernetes environments for productive and resource-constrained scenarios, including big data and edge [16]. At the same time, the Ingress layer in Kubernetes remains an active field of comparison (implementations, algorithms, overhead), where recent IEEE results [9] allow the experiment to be enriched with practical configurations. Together with resource consumption models [6], this provides a methodologically correct correlation of observability, balancing, and performance in a microservice SCS.

The purpose of this article is to study the effectiveness of open-source solutions for monitoring and load balancing in microservice applications running on specialized computer systems.

To achieve this goal, the following tasks must be solved:

– review modern open-source tools in the field of monitoring and balancing;

– determine the criteria for evaluating their effectiveness in the context of microservice application performance;

– build an experimental environment using a test microservice application;

– conduct a series of load tests using different tools and record the results;

– perform a comparative analysis of the data obtained and formulate practical recommendations.

Thus, the work aims to bridge the gap between the theoretical capabilities of open-source tools and their actual effectiveness when used in SCS.

Presentation of the main material

1. Theoretical aspects

1.1. Microservice architecture and performance

Microservice architecture is one of the leading approaches to building modern distributed applications. A microservice is an independent software component that implements a separate business function and interacts with other components through well-defined interfaces, primarily via the API interface of the HTTP/gRPC remote procedure call system or asynchronous message queues. This architecture provides development flexibility, the ability to choose the optimal technologies for each service, as well as deployment isolation and scaling independence. For SCS, the microservice approach is particularly valuable because it allows computing resources to be distributed across subsystems and complex data flows to be managed efficiently.

In the context of microservice applications, system performance is determined by its ability to process the required volume of requests with minimal delays and optimal use of computing resources. Important performance characteristics include average and percentile request processing delays (latency), throughput, and the level of utilization of the processor, RAM, and network resources. In SCS, performance is considered not only as a quantitative characteristic, but also as stability under peak loads, which requires the ability to automatically scale and adapt to dynamic conditions.

To formalize the task of performance evaluation, we introduce the following mathematical definitions and notations, which will be used in the experimental part of the study.

Definition 1. Let the system consist of a set of microservices $S = \{s_1, s_2, ..., s_n\}$, where each service s_i is characterized by a vector of performance parameters $P_i = (L_i, T_i, R_i)$, where L_i is the request processing delay, T_i is the throughput, and R_i is the resource consumption.

Definition 2. The request processing delay L is defined as the time interval between the moment the client sends the request $t_{request}$ and the moment the response is received $t_{response}$:

$$L = t_{response} - t_{request}. \quad (1)$$

Since the distribution of delays in microservice systems usually has a long tail, using only the average value is insufficient for an objective assessment. Therefore, the study uses percentile analysis, which allows us to evaluate the stability of the system for the vast majority of users.

Definition 3. The average delay is calculated as the arithmetic mean of all measurements:

$$\bar{L} = \left(\frac{1}{n}\right) \times \Sigma_{i=1^n} L_i. \quad (2)$$

Definition 4. The delay percentile p (e.g., $p95$, $p99$) is defined as the delay value that is not exceeded for p percent of all requests:

$$L_{p95} = L_{([0,95 \times n])}. \quad (3)$$

Definition 5. The throughput of a system T is defined as the number of successfully processed requests per unit of time:

$$T = N_{requests} / t_{interval}. \quad (4)$$

Definition 6. The resource overhead coefficient $O_{resource}$ characterizes the additional consumption of resources (CPU or RAM) by a monitoring or balancing tool relative to the baseline:

$$O_{resource} = \left(R_{with\_tool} - R_{baseline}\right) / R_{baseline} \times 100\%. \quad (5)$$

For a comprehensive assessment of the tool's effectiveness, an integrated indicator is proposed that takes into account all key characteristics with corresponding weighting coefficients:

$$E = \alpha \times \left(1/L_{p99}\right) + \beta \times T + \gamma \times \left(1/O_{resource}\right), \quad (6)$$

where α, β, γ are weighting coefficients determined by the priorities of a specific specialized computer system (SCS). For example, for latency-critical systems, $\alpha > \beta > \gamma$, while for resource-constrained edge systems, $\gamma > \alpha \approx \beta$ is appropriate.

1.2. Monitoring and observability

Monitoring is another important component, which involves the systematic collection, aggregation, and analysis of data on the functioning of the system. In microservice architectures, it is key to achieving observability, i.e., the ability to draw conclusions about the internal state of the system based on its external manifestations. Effective monitoring allows for the timely detection of performance degradation, service interaction failures, or security-threatening attacks.

According to the concept of the "three pillars of observability" [1–3], effective monitoring of microservice systems is based on three complementary components: metrics (quantitative indicators of the system's state), logs (event records), and tracing (tracking the path of a request through the system).

Based on an analysis of scientific sources [1–6] and the practical needs of the SCS, a system of criteria was developed to evaluate the effectiveness of open-source monitoring and load balancing tools (Table 1).

Table 1. *Criteria for evaluating the effectiveness of tools*

Criterion	Description	Metric	Priority
Latency	Request processing time from receipt to response	p95, p99 (ms)	High
Bandwidth	Number of requests processed per unit of time	RPS	High
Resource overhead	Additional CPU and RAM consumption by tools	% CPU, MB RAM	Medium

Continuation of the table 1

Criterion	Description	Metric	Priority
Scalability	Horizontal scaling capability	Number of nodes	Medium
Integration complexity	Time and effort required to implement the tool	Hours/days	Low
Stability	Ability to maintain performance under load	σ (deviation)	High

Source: developed by the authors

2. Open-source solutions

Among the most common open-source monitoring tools, it is worth highlighting Prometheus for collecting metrics, Grafana for visualization, ELK stack for centralized logging, as well as service mesh components, in particular Istio and Linkerd, which provide distributed request tracing. For a systematic analysis of existing solutions, open-source tools were classified according to their functional purpose (Table 2).

Table 2. *Classification of open-source tools according to their functional purpose*

Category	Tool	Main function	OSI Level
Monitoring (metrics)	Prometheus	Metrics collection and storage	Application (L7)
	Grafana	Visualization and alerting	Application (L7)
Monitoring (logs)	Elasticsearch	Indexing and search	Application (L7)
	Logstash	Collection and transformation	Application (L7)
	Kibana	Log visualization	Application (L7)
Load balancing	HAProxy	High-performance proxy	Transport/Application (L4/L7)
	Nginx	Web server and reverse proxy	Application (L7)
	Traefik	Cloud-native edge router	Application (L7)
Service Mesh	Istio	Full-featured mesh	Application (L7) + mTLS
	Linkerd	Lightweight mesh	Application (L7) + mTLS

Source: developed by the authors based on [2, 7–14]

2.1. Monitoring tools

Prometheus + Grafana. Prometheus implements a pull model for collecting metrics, in which the server periodically polls application endpoints. This approach provides centralized control over the frequency of collection and allows new services to be automatically discovered through the service discovery mechanism [17, 18].

Prometheus stores data in its own time series database (TSDB), optimized for fast recording and aggregation of metrics.

The PromQL query language provides powerful tools for aggregating and analyzing metrics. For example, to calculate the 95th percentile of HTTP request latency over the last 5 minutes, the following query is used:

histogram_quantile(0.95, rate(http_request_duration_seconds_bucket[5m]))

This query directly corresponds to the Lp95 metric defined in formula (3). Grafana provides visualization of collected metrics through a flexible dashboard system and supports threshold-based alert configuration.

ELK Stack (Elasticsearch, Logstash, Kibana) provides centralized log management and deep event analytics [3]. Elasticsearch uses an inverted index for fast full-text search, allowing millions of records to be analyzed in seconds. Logstash acts as a data aggregator and transformer, supporting over 200 plugins for various sources. Kibana provides an interface for log visualization and analysis. In terms of log analytics, the focus on the ACM Computing Surveys log review [3] allows ELK to be aligned with modern parsing methods, anomaly detection, and validation datasets.

A comparative overview of the monitored tools is provided in Table 3.

Table 3. *Comparative characteristics of monitoring tools*

Parameter	Prometheus + Grafana	ELK Stack
Data type	Time series metrics	Unstructured and structured logs
Collection model	Pull (HTTP scraping)	Push (via Beats/Logstash)
Query language	PromQL	Lucene / KQL
Storage	TSDB (local, up to 15 days)	Elasticsearch (distributed)
K8s integration	Native (service discovery)	Via Filebeat/Metricbeat
Alerting	Alertmanager (flexible rules)	Watcher / ElastAlert
Resource capacity	Low–medium	High (especially Elasticsearch)
License	Apache 2.0 / AGPL	Elastic License 2.0

Source: developed by the authors based on [2, 3, 17, 18]

2.2. Load balancers

When choosing open-source load balancers at the L4/L7 level (HAProxy, Nginx, Traefik, Envoy), pay attention to key parameters such as latency, throughput, and stability under high load. Practical measurements in server WWW infrastructures showed similar results for HAProxy and Nginx under high loads, with the best performance observed in Linux Virtual Server (LVS approach) [7].

A key aspect of the comparison is the load balancing algorithms, which directly affect the uniformity of request distribution and, accordingly, latency metrics. The main algorithms are formalized as follows.

Round Robin (RR) – cyclic distribution of requests between servers with equal weight:

$$server_{next} = \left(server_{current} + 1\right) \bmod N, \tag{7}$$

where N is the number of available servers.

Least Connections (LC) – selection of the server with the fewest active connections:

$$server_{next} = arg\ min_{i \in \{1..N\}} \left(connections_i\right). \tag{8}$$

HAProxy is optimized for maximum performance and supports operation at the L4 (TCP) and L7 (HTTP) levels. According to studies [7, 8], HAProxy demonstrates the lowest latency among the load balancers considered when operating at the L4 level. Some studies offer

mathematical models for analyzing HAProxy performance, including MMAP/PH/M/N queues and Monte Carlo simulations confirmed by test bench measurements [8].

Nginx is positioned as a universal web server with reverse proxy and load balancing functions. Its asynchronous event-based architecture ensures efficient processing of a large number of concurrent connections. Advanced SSL termination and caching capabilities make it ideal for web-oriented applications [9].

Traefik is designed as a cloud-native edge router with native support for Docker and Kubernetes. Key advantages include automatic service discovery and dynamic configuration without rebooting, as well as built-in integration with Let's Encrypt for automatic TLS certificate management [10]. For Kubernetes environments, Ingress controllers were additionally considered: the 2024 IEEE paper compares implementations and balancing algorithms specifically in the Kubernetes cluster, which directly coincides with our experimental setup [9–11].

A comparative analysis of the studied balancers is presented in Table 4.

Table 4. *Comparative characteristics of load balancers*

Parameter	HAProxy	Nginx	Traefik
OSI level	L4 / L7	L7	L7
Algorithms	RR, LC, Source, URI	RR, LC, IP-hash	WRR, Mirroring
SSL termination	Yes	Yes (extended)	Yes (Let's Encrypt)
Health checks	TCP, HTTP, Agent	Passive, Active	Docker/K8s native
K8s integration	Ingress Controller	Ingress Controller	Native (CRDs)
Configuration	Static (file)	Static (file)	Dynamic (auto-discovery)
Optimization for	Max. throughput	Web services	DevOps/Cloud-native
License	GPL v2	BSD-2	MIT

Source: developed by the authors based on [7–11]

2.3. Service Mesh solutions

Service mesh adds L7 routing, observability, and security policies (mTLS mutual authentication) to the system, but introduces latency and CPU/memory consumption. A detailed analysis of sidecar proxy overhead with measured effects on latency (increase to ~2.7× on certain configurations) and virtual central processing unit (vCPU) [12] shows that the amount of overhead depends heavily on the configuration (TCP proxying vs. protocol parsing) and workload.

Istio is the most feature-rich solution based on the Envoy proxy. It provides complete control over traffic through VirtualService and DestinationRule, including canary deployments, circuit breaking, and fault injection [12]. Research [14] has shown that enabling mTLS increases latency by 15–20%, which corresponds to formula (5) for calculating overhead. The performance of a mesh cluster also depends on related Kubernetes components, such as etcd configuration, which affects the overall behavior of the control plane and the application [15].

Linkerd is positioned as a lightweight service mesh with an emphasis on simplicity and minimal overhead. The linkerd2-proxy, written in Rust, ensures low resource consumption. A comparison of Istio and Linkerd in edge conditions showed Linkerd to be more

"edge-friendly" due to lower overhead costs [13]. A separate technical report on the impact of mTLS in different implementations (Istio, Ambient Istio, Linkerd, Cilium) helps to separate security and performance effects in experimental design [14].

A comparative analysis of Istio and Linkerd is presented in Table 5.

Table 5. *Comparative characteristics of Service Mesh solutions*

Parameter	Istio	Linkerd
Proxy architecture	Envoy (C++)	linkerd2-proxy (Rust)
mTLS	Yes (configured)	Yes (default)
Traffic management	Full (VirtualService, DestinationRule)	Basic (TrafficSplit)
Observability	Metrics, Logs, Traces	Metrics, Traces (golden metrics)
Latency overhead	2–3 ms (up to ~2.7× without mTLS)	<1 ms
CPU overhead (sidecar)	~100–150 mCPU	~20–50 mCPU
RAM overhead (sidecar)	~50–100 MB	~10–20 MB
Complexity of implementation	High	Low
Recommended environment	Enterprise, complex policies	Edge, resource-constrained

Source: developed by the authors based on [12–15]

2.4. Summarizing the results of theoretical analysis

A separate class of works focuses on container platforms and their interaction with balancing/monitoring. Review and applied articles propose load balancing mechanisms in Docker/Swarm/Kubernetes environments for productive and resource-constrained scenarios, including big data and edge [16]. Together with resource consumption models [6], this provides a methodologically correct correlation of observability, balancing, and performance in a microservice SCS.

Based on the analysis, a matrix of tool compatibility with typical usage scenarios (Table 6) has been formed, which allows for a reasoned approach to the selection of solutions for specific SCS.

Table 6. *Matrix of tool compatibility with usage scenarios*

Scenario	HAProxy	Nginx	Traefik	Istio	Linkerd
Highly loaded systems	**+++**	++	+	+	++
Web applications (e-commerce)	++	**+++**	++	+	+
DevOps/CI-CD environments	+	+	**+++**	++	++
Enterprise with security requirements	+	+	+	**+++**	++
Edge/IoT with limited resources	++	+	+	–	**+++**

Note: +++ – best solution; ++ – good; + – acceptable; – not recommended
Source: developed by the authors

Thus, the work aims to bridge the gap between the theoretical capabilities of open-source tools and their actual effectiveness when applied in SCS.

The theoretical analysis allows us to formulate hypotheses about the expected results of the experimental study:

1. HAProxy will demonstrate the lowest latency values (p95, p99) among load balancers due to optimization for L4 operation;

2. Istio with mTLS enabled will have the highest resource overhead due to additional encryption operations;

3. Linkerd will provide the optimal balance of functionality and overhead for edge environments;

4. Traefik will demonstrate the greatest ease of integration into the Kubernetes environment thanks to its auto-discovery mechanism.

These hypotheses are tested in the experimental part of the study, the methodology and results of which are presented in the next section.

3. Testing the microservice application

To test the effectiveness of open-source solutions in the field of monitoring and load balancing, a test environment was created that replicates the typical operating conditions of an SCS with a microservice architecture.

Description of the environment. The test application consisted of a set of microservices implemented in Docker containers and deployed in a Kubernetes cluster. The architecture included an authentication service, a user management service, a business logic service, and a database accessible through a separate data access service.

To reproduce a real-world load scenario, the application was wrapped in an API gateway that provided external access to the system.

Tool integration. The following open-source solutions were gradually integrated into the cluster:

– Prometheus + Grafana – for collecting and visualizing metrics (latency, throughput, CPU, RAM, network I/O);

– ELK stack – for centralized log collection and analysis;

– HAProxy, Nginx, and Traefik – as external load balancers at the L4 and L7 levels;

– Istio and Linkerd – as service mesh implementations for internal load balancing and traffic monitoring.

Testing methodology. Apache JMeter and k6 tools were used to generate load, which allowed us to reproduce various user activity scenarios. Testing was conducted in two modes:

1. A stable environment with a uniform load over a long period of time (baseline tests).

2. Stress tests with sharp peaks in request intensity, simulating peak periods of operation.

Metrics and evaluation criteria. System performance was evaluated based on the following indicators:

– latency (average, p95, p99);

– throughput (number of requests per second);

– resource consumption (CPU, RAM) by monitoring and balancing tools;

– operational efficiency, i.e., ease of tool integration and speed of configuration.

Experiment scenarios. For each tool, a series of tests was conducted with varying parameters:

– in the case of HAProxy/Nginx/Traefik, round-robin, least-connections, and IP-hash algorithms were compared;

– in the case of Istio/Linkerd, the impact of enabling mTLS, retry mechanisms, and rate limiting policies was evaluated;

– for Prometheus/Grafana and ELK stack – the overhead of data collection at different intervals and log volumes was analyzed.

Expected results. The experiment was expected to yield quantitative data that would allow comparing the effectiveness of different approaches to balancing and monitoring. In particular, we planned to determine:

– which tools provide the least overhead under high load;

– which configurations provide the optimal balance between performance and management flexibility;

– how much the implementation of a service mesh affects delays compared to classic load balancers.

Fig. 1 shows a diagram of the experimental microservices environment, which shows the user, load balancers (HAProxy/Nginx/Traefik), API Gateway, services, database, as well as the integration of monitoring tools (Prometheus + Grafana, ELK) and service mesh (Istio/Linkerd).

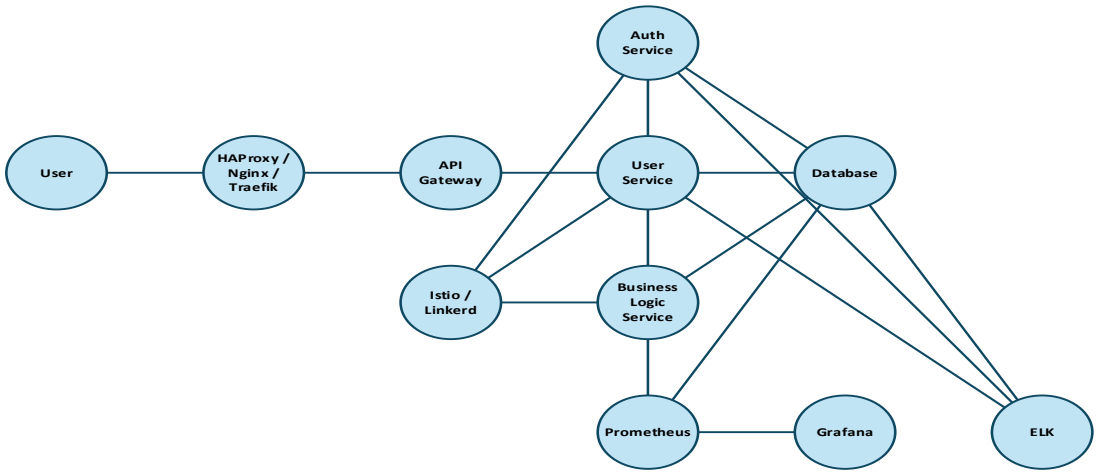


Fig. 1. Diagram of the experimental microservices environment

Source: developed by the authors

During experimental research, a series of tests was performed in two scenarios: stable load (Baseline) and stress tests with periodic intensity peaks (more than 600 measurements were performed).

1) Scenario characteristics

In baseline mode, the system demonstrated an average intensity of ≈1500 requests per second (RPS) with fluctuations of no more than ±5%. The average p95 latency was 90±3 ms, p99 – 130±5 ms, which

corresponds to normal operating mode. During stress tests, the intensity ranged from 1400 to 3500 RPS, and p95 delays increased by 20–40 ms, p99 – by 30–70 ms, depending on the balancing tool (Fig. 2).

This confirms the typical pattern of performance degradation under peak loads. This behavior allowed us to evaluate the ability of the tools to maintain stability during short-term overloads.

The graph in Fig. 3 shows that in the baseline scenario, p95 remained stable (≈90 ms), while in the stress scenario, it increased to 120–140 ms at peaks. This indicates a typical response of microservice architecture to uneven load, where some services become a "bottleneck".

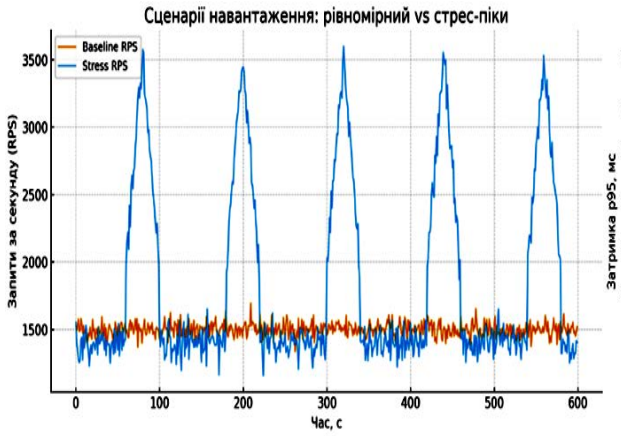


Fig. 2. RPS load scenarios over time
Source: developed by the authors

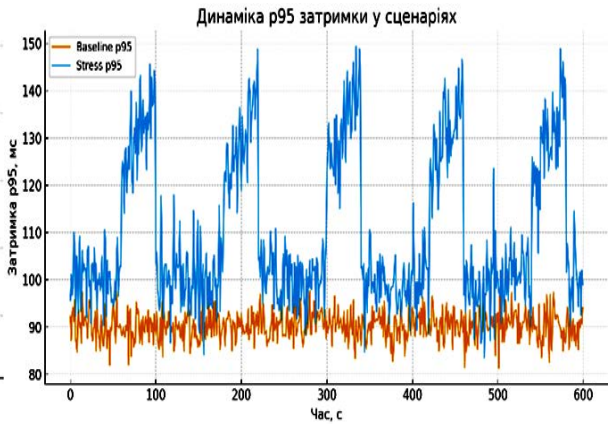


Fig. 3. Dynamics of p95 delays (baseline vs stress)
Source: developed by the authors

2) Comparison of tools

All tools showed differences in both average values (Fig. 4) and stability of results (Fig. 5) when averaged over a 10-minute test.

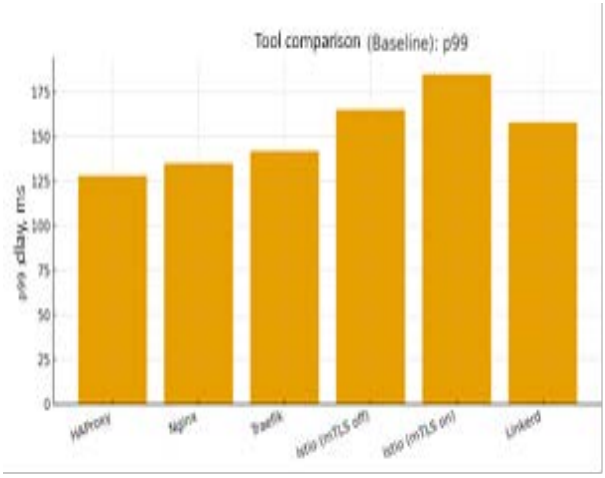


Fig. 4. Comparison of tools (Baseline p99)
Source: developed by the authors

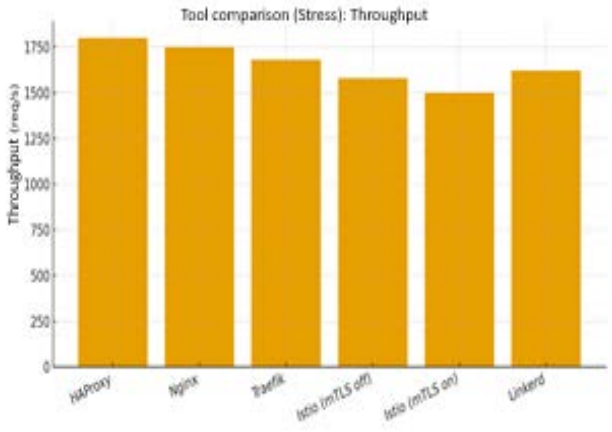


Fig. 5. Comparison of tools by throughput in a stress scenario
Source: developed by the authors

HAProxy showed the best results for latency (p95 = 88±2 ms; p99 = 128±4 ms in baseline) and maintained the lowest values even under stress load (p95 = 108±4 ms). Throughput remained stable (≈1800 RPS at peaks).

Nginx was slightly inferior to HAProxy, especially in terms of p99 (135±5 ms), but provided greater flexibility in routing, which is important for web-oriented services.

Traefik showed greater overhead: p99 in baseline – 142±5 ms, in stress tests – 190±7 ms, but stood out for its ease of integration with Kubernetes and automation of TLS certification.

Linkerd showed the best balance between performance and resource consumption (baseline p95 ≈ 105±3 ms; p99 ≈ 158±6 ms), making it suitable for edge environments.

Istio showed higher latencies (baseline p95 = 110±4 ms; p99 = 165±6 ms), and even higher with mTLS enabled (p95 = 125±5 ms; p99 = 185±8 ms).

However, its functionality (detailed access policies, security, monitoring) significantly exceeds that of alternatives.

The effectiveness of open-source tools for monitoring and load balancing depends on the usage scenario and the requirements of the specialized computer system.

3) Resource overhead costs

The data confirm (Table 7) that in the baseline scenario, the difference between the instruments is less noticeable, but during stress loading, the differences become significant.

Table 7. *Summary results by tools*

Tool	Baseline p95 (ms)	Baseline p99 (ms)	Baseline RPS	Stress p95 (ms)	Stress p99 (ms)	Stress RPS	CPU overhead (%)	RAM overhead (MB)
HAProxy	88	128	1500	108	165	1800	6.5	180
Nginx	92	135	1470	116	178	1750	7.2	220
Traefik	95	142	1420	122	190	1680	8.0	260
Istio (mTLS off)	110	165	1350	140	225	1580	12.5	420
Istio (mTLS on)	125	185	1300	165	260	1500	16.0	520
Linkerd	105	158	1380	135	210	1620	10.5	360

Source: developed by the authors

The overhead graphs (Fig. 6; 7) show:

– the lowest CPU and RAM consumption was recorded in HAProxy (≈6.5% CPU, 180 MB RAM) and Nginx (≈7.2% CPU, 220 MB RAM);

– Traefik required more resources (≈8% CPU, 260 MB RAM), which is explained by the dynamic nature of the configuration;

– Linkerd had an average overhead (≈10.5% CPU, 360 MB RAM);

– Istio created the most load, especially with mTLS (≈16% CPU, 520 MB RAM), which can be critical in resource-constrained environments.

No tool is universal; the choice should be made based on the balance between performance, functionality, and resource capabilities of a particular architecture.

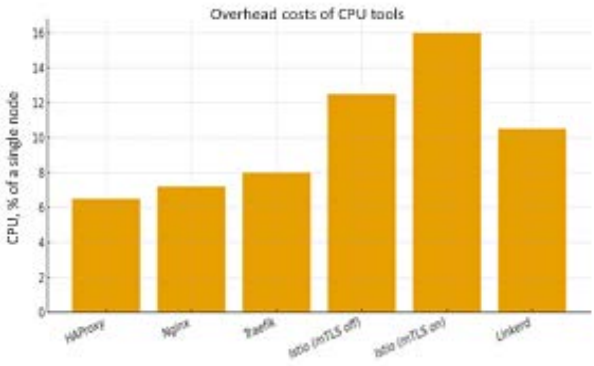


Fig. 6. Overhead CPU
Source: developed by the authors

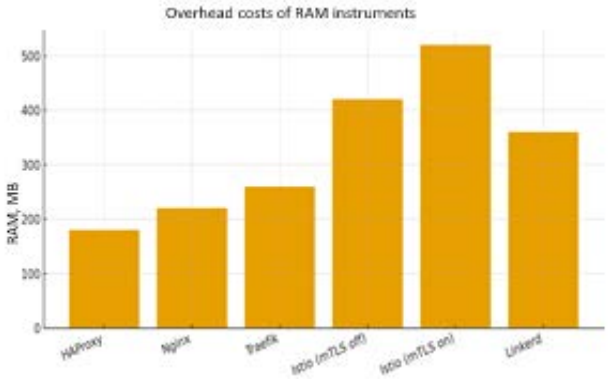


Fig. 7. Overhead RAM
Source: developed by the authors

Conclusions

Thus, a study was conducted that confirmed the relevance of using open-source solutions for monitoring and load balancing in microservice applications operating in SCS. The increasing complexity of architectural solutions and the demand for high performance require tools that can simultaneously provide observability, flexible traffic management, and efficient resource utilization.

Analysis of theoretical aspects showed that monitoring and balancing remain key elements in supporting the performance of microservice environments. The use of the Prometheus + Grafana combination allows for the effective collection and visualization of metrics, while the ELK stack provides centralized log management and deep event analytics. The HAProxy, Nginx, and Traefik balancing tools demonstrated different strengths: from maximum performance (HAProxy) to flexible routing (Nginx) and ease of integration into DevOps processes (Traefik). The Istio and Linkerd service mesh solutions showed advanced traffic and security management capabilities, but at the cost of higher resource overhead.

The experimental part of the study made it possible to evaluate the real effectiveness of using various open-source tools in microservice architectures. In stable load scenarios, HAProxy demonstrated the lowest latency and highest stability, confirming its focus on high performance and optimization for handling large numbers of concurrent connections. This balancer demonstrated the ability to maintain stable throughput even under heavy load conditions, ensuring low p95 and p99 latency values.

In situations with sharp load spikes, HAProxy also remained the most stable, while Istio with mTLS enabled showed a noticeable degradation in performance.

This is due to the additional overhead of traffic encryption and access policy management, which significantly impacted latency and system throughput. Nginx and Traefik took intermediate positions, providing a compromise between performance and functionality.

Nginx proved to be convenient for web-oriented systems, as it provides flexible routing capabilities and supports SSL termination. Traefik, in turn, stood out for its ease of integration with Kubernetes and automatic TLS certificate management, making it attractive for DevOps environments.

Linkerd deserves a special mention, as it showed the best balance of performance and resource efficiency compared to Istio. Although it has slightly less functionality, it provides lower latency and less overhead, making it suitable for resource-constrained environments or edge systems.

The results of the experimental study made it possible not only to evaluate the performance of open-source tools, but also to formulate a number of practical recommendations for their application in real-world conditions.

For highly loaded specialized systems, where the primary task is to minimize latency and ensure maximum stability during peak loads, HAProxy is the most appropriate choice. This balancer showed the best p95 and p99 performance and confirmed its ability to maintain high throughput without critical degradation.

For web-oriented services that serve end users and require complex routing rules, Nginx is the optimal solution. Its ability to perform

SSL termination and support for advanced HTTP routing scenarios makes it convenient for e-commerce, portals, or online services.

For DevOps environments focused on containerization and CI/CD process automation, Traefik is the most effective. Thanks to its dynamic integration with Kubernetes and Docker, it greatly simplifies traffic management.

Finally, Linkerd has proven itself suitable for resource-constrained environments where a balance between performance and resource efficiency is important. At the same time, Istio should be used in large enterprise systems with high requirements for security, access control policies, and comprehensive monitoring, even despite its higher overhead.

Thus, the results of the study confirm that open-source tools are capable of providing effective monitoring and load balancing in microservice architectures. The choice of a specific solution should be based on a balance between performance, functionality, and resource constraints of a specialized computer system.

The analysis outlined three key areas for further research aimed at improving the efficiency of SCS:

1. Modeling and minimizing service mesh overhead. It is critically important to develop detailed mathematical models for accurate prediction of sidecar proxy overhead (Istio, Linkerd) in latency-critical environments, taking into account dynamic configurations (mTLS, L7 filters), as well as comparisons with new architectures such as Ambient Mesh.

2. Development of adaptive load balancing algorithms driven by metrics (Metric-Aware Balancing). It is necessary to implement dynamic routing that uses real-time service quality metrics from Prometheus, ensuring the transition to service quality balancing.

3. Application of machine learning (ML) methods for proactive monitoring and scaling management. ML integration will enable the creation of models to predict performance degradation and detect anomalies in logs (ELK stack) before a failure occurs, which is critical for improving the resilience of modern SCS.

References

1. Waseem, M., Liang, P., Shahin, M., Di Salle, A., Márquez, G. (2021), "Design, Monitoring, and Testing of Microservices Systems: The Practitioners' Perspective", *Journal of Systems and Software*, No. 182, P. 111061. DOI: https://doi.org/10.1016/j.jss.2021.111061
2. Giamattei, L., Guerriero, A., Pietrantuono, R., et al. (2024), "Monitoring tools for DevOps and microservices: A systematic grey literature review", *Journal of Systems and Software*, No. 208, P. 111906. DOI: https://doi.org/10.1016/j.jss.2023.111906
3. He, S., Zhu, J., He, P., Lyu, M.R. (2021), "A Survey on Automated Log Analysis for Reliability Engineering", *ACM Computing Surveys*, No. 54(6), P. 123. DOI: https://doi.org/10.1145/3460345
4. Burns, B., Grant, B., Oppenheimer, D., Brewer, E., Wilkes, J. (2016), "Borg, Omega, and Kubernetes", *Communications of the ACM*, No. 59(5), P. 50–57. DOI: https://doi.org/10.1145/2890784
5. Amaral, M., Polo, J., Carrera, D., et al. (2015), "Performance Evaluation of Microservices Architectures using Containers", *IEEE NCA*, P. 93–100. DOI: https://doi.org/10.1109/NCA.2015.49
6. Turin, G., Borgarelli, A., Donetti, S., Damiani, F., Johnsen, E.B., Tapia Tarifa, S.L. (2023), "Predicting Resource Consumption of Kubernetes Container Systems Using Resource Models", *Journal of Systems and Software*, No. 203, P. 111750. DOI: https://doi.org/10.1016/j.jss.2023.111750
7. Dymora, P., Mazurek, M., Sudek, B. (2021), "Comparative Analysis of Selected Open-Source Solutions for Traffic Balancing in Server Infrastructures Providing WWW Service", *Energies*, No. 14(22), P. 7719. DOI: https://doi.org/10.3390/en14227719
8. Sokolov, A. (2024), "Application of Queueing Theory to Investigation of HaProxy Load Balancer Performance Characteristics", DCCN 2023, CCIS 2129, Springer, P. 89–100. DOI: https://doi.org/10.1007/978-3-031-61835-2_7
9. Rathi, G., Amin, S. (2024), "Performance Analysis of Different Ingress Controllers Within the Kubernetes Cluster", *IEEE ICITEICS 2024*. DOI: http://dx.doi.org/10.1109/ICITEICS61368.2024.10625280
10. Khamdani, A.R., Muslikh, A.R., Affandi, A.S. (2025), "Comparative Analysis of Performance and Efficiency of Load Balancing Algorithms on Ingress Controller", *Jurnal Teknik Informatika*, Vol. 6, No. 1, P. 453–468. DOI: https://doi.org/10.52436/1.jutif.2025.6.1.4040
11. Nguyen, N., Kim, T. (2020), "Toward Highly Scalable Load Balancing in Kubernetes Clusters", *IEEE Communications Magazine*, No. 58(7), P. 78–83. DOI: https://doi.org/10.1109/MCOM.001.1900660
12. Zhu, X., She, G., Xue, B., et al. (2023), "Dissecting Overheads of Service Mesh Sidecars", ACM SoCC '23: Proceedings of the 2023 ACM Symposium on Cloud Computing, P. 142–157. DOI: https://doi.org/10.1145/3620678.3624652
13. Elkhatib, Y., Salmon, B., Harkous, H., et al. (2023), "An Evaluation of Service Mesh Frameworks for Edge Systems", *EdgeSys '23: Proceedings of the 6th International Workshop on Edge Systems, Analytics and Networking*, P. 19–24. DOI: https://doi.org/10.1145/3578354.3592867
14. Bremler-Barr, A., Lavi, O., Naor, Y., Rampal, S., Tavori, J. (2024), "Performance Comparison of Service Mesh Frameworks: the mTLS Test Case", arXiv (Tech. Report). DOI: https://doi.org/10.48550/arXiv.2411.02267

15. Larsson, L., Tärneberg, W., Klein, C., Elmroth, E., Kihl, M. (2020), "Impact of etcd Deployment on Kubernetes, Istio, and Application Performance", *Software: Practice and Experience*. DOI: https://doi.org/10.1002/spe.2885
16. Singh, N., Tanwar, S., Gupta, R., Kumar, N. (2023), "Load balancing and service discovery using Docker Swarm for big data applications in microservice architecture", *Journal of Cloud Computing*, No. 12, P. 77. DOI: https://doi.org/10.1186/s13677-022-00358-7
17. Jani, Y. (2024), "Unified Monitoring for Microservices: Implementing Prometheus and Grafana for Scalable Solutions", *Journal of Artificial Intelligence*, Machine Learning and Data Science, No. 2(1), P. 848–852. DOI: http://dx.doi.org/10.51219/JAIMLD/yash-jani/206
18. Elrad, M.D. (2025), "Prometheus & Grafana: A Metrics-focused Monitoring Stack", *Journal of Computer Allied Intelligence*, No. 3(3), P. 28–39. DOI: https://doi.org/10.69996/jcai.2025015

Received (Надійшла) 07.11.2025
Accepted for publication (Прийнята до друку) 03.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Glavchev Maksym – PhD (Economic Sciences), Associate Professor, National Technical University "Kharkiv Polytechnic Institute", Professor at the Department of Computer Engineering and Programming, Kharkiv, Ukraine; e-mail: Maksym.Glavchev@khpi.edu.ua; ORCID ID: https://orcid.org/0000-0001-9670-9118

Hlavchev Dmytro – PhD, National Technical University "Kharkiv Polytechnic Institute", Associate Professor at the Department of Computer Engineering and Programming, Kharkiv, Ukraine; e-mail: Dmytro.Hlavchev@khpi.edu.ua; ORCID ID: https://orcid.org/0000-0003-4248-4819

Panchenko Volodymyr – National Technical University "Kharkiv Polytechnic Institute", Senior Lecturer at the Department of Computer Engineering and Programming, Kharkiv, Ukraine; e-mail: Volodymyr.Panchenko@khpi.edu.ua; ORCID ID: https://orcid.org/0000-0003-3364-3398

Главчев Максим Ігорович – кандидат економічних наук, доцент, Національний технічний університет "Харківський політехнічний інститут", професор кафедри "Комп'ютерна інженерія та програмування", Харків, Україна.

Главчев Дмитро Максимович – доктор філософії, Національний технічний університет "Харківський політехнічний інститут", доцент кафедри "Комп'ютерна інженерія та програмування", Харків, Україна.

Панченко Володимир Іванович – Національний технічний університет "Харківський політехнічний інститут", старший викладач кафедри "Комп'ютерна інженерія та програмування", Харків, Україна.

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ *OPEN-SOURCE*-РІШЕНЬ ДЛЯ МОНІТОРИНГУ Й БАЛАНСУВАННЯ НАВАНТАЖЕННЯ В МІКРОСЕРВІСНИХ ЗАСТОСУНКАХ

Предметом дослідження є *open-source*-рішення та їх упровадження для моніторингу й балансування навантаження в мікросервісних застосунках, що функціонують у спеціалізованих комп'ютерних системах. Дослідження охоплює широкий спектр інструментів, зокрема системи збору метрик, централізованого логування й розподіленого трасування запитів. Актуальність роботи зумовлена постійним зростанням складності розподілених архітектур і критичною потребою в ефективному контролі продуктивності (спостережуваності) й стабільному розподілі трафіку. **Мета роботи** – комплексне емпіричне оцінювання й порівняння ключових *open-source*-інструментів моніторингу й балансування навантаження, зокрема *Prometheus / Grafana* (для метрик), *ELK stack* (для логів), *HAProxy, Nginx, Traefik* (балансувальники), а також *Istio* та *Linkerd* (*service mesh*), з метою розроблення практичних рекомендацій щодо проєктування й експлуатації мікросервісних систем. **Завдання:** проаналізувати поширені *open-source*-інструменти; визначити критерії їх ефективності; створити тестове середовище на базі *Kubernetes*; провести серію навантажувальних тестів з різними конфігураціями; кількісно оцінити ключові показники продуктивності, зокрема затримку, пропускну здатність і використання ресурсів. **Методи дослідження.** Застосовано системний аналіз, емпіричне моделювання та бенчмаркінг. Для об'єктивного порівняння впроваджено методи навантажувального тестування (*baseline* та стрес-сценарії) в кластері *Kubernetes*. Ключові критерії оцінювання: затримка оброблення запитів, пропускна здатність і ресурсні накладні витрати самих інструментів. **Результати** підтверджують, що *open-source*-рішення здатні забезпечити високий рівень спостережуваності та ефективне балансування навантаження в спеціалізованих комп'ютерних системах, водночас залишаючись економічно вигідною альтернативою комерційним продуктам. Дослідження виявило переваги й недоліки кожного з інструментів, що дає змогу обґрунтовано підходити до їх вибору залежно від специфічних вимог проєкту. **Висновки:** підтверджено здатність *open-source*-інструментів ефективно забезпечувати спостережуваність і управління навантаженням у спеціалізованих комп'ютерних системах і водночас залишатися економічно вигідною альтернативою комерційним продуктам. Сформульовані висновки дають змогу розробити практичні рекомендації для проєктування й експлуатації мікросервісних застосунків із зосередженням на стабільності та продуктивності. Результати дослідження можуть бути впроваджені в прийнятті архітектурних рішень для розподілених систем різного масштабу.

Ключові слова: *open-source*; *service mesh*; мікросервіс; моніторинг; балансування навантаження; спеціалізована комп'ютерна система.

Bibliographic descriptions / Бібліографічні описи

Glavchev, M., Hlavchev, D., Panchenko, V. (2025), "Evaluating the effectiveness of open-source solutions for monitoring and load balancing in microservice applications", *Management Information Systems and Devises*, No. 4 (187), P. 182–199. DOI: https://doi.org/10.30837/0135-1710.2025.187.182

Главчев М. І., Главчев Д. М., Панченко В. І. Дослідження ефективності впровадження *open-source*-рішень для моніторингу й балансування навантаження в мікросервісних застосунках. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 182–199. DOI: https://doi.org/10.30837/0135-1710.2025.187.182

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.054:004.056.53 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.200

A. Abakumov, V. Kharchenko

COMBINED METHOD OF UAV CYBER ASSETS SECURITY ASSESSMENT BY USE OF PROCEDURES IMECA AND PENETRATION TESTING

Subject of the research is determined as methods and tools for security assessment of cyber assets of unmanned aerial vehicles (UAVs). The conclusion about a certain gap between theoretical risk assessment methods and practical penetration testing (PT) tools for UAV cyber assets was drawn based on an analysis of publications. Existing PT tools focus on attack reproduction, although they do not provide a methodology for the assessment of their impact in dynamic application environments. **Objective of the research** is to develop a combined method and elements of technology for reliable vulnerability detection with the possibility to verify the process and reason the selection of countermeasures. **Research tasks** include feasibility reasoning of combined use of various methods and tools for assessing UAV cybersecurity; development of models and an assessment method combining analytical and experimental procedures; practical execution of the method using a test environment. **Methods used:** Intrusion Modes and Criticality Analysis (IMECA) and PT. **Research results**: structure and sequence of combined cybersecurity assessment; functional IDEF0 model that ensures the completeness of the security assessment of UAV cyber assets and consists of the following stages: information gathering and vulnerability assessment, intrusion modes replication, IMECA-analysis (including preliminary and a posteriori IMECA-analysis), and countermeasure selection; deployed and tested test environment based on DVD simulator, capable of reproducing 80 % of priority intrusion modes. **Conclusions:** the combined method for security assessment of UAV cyber assets integrates IMECA procedures with PT practices, which allows to overcome the limitations of static risk assessment methods and isolated technical tests, creating a closed loop of verification and protection of on-board systems. Further research involves conducting a full-scale series of experiments for all identified intrusion modes, constructing a posteriori criticality matrices, and selecting countermeasures, as well as integration of the proposed method with the task of the functional safety assessment of unmanned systems.

Keywords: UAV cyber assets; IMECA-analysis; combined method; intrusion mode simulation; penetration testing.

1. Introduction

The rapid growth in the use of small unmanned aerial vehicles (UAVs), better known as "drones", can be observed in the following areas [1]:

- monitoring of hard-to-reach areas, real-time disaster relief;
- provision of services for smart cities;
- aerial photography, cinematography;
- precision agriculture (irrigation, phytopathology, soil mapping, farmer data analysis);
- traffic monitoring to obtain real-time information on road conditions;
- technical inspection and security monitoring of critical infrastructure facilities;
- reconnaissance, patrolling, logistics, demining, fire correction in combat conditions.

Since 2022, such UAVs have been actively used by the Armed Forces of Ukraine in countering Russia's full-scale aggression against Ukraine. Experience with the use of [2–6] UAVs has shown that high-tech devices can become targets of successful cyber attacks, not to mention civilian UAVs that are adapted for use in military operations. In combat zones, massive signal interference leads to significant UAV losses. For example, thousands of UAVs are shot down every month in Ukraine, mainly due to attacks that jam GPS navigation and control channels [7]. Zachary Kallenborn, a research associate at George Mason University, writes: "You can use them more aggressively because you don't care if they get lost" [2].

However, commercial UAVs require additional adaptation for military use, such as the use of special firmware that prevents detection by enemy aerial reconnaissance systems. In addition, it should be noted that software (SW) is constantly being updated, so outdated modified firmware becomes unusable in newer versions of UAVs.

These examples only emphasize the need for a systematic assessment of the security of cyber-physical components of UAVs [8], namely the analysis of potential threats and the identification of vulnerabilities (including zero-day vulnerabilities) before they are exploited by attackers, which could not only lead to the loss of the device itself during a mission, but also pose a threat to the lives of operators.

2. Analysis of recent studies and publications

An analysis of sources on the assessment of UAV cybersecurity (CS) and existing penetration testing (PT) methodologies adapted to the specifics of UAVs is provided based on a study of leading scientific databases, including Scopus, IEEE Xplore, and Google Scholar, published after 2020.

Table 1 provides the key ones, in the authors' opinion.

Table 1. *Classification of reviewed sources by research areas*

Research area	Sources
Threats, vulnerabilities of UAV components	[1, 5, 8–10]
Risk-oriented analysis and assessment of UAV vulnerabilities	[11–14]
UAV penetration testing	[15]

The paper [5] discusses numerous examples of malicious use of UAVs and analyzes possible attack vectors in civil and military scenarios. It shows that UAVs are vulnerable to a wide range of attacks and emphasizes the importance of implementing measures to detect and prevent them. [9] argues that UAV design problems are becoming increasingly apparent with the transition to mass military use, systematizes risks according to CIA aspects, and methods for analyzing vulnerabilities in UAV software.

The author of [10] presented a comprehensive classification of cyberattacks on UAVs, which can be used as a basis for threat modeling.

The work [11] considers the issue of assessing the design of multifunctional UAV fleets, identifies threats, vulnerabilities, and the potential consequences of cyberattacks, considering the characteristics of the interaction of system elements. The authors proposed a multi-level model of threats and attack scenarios, taking into account the functional distribution in the UAV infrastructure. A key methodological component of the study is the use of the Intrusion Modes and Effects Criticality (IMECA) method, which allows threats to be classified by level of criticality, the consequences of attacks to be modeled, and recommendations for improving system security to be formulated. [12] presents a model for the security of Internet of Drones (IoD) systems, focused on monitoring critical infrastructure. The authors analyzed so-called radio frequency vulnerabilities and applied IMECA to construct a risk matrix that considers the probability and criticality of intrusions.

Article [13] addresses the problem of the lack of a standardized method for assessing the overall security level of UAVs. The authors propose D3S (Drone Security Scoring System) – a methodology for assessing and assigning a security score to specific drone models based on the analysis of their components and resistance to attacks. In [14], the critical need for a structured methodology for assessing the security of UAVs is justified, given their integration into cyber-physical systems and the Internet of Things. The authors propose a systematic step-by-step approach that combines threat modeling, vulnerability analysis, assessment, and selection of appropriate countermeasures based on the assessment results.

Drone Attack Tool (DRAT) is a UAV PT framework proposed in [15] and designed to automate the process of finding vulnerabilities in commercial UAVs. The main goal of the tool is to reduce dependence on the operator's deep expertise and manual execution of complex attack scenarios by combining the necessary resources in a single graphical interface.

3. Research objectives and tasks

The analysis of sources indicates a gap between theoretical risk assessment methods and practical UAV penetration testing tools. Existing approaches to risk assessment (e.g., D3S) do not consider the actual exploitation of vulnerabilities in the dynamic conditions of UAV use. At the same time, existing PT tools (e.g., DRAT) focus on reproducing attacks but do not provide a methodology for assessing their impact.

Therefore, the purpose of the study is to develop a combined method and elements of technology for reliable vulnerability detection with the possibility of verifying the process and justifying the choice of countermeasures.

Research objectives:

- to justify the feasibility of the combined use of various methods and means of assessing the cybersecurity of UAVs;
- to develop an assessment method that includes analytical and experimental procedures;
- to test the method in practice using an appropriate test environment.

4. Materials and methods

One of the previous studies [16] analyzed several combinations of analytical and experimental methods for assessing security and cybersecurity (CS), considering such indicators as completeness, execution time, cost, and reliability of such assessment.

The analysis showed that the combination of the risk-oriented IMECA method [11, 12, 17] with PT [18, 19] best meets the requirements for CS analysis of unmanned intelligent systems (UIS).

Accordingly, the proposed combined method for assessing the security of UAV cyber assets using IMECA-analysis and PT procedures (hereinafter referred to as the combined method) can also be applied to assess the CS of unmanned aerial vehicles (UAVs). In the context of this study, UAV cyber assets are understood as a complex set of components that ensure the functioning of the system: a digital flight platform (including sensors and onboard software), a ground control station, as well as data and command transmission channels [20].

The essence of the combined method is to assess the security of UAV cyber assets by integrating IMECA-analysis procedures in preparatory and a posteriori forms with PT procedures to create an end-to-end cycle of verification of compliance with asset security requirements.

This approach determines its advantages, namely the reduction of residual vulnerability risks through improved test coverage and, as a result, the reduction of successful intrusion risks.

This work provides a detailed description of the processes of evaluating CS using the proposed method. To formalize and structurally describe the proposed method, the Integrated Definition for Function Modeling (IDEF0) methodology was chosen. This choice is due to the need to detail the processes of transforming input information (data on architecture, application scenarios, constraints) into results (risk assessments and a set of countermeasures) with a clear definition of control elements and necessary resources.

The hierarchical nature of IDEF0 provides the ability to step-by-step detail (decompose) complex assessment procedures, which allows maintaining the logical integrity of the method when integrating heterogeneous components: analytical IMECA-analysis and experimental PT.

4.1. General model of the combined method

The proposed combined method, presented in Figure 1 in the form of an IDEF0 context diagram, is based on a holistic process aimed at identifying, analyzing, confirming, and minimizing cyber risks.

At the entrance, information about the object of study is formed: the architecture and components of the UAV (I-ARCH), scenarios for its use (I-SCEN), as well as legal, operational, and technical restrictions (I-LIM).

The assessment process is implemented through a sequence of interrelated stages, which are provided by the corresponding set of tools (mechanisms) marked with red arrows in the diagram.

The process is strictly regulated by a set of control elements, which are shown in the diagram with blue arrows.

The result is a package of output data that includes a justified set of countermeasures (O-COUNTER), residual risk matrices (O-MATRIX), and a detailed report on the assessment of the impact on the UAV mission (O-IMPACT).

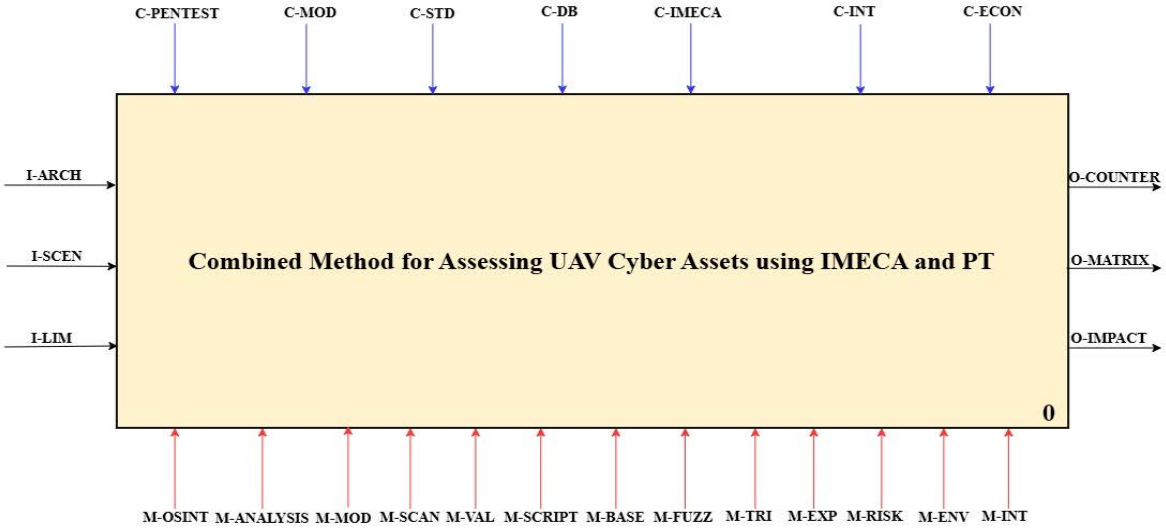


Fig. 1. Context diagram IDEF0 of the combined method (level A0)

4.2. Model decomposition and main stages of the combined method

Figure 2 shows the decomposed model of the combined method (level A1). It combines the classic stages of PT: information gathering and system analysis (1), assessment of known (2) and detection of unknown vulnerabilities (3), intrusion mode replication (5), as well as integrated IMECA-analysis in its preliminary (4) and a posterior (6) forms into a continuous process.

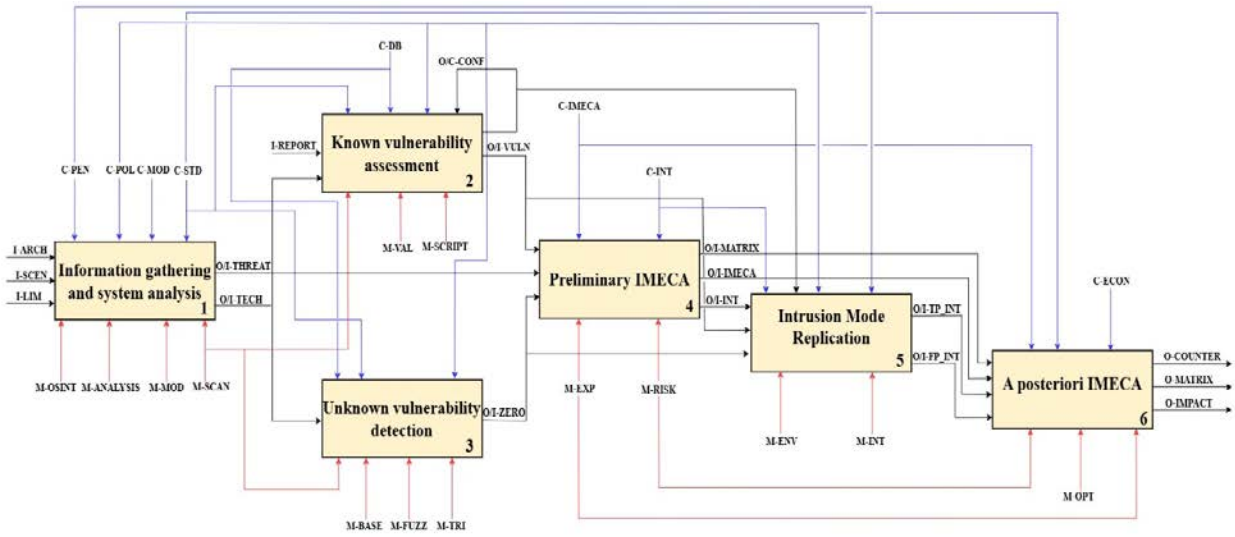


Fig. 2. Decomposed model of the combined method (level A1)

At the first stage of information gathering and system analysis, the research context is formed, and vulnerabilities and potential threats to the UAV under study are identified. Using OSINT tools (M-OSINT), automated scanners (M-SCAN), modeling tools (M-MOD), and analysis tools (M-ANALYSIS), a stack of UAV component technologies (O/I-TECH) and a list of potential threats (O/I-THREAT) are formed. The actions of researchers are guided by the C-PEN methodology, defined by C-MOD modeling frameworks, regulated by CS standards (C-STD), and governed by the terms of use of OSINT and automated scanning tools (C-POL), which impose additional technical and legal restrictions to avoid ethical violations. The further process branches into two parallel procedures: assessment of known vulnerabilities and detection of unknown vulnerabilities.

The goal of the second stage is to assess known vulnerabilities (O/I-VULN) by comparing UAV components with the vulnerability database (C-DB) and community reports (I-REPORT). At this stage, researchers actively use automated scanners (M-SCAN), vulnerability validation tools (M-VAL), and scripts to retrieve information from DBs (M-SCRIPT).

The functional purpose of the third stage is to identify zero-day vulnerabilities (O/I-ZERO) in UAV components that cannot be detected by automated means. Based on the input list of threats (O/I-THREAT) and the technology stack (O/I-TECH), researchers form a reference behavior model and analyze attack surfaces using basic and static analysis tools (M-BASE). Next, dynamic fuzzing (M-FUZZ) is performed to provoke failures, followed by triage and analysis of the root causes of anomalies (M-TRI) to confirm the criticality of the vulnerabilities found. The entire process is regulated by CS standards (C-STD) and terms of use (C-POL). Identified known and newly discovered vulnerabilities are consolidated and transferred to the preliminary IMECA-analysis input.

The purpose of the fourth stage is to analytically transform vulnerability data into an assessment of the risks to UAV missions. Based on the input lists of threats (O/I-THREAT), known vulnerabilities (O/I-VULN), and zero-day vulnerabilities (O/I-ZERO), attack vectors are mapped to intrusion modes. This process is regulated by the IMECA methodology and its assessment scales (C-IMECA), as well as intrusion models (C-INT). Using expert analysis (M-EXP) and risk assessment tools (M-RISK), a hypothesis about the level of danger is formed and a preliminary assessment of the probability, complexity of implementation, and severity of consequences is carried out. The result of this stage is the formation of preliminary criticality matrices (O-MATRIX) and prioritized attack scenarios (O/I-INT).

The fifth stage consists of practical verification of theoretical attack vectors in a controlled laboratory environment. Based on prioritized modes (O/I-INT), researchers reproduce scenarios using a specialized environment and equipment (M-ENV), including UAV simulator and PT operating systems, as well as appropriate intrusion tools (M-INT). The exploitation process is regulated by intrusion models (C-INT) and based on commonly accepted PT methodologies (C-PEN), such as PTES [21] or OSSTMM [22]. The result of this stage is an objective differentiation between successful confirmed intrusions (O/I-TP_INT) and refuted false positives (O/I-FP_INT).

The last stage consists of a final synthesis of the results and decisions on system protection. Based on empirical data on successful (O/I-TP_INT) and refuted (O/I-FP_INT) intrusions,

as well as the initial matrix (O/I-MATRIX), the criticality of threats is reassessed. The key mechanism of this stage is optimization algorithms (M-OPT), which allow the selection of countermeasures to be automated. The process is managed taking into account cost-effectiveness criteria (C-ECON), which ensure that costs are minimized while achieving the required level of security. The result of the work is an updated criticality matrix (O-MATRIX), an impact assessment report (O-IMPACT), and a final set of recommended countermeasures (O-COUNTER), which guarantees an acceptable level of residual risk.

For a deeper understanding of the algorithm of actions when applying the combined method, a detailed description of each stage of the presented IDEF0 model is given below.

4.3. Details of the stages of the combined method

4.3.1. Information gathering and system analysis

Figure 3 shows the decomposition of the information gathering and system analysis stage. The model details the process by breaking it down into four functional blocks: passive reconnaissance (1), active reconnaissance (2), system components mapping (3), and identification of potential threats (4).

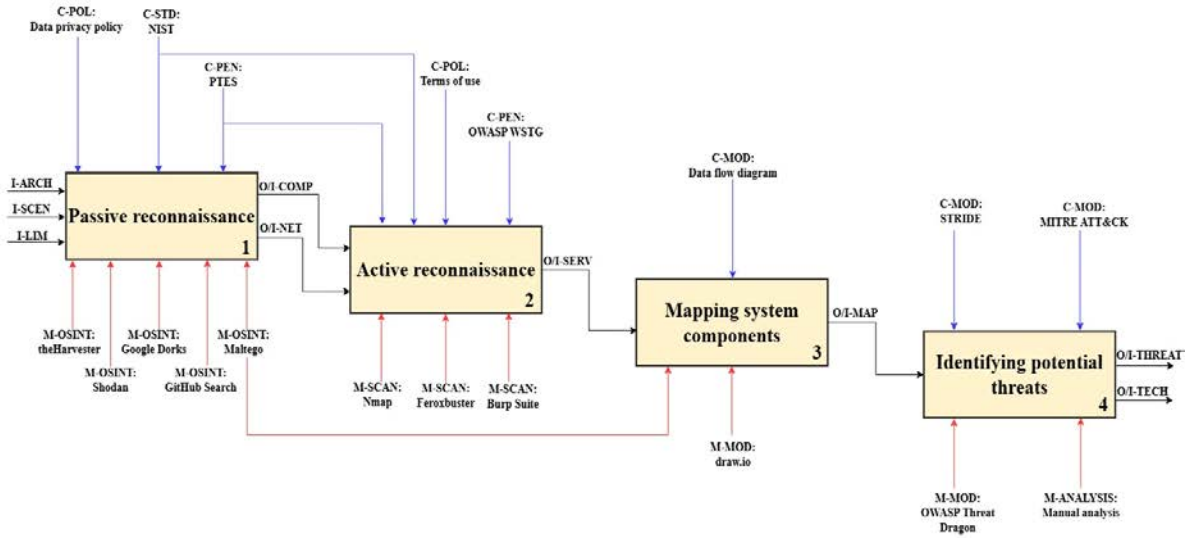


Fig. 3. Decomposed model of the information gathering and system analysis stage

Based on input data about the architecture (I-ARCH), usage scenarios (I-SCEN), and existing limitations (I-LIM), researchers use OSINT tools to form a preliminary network map (O/I-NET) and a list of UAV components (O/I-COMP). This includes searching for vulnerabilities via Shodan and theHarvester, analyzing data leaks using Google Dorks, researching code repositories via GitHub Search, and visualizing relationships in Maltego. Activities are strictly regulated by NIST standards [23] (C-STD) and PTES methodology (C-PEN), with the key restriction being the data privacy policy (C-POL), which excludes privacy violations.

The purpose of the active reconnaissance phase is to verify the collected data and identify entry points through direct interaction with UAV interfaces. The input data consists of identified components and a network map from the previous phase. Nmap is used to scan ports and services, and Feroxbuster and Burp Suite (M-SCAN) tools are used to analyze web interfaces and APIs. The process is governed by PTES [21] and OWASP Web Security Testing Guide (OWASP WSTG) [24] (C-PEN) methodologies. A critical aspect of management is compliance with the terms of use (C-POL), which define the permissible limits of interference to avoid destabilizing the operation of the UAV. The result is a confirmed list of active services (O/I-SERV).

The functional purpose of the system component mapping stage is to build a detailed model of UAV component interaction based on data about active services (O/I-SERV). Using the draw.io modeling tool (M-MOD), researchers systematize the information obtained in the form of a component map (O/I-MAP). The construction process is guided by the principles of creating data flow diagrams (C-MOD), which allows visualizing the vectors of information transfer between system modules and preparing the groundwork for threat modeling.

The final stage is aimed at identifying potential threats (O/I-THREAT) and forming a technology stack (O/I-TECH). Based on the component map (O/I-MAP), a security analysis is performed using the automated OWASP Threat Dragon tool (M-MOD) and manual analysis (M-ANALYSIS). The threat modeling process is regulated by the STRIDE methodology (for classifying attack types) and the MITRE ATT&CK knowledge base (C-MOD), which covers a wide range of known attacker tactics and techniques.

4.3.2. Assessment of known vulnerabilities

Figure 4 shows a model that details the process of assessing known vulnerabilities into three functional blocks: matching components with vulnerabilities DB (1), automated scanning (2), and scanning results validation (3).

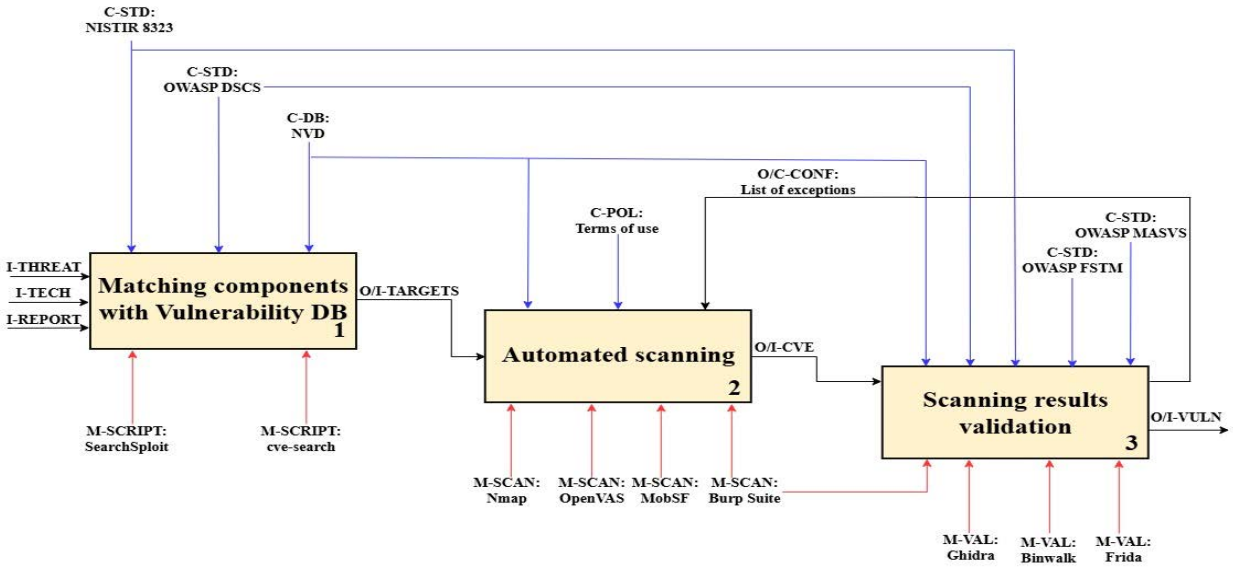


Fig. 4. Decomposed model of the known vulnerability assessment stage

The functional purpose of the first stage is to analytically identify potential vulnerabilities by correlating detected UAV components with known records in the database. Based on input data about threats (I-THREAT), technology stacks (I-TECH), and community reports (I-REPORT), researchers use exploit search tools and online database queries, such as SearchSploit and cve-search (M-SCRIPT). The search is regulated by the NISTIR 8323 standard (C-STD) and the recommendations of the specialized OWASP Drone Security Cheat Sheet (OWASP DSCS) reference guide. The result of the work is a list of potential targets (O/I-TARGETS) for further scanning verification.

The purpose of automated scanning is to actively check identified targets for vulnerabilities. Researchers use the M-SCAN suite of tools, which includes Nmap (port scanner), OpenVAS (vulnerability scanner), MobSF (mobile application analysis), and Burp Suite (web vulnerability proxy scanner). A critical aspect of management is strict adherence to terms of use (C-POL) to prevent system destabilization, as well as consideration of the list of exceptions (O/C-CONF). The result is a list of vulnerabilities and their CVE identifiers (O/I-CVE).

The final stage is aimed at validating the identified vulnerabilities to filter out false positives. For this purpose, reverse engineering (Ghidra), firmware analysis (Binwalk), and dynamic instrumentation (Frida) tools are used (M-VAL). The validation process is based on specialized standards for testing firmware, such as OWASP Firmware Security Testing Methodology (OWASP FSTM) and mobile applications – OWASP Mobile Application Security Verification Standard (OWASP MASVS), as well as the general standard NISTIR 8323. The result is a list of confirmed known vulnerabilities (O/I-VULN) and an updated list of exceptions (O/C-CONF), which is returned to the automatic scanning stage to improve its accuracy in subsequent iterations.

4.3.3. Detection of unknown vulnerabilities

The decomposition of the third stage of the combined method is shown in Figure 5. The model details the process of detecting unknown vulnerabilities into three functional blocks: attack surface analysis (1), fuzzing and anomaly detection (2), triage and root cause analysis (3).

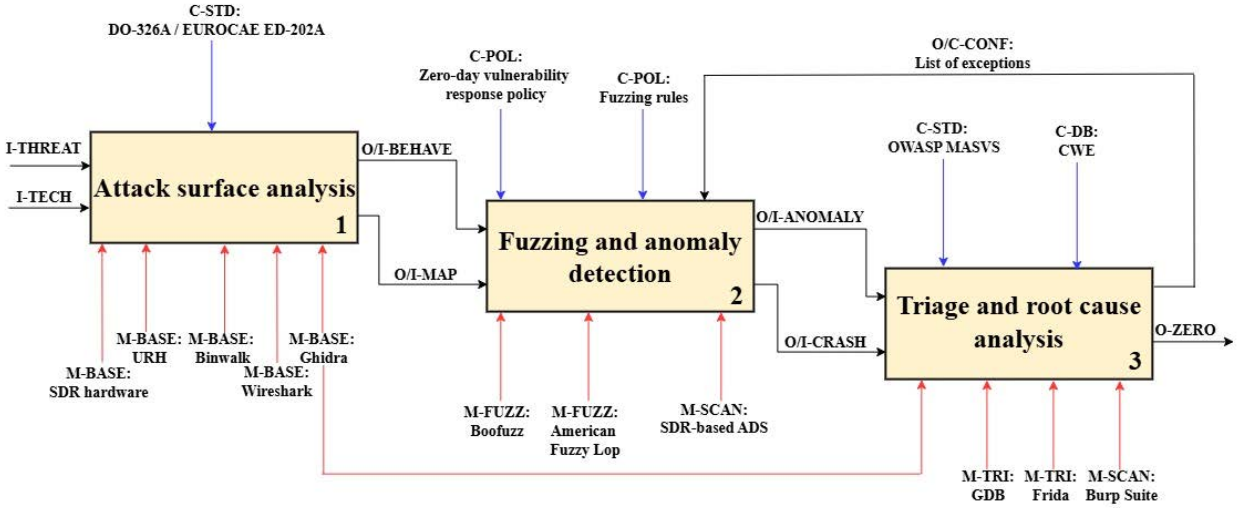


Fig. 5. Decomposed model of the stage of identifying unknown vulnerabilities

The purpose of the attack surface analysis stage is to create a model of normal UAV behavior and to map entry points in detail. Based on threat data (I-THREAT) and technology stack (I-TECH), a reference behavior profile (O/I-BEHAVE) and an attack surface map (O/I-MAP) are generated. Technical implementation is provided by the M-BASE suite of tools: SDR hardware is used for physical signal interception, which is then processed in URH and Wireshark for protocol analysis. Binwalk and Ghidra are used in parallel for static firmware analysis. The process is strictly regulated by DO-326A / EUROCAE ED-202A (C-STD) airworthiness safety standards, which define the requirements for security architecture.

The fuzzing and anomaly detection stage is a dynamic phase of active provocation of system malfunctions. Using a reference profile and attack map, as well as considering a list of exceptions from previous iterations (O/C-CONF), researchers apply fuzzing tools (M-FUZZ): the Boofuzz framework and the American Fuzzy Lop phaser to generate mutated data. Simultaneously with the attack, an SDR-based anomaly detection system (ADS) (M-SCAN) operates, performing external monitoring of the airwaves to record deviations. The process is managed in accordance with the zero-day vulnerability response policy and fuzzing rules (C-POL). The result is recorded reports of detected anomalies (O/I-ANOMALY) and critical failure logs (O/I-CRASH).

The final stage is aimed at verifying the anomalies found and determining the technical cause of the failure. Using M-TRI tools (GDB and Frida), researchers perform debugging and dynamic instrumentation of processes, and Burp Suite (M-SCAN) is used to analyze web vulnerabilities. The classification of detected defects is carried out based on CWE (C-DB) and in accordance with the OWASP MASVS standard (C-STD). Confirmed critical defects are recorded as zero-day vulnerabilities (O-ZERO), and false positives are added to the list of exceptions (O/C-CONF), which is returned to the fuzzing stage for testing optimization.

4.3.4. Preliminary IMECA-analysis

Figure 6 shows the decomposition of the fourth stage of the model – preparatory IMECA analysis, which consists of three functional blocks: identification of intrusion modes (1), assessment of intrusion parameters (2), and risk prioritization (3).

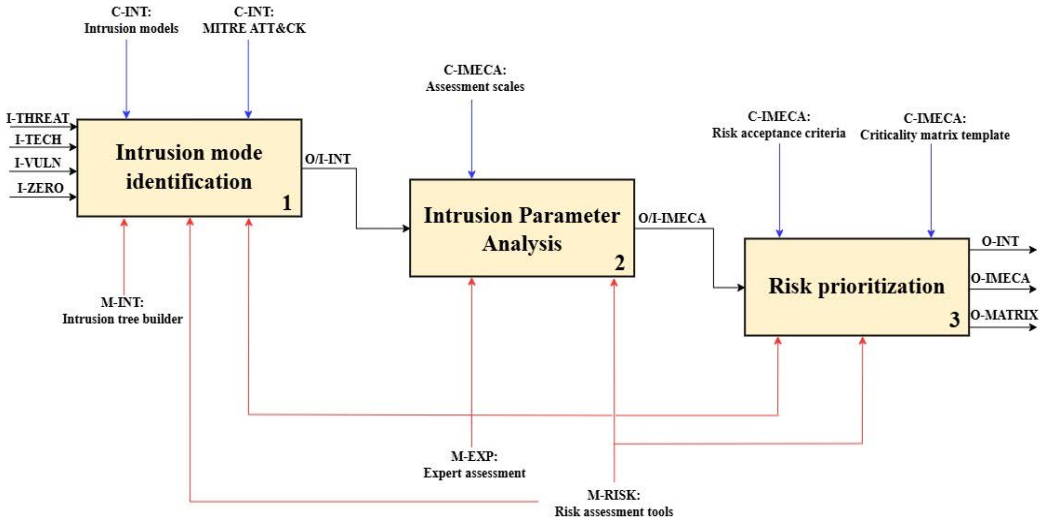


Fig. 6. Decomposed model of the preparatory IMECA-analysis stage

The purpose of intrusion mode identification is to synthesize technical data on vulnerabilities with abstract threat models to form specific intrusion scenarios. Based on input data about threats (I-THREAT), technology stack (I-TECH), known (I-VULN) and unknown (I-ZERO) vulnerabilities, experts transform vulnerability information into logical chains of attacker actions. This process is carried out using an intrusion tree constructor (M-INT) and expert assessment (M-EXP), guided by the MITRE ATT&CK tactics DB and intrusion models (C-INT). The result of this stage is a comprehensive list of identified intrusion modes (O/I-INT), which considers the specifics of combined attacks.

The next stage involves a preliminary quantitative and qualitative assessment of the identified intrusion modes. Experts (M-EXP) use approved assessment scales (C-IMECA) to determine key risk parameters: probability of occurrence, complexity of implementation, and severity of consequences. To reduce the subjectivity of expert judgments and automate calculations, risk assessment tools (M-RISK) are used, such as AXMEA [25]. The output of this stage is a structured table with preliminary assessments (O/I-IMECA).

The final stage is aimed at ranking threats to determine the focus of further experiments. Based on the completed table (O/I-IMECA) and using risk assessment tools (M-RISK), threats are visualized on a "probability-severity" plane according to the criticality matrix template (C-IMECA). The key control element is the risk acceptance criteria (C-IMECA), which define threshold values: scenarios that fall into the unacceptable risk zone are highlighted in a priority list (O-INT) for mandatory replication, and a criticality matrix (O-MATRIX) is formed.

4.3.5. Replication of intrusion modes

Figure 7 shows the decomposition of the fifth stage of the combined method. The model details the stage of replication of intrusion modes into three functional blocks: environment setup (1), operation (2), and post-operation (3).

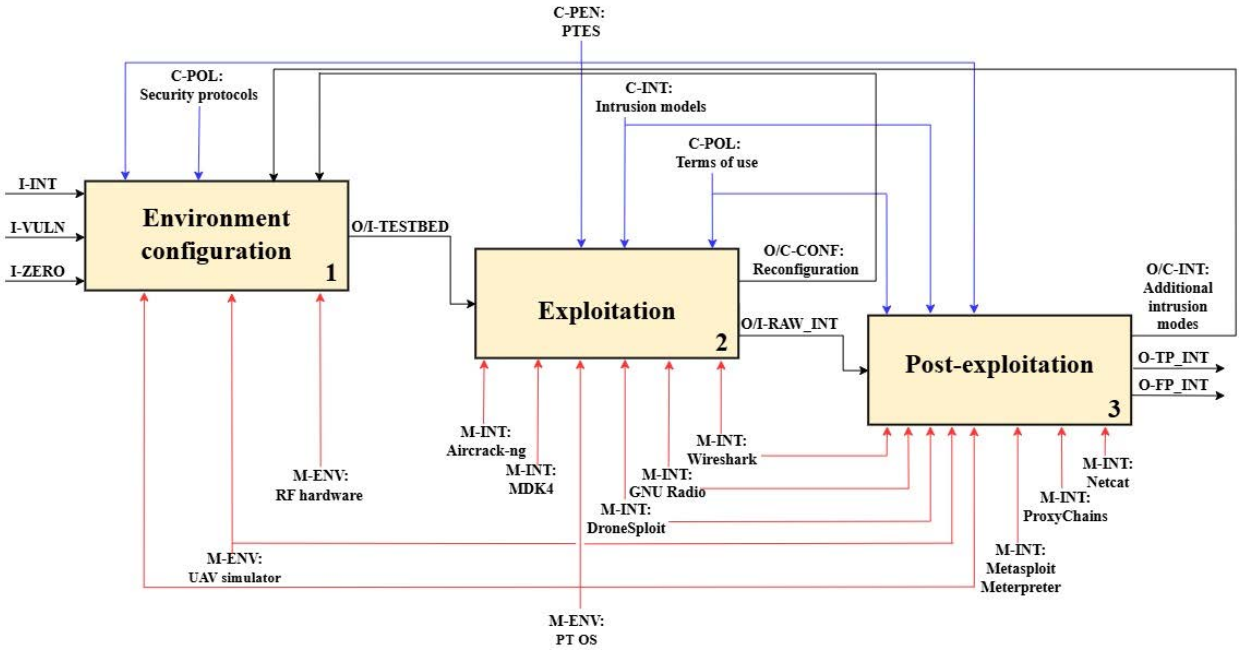


Fig. 7. Decomposed model of the replication stage of intrusion modes

The goal of the first stage is to prepare an isolated and controlled environment for the safe reproduction of intrusions. Based on prioritized modes (I-INT) and vulnerability data (I-VULN, I-ZERO), a test bed (O/I-TESTBED) is deployed. Technical implementation is provided by a set of M-ENV tools: deployment of a specialized OS (e.g., Kali Linux), configuration of a UAV simulator to create digital twins, and calibration of RF hardware. The process is strictly regulated by security protocols (C-POL) for test isolation and rules of use, and feedback from the next stage (O/C-CONF) allows for quick adjustments to the test bed configuration.

The operational stage is aimed at the direct implementation of intrusions on the prepared test bed. Researchers use a wide range of M-INT tools: Aircrack-ng and MDK4 are used for Wi-Fi attacks, the DroneSploit framework is used to exploit specific UAV vulnerabilities, and GNU Radio is used to work with radio waves. Traffic monitoring and analysis is performed using Wireshark. Actions are managed according to the PTES (C-PEN) methodology and approved intrusion models (C-INT). The result of this stage is the receipt of preliminary "raw" intrusion results (O/I-RAW_INT) or a request to reconfigure the environment (O/C-CONF) in case of failure.

The final stage focuses on securing the system, assessing the depth of penetration, and collecting evidence. Using M-INT tools, researchers establish persistent control over the device (via Metasploit Meterpreter and Netcat) and use ProxyChains to tunnel traffic. The process is governed by PTES rules and term of use (C-POL). If new attack vectors are discovered during the operation, they are returned to the beginning of the cycle as additional modes (O/C-INT). The result is verified data: confirmed successful intrusions (O-TP_INT) and refuted false hypotheses (O-FP_INT), which are transferred to the a posteriori IMECA-analysis.

4.3.6. A posteriori IMECA-analysis

Figure 8 shows the decomposition of the last stage of the combined method. The model details the posterior IMECA-analysis into three functional blocks: intrusion criticality reassessment (1), countermeasure selection (2), and residual risk assessment (3).

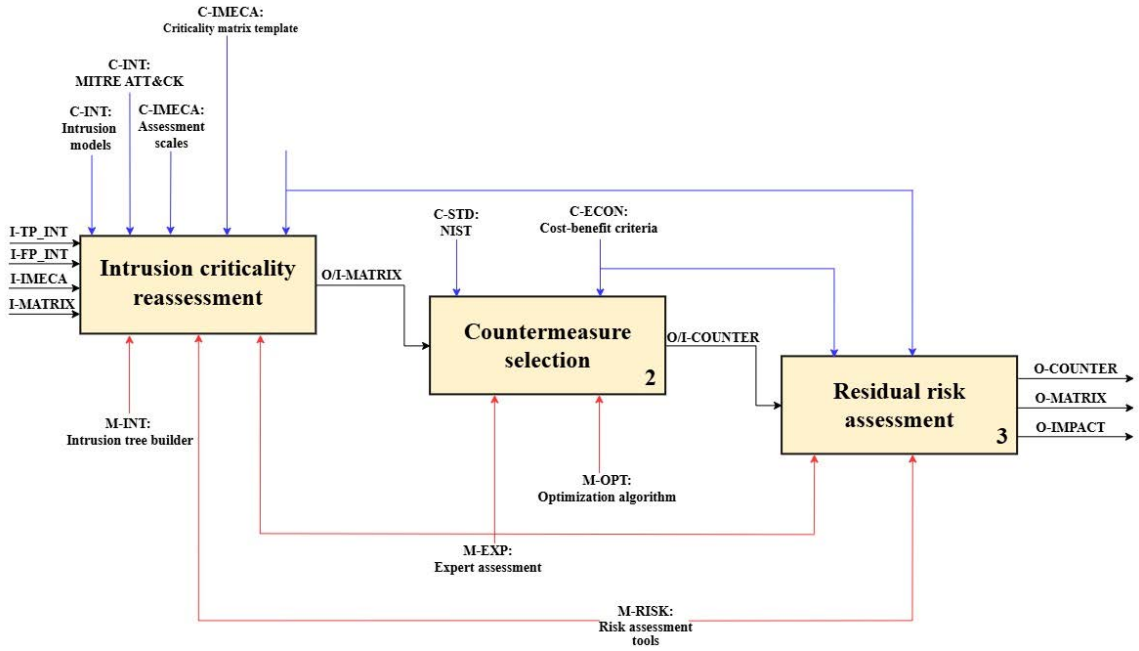


Fig. 8. Decomposed model of the a posteriori IMECA analysis stage

The goal of the first stage is to update risk assessments based on empirical data obtained during the replication of intrusion modes. Input data on confirmed (I-TP_INT) and false (I-FP_INT) attacks, together with initial tables (I-IMECA, I-MATRIX), are processed by experts (M-EXP) using intrusion tree builders (M-INT). This makes it possible to visualize verified compromise paths and adjust probability parameters to real values. The process is regulated by approved assessment scales, risk acceptance criteria (C-IMECA), and intrusion models (C-INT). The result is a validated criticality matrix (O/I-MATRIX) that reflects the actual state of system security.

The second stage involves the formation of a set of measures to neutralize critical risks. Using optimization algorithms (M-OPT) and expert assessment (M-EXP), researchers select countermeasures from security standards catalogs such as NIST (C-STD). The key constraint is the cost-benefit criteria (C-ECON), which ensure the implementation of the "minimum cost acceptable risk" principle. This allows us to weed out overly expensive solutions for protection against unlikely threats. The result is a pre-formed set of recommended countermeasures (O/I-COUNTER).

The final stage involves modeling the state of the system after the hypothetical implementation of the selected countermeasures. Risk assessment tools (M-RISK) and expert analysis (M-EXP) are used to calculate the residual risk for each scenario. If the risk level meets the acceptance criteria (C-IMECA) and is in the "green zone" of the matrix, the process ends with the formation of a final set of countermeasures (O-COUNTER), a residual risk matrix (O-MATRIX), and an impact assessment report (O-IMPACT).

5. Research results and their discussion

5.1. Analysis of preliminary results

The experimental study used the results of a preliminary study [19], in which IMECA tables of intrusion modes and criticality matrices were constructed based on the developed threat models for UAV application scenarios.

Analysis of the results showed that the most critical intrusion modes in UAV mission scenarios in terms of severity of consequences are: spoofing, jamming, man-in-the-middle and replay attacks, session hijacking, and optical blinding. Therefore, these intrusion modes were selected as priorities for further practical replication.

5.2. Building a test environment

Given the current martial law conditions in Ukraine, as well as the lack of safe areas for field testing and the instability of the power supply, it was decided to verify vulnerabilities in a simulated environment.

The Damn Vulnerable Drone (DVD) [26] specialized simulator was selected as the target system, which allows emulating UAV vulnerabilities. This tool was chosen because of its architectural advantages, which are based on the Software-in-the-Loop (SITL) principle,

as this approach to simulation allows real UAV software to be run in a virtual environment that is as close as possible to its execution on physical equipment.

Advantages of using the simulator in the context of further research:

- The use of ArduPilot in the DVD simulator allows you to execute real binary firmware code, which ensures the sensitivity of UAV components to various commands and input data.
- Integration with Gazebo provides realistic physical simulation of flight and interaction with the environment, allowing for accurate analysis of possible consequences and intrusion scenarios.
- Although the proposed simulator does not reproduce the architecture of every real-world UAV model, the vulnerabilities injected into it are characteristic of many types of UAVs.

An analysis of the documentation and a review of the simulator's functionality confirmed the possibility of reproducing most of the above-mentioned intrusion modes:

- Despite the impossibility of physical RF spoofing, DVD allows logical spoofing by entering artificial GPS coordinates directly into the data stream, forcing the UAV to accept the false location as valid.
- Jamming can be implemented in DVD as a denial-of-service attack at the channel or application levels, resulting in loss of control, similar to the action of electronic warfare (EW).
- In the DVD simulator, the connection between the ground station and the UAV is emulated via a TCP/UDP connection, which is suitable for reproducing man-in-the-middle attacks and allows telemetry to be intercepted and flight commands to be modified in real time.
- The ability to capture traffic in the DVD environment allows for the replay of commands, verifying the absence of replay protection in the protocols used.
- The simulator is vulnerable to control session interception, especially in scenarios using unprotected protocols (such as Telnet, FTP) or through Wi-Fi session interception.

Three approaches were used in the process of deploying the test environment:

- Virtualization (Windows 11 + Hyper-V): An attempt to deploy via GPU Passthrough revealed a conflict between Nvidia drivers and the xRDP subsystem in the Kali Linux environment. As a result, only the Lite version of the simulator could be launched, which did not provide full functionality for replicating complex intrusion modes.
- ARM architecture (M1-based Macbook + Parallels): A fundamental incompatibility of a number of simulator components with the ARM architecture was discovered.
- Dedicated station (Bare Metal Kali Linux): The most effective solution was to deploy the simulator on a separate desktop PC running Kali Linux OS. This allowed us to avoid problems with hardware resource virtualization and gain access to the full version of the simulator. The specifications of the current test environment are shown in Table 2.

Limitations of the current configuration:

- Despite its successful deployment, this stand is characterized by low mobility and requires a continuous power supply, which limits the possibility of conducting experiments in the absence of electricity.
- The performance of the current CPU is quite low, especially when emulating complex intrusion modes while using attack and monitoring tools.

Table 2. *TTX of the test environment*

Component	**Name**
CPU	Intel Core i5-6700
GPU	Nvidia GeForce GTX 1660 Ti
RAM	16 Гб
Storage	SSD 256 Гб
PT OS	Kali Linux
UAV simulator	DVD based on ArduPilot/MAVLink architecture

Additional limitations of the test environment:

• Attacks on physical sensors, such as optical blinding of a camera with a laser, cannot be reproduced because the simulator's virtual camera receives images from a graphics engine and does not have a physical matrix sensitive to light saturation.

• The simulator does not reproduce the battery discharge process, so attacks aimed at preventing the transition to sleep mode or increasing the load on the CPU will not work.

• In the simulator, attacks on communication channel availability are implemented through packet injection or software connection termination. However, this does not take into account the physical aspects of electronic warfare (signal strength, weather conditions, terrain, and antenna characteristics).

These limitations narrow the scope of testing the combined method in the current iteration to the logical and network levels. Assessing physical resilience and protection against kinetic and/or energy attacks requires a transition to hybrid Hardware-In-The-Loop (HITL) modeling in subsequent stages of the study.

5.3. Practical testing of the test environment

To confirm the ability of the deployed test environment to replicate intrusion modes, an attack scenario on the UAV wireless control channel was implemented. The purpose of the experiment was to test the possibility of unauthorized access to the control network protected by the WPA2 protocol, which corresponds to the intrusion mode – "session interception".

The use of the built-in airodump-ng utility allowed us to scan the airwaves and identify the target network (SSID), its BSSID (MAC address), operating channel, and encryption type (WPA2-PSK/CCMP). A ground station connected to the network was also identified.

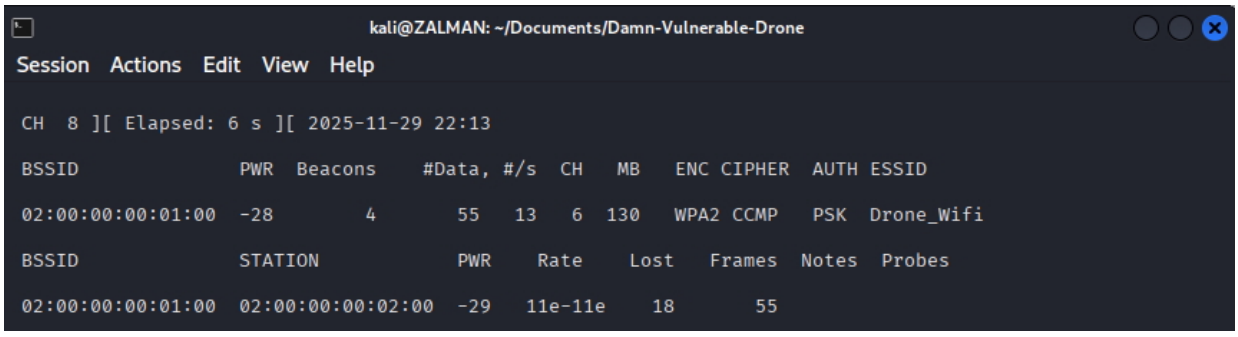


Fig. 9. Result of scanning with the airodump-ng utility

To obtain the cryptographic handshake required to crack the password, a deauthentication attack was initiated. Using the aireplay-ng tool, a series of deauthentication packets were sent to the access point and client addresses. This resulted in a forced disconnection. When the client attempted to automatically reconnect, the airodump-ng utility successfully intercepted and saved the WPA handshake to a *.cap file.

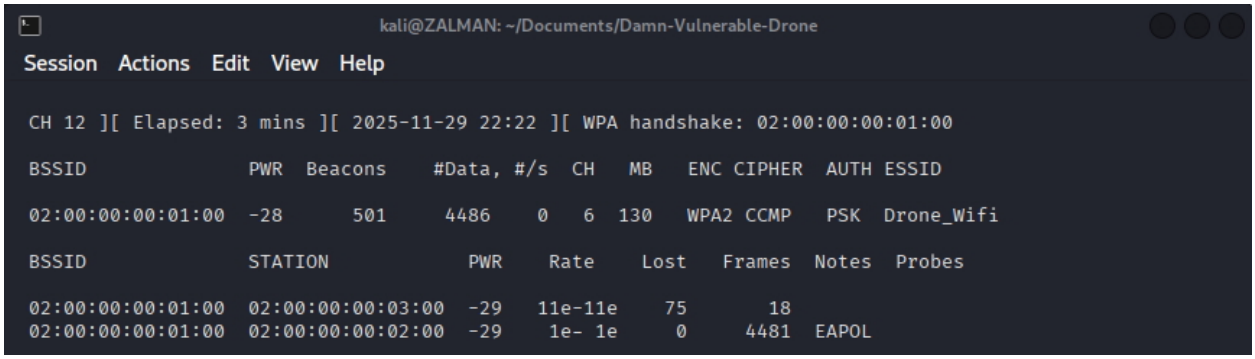


Fig. 10. Interception of WPA handshake using the airodump-ng utility

The resulting password hash was subjected to a dictionary attack using the aircrack-ng tool version 1.7. Using the standard rockyou.txt password dictionary, the network's PSK key was successfully cracked.

Fig. 11. Result of a dictionary attack

As a result of the experiment, a valid password for accessing the UAV control wireless network was obtained. The attacker successfully connected to the network, obtaining an IP address in the same range as the flight controller. Compromising the network allows moving on to the post-exploitation stage, in particular: performing "man-in-the-middle" attacks to intercept telemetry, connect to open services (SSH/Telnet), or inject fake control commands. The experiment confirmed that the probability of implementing this intrusion mode is very high when using a weak password. This indicates the need to increase the criticality level of the intrusion mode in the matrix and justifies the need to implement complex password policies or switch to more secure communication protocols (e.g., WPA3) as a priority countermeasure.

6. Conclusions and prospects for further research

The paper presents and justifies a combined method for assessing the security of UAV cyber assets, which integrates IMECA analysis procedures with PT practice. This allows overcoming the limitations of static risk assessment methods and isolated technical tests, creating a closed cycle of system verification and protection.

Main results of the study:

- A functional model of a combined method (IDEF0) is proposed, which ensures the completeness of the security assessment of UAV cyber assets and consists of the following stages: information gathering and vulnerability assessment, intrusion mode replication, IMECA analysis (including preparatory and a posteriori IMECA analysis), and countermeasure selection.
- A test environment based on a DVD simulator has been deployed and verified. Analytical testing has confirmed the stand's ability to reproduce 80% of priority intrusion modes, making it an effective tool for secure research under existing constraints.
- Practical testing was carried out, during which an attack scenario on the wireless control channel was successfully implemented, confirming the possibility of gaining unauthorized access to the flight control system. This experiment proved the ability of the selected test environment to replicate intrusion modes.

Theoretical and experimental studies confirm the advantages of the proposed combined method compared to known analytical [11] and experimental [15] methods, which consist in improving the completeness and reliability of assessing the security of UAV cyber assets. This is achieved by adjusting the results of the preliminary IMECA analysis by reviewing the results of the PT and forming the final IMECA tables, which makes it possible to justify the choice of countermeasures and, consequently, to increase cyber and functional security as a whole.

Prospects for further research are related to conducting a full-scale series of experiments for all identified intrusion modes, constructing a posteriori criticality matrices and selecting countermeasures, as well as integrating the proposed method with the task of assessing the functional safety of unmanned systems [25].

For further research, it is recommended to switch to a mobile workstation (laptop) with a discrete Nvidia RTX series video card with CUDA driver support to ensure autonomy, increase productivity, and reduce the time required for experiments.

References

1. Tlili, F., Fourati, L. C., Ayed, S., Ouni, B. (2022), "Investigation on vulnerabilities, threats and attacks prohibiting UAVs charging and depleting UAVs batteries: Assessments & countermeasures". *Ad Hoc Networks*. Vol. 129, p. 102805. DOI: 10.1016/j.adhoc.2022.102805
2. Freedberg Jr, S. J. (2023), "Dumb and cheap: When facing electronic warfare in Ukraine, small drones' quantity is quality". *BreakingDefense.com.* Available at: https://breakingdefense.com/2023/06/dumb-and-cheap-when-facing-electronic-warfare-in-ukraine-small-drones-quantity-is-quality/ (Accessed: 30 November 2025).
3. Hartmann, K., Giles, K. (2016), "UAV exploitation: A new domain for cyber power". *2016 8th International Conference on Cyber Conflict (CyCon)*, Tallinn, Estonia, pp. 205–221. DOI: 10.1109/CYCON.2016.7529436
4. FP Explainers (2025), "Indian Army's 'Make in India' drones hacked in border areas: Report". *Firstpost.com.* Available at: https://www.firstpost.com/india/indian-army-make-in-india-drones-hacked-in-border-areas-report-13859474.html (Accessed: 30 November 2025).
5. Yaacoub, J.-P., Noura, H., Salman, O., Chehab, A. (2020), "Security analysis of drones systems: Attacks, limitations, and recommendations". *Internet of Things.* Vol. 11, p. 100218. DOI: 10.1016/j.iot.2020.100218
6. The New Geopolitics Research Network (2024), "Ukrainian Drones vs Russian Jamming". *NewGeopolitics.org.* Available at: https://www.newgeopolitics.org/2024/06/10/ukrainian-drones-vs-russian-jamming/ (Accessed: 30 November 2025).
7. Royal United Services Institute for Defence and Security Studies (2023), "Meatgrinder: Russian Tactics in the Second Year of Its Invasion of Ukraine". *RUSI*. Available at: https://static.rusi.org/403-SR-Russian-Tactics-web-final.pdf (Accessed: 30 November 2025).
8. Mekdad, Y., Arış, A., Babun, L., El Fergougui, A., Conti, M., Lazzeretti, R., Uluagac, S. (2023), "A survey on security and privacy issues of UAVs". *Computer Networks*. Vol. 224, p. 109626. DOI: 10.1016/j.comnet.2023.109626
9. Yu, Z., Wang, Z., Yu, J., Liu, D., Song, H. H., Li, Z. (2024), "Cybersecurity of Unmanned Aerial Vehicles: A Survey". *IEEE Aerospace and Electronic Systems Magazine.* Vol. 39, No. 9, pp. 182–215. DOI: 10.1109/MAES.2023.3318226
10. Kong, P.-Y. (2021), "A Survey of Cyberattack Countermeasures for Unmanned Aerial Vehicles". *IEEE Access*. Vol. 9, pp. 148244–148263. DOI: 10.1109/ACCESS.2021.3124996
11. Zemlianko, H., Kharchenko, V. (2023), "Cybersecurity risk analysis of multifunctional UAV fleet systems: a conceptual model and IMECA-based technique". *Radioelectronic and Computer Systems*. No. 4, pp. 152–170. DOI: 10.32620/reks.2023.4.11
12. Torianyk, V., Kharchenko, V., Zemlianko, H. (2021), "IMECA based assessment of Internet of Drones systems cyber security considering radio frequency vulnerabilities". *Proc. 2nd Int. Workshop on Intelligent Information Technologies and Systems of Information Security (IntelITSIS'2021),* Khmelnytskyi, Ukraine. Available at: https://ceur-ws.org/Vol-2853/paper50.pdf (Accessed: 30 November 2025).
13. Branco, B., Silva, J. S., Correia, M. (2024), "D3S: A Drone Security Scoring System". *Information*. Vol. 15, No. 12, p. 811. DOI: 10.3390/info15120811
14. Ficco, M., Granata, D., Palmieri, F., Rak, M. (2024), "A systematic approach for threat and vulnerability analysis of unmanned aerial vehicles". *Internet Things*. Vol. 26, p. 101180. DOI: 10.1016/j.iot.2024.101180
15. Veerappan, C. S., Keong, P. L. K., Balachandran, V., Fadilah, M. S. B. M. (2021), "DRAT: A Penetration Testing Framework for Drones". *2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA)*, Chengdu, China, pp. 498–503. DOI: 10.1109/ICIEA51954.2021.9516363
16. Abakumov, A., Kharchenko, V. (2023), "Combining experimental and analytical methods for penetration testing of AI-powered robotic systems". *Proc. 7th Int. Conf. on Computational Linguistics and Intelligent*

Systems (COLINS 2023), Kharkiv, Ukraine. Vol. 3403, pp. 470–481. Available at: https://ceur-ws.org/Vol-3403/paper40.pdf (Accessed: 08 April 2025).

17. Veprytska, O., Kharchenko, V. (2023), "Extended IMECA Technique for Assessing Risks of Successful Cyberattacks". *2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Athens, Greece, pp. 1–7. DOI: 10.1109/DESSERT61349.2023.10416447
18. Abakumov, A., Kharchenko, V. (2022), "Combining IMECA analysis and penetration testing to assess the cybersecurity of industrial robotic systems". *2022 12th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Athens, Greece, pp. 1–6. DOI: 10.1109/DESSERT58054.2022.10018823
19. Abakumov, A., Kharchenko, V., Popov, P. (2025), "A Hybrid Cybersecurity Assessment Framework for Unmanned Aircraft Vehicles Based on IMECA and Penetration Testing". *2025 55th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W),* Naples, Italy, pp. 7–14. DOI: 10.1109/DSN-W65791.2025.00032
20. Sanghavi, P., Kaur, H. (2023), "A Comprehensive Study on Cyber Security in Unmanned Aerial Vehicles". *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, pp. 804–809. Available at: https://ieeexplore.ieee.org/document/10112549 (Accessed: 09 December 2025).
21. The Penetration Testing Execution Standard (2017), "PTES Technical Guidelines". *Pentest-standard.org.* Available at: http://www.pentest-standard.org/index.php/PTES_Technical_Guidelines (Accessed: 30 November 2025).
22. Herzog, P. (2010), "OSSTMM 3: The Open-Source Security Testing Methodology Manual – Contemporary Security Testing and Analysis". *ISECOM.* Available at: https://www.isecom.org/OSSTMM.3.pdf (Accessed: 30 November 2025).
23. Scarfone, K., Souppaya, M., Cody, A., Orebaugh, A. (2008), "Technical Guide to Information Security Testing and Assessment: Recommendations of the National Institute of Standards and Technology". *NIST Special Publication 800-115.* Gaithersburg, MD: National Institute of Standards and Technology. Available at: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf (Accessed: 30 November 2025).
24. OWASP (2024), "Web Security Testing Guide". *owasp.org.* Available at: https://owasp.org/www-project-web-security-testing-guide/ (Accessed: 30 November 2025).
25. Babeshko, I., Illiashenko, O., Kharchenko, V., Leontiev, K. (2022), "Towards Trustworthy Safety Assessment by Providing Expert and Tool-Based XMECA Techniques". *Mathematics*. Vol. 10, p. 2297. DOI: 10.3390/math10132297
26. Aleks, N. (2023), "Damn Vulnerable Drone (DVD)". *GitHub repository.* Available at: https://github.com/nicholasaleks/Damn-Vulnerable-Drone (Accessed: 29 November 2025).

Received (Надійшла) 01.12.2025
Accepted for publication (Прийнята до друку) 09.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Abakumov Artem – National Aerospace University "Kharkiv Aviation Institute", Master of Science in Information measuring technologies, PhD Student at the Department of Computer Systems, Networks, and Cybersecurity, Kharkiv, Ukraine; e-mail: a.i.abakumov@csn.khai.edu; ORCID ID: https://orcid.org/0000-0002-7742-6515

Kharchenko Vyacheslav – Corr.-member of National Academy of Science, Doctor of Sciences (Engineering), Professor, National Aerospace University "Kharkiv Aviation Institute", Head at the Department of Computer Systems, Networks, and Cybersecurity, Kharkiv, Ukraine; e-mail: v.kharchenko@csn.khai.edu; ORCID ID: https://orcid.org/0000-0001-5352-077X

Абакумов Артем Ігорович – Національний аерокосмічний університет "Харківський авіаційний інститут", фахівець зі спеціальності "Інформаційні вимірювальні системи", аспірант кафедри комп'ютерних систем, мереж і кібербезпеки, Харків, Україна.

Харченко Вячеслав Сергійович – член-кореспондент Національної академії наук України, доктор технічних наук, професор, Національний аерокосмічний університет "Харківський авіаційний інститут", завідувач кафедри комп'ютерних систем, мереж і кібербезпеки, Харків, Україна.

КОМБІНОВАНИЙ МЕТОД ОЦІНЮВАННЯ БЕЗПЕКИ КІБЕРАКТИВІВ БпЛА З ВИКОРИСТАННЯМ IMECA-ПРОЦЕДУР І ТЕСТУВАННЯ НА ПРОНИКНЕННЯ

Предмет вивчення – методи й засоби оцінювання безпеки кіберактивів безпілотних літальних апаратів (БпЛА). На підставі аналізу публікацій зроблено висновок про певний розрив між теоретичними методами оцінювання ризиків і практичними інструментами тестування на проникнення (ТнП) кіберактивів БпЛА. Наявні інструменти ТнП призначені для відтворення атак, але не надають методології оцінювання їх впливу в динамічних умовах застосування. **Мета дослідження** – розробити комбінований метод і елементи технології достовірного виявлення вразливостей з можливістю верифікації процесу й обґрунтування вибору контрзаходів. **Завдання роботи:** обґрунтування доцільності комбінованого застосування різних методів і засобів оцінювання кібербезпеки БпЛА; розроблення моделей і методу оцінювання, який поєднує аналітичні та експерементальні процедури; практичне відпрацювання методу з використанням тестового середовища. **Упроваджені методи**: ризик-орієнтований аналіз режимів вторгнення, наслідків і критичності (IMECA-аналіз) і ТнП. **Результати дослідження**: обґрунтовано структуру й послідовність комбінованого оцінювання кібербезпеки; запропоновано функціональну IDEF0-модель, що забезпечує повноту оцінювання безпеки кіберактивів БпЛА й містить такі етапи: збір інформації та оцінювання вразливостей, реплікація режимів вторгнення, IMECA-аналіз (зокрема з підготовчим і апостеріорним IMECA-аналізом) і вибір контрзаходів; розгорнуто й апробовано тестове середовище на базі симулятора DVD, який здатний відтворювати 80 % пріоритетних режимів вторгнення. **Висновки:** комбінований метод оцінювання безпеки кіберактивів БпЛА інтегрує процедури IMECA з практикою ТнП, що дає змогу подолати обмеження статичних методів оцінювання ризиків та ізольованих технічних тестів, створюючи замкнутий цикл верифікації та захисту бортових систем. Подальші дослідження пов'язані з проведенням повномасштабної серії експериментів для всіх ідентифікованих режимів вторгнення, побудовою апостеріорних матриць критичності та вибору контрзаходів.

Ключові слова: кіберактиви БпЛА; IMECA-аналіз; комбінований метод; симуляції режимів вторгнення; тестування на проникнення.

Bibliographic descriptions / Бібліографічні описи

Abakumov, A., Kharchenko, V. (2025), "Combined method of uav cyber assets security assessment by use of procedures imeca and penetration testing", *Management Information Systems and Devises*, No. 4 (187), P. 200–219. DOI: https://doi.org/10.30837/0135-1710.2025.187.200

Абакумов А. І., Харченко В. С. Комбінований метод оцінювання безпеки кіберактивів БпЛА з використанням IMECA-процедур і тестування на проникнення. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 200–219.
DOI: https://doi.org/10.30837/0135-1710.2025.187.200

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.7:004.38:621.373.8:621.391 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.220

O. Petrenko, O. Petrenko, A. Bidun, Z. Ostrovskyi

MODEL FOR ASSESSING THE LEVEL AND PRIORITIZING MULTI-PARAMETER THREATS USING THE MAMDANI FUZZY LOGIC ALGORITHM OF THE FIRST TYPE

The subject of the study is a model for assessing threats and determining their priorities based on fuzzy logic methods. The first-type Mamdani algorithm was used to build the model. The developed threat assessment model was tested on a static scenario, as well as on dynamic real-time attack scenarios. The problem was solved using fuzzy logic methods. *The Fuzzy Logic Toolbox* (a MATLAB extension) was used to model the system, which contains tools for designing systems based on fuzzy logic. Block diagrams of the static and dynamic fuzzy threat assessment models are presented in the *Simulink* application. The purpose of the study is to develop and analyze a fuzzy model for assessing threats and determining their priorities in order to make decisions on the sequence of measures to counter these threats. **The objectives of the work** include justifying the feasibility and effectiveness of using fuzzy logical expressions and fuzzy logic operations for a formalized description of expert requirements for determining threat priorities. Fuzzy logic methods are widely implemented in various control systems, particularly in the following areas: nonlinear process control, self-learning systems, risk and critical situation analysis, pattern recognition, financial analysis, corporate repository information research, and management and coordination strategy optimization. **Methods used in the study:** probability theory, fuzzy logic theory, modeling. **Results achieved.** The possibility of using fuzzy logical expressions and fuzzy logic operations for a formalized description of expert criteria for determining threat priorities is considered. This approach provides numerical assessments of threats based on specified parameters, which contributes to accuracy and flexibility in the analysis process. The possibility of using fuzzy logical expressions and fuzzy logic operations for a formalized description of expert requirements for determining threat priorities is justified. This makes it possible to obtain numerical threat assessments based on specified input parameters, ensuring accuracy and adaptability in the analysis process. The article proposes an algorithm for rating threats on a scale from 0 to 1 using a fuzzy logic system, which contributes to accurate results. **Conclusions.** The developed procedure for prioritizing threats, based on a fuzzy set model, significantly expands the functionality and allows determining threat levels. This, in turn, creates the basis for making effective decisions on the implementation of measures to counter these threats and is the main result of the study.

Keywords: model; fuzzy logic; membership function; threat level assessment; threat prioritization; decision support; uncertainty; linguistic variables; fuzzy inference.

Introduction

Problem statement

In most cases, security specialists assess threats based on their own experience, converting threat levels into numerical values. However, this approach to assessing threat levels significantly limits the overall capabilities of the methodology, as the reliability of expert conclusions often gives rise to conflicting opinions. In today's world, the rapid development of information technology and the increasing complexity of decision-making processes bring to the fore methods that take into account factors of uncertainty and insufficient data. Approaches based on fuzzy logic, which allows for the formalization and analysis of complex systems where traditional methods lose their effectiveness, are particularly important in this context.

The article focuses on the development of methodological foundations and practical recommendations for the implementation of threat prioritization systems using fuzzy logic. Fuzzy set methods are particularly useful in situations where it is impossible to construct an accurate mathematical model of the system's functioning.

Thanks to fuzzy set theory, it becomes possible to use imprecise and subjective expert knowledge about the subject area to make decisions without the need to formalize them in the form of traditional mathematical models.

Thus, the implementation of fuzzy logic methods for assessing the level of threats with their subsequent prioritization for timely and balanced countermeasures is a relevant scientific task.

Analysis of recent studies and publications

Fuzzy logic first appeared in the mid-1960s thanks to the work of Lotfi Zadeh [1], an American mathematician and logician who first introduced the concept. Since then, its theoretical foundations and models have continued to evolve and remain one of the most widely used methods today.

The practical application of fuzzy set theory actually began in 1975, when E. Mamdani created the first fuzzy controller [2]. Fuzzy logic methods are widely used in various control systems, particularly in the following areas: control of nonlinear processes, self-learning systems, analysis of risky and critical situations, pattern recognition, financial analysis, research of data from corporate repositories, optimization of management strategies, and coordination of actions [3, 4].

Works [5, 6] discuss the development of an automated decision support system that uses fuzzy networks to analyze and assess the air situation from the perspective of threats.

An analysis of scientific literature demonstrates the active and effective application of fuzzy logic methods for threat assessment in various fields, including information security, cybersecurity, and critical infrastructure risk management [7, 8].

The main advantage of using fuzzy logic is its ability to effectively process inaccurate, fuzzy, or incomplete input data and expert assessments, which are often encountered during threat analysis.

Numerous studies [9, 10, 11, 12] consider the practical implementation of the presented models, in particular using the Fuzzy Logic Toolbox environment in MatLab, as well as the creation of test sets of fuzzy rules. Research shows that fuzzy logic methods are an important and effective tool for threat assessment, especially in situations with significant uncertainty. These methods facilitate the development of reliable and adaptive models that take expert knowledge into account, ensuring more informed decision-making in the field of security.

Other studies in this area cover such areas as guided fuzzy clustering [13], rule merging [14], and multi-criteria optimization.

The purpose of this article is to develop and analyze a fuzzy model for assessing threats and determining their priorities in order to make decisions on the sequence of measures to counter these threats.

Presentation of main material

To eliminate the shortcomings of existing risk analysis and assessment methods, the use of fuzzy logic methods is proposed. Fuzzy logic demonstrates high efficiency in cases where there is insufficient understanding of the characteristics of the system under study, limited access to the necessary amount of data, and risk assessment is based on expert information, where the input data may be insufficiently accurate or incorrectly presented. The flexibility and ease of use of fuzzy logic as a methodology for solving problems ensure its effective implementation in data control and analysis systems. At the same time, human intuition and operator experience are also involved [15, 16].

Fuzzy logical inference assumes that a set of rules must be applied to evaluate the activated membership function. In the context of fuzzy logical inference, such a set is called a rule base or knowledge base for a specific subject area. The use of a set of rules contributes to a more complete coverage of the reference space, while ensuring the reliability of the conclusions obtained [17]. Based on the set of rules, a fuzzy logical inference system is built, as shown in Fig. 1.

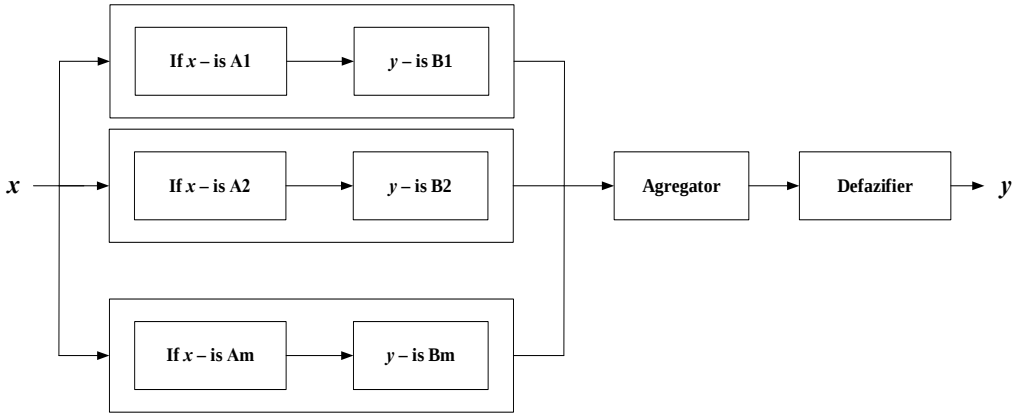


Fig. 1. Structural diagram of a fuzzy logic inference system
Source: [17]

The fuzzy inference process, based on fuzzy set theory, involves the use of fuzzy logic to form a correspondence between a given input signal and an output result. This mapping serves as the basis for decision-making or pattern recognition. This process takes into account all key elements: membership function, logical operations, and if-then rules [18]. This article proposes an algorithm based on fuzzy inference rules using the Mamdani algorithm, which is designed to assess the level of threat. In the algorithm, several arrays of input data are processed to determine the initial value of the threat level.

The Mamdani algorithm is one of the first to be successfully implemented in fuzzy inference systems. It was developed in 1975 by English mathematician Ebrahim Mamdani as an approach to controlling the operation of a steam engine [19]. Fuzzy inference using the Mamdani method was first proposed for the development of control systems based on the synthesis of linguistic rules formulated by experienced experts [2]. In this system, the output of each rule is represented as

a fuzzy set of all possible values of a linguistic variable. Mamdani systems, due to their intuitiveness and simpler rule base structure, are ideal for use in expert systems in which rules are formed based on the knowledge and experience of specialists.

The Mamdani algorithm works as follows:

- for each input parameter, the degree of its membership in the corresponding fuzzy set is determined;
- based on each fuzzy logical rule, the degree of correspondence of the rules to the obtained data is evaluated;
- the degree of membership of each conclusion derived from fuzzy logical rules is calculated;
- calculations are performed to determine the values of the conclusions.

The obtained values have the form of a fuzzy quantity representing the result of logical analysis. To convert this result into a clear value, a defuzzification procedure is used.

The inference process for the Mamdani system is summarized in Fig. 2.

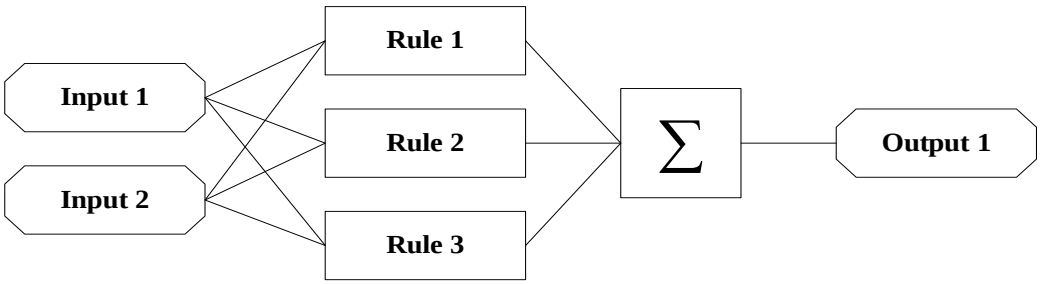


Fig. 2. Fuzzy inference process for the Mamdani system
Source: [20]

The MATLAB Fuzzy Logic Toolbox and Simulink extension package was used to model the system. A triangular shape was chosen for the membership functions. After defining the input variables, the graphical interface of the membership function editor is shown in Fig. 3.

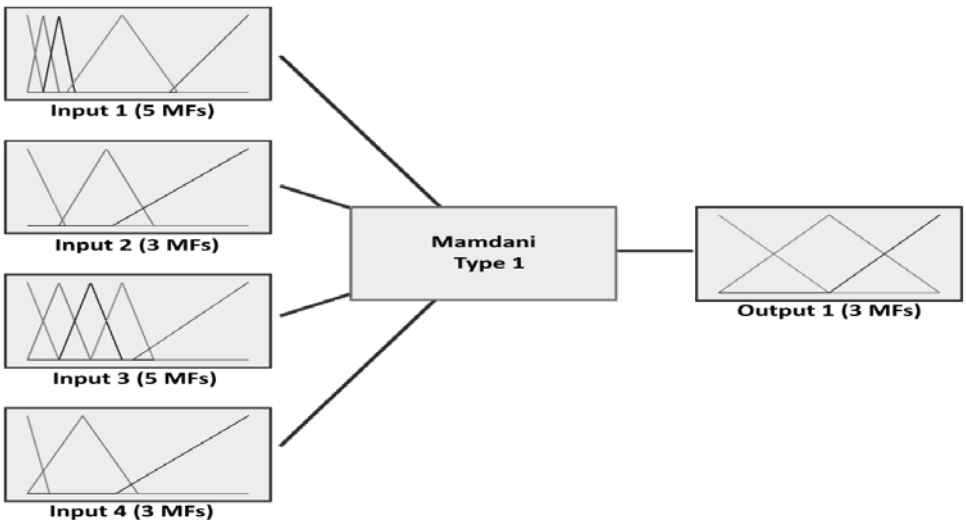


Fig. 3. Graphical interface of the membership function editor
Source: developed by the authors

Figure 4 shows the membership functions for four possible types of input parameters. These functions show how each point of influence of the input parameter determines the membership value in the range from 0 to 1.

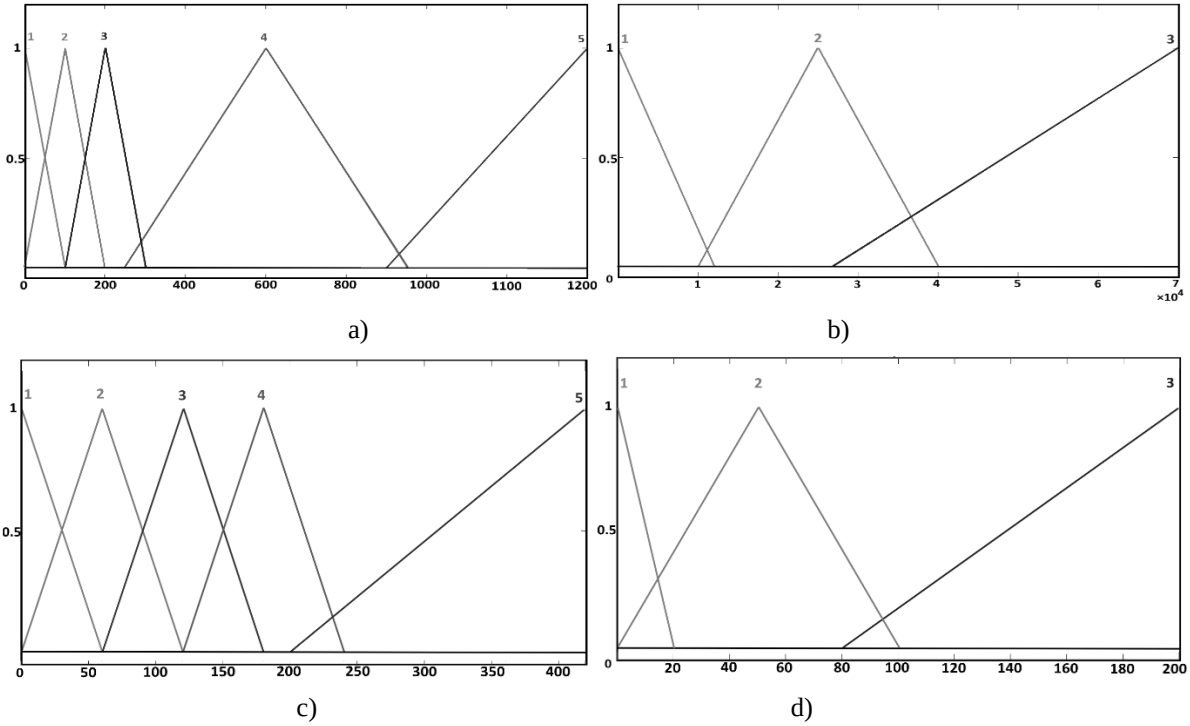


Fig. 4. Membership functions for input parameters of the first type (a), second type (b), third type (c), and fourth type (d)
Source: developed by the authors

The output parameter of the fuzzy model for threat assessment is the threat priority, which varies from 0 to 1. This is illustrated in Fig. 5.

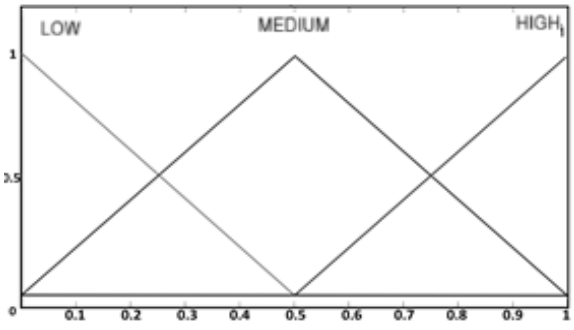


Fig. 5. Membership functions for threat prioritization
Source: developed by the authors

In the course of work, based on available standard data and expert comments on the relationship between input and output parameters, it is necessary to define fuzzy inference rules. Initial rules were formulated and evaluated for reliability in both static conditions and real-time scenarios. The input data allows the priority of threats to be adapted by applying specific rules.

Within the framework of the presented model, 226 rules were formulated, confirming its stability and effectiveness. Some of the fuzzy inference rules used are described in detail in Table 1.

Table 1. *Basic rules of fuzzy inference applied in this article*

Rule	Description
Rule 1 (low priority)	IF (Input1 is mf1) AND (Input2 is mf3) AND (Input3 is mf5) AND (Input4 is mf1) THEN (Output1 is mf1) (Weight: 1)
Rule 2 (low priority)	IF (Input1 is mf2) AND (Input2 is mf2) AND (Input3 is mf4) AND (Input4 is mf2) THEN (Output1 is mf1) (Weight: 1)
Rule 3 (low priority)	IF (Input1 is mf3) AND (Input2 is mf3) AND (Input3 is mf3) AND (Input4 is mf3) THEN (Output1 is mf1) (Weight: 1)
Rule 4 (medium priority)	IF (Input1 is mf4) AND (Input2 is mf2) AND (Input3 is mf3) AND (Input4 is mf3) THEN (Output1 is mf2) (Weight: 1)
Rule 5 (medium priority)	IF (Input1 is mf4) AND (Input2 is mf2) AND (Input3 is mf3) AND (Input4 is mf2) THEN (Output1 is mf1) (Weight: 1)
Rule 6 (high priority)	IF (Input1 is mf5) AND (Input2 is mf1) AND (Input3 is mf1) AND (Input4 is mf3) THEN (Output1 is mf3) (Weight: 1)
Rule 7 (high priority)	IF (Input1 is mf5) AND (Input2 is mf2) AND (Input3 is mf1) AND (Input4 is mf3) THEN (Output1 is mf3) (Weight: 1)
Rule 8 (high priority)	IF (Input1 is mf5) AND (Input2 is mf1) AND (Input3 is mf2) AND (Input4 is mf3) THEN (Output1 is mf3) (Weight: 1)

Let us assume that, after analyzing the system's performance, experts evaluated the input parameters according to the following indicators: first type – 70 points, second type – 30,000 points, third type – 180 points, and fourth type – 80 points. In accordance with the defined rules and using Mamdani's fuzzy inference algorithm, an initial threat assessment of 0.202 was obtained (Fig. 6).

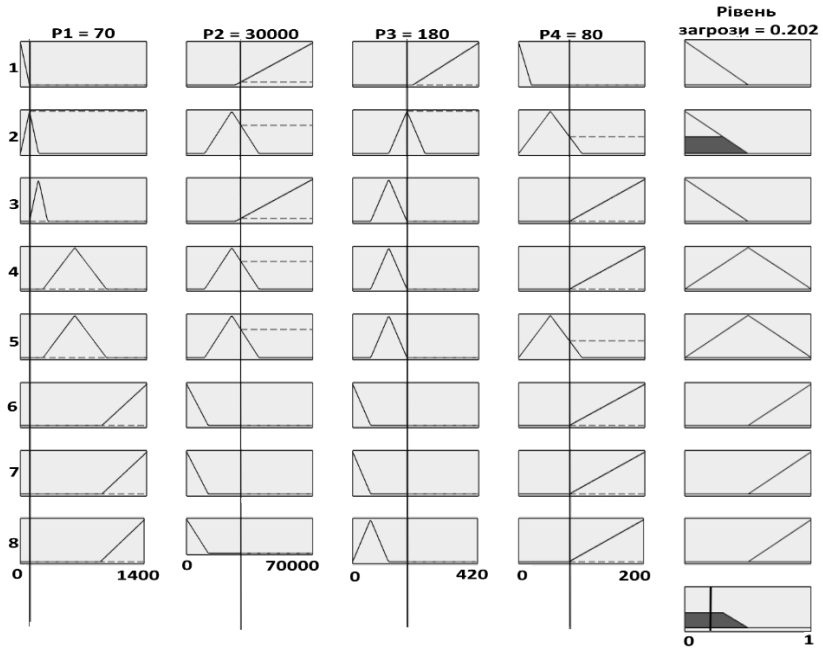


Fig. 6. Graphical interface of the program for viewing rules (Fuzzy Logic Designer Rule Inference) after completing the fuzzy inference procedure
Source: developed by the authors

The diagram of the proposed fuzzy model for threat assessment, developed using MATLAB software, is shown in Fig. 7. The figure shows a static scenario that processes input parameters as constant information for each moment in time. The basic fuzzy inference system assesses the threat level for each set of input parameters.

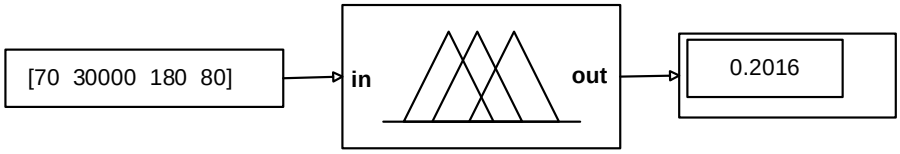


Fig. 7. Static model of fuzzy logic for threat assessment in the MATLAB environment
Source: developed by the authors

A threat with a higher priority characterizes a more dangerous set of input parameters. The threat priority value itself influences decisions on the use of protective measures to neutralize this threat. Table 2 presents the results of modeling in a static test scenario performed over eight time points for a set of four input parameters.

Table 2. *Results of modeling in a static test scenario with an 8-cycle run*

№	P_1	P_2	P_3	P_4	T
1	70	45000	250	80	0.202
2	187	38000	220	90	0.5
3	295	33700	150	100	0.4963
4	450	27700	100	150	0.5
5	792	17000	70	70	0.5
6	955	10200	45	65	0.7661
7	1110	7690	30	20	0.7942
8	1224	5960	20	15	0.8085

The results of testing this model with the parameters shown in Table 2 are shown in Fig. 8.

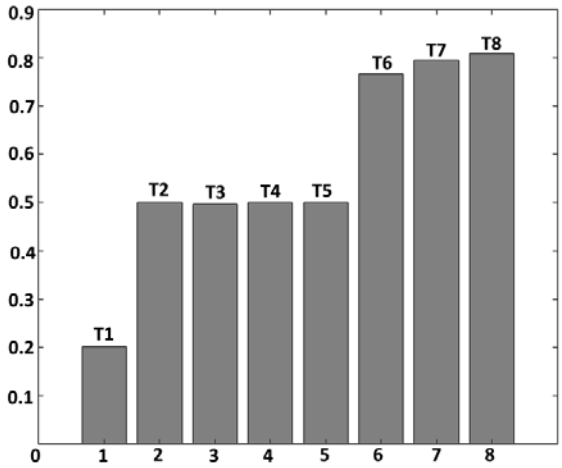


Fig. 8. Threat level assessment during static scenario testing

Assessing the threat level during static scenario testing is crucial for ensuring system security and reliability. It allows you to identify potential risks, detect weaknesses in the architecture or code, and predict possible damage scenarios.

Proper testing helps prevent critical problems, reduces the likelihood of failure, and promotes more informed decisions about protection, such as information or data.

Let's consider several scenarios for modeling dynamic attacks based on given input parameters in order to assess the threats that may arise within their limits. To study the stability and effectiveness of the fuzzy model, a comparison of the results obtained when implementing different scenarios was performed.

Figure 9 shows an example of a dynamic scenario that adaptively analyzes input parameters and uses them as data relevant to solving real-world problems that change over time.

The block diagram of this scenario was developed using the MATLAB software environment.

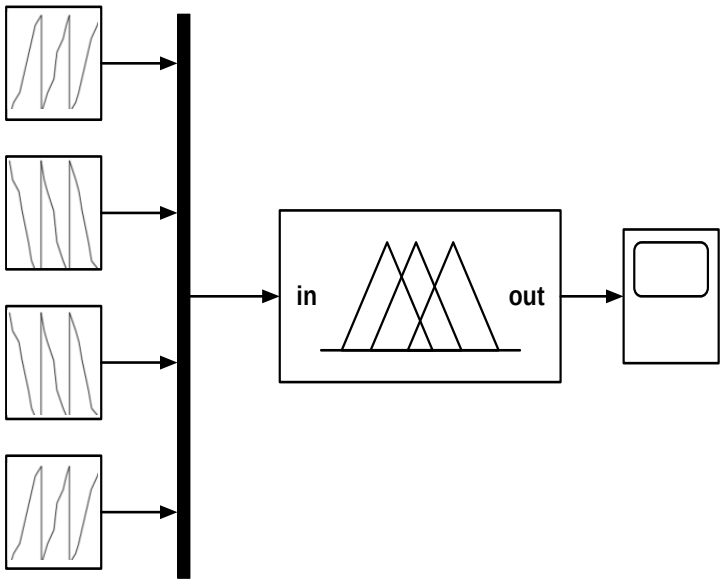


Fig. 9. Dynamic model of fuzzy logic for threat assessment in the MATLAB environment
Source: developed by the authors

In all scenarios considered, the input data of the fuzzy model is formed as a set of input parameters that are received in real time. In the first scenario, each of the parameters changes in a certain way.

In particular, parameter P_1 is characterized by increasing variable values, which at eight time points take the form [70, 187, 295, 450, 792, 955, 1110, 1224].

Parameter P_2 shows a decrease in variable values, which at the same time points are equal to [45000, 38000, 33700, 27700, 17000, 10200, 7690, 5960]. In turn, parameters P_3 and P_4 also decrease over time.

Their values in eight time intervals are [250, 220, 150, 100, 70, 45, 30, 20] and [200, 170, 140, 110, 80, 50, 20, 10], respectively. The characteristics of changes over time for each of the four parameters – P_1, P_2, P_3, and P_4 – used in this scenario are clearly illustrated in Fig. 10.

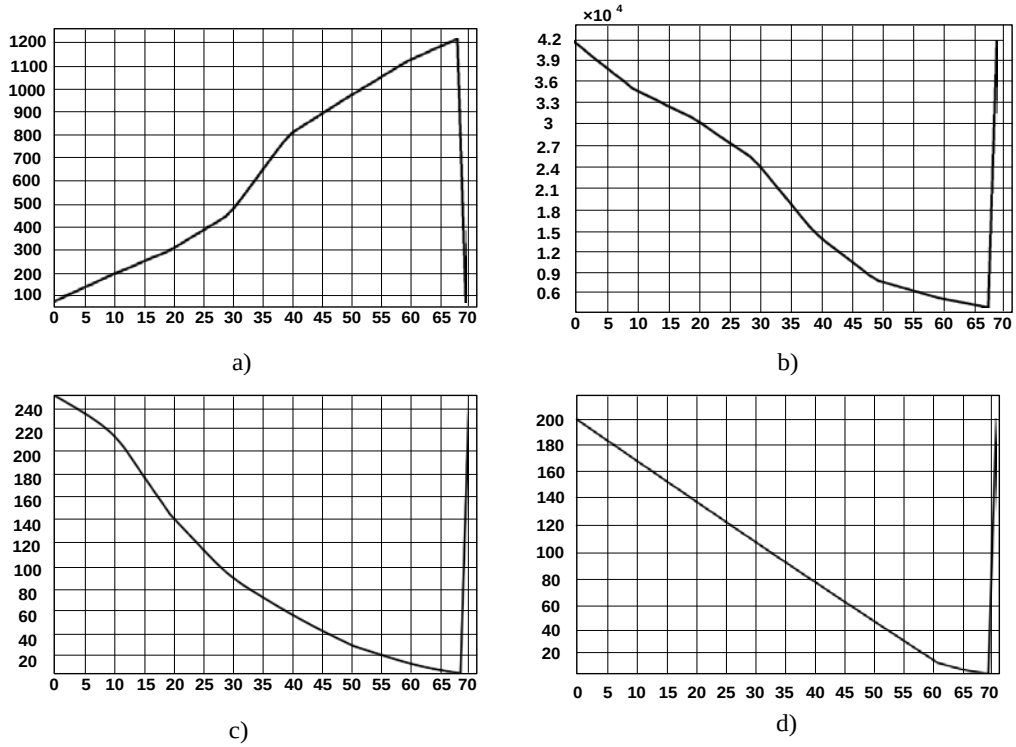


Fig. 10. Dependence of input parameters of the first type (a), second type (b), third type (c), and fourth type (d) on time

Source: developed by the authors

Fig. 11 shows the results of assessing the threat to a fuzzy system. As can be seen, the threat value increases significantly with the growth of parameter P_1, while the other three parameters show a downward trend. In the first scenario, the final threat value is 0.8014, which indicates a fairly high level of threat.

Fig. 11. Output of the fuzzy logic model for assessing threats for the first scenario

Source: developed by the authors

The second scenario assumes that each of the specified parameters changes according to certain patterns over eight different points in time. In particular, parameter P_1 shows a gradual decrease in its values, taking the following indicators: [1500, 1000, 800, 700, 300, 200, 100, 50]. At the same time, parameter P_2, on the contrary, is characterized by an increase in values and is determined by the following values: [2000, 3000, 7000, 10000, 12000, 15000, 16000, 19000]. As for parameters P_3 and P_4, both also show a tendency to increase in value over time, reaching values of [15, 25, 35, 75, 115, 140, 150, 160] for P_3 and [0, 5, 10, 15, 20, 30, 40, 50] for P_4. The dynamics of change for each of these four parameters – P1, P2, P3, and P4 – according to the scenario are illustrated in Fig. 12.

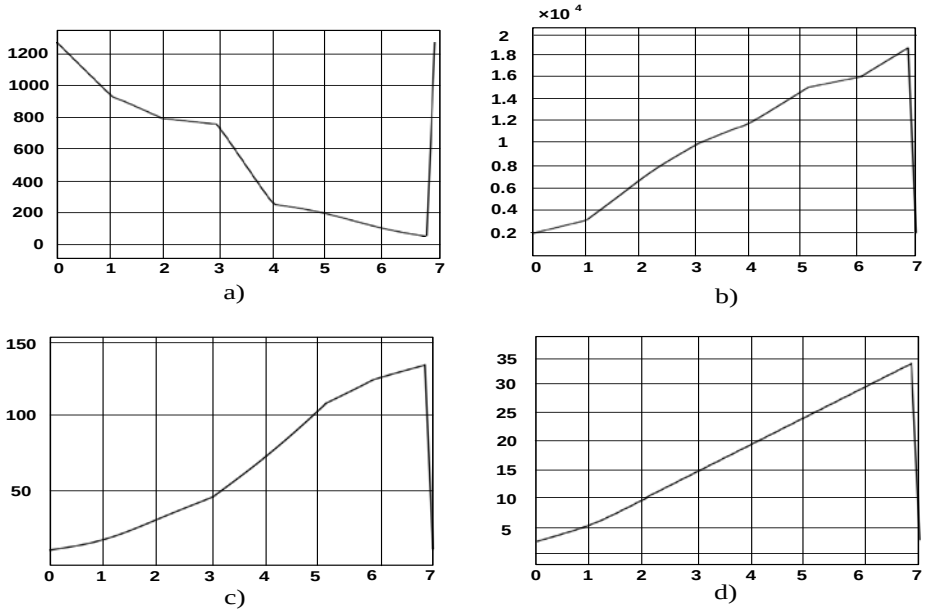


Fig. 12. Dependence of input parameters of the first type (a), second type (b), third type (c), and fourth type (d) on time
Source: developed by the authors

Fig. 13 shows the results of the fuzzy system threat assessment. Given the above conditions, the threat value is significantly reduced, as parameter P_1 tends to decrease, while the other three parameters show an increase.

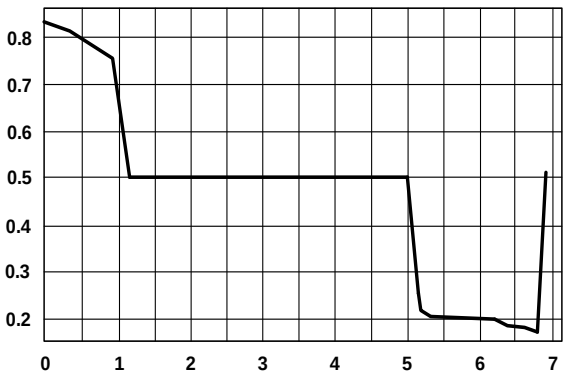


Fig. 13. Output of the fuzzy logic model for assessing threats for the second scenario
Source: developed by the authors

Conclusions

The study presents a detailed description of the approach to threat assessment using fuzzy set theory. An analysis of the parameters necessary for calculating the threat level was carried out, and a multifunctional approach to decision-making based on fuzzy logic was determined. Such a system is an effective tool that optimizes the decision support process, greatly facilitating the work of the specialist conducting the assessment.

The article proposes an algorithm for rating threats on a scale from 0 to 1 using a fuzzy logic system, which ensures high accuracy of results. The developed threat assessment model was tested on a static scenario, as well as on dynamic real-time attack scenarios. A comparison of the simulation results in Table 2 for static threats with the results in Figure 11 for the dynamic scenario demonstrates the high accuracy, reliability, and minimal error rate of the created model. This indicates its effectiveness and potential for use in various conditions.

The developed threat prioritization procedure, based on a fuzzy set model, significantly expands the functionality and allows determining threat levels. This, in turn, creates a basis for making informed decisions on the implementation of measures to counter these threats.

The use of the obtained results as statistical data for calculating the probability of a threat and the method of refining the probabilities of events is appropriate when constructing a functional model for threat detection based on a Bayesian network [21, 22], which will allow refining the probability of its occurrence.

References

1. Zadeh, L. A. (2015), "Fuzzy logic – a personal perspective", *Fuzzy Sets and Systems*. Vol. 281. DOI: https://doi.org/10.1016/j.fss.2015.05.009.
2. Mamdani, E.H., Assilian, S. (1975), "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", *International Journal of Man-Machine Studies*. Vol. 7, No. 1, P. 1-13. DOI: https://doi.org/10.1016/S0020-7373(75)80002-2.
3. Chen G., Trung T. (2000), "Introduction to fuzzy sets", *fuzzy logic, and fuzzy control systems*, Boca Raton, London, New York, No. 1, P. 316. DOI: https://doi.org/10.1201/9781420039818.
4. Espinosa, J., Vandewalle, J., Wertz, V. (2005), "Fuzzy Logic, Identification and Predictive Control", *Vincent Wertz*. – USA: Springer-Verlag London Limited, P. 263. DOI: https://doi.org/10.1007/b138626.
5. Volkov, A., Bazilo, S., Tokar, O., Horbachov, K., Lutsyshyn, A., Zaitsev, I., Iasechko, M. (2022), "Automated assessment of the air situation during the preparation and conduct of combat operations using a decision support system based on fuzzy networks of target installations", *International Journal of Computer*, Vol. 22, No. 11, P. 184–188. DOI: https://doi.org/10.22937/IJCSNS.2022.22.11.26.
6. Volkov, A., Stadnichenko, V., Yaroshchuk, V., Halkin, Y., Tokar, O. (2024), "Proposals for the implementation of a decision support system for air defence fire control based on fuzzy networks of targets", *Systemy Logistyczne Wojsk*, No. 61, P. 211-228. DOI: https://doi.org/10.37055/slw/203558.

7. Azimirada, E., Haddadniab, J. (2015), "Target threat assessment using fuzzy sets theory", *International Journal of Advances in Intelligent Informatics*, Vol 1, No. 2, P. 57–74. DOI: https://doi.org/10.26555/ijain.v1i2.18.
8. Coskun, M., Tasdemir, S. (2022), "Fuzzy logic-based threat assessment application in air defense systems", *IEEE Transactions on Aerospace and Electronic Systems*, No. 59 (3), P. 2245–2251. DOI: https://doi.org/10.1109/TAES.2022.3212032.
9. Murasov, R., Nikitin, A., Meshcheriakov, I. (2024), "Mathematical model of risk assessment of the operation of critical infrastructure objects based on the theory of fuzzy logic", *Social Development and Security*, No. 14 (5), P. 166-174. DOI: https://doi.org/10.33445/sds.2024.14.5.17.
10. Хавіна, І.П., Цуранов, М.В. (2023), "Дослідження механізму нечіткої логіки для оцінки інформаційних ризиків підприємства", *Modern research in science and education*: The 4th International scientific and practical conference (Chicago, USA, December 7-9, 2023), P. 329–335. DOI: https://dspace.univd.edu.ua/handle/123456789/19547.
11. Кочетков, О.В., Гаур, Т.О., Машін, В.М. (2019), "Система оцінки ризиків інформаційної безпеки підприємства на основі нечіткої логіки", *Наукові праці ОНАЗ ім. О. С. Попова*, № 1, С. 97–104. http://nbuv.gov.ua/UJRN/Nponaz_2019_1_14.
12. Amini, A., Jamil, N., Ahmad, A.R., Sulaiman, H. (2017), "A Fuzzy Logic Based Risk Assessment Approach for Evaluating and Prioritizing Risks in Cloud Computing Environment", *Recent Trends in Information and Communication Technology Lecture Notes on Data Engineering and Communications Technologies*, P. 650-659. DOI: https://doi.org/10.1007/978-3-319-59427-9_67.
13. Tuncer, O., Cirpan, H.A. (2023), "Adaptive fuzzy based threat evaluation method for air and missile defense systems", *Information Sciences*, Vol. 643, P. 119-191. DOI: https://doi.org/10.1016/j.ins.2023.119191.
14. Bhattacharyya R., Mukherjee S. (2021), "Fuzzy Membership Function Evaluation by Non-Linear Regression", *An Algorithmic Approach, Fuzzy Information and Engineering*, No. 4, P. 412-434. DOI: https://doi.org/10.1080/16168658.2021.1911567
15. Kecman V. (2001), "Learning and Soft Computing", *Support Vector Machines, Neural Networks, and Fuzzy Logic Models, Massachusetts Institute of Technology Press, Cambridge, MA*, P. 578. DOI: https://doi.org/10.1016/S0925-2312(01)00685-3.
16. Ross T. J. (2004), "Fuzzy Logic with Engineering Applications", P. 628. DOI: https://doi.org/10.1002/9781119994374.
17. Шубін, І.Ю., Ашурова, О. (2020), "Нечіткі множини та нечітка логіка як інструмент формалізації вимог", *Інформаційні системи та технології*: матеріали 9-ї Міжнар. наук.-техн. конф., 17-20 листопада 2020 р. Харків: Друкарня Мадрид, С. 64–68. DOI: https://openarchive.nure.ua/handle/document/16161.
18. Yun, J., Hong, S.-S., Han, M.-M. (2012), "A dynamic neuro fuzzy knowledge-based system in threat evaluation", 13th International Symposium on Advanced Intelligent Systems (ISIS), P. 1601-1605. DOI: https://doi.org/10.1109/SCIS-ISIS.2012.6505178.
19. Mamdani E.H. (1974), "Application of fuzzy algorithms for the control of a simple dynamic plant", In Proc IEEE, P. 121-159. DOI: https://doi.org/10.1049/piee.1974.0328.
20. Fuzzy Inference Process https://www.mathworks.com/help/fuzzy/fuzzy-inference-process.html.
21. Петренко О.С., Петренко О.Є., Бідун А.К. (2025), "Виявлення загроз з застосуванням мережі Байєса", *Системи озброєння і військова техніка*, № 3 (83), С. 129-134. DOI: https://doi.org/10.30748/soivt.2025.83.15.
22. Kozhukhivskyi A., Kozhukhivska O. (2022), "RISK ASSESSMENT MODELING OF ERP-SYSTEMS", *Radio Electronics, Computer Science, Control*. No. 4, P. 149-161. DOI: https://doi.org/10.15588/1607-3274-2022-4-12.

Received (Надійшла) 15.11.2025
Accepted for publication (Прийнята до друку) 30.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Petrenko Oleksii – PhD (Engineering Sciences), Ivan Kozhedub Kharkiv National Air Force University, Senior Research Scientist, Professor at the Department of Combat Use of GBAD Systems with Open Architecture, Kharkiv, Ukraine; e-mail: alexwgs78@gmail.com;
ORCID ID: https://orcid.org/0000-0001-9903-7388

Petrenko Olha – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Information Technology Security, Kharkiv, Ukraine; e-mail: olha.petrenko@nure.ua; ORCID ID: https://orcid.org/0000-0002-7862-5399

Bidun Andrii – Ivan Kozhedub Kharkiv National Air Force University, Teacher at the Department of Combat Use of GBAD Systems with Open Architecture, Kharkiv, Ukraine; e-mail: andriybidun2@gmail.com; ORCID ID: https://orcid.org/0000-0002-4789-9397

Ostrovskyi Zakhar – Ivan Kozhedub Kharkiv National Air Force University, Full-time attendee of the Faculty of Anti-aircraft Missile Forces, Kharkiv, Ukraine; e-mail: ostrovskiyzakhar1@gmail.com; ORCID ID: https://orcid.org/0000-0002-5215-0620

Петренко Олексій Сергійович – кандидат технічних наук, Харківський національний університет Повітряних Сил ім. І. Кожедуба, старший науковий співробітник, професор кафедри бойового застосування зенітного ракетного озброєння з відкритою архітектурою, Харків, Україна.

Петренко Ольга Євгенівна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри безпеки інформаційних технологій, Харків, Україна.

Бідун Андрій Костянтинович – Харківський національний університет Повітряних Сил ім. І. Кожедуба, викладач кафедри бойового застосування зенітного ракетного озброєння з відкритою архітектурою, Харків, Україна.

Островський Захар Назарович – Харківський національний університет Повітряних Сил ім. І. Кожедуба, слухач факультету зенітних ракетних військ, Харків, Україна.

ОЦІНЮВАННЯ РІВНЯ ТА ПРІОРИТИЗАЦІЇ БАГАТОПАРАМЕТРИЧНИХ ЗАГРОЗ ІЗ ВИКОРИСТАННЯМ АЛГОРИТМУ НЕЧІТКОЇ ЛОГІКИ МАМДАНІ ПЕРШОГО ТИПУ

Предметом дослідження є модель оцінювання загроз і визначення їх пріоритетів на основі методів нечіткої логіки. Для побудови моделі використано алгоритм Мамдані першого типу. Розроблену модель оцінювання загроз протестовано на статичному сценарії, а також на динамічних сценаріях атак у реальному часі. Поставлене питання розв'язано із застосуванням методів нечіткої логіки. Для моделювання системи використано *Fuzzy Logic Toolbox* (розширення MATLAB), що містить інструменти для проєктування систем на основі нечіткої логіки. Блок-схеми статичної та динамічної нечіткої моделі оцінювання загроз подано в застосунку *Simulink*. **Мета дослідження** – розроблення й аналіз нечіткої моделі оцінювання загроз і визначення їх пріоритетів для прийняття рішення щодо послідовності заходів з протидії цим загрозам. **Завдання роботи** передбачають обґрунтування

доцільності та ефективності застосування нечітких логічних виразів і операцій нечіткої логіки для формалізованого опису експертних вимог до визначення пріоритетів загроз. Методи нечіткої логіки широко впроваджуються в різноманітних системах управління, зокрема в таких сферах: управління нелінійними процесами, системи із самонавчанням, аналіз ризикових і критичних ситуацій, розпізнавання образів; фінансовий аналіз, дослідження інформації із корпоративних сховищ, оптимізація стратегій управління та координації дій. **Методи, використані в дослідженні:** теорія ймовірності, теорія нечіткої логіки, моделювання. **Досягнуті результати.** Розглянуто можливість застосування нечітких логічних виразів і операцій нечіткої логіки для формалізованого опису експертних критеріїв щодо визначення пріоритетності загроз. Такий підхід забезпечує отримання числових оцінок загроз на основі заданих параметрів, що сприяє точності та гнучкості в процесі їх аналізу. Обґрунтовано можливість застосування нечітких логічних виразів і операцій нечіткої логіки для формалізованого опису експертних вимог до визначення пріоритетів загроз. Це дає змогу отримати числові оцінки загроз на основі заданих вхідних параметрів, забезпечуючи точність і адаптивність у процесі аналізу. У статті запропоновано алгоритм рейтингової оцінки загроз за шкалою від 0 до 1 за допомогою системи нечіткої логіки, що сприяє точним результатам. **Висновки.** Розроблена процедура пріоритизації загроз, побудована на моделі нечітких множин, значно розширює функціональні можливості й дає змогу визначати рівні загроз. Це зі свого боку створює підґрунтя для ухвалення ефективних рішень щодо впровадження заходів із протидії цим загрозам і є основним результатом дослідження.

Ключові слова: модель; нечітка логіка; функція належності; оцінка рівня загроз; визначення пріоритетів загроз; підтримка прийняття рішень; невизначеність; лінгвістичні змінні; нечіткий висновок.

Bibliographic descriptions / Бібліографічні описи

Petrenko, O., Petrenko, O., Bidun, A., Ostrovskyi, Z. (2025), "Model for assessing the level and prioritizing multi-parameter threats using the Mamdani fuzzy logic algorithm of the first type", *Management Information Systems and Devises*, No. 4 (187), P. 220–233. DOI: https://doi.org/10.30837/0135-1710.2025.187.220

Петренко О. С., Петренко О. Є., Бідун А. К., Островський З. Н. Оцінювання рівня та пріоритизації багатопараметричних загроз із використанням алгоритму нечіткої логіки Мамдані першого типу. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 220–233. DOI: https://doi.org/10.30837/0135-1710.2025.187.220

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

УДК 004.056.55:004.272:519.21 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.234

Д. Прокопович-Ткаченко, Л. Рибальченко,
О. Черкаський, Д. Черкаський, Н. Зубченко

ІТЕРАЦІЙНІ ФРАКТАЛЬНІ ВІДОБРАЖЕННЯ ЯК ЗАСІБ ПОСИЛЕННЯ ЕНТРОПІЇ ТА НАДІЙНОСТІ ПРОЦЕСІВ ФОРМУВАННЯ КРИПТОГРАФІЧНИХ КЛЮЧІВ

Предметом дослідження є підвищення надійності формування криптографічних ключів для забезпечення надійної системи захисту. **Мета роботи** – підвищення ефективної ентропії та відтворюваності процесів формування криптографічних ключових матеріалів способом використання фрактальних відображень як ентропійних кондиціонерів у ланцюгу генерації випадкових чисел. З огляду на окреслену мету необхідно виконати такі **завдання**: провести комплексне дослідження сучасних методів генерації випадкових послідовностей і механізмів їх кондиціонування; порівняти традиційні методи нормалізації з фрактальними перетвореннями; розробити модель фрактального підсилення ентропії та визначити її оптимальні параметри для різних типів апаратних платформ. **Упроваджені методи.** У статті вдосконалено метод формування криптографічних ключових матеріалів, оснований на інтеграції фізичних джерел ентропії з фрактальними математичними моделями. **Результати дослідження.** Розроблено структурно-функціональну схему фрактального формування ключів, у якій апаратний генератор істинно випадкових чисел (*True Random Number Generator,* TRNG) забезпечує початкову ентропію, а фрактальний кондиціонер реалізує ітераційні відображення на основі множин Мандельброта, Жулія та систем ітераційних функцій (*Iterated Function Systems*, IFS). У роботі продемонстровано підхід для підвищення стохастичності потоку, зменшення локальної кореляції та стабілізації мінімальної ентропії без втручання в базовий механізм генерації. Запропоновано кондиціонування, що передбачає нормалізацію, пермутаційне змішування та криптографічне стискання з використанням функцій SHAKE128/256 (*Secure Hash Algorithm Keccak Extendable*), що забезпечує рівномірний розподіл бітів і підготовку матеріалу для модулів HKDF (*HMAC-based Key Derivation Function*) або *Argon2id* (*Password-Based Key Derivation Function*). Описана в статті система, що підтримує захищене журналювання параметрів, контроль відтворюваності через часові мітки та односторонні відбитки, полегшує аудит і сертифікацію за стандартами FIPS 140-3 (*Federal Information Processing Standard*) й NIST SP 800-90B (*National Institute of Standards and Technology*). **Висновки.** Проведені експерименти на апаратних платформах *Cortex*-M4F і x86_64 підтвердили зростання мінімальної ентропії з 0.72 до 0.99 і покращення середнього *p-value* в тестах NIST STS з 0.43 до 0.54. Це свідчить про ефективність фрактального кондиціонування як способу підсилення ентропійних властивостей без збільшення апаратних витрат. Отже, розроблена в межах дослідження технологія може бути впроваджена в державні цифрові сервіси, електронні реєстри, платформи електронного підпису й апаратні модулі безпеки та забезпечити високу надійність, прозорий аудит і відповідність міжнародним стандартам криптографічної безпеки.

Ключові слова: фрактальна криптографія; генерація випадкових чисел; ентропійне кондиціонування; формування ключів; криптографічна стійкість.

1. Вступ

Надійність сучасних криптографічних систем безпосередньо залежить від якості ключового матеріалу, який формується на основі випадкових послідовностей [1]. Недостатня ентропія або передбачуваність джерела випадковості призводить до втрати криптографічної стійкості [2], компрометації даних [3] та унеможливлює сертифікацію

систем безпеки [4]. Проблема полягає в тому, що навіть сертифіковані апаратні генератори істинно випадкових чисел (*True Random Number Generator*, TRNG), які відповідають вимогам FIPS 140-3, залишаються вразливими до деградації компонентів [5], зовнішніх електромагнітних впливів [6] і старіння елементної бази [7]. Це може спричиняти появу статистичних дефектів і зниження ефективної мінімальної ентропії [8], що особливо небезпечно для державних цифрових сервісів та інфраструктур електронного підпису [9].

Попри значний прогрес у розробленні квантових генераторів випадкових чисел [11], мемристорних структур [12] і гібридних архітектур [13], наукові публікації останніх років мають низку нерозв'язаних питань [14]. Зокрема залишається складним завдання забезпечення повної відтворюваності процесів генерації [16], стабільності ентропійних параметрів [15] і надійного контролю стану джерел випадковості під час експлуатації [16, 17].

Теоретична актуальність цього дослідження полягає в створенні математичних моделей кондиціонування випадкових потоків, здатних зменшувати локальні кореляції без втрати відтворюваності [18, 19].

Прикладна актуальність зумовлена потребою підвищити якість ключового матеріалу в криптографічних системах без суттєвого ускладнення апаратної частини [20] і з дотриманням вимог стандартів NIST SP 800-90B (*National Institute of Standards and Technology*) [21] та ISO/IEC 20543 [22]. Одним із перспективних напрямів розв'язання окресленої проблеми є використання фрактальних відображень як ентропійних кондиціонерів [23].

Множини Мандельброта [23], Жулія [24] та системи ітераційних функцій (*Iterated Function Systems*, IFS) [25] визначаються високою чутливістю до початкових умов [26] і складною спектральною структурою, що забезпечує ефективне згладжування статистичних нерівномірностей.

На відміну від хаотичних генераторів, фрактальні моделі є детермінованими, тому зберігають відтворюваність і контрольованість процесів, що є необхідною умовою для аудиту [27], стандартизованої верифікації [28] та сертифікації криптографічних модулів [29].

Використання таких відображень, як нелінійних фільтрів між апаратним генератором (TRNG) та функцією формування ключів (*Key Derivation Function*, KDF) дає змогу підвищити ефективну мінімальну ентропію без утручання в базовий механізм генерації [27].

Отже, нерозв'язаною залишається проблема підвищення ефективної ентропії генераторів випадкових чисел у спосіб, який би поєднував математичну строгість [1], стохастичну варіативність [3] і відповідність міжнародним стандартам безпеки [4]. Її розв'язання дає змогу забезпечити стабільне формування криптографічного ключового матеріалу для систем державного значення [10], підвищити стійкість до гібридних кібератак [24] і створити основу для побудови довірчих цифрових сервісів у межах концепції *Zero Trust* (*Zeroization* – безпечне вилучення даних із пам'яті) [25].

2. Аналіз сучасних наукових публікацій і визначення проблеми дослідження

У наукових роботах останнього десятиліття активно досліджуються фізичні генератори істинно випадкових чисел (TRNG), основані на різних природних ефектах: тепловому шумі [1], лавинному шумі діодів [2], флуктуаціях напруги [3], фазових варіаціях осциляторів [4] і квантових процесах [5]. Квантові генератори нового покоління досягли пропускної здатності понад 100 Гбіт/с [6], що підтверджує потенціал таких систем для високошвидкісних застосувань. Водночас у сучасних працях досліджуються гібридні рішення, які поєднують квантові ефекти з мемристорними структурами [7] або спін-орбітальними тунельними переходами [8], що розширює можливості інтеграції TRNG у мікроелектроніку.

Однак низка дослідників звертає увагу на практичні обмеження апаратних генераторів – деградацію елементної бази [9], вплив температури [10], коливання живлення [11] й старіння компонентів [12]. Навіть сертифіковані TRNG-модулі, створені відповідно до вимог FIPS 140-3, демонструють вразливість до статистичних дефектів і активних атак, зокрема електромагнітного впливу або модифікації джерела шуму [13].

Деякі автори доводять, що навіть у сертифікованих реалізаціях можуть виникати кореляції між бітами, які знижують ефективну мінімальну ентропію потоків [14]. Це підтверджує необхідність подальшого вдосконалення процесів ентропійного кондиціонування. Проблема достовірного оцінювання ентропії джерел випадковості посідає ключове місце в сучасних міжнародних стандартах. Документи NIST SP 800-90B [15], FIPS 140-3 (*Federal Information Processing Standard*) [16] і рекомендації ISO/IEC 20543 [17] вимагають наявності прозорих і відтворюваних методик контролю якості ентропії під час експлуатації генераторів. Це стимулює появу математичних моделей кондиціонування сирих бітових потоків, здатних підвищити якість випадковості без втрати відтворюваності результатів [18].

Одним із перспективних напрямів у цій сфері є використання нелінійних динамічних систем, зокрема хаотичних і фрактальних відображень [19]. Фрактальні структури, зокрема множини Мандельброта [20], Жулія [21] або системи ітераційних функцій (IFS) [22], мають високу чутливість до початкових умов [23], складну топологію та спектральну насиченість, що робить їх придатними для стохастичного перетворення даних [24]. Окремі публікації доводять, що використання фрактальних моделей як ентропійних кондиціонерів дає змогу зменшити локальні кореляції, підвищити непередбачуваність і стійкість випадкових послідовностей [25].

Водночас дослідники наголошують, що застосування фрактальних процесів у криптографії потребує суворої верифікації [26] та стандартизованого аудиту [27]. Попередні роботи у сфері хаос-орієнтованих генераторів демонструють можливість суттєвого покращення статистичних показників у тестах NIST STS за правильної параметризації [28]. Однак нестабільність параметрів і складність сертифікації обмежують їх практичне застосування [29]. Тому в останні роки фрактальні методи розглядаються не як самостійне джерело випадковості, а як керований нелінійний фільтр між апаратним

генератором і модулем формування ключів [30], що забезпечує відтворюваність, журналювання параметрів і стандартизований контроль якості [31].

Узагальнюючи результати аналізу сучасних публікацій, можемо зробити висновок, що нерозв'язаною проблемою залишається відсутність адаптивних математичних моделей, здатних стабільно підвищувати ефективну мінімальну ентропію без порушення вимог сертифікації та без збільшення апаратних витрат. Це створює наукову й практичну передумову для розроблення фрактального ентропійного кондиціонера, що поєднує фізичні та математичні механізми генерації випадковості в єдиній архітектурі з підтримкою аудиту й контролю відтворюваності.

3. Мета й завдання дослідження

Аналіз сучасних наукових публікацій засвідчив, що проблема підвищення ефективної ентропії генераторів випадкових чисел у криптографічних системах залишається нерозв'язаною. Наявні підходи, навіть у використанні апаратних генераторів істинно випадкових чисел, не забезпечують достатнього рівня стабільності й відтворюваності, що унеможливлює повну сертифікацію відповідно до вимог FIPS 140-3 і NIST SP 800-90B [1, 15].

Значна частина сучасних рішень орієнтована або на складні квантові процеси, або на апаратні структури, що мають обмежений рівень адаптивності й потребують складного калібрування. Найчастіше відсутні математично формалізовані моделі, здатні в режимі реального часу компенсувати коливання фізичних параметрів генератора й стабілізувати ентропійні показники без втрати продуктивності [2].

Метою цього дослідження є підвищення ефективної ентропії та відтворюваності процесів формування криптографічних ключових матеріалів способом використання фрактальних відображень як ентропійних кондиціонерів у ланцюгу генерації випадкових чисел [3].

Для реалізації окресленої мети необхідно провести комплексне дослідження сучасних методів генерації випадкових послідовностей і механізмів їх кондиціонування, порівняти традиційні методи нормалізації з фрактальними перетвореннями, розробити теоретичну модель фрактального підсилення ентропії та визначити її оптимальні параметри для різних типів апаратних платформ [4, 19, 21].

Особливу увагу приділено створенню архітектури програмно-апаратного конвеєра, який інтегрує фрактальний кондиціонер між TRNG і KDF з підтримкою захищеного журналювання, контролю відтворюваності та сумісності з профілями безпеки FIPS та NIST [5]. Важливим складником роботи стало розроблення методики експериментальної перевірки, що охоплює вимірювання мінімальної ентропії та статистичних показників *p-value* в тестах NIST STS і SP 800-90B, а також оцінювання продуктивності, енергоспоживання та стабільності отриманих результатів на різних класах платформ – від мікроконтролерів *Cortex*-M4F до серверних систем x86_64 [6, 16].

У межах дослідження змодельовано процеси фрактального перетворення з використанням MATLAB *Mobile*, побудовано карти *p-value*, спектральні діаграми та гістограми розподілу бітів для оцінювання стохастичних властивостей потоків.

Результати дали змогу визначити області параметрів, що забезпечують оптимальний баланс між статистичною рівномірністю, швидкодією та енергетичною ефективністю системи [19, 23].

Окремо порівняно фрактальне кондиціонування з класичними методами вирівнювання, такими як схема фон Неймана, універсальне гешування або криптографічне стискання SHA-3/SHAKE (*Secure Hash Algorithm* 3 / *Secure Hash Algorithm Keccak Extendable*), що допомогло визначити переваги фрактального підходу щодо гнучкості параметрів і відтворюваності результатів [10, 15].

На підставі досягнутих результатів сформульовано практичні рекомендації з упровадження фрактального ентропійного кондиціонера в державні інформаційні системи, цифрові реєстри, протоколи TLS 1.3/QUIC й апаратні модулі безпеки HSM (*Hardware Security Module*) і TPM (*Trusted Platform Module*). Запропонований підхід узгоджується з концепцією *Zero Trust* і уможливлює створення контрольованих, відтворюваних і сертифікованих процесів формування ключового матеріалу [24, 25].

Реалізація запропонованої моделі дасть змогу підвищити надійність і стабільність генераторів випадкових чисел, забезпечити прозорий аудит криптографічних процесів і покращити загальний рівень інформаційної безпеки державних і промислових цифрових сервісів. Отже, дослідження передбачає створення концептуально нового підходу до підсилення ентропійних властивостей генераторів випадкових чисел, що поєднує математичну елегантність фрактальних процесів із вимогами криптографічної надійності та регуляторної відповідності.

4. Матеріали й методи дослідження

Після аналізу наявних підходів до генерації ключового матеріалу стає очевидним, що навіть найсучасніші апаратні генератори істинно випадкових чисел мають обмеження, пов’язані з коливаннями фізичних параметрів, статистичними відхиленнями й складністю забезпечення повної відтворюваності результатів [1–4, 10]. Це зумовлює потребу в упровадженні додаткових рівнів ентропійного кондиціонування, здатних зменшувати кореляції в потоках випадкових бітів, вирівнювати розподіл і підвищувати ефективну мінімальну ентропію без втручання в сам механізм генерації [6, 15, 16, 27]. У цьому дослідженні фрактальні відображення розглядаються як математична основа для побудови такого кондиціонера.

Їх ключова властивість полягає у високій чутливості до початкових умов і параметрів, що створює значне різноманіття вихідних послідовностей навіть за мінімальних збурень вхідних даних [19–22]. На відміну від хаотичних генераторів, фрактальні системи забезпечують керованість і повторюваність ітераційного процесу, що є критично важливим для аудиту й сертифікації криптографічних модулів [1, 3, 4].

Мета методу – інтеграція фрактального перетворення в ланцюг між апаратним генератором випадкових чисел і криптографічною функцією формування ключів. Такий підхід дає змогу зменшити локальні залежності у вихідних послідовностях, підвищити якість статистичних показників у тестах NIST STS, забезпечити стабільну мін-ентропію

потоку й покращити стійкість до аналітичних атак, спрямованих на відновлення стану генератора [3, 6, 16, 19–23].

У межах подальшого методологічного дослідження буде запропоновано теоретичну модель фрактального підсилення ентропії, описано алгоритмічні перетворення та оптимізаційні параметри, що дають змогу формалізувати інтеграцію фрактального кондиціонера в типовий криптографічний конвеєр "випадковість – нормалізація – стискання – формування ключа". Така модель поєднує чіткі вимоги інформаційної безпеки зі стохастичними властивостями нелінійних систем, забезпечуючи баланс між математичною обґрунтованістю та практичною придатністю для реалізації в мікроконтролерах і серверних середовищах [10, 15, 19–23, 27–28].

Розглянемо наявні підходи до генерації ключового матеріалу. Джерела випадковості поділяються на фізичні (TRNG) й детерміновані (*Deterministic Random Bit Generator*, DRBG) [1–3]. TRNG використовують фізичні феномени (тепловий шум, лавинний шум діодів, флуктуації фаз) і потребують кондиціонування й оцінювання ентропії [1, 27, 28]. DRBG (HMAC/SHA-2, CTR-AES, Hash-DRBG) потребують якісних *seeds* та періодичного *reseeding* [2, 10]. Для усунення зміщення застосовують схему фон Неймана (дебайасинг), універсальне гешування (*leftover hash lemma*), криптографічне стискання (SHA-3/SHAKE) [6, 15, 16]. Альтернативні ідеї передбачали хаос- та фрактал-орієнтовані генератори, однак їх критика зосереджена на складності коректного безпекового аналізу й ризику некоректної параметризації [20–23]. У цій роботі фрактали *не* замінюють RNG, а є *нелінійним кондиціонером* між TRNG і KDF, контрольованим і аудитовним за профілями NIST/FIPS [1–4, 6].

Подамо теоретичну модель фрактального підсилення ентропії. Позначимо сирий потік TRNG як $X \in \{0,1\}^n$ із щільністю розподілу P_X та оцінкою мін-ентропії $H_\infty(X) = -\log_2\left(\max_x P_X(x)\right)$ [1, 3]. Нехай $F_\theta : \{0,1\}^n \to \{0,1\}^n$ – родина *фрактальних кондиціонерів*, де параметри θ функціонально залежать від вектора середовищних вимірів E (температура, частота, напруга, шум, часові мітки), а також від короткої публічної солі s:

$$Y = F_\theta(X, s), \quad \theta = \Phi(E, s), \tag{1}$$

де F_θ реалізує T ітерацій над комплексним станом $z_{t+1} = z_t^2 + c$, $z_0 = \Psi(x_t)$ з квантуванням біта (-ів) з підпростору фазових портретів [19–21].

Для IFS використовується множина афінних відображень $\{w_i\}_{i=1}^m$ з вагами $p_i(\theta)$ і вибіркою наступного символу $i_t \sim p(\cdot \mid \theta)$, після чого біт формується способом порівняння координат або за знаком [20, 22].

Властивості. Для будь-якої фіксованої θ перетворення F_θ є детермінованим, отже, за інформаційною монотонністю $H_\infty(Y) \le H_\infty(X)$ [10, 16]. Мета – не *нарощування* ентропії в строгому сенсі, а зниження кореляцій / зміщень, поліпшення *ефективної* мін-ентропії оцінками NIST 800-90B, спрощення параметричного аудиту через логування θ та відбитків $h(\theta)$ [1, 3, 16]. Після F_θ застосовується універсальне

гешування h_S із секретним сидом S, наприклад, HKDF-SHA-3 або XOF SHAKE (XOF – *eXtendable Output Function*):

$$K = h_S(Y), \quad |K| \le H_\infty^\varepsilon(Y) - 2\log_2 \frac{1}{\varepsilon}, \tag{2}$$

де H_∞^ε – згладжена мін-ентропія, а (2) випливає з леми про залишковий геш [16].

Стохастична параметризація. Параметр $c = a + ib$ для множин Мандельброта/ Жулія обирається як

$$a = \text{scale}_a\left(\text{ADC}(E) \oplus \text{slice}(X)\right), \quad b = \text{scale}_b\left(\text{TSC} \oplus \text{mix}(X)\right), \tag{3}$$

де $\text{ADC}(E)$ – квантизовані сенсори середовища; TSC – часовий лічильник; $\text{mix}(\cdot)$ – пермутаційне змішування (наприклад, інверсії Фішера – Єтса). Ітераційна глибина T й мапа квантування бітів Q визначають компроміс продуктивності та якості.

Оптимізаційна постановка. Нехай $\hat{H}_\infty(\theta)$ – оцінка мін-ентропії NIST 800-90B для виходу Y. Тоді вибір θ можна формалізувати як

$$\max_{\theta \in \Theta} \hat{H}_\infty(\theta) \quad \text{s.t.} \quad p_j(\theta) \in \left[\alpha_j, \beta_j\right], \ \forall j, \quad \text{pval}_i(\theta) \ge \tau, \ \forall i \in \mathsf{T}, \tag{4}$$

де p_j – параметри стабільності процесу; T – підмножина тестів NIST STS; τ – поріг *p-value* [3]. Розв'язання (4) здійснюється через *grid search* або баєсівську оптимізацію на калібрувальних трасах.

Архітектура системи фрактального формування ключів. Пропонується конвеєр зі стандартними інтерфейсами:

$$\text{TRNG} \xrightarrow{\text{дебайасинг/пермутації}} \text{X} \xrightarrow{F_\theta} \text{Y} \xrightarrow{\text{SHA}-3/\text{XOF}} \text{Z} \xrightarrow{\text{HKDF/Argon2id}} \{\text{K}, \text{salt}, \text{IV}\}.$$

TRNG – апаратний (лавинний/RO/часовий шум) [1, 27, 28]. У блоці нормалізації відбувається пермутаційне змішування блокових підпослідовностей і стискання SHAKE128/256 [6, 15]. Фрактальний кондиціонер F_θ – одна з реалізацій: Мандельброт/Жулія/IFS [19–22]. KDF-модулі: HKDF–SHA-3 (RFC 5869 із заміною на SHA-3) і/або *Argon2id* для стяжіння з паролями/PEM [5, 6, 8].

Журналювання й відтворюваність. Для аудиту логуються публічні артефакти: $meta = \{\text{ver}, s, h(\theta), T, \text{XOF\_mode}, \text{time}, \text{deviceID}\}$ і захищаються через автентифіковане шифрування з асоційованими даними (*Authenticated Encryption with Associated Data,* AEAD) (наприклад, *ChaCha*20–*Poly*1305) з ключем керування, відокремленим від генерації [7, 24]. Секретні дані X, Y, Z не зберігаються. Перевірка сертифікаційною лабораторією виконується відтворенням процесу на *штучних* трасах із симульованими E та валідацією *p-values*/мін-ентропії [1, 3, 4].

Взаємодія з криптосервісами. Інтерфейси для TLS 1.3/QUIC, сховищ ключів HSM/TPM/TEE (*Hardware Security Module* / *Trusted Platform Module* / *Trusted Execution Environment*), державних реєстрів: формування *Salt*/IV/*nonce* (*Salt / Initialization Vector / Number Used Once*); періодичний *reseeding* DRBG; політика *key rotation*; безпечне вилучення буферів [4, 10].

На рис. 1 подано схему формування криптографічних ключів із підсиленням ентропійних властивостей за допомогою фрактальних відображень. Схема демонструє послідовність функціональних блоків, що утворюють єдиний процес оброблення випадкових даних.

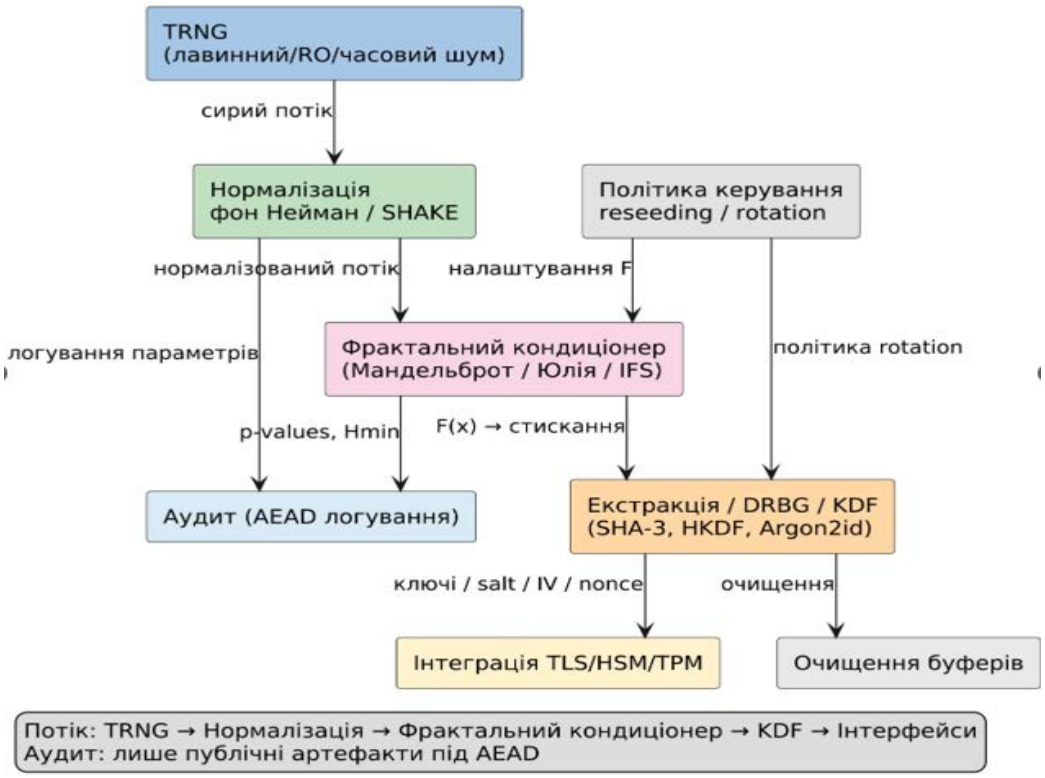


Рис. 1. Схема фрактального формування криптографічних ключів

Схема фрактального формування криптографічних ключів містить такі функціональні блоки:

– TRNG-модуль – джерело первинної ентропії, що генерує випадкові біти на основі фізичних процесів (лавинний, кільцевий або часовий шум);

– блок нормалізації – виконує функції усунення зміщення (фон Нейман), пермутаційного змішування (Фішер – Єтс) та криптографічного стискання (SHAKE128/256);

– фрактальний кондиціонер *F(x)* – центральна функціональна частина схеми, що реалізує ітераційні фрактальні відображення (Жулія, IFS) для зниження кореляцій і підсилення стохастичних властивостей потоку;

– модуль KDF/екстракції – поданий для формування ключів на основі SHA-3, HKDF або *Argon2id*; виконує екстракцію, стискання й нормування даних;

– підсистема журналювання й аудиту – забезпечує контроль відтворюваності процесів за допомогою збереження публічних артефактів (мітки часу, геші параметрів) і реалізує криптографічний захист AEAD (*ChaCha*20-*Poly*1305);

– керувальний блок (*policy_F*) – відповідає за параметри фрактального кондиціонера, порогові *p-value, reseeding* і ротацію ключів;

– модуль очищення пам'яті (ZERO) – виконує безпечне занулення тимчасових даних після завершення циклу формування;

– інтерфейс із зовнішніми криптосервісами – передає сформовані ключі, сіль, IV та *nonce* до HSM, TPM, TEE (*Trusted Execution Environment*) та протоколів TLS 1.3/QUIC.

Схема відтворює послідовність взаємозв'язку між складниками формування криптографічних ключів у процесі оброблення випадкових даних.

5. Результати дослідження

Експериментально перевірено ефективність фрактального кондиціонування під час формування криптографічних ключів.

Основна мета полягає в кількісному підтвердженні того, що інтеграція фрактальних відображень у ланцюг генерації випадкових чисел сприяє зменшенню локальних кореляцій, підвищенню оцінки мін-ентропії та забезпеченню більш рівномірного розподілу бітів у потоках даних.

Для цього проведено серію експериментів на апаратних платформах різного класу – від мікроконтролерів *Cortex*-M4F до серверних систем x86_64 – з метою оцінювання стабільності, продуктивності та енергоспоживання за різних параметрів фрактальних ітерацій.

Експерименти виконувались у кількох режимах, що охоплювали застосування множин Жулія та систем ітераційних функцій (IFS) як математичних основ фрактального кондиціонера. Для кожного сценарію обчислювалися *p-value* в шести базових тестах пакета NIST STS (*Frequency, Runs*, FFT, *ApproxEntropy, Serial, Cumulative Sums*) і оцінювалась мінімальна ентропія відповідно до методики NIST SP 800-90B.

Порівняння результатів до та після кондиціонування дало змогу визначити ступінь покращення статистичних властивостей потоку.

Додаткову увагу приділено комплексній візуалізації процесів, зокрема картам *p-value*, графікам мінімальної ентропії, спектральним густинам і гістограмам розподілу бітів. Ці засоби допомогли оцінити поведінку фрактального кондиціонера як стохастичної системи, виявити області параметрів, що призводять до періодичності, та визначити оптимальні зони стійкого підсилення ентропії.

Отже, розділ поєднує формальні числові результати з інтерпретацією динаміки процесів, що забезпечує повну уяву про якість і відтворюваність фрактального підходу.

Платформи, використані в дослідженні: MCU класу Cortex-M4F (96 MHz, 192 KB RAM); сервер x86_64 (2.4 GHz, 64 GB). TRNG: RO-масив і шум АЦП. KDF: *Keccak* (SHA-3/SHAKE), HKDF, Argon2id [6, 8].

У табл. 1 подано шаблони параметрів налаштування фрактальних відображень. Межі обрано з огляду на чутливість до E та стійкість до *lock-in* у періодичних орбітах [19–22].

Таблиця 1. *Параметри фрактальних відображень*

Відображення	Параметри	Діапазон	Глибина	Квантування
Мандельброт	$a,\ b$	64…256	знак(), знак()	---
Жулія	$a,\ b$	64…256	порогове порівняння координат	---
IFS	$\{A_i,\ p_i\}$; афінні	512…2048	…	індекс за модулем 2

У таблиці продемонстровано базові налаштування параметрів, використані в моделюванні фрактальних відображень різних типів – множин Жулія та систем ітераційних функцій (IFS). Кожне з цих відображень має власну математичну структуру, що визначає спосіб формування випадкових бітів.

Для множини Мандельброта квантування здійснюється за знаком дійсної та уявної частин комплексного числа після кожної ітерації, що формує бітові значення з високою чутливістю до початкових умов. Для множини Жулія застосовується порогове порівняння координат, що дає змогу отримати більш рівномірний розподіл у межах фазового простору.

Система IFS базується на виборі однієї з афінних функцій з набору й на визначенні біта відповідно до індексу перетворення за модулем 2. Діапазони глибини ітерацій (від 64 до 2048) визначають баланс між якістю ентропії та швидкодією системи: збільшення глибини підвищує статистичну рівномірність результату, але зменшує пропускну здатність. Отже, табл. 1 відтворює вибір параметрів, які забезпечують стабільність, відсутність періодичних орбіт і придатність фрактальних структур для використання в криптографічному кондиціонуванні.

Подамо методику оцінювання. Запуск NIST STS (*Frequency, Runs,* FFT, *ApproxEntropy, Serial, Cumulative Sums*) на блоках $\geq 10^6$ біт; усереднення *p-values*; контроль $\alpha = 0,01$ [3]. Оцінки SP 800-90B (*Most Common Value, Collision, Markov, Compression*) до і після кондиціонера [1]. Пропускна здатність (Mb/s), енергія на біт (мкДж/біт), пам’ять (КВ).

У табл. 2 зведено типові результати для MCU (вибірка 64 трас). Спостерігається вирівнювання *p-values* і зростання найнижчої оцінки мін-ентропії на біт.

Таблиця 2. *Узагальнені результати NIST STS (MCU,* 10^6 *біт)*

Метрика	До	Після	Після SHAKE
Середня *p-value* (6 тестів)	0.43	0.52	0.54
Мінімальна *p-value*	0.008	0.021	0.036
$H_{\min}$ (біт/біт)	0.72	0.84	0.99

У табл. 2 узагальнено результати статистичного тестування випадкових послідовностей, отриманих на мікроконтролерній платформі *Cortex*-M4F.

Порівняємо три стани генерації:

1) до застосування фрактального кондиціонера – сирий потік з апаратного TRNG;

2) після фрактального кондиціонера – потік після оброблення нелінійними ітераційними функціями;

3) після SHAKE-стискання – остаточний результат після криптографічного екстрактора.

Основними метриками є середнє й мінімальне значення *p-value* (за шістьма тестами пакета NIST STS) та оцінка мінімальної ентропії H_{min} у бітах на біт. Спостерігається зростання середнього *p-value* з 0.43 до 0.52, а після застосування SHAKE – до 0.54, що свідчить про підвищення якості випадковості. Мінімальна ентропія збільшилася з 0.72 до 0.99, що практично відповідає ідеальній рівномірності розподілу. Отже, у табл. 2 продемонстровано, що фрактальне кондиціонування ефективно зменшує кореляції між бітами, підвищує стохастичність потоку й покращує результати за стандартами NIST SP 800-90B.

Формальне нарощення ентропії детермінованим перетворенням неможливе [10, 16]; покращення пов'язане з *усуненням локальних залежностей* X і підвищенням якості оцінок у сенсі NIST 800-90B унаслідок змішування та стискання. Фінальний екстрактор (SHA-3/HKDF) забезпечує інформаційно-теоретичну гарантію (2) [5, 6, 16].

Зважаючи на параметри продуктивності, необхідно зазначити, що на MCU середня пропускна здатність конвеєра (разом з $T = 128$) становить 2.6 Mb/s; на x86_64 – 180 Mb/s (SHAKE апаратно оптимізований). Енергетичні виміри для MCU: ≈ 0.19 мкДж/біт за умови $V = 3.3$ В.

Комплексна візуалізація результатів моделювання є важливим етапом аналізу процесів формування криптографічних ключів на основі фрактальних відображень. Вона дає змогу не лише якісно оцінити стохастичні властивості сформованих потоків, але й виявити приховані закономірності, зони стабільності та чутливість моделі до варіацій параметрів. В умовах гібридних кібератак, коли порушення випадковості може спричинити деградацію криптографічної стійкості, візуальний аналіз стає ефективним інструментом додаткового контролю коректності ентропійного кондиціонування.

Другим аспектом є інтеграція візуалізації в цикл дослідження: MATLAB *Mobile* використовується як середовище для оперативного моніторингу параметрів фрактального процесу в реальному часі. Це дає змогу виконувати калібрування констант ітерацій, спостерігати зміну *p-value* в тестах NIST та досліджувати спектральні особливості сигналів, сформованих після проходження фрактального кондиціонера. Кольорові карти, побудовані на основі двовимірних матриць параметрів, допомагають визначати області оптимальної ентропії та уникати періодичних орбіт.

Третій аспект полягає в поєднанні аналітичних і візуальних методів. На відміну від чисто статистичного аналізу, графічне подання (карти *p-value*, діаграми ентропії, поверхні фазових станів) дає змогу інтерпретувати поведінку системи як цілісного динамічного об'єкта. Це уможливлює зіставлення результатів різних конфігурацій фрактальних відображень (Мандельброта, Жулія, IFS) та порівняння їх впливу на рівномірність розподілу випадкових бітів.

На рис. 2 подано карту *p-value* – двовимірну теплокарту, де по осях *a* і *b* відкладені параметри фрактального відображення (наприклад, дійсна й уявна частини константи для множини Мандельброта). Колір комірки відповідає середньому значенню *p-value* в тестах NIST STS. Темні ділянки свідчать про зниження випадковості, а світлі – про оптимальні області параметрів.

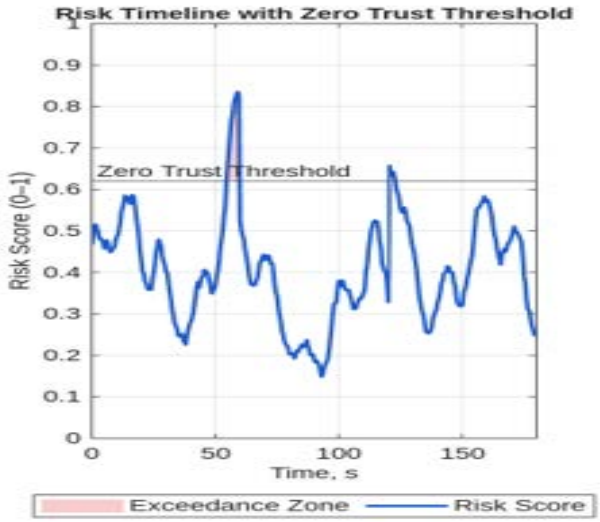


Рис. 2. Карта *p-value* – двовимірна теплокарта

Графік мінімальної ентропії – лінійна залежність H_{min}(t) від номера ітерації або часу (рис. 3). Він демонструє, як змінюється ефективна мін-ентропія потоку під час повторних запусків алгоритму, що дає змогу виявити стабільність процесу кондиціонування.

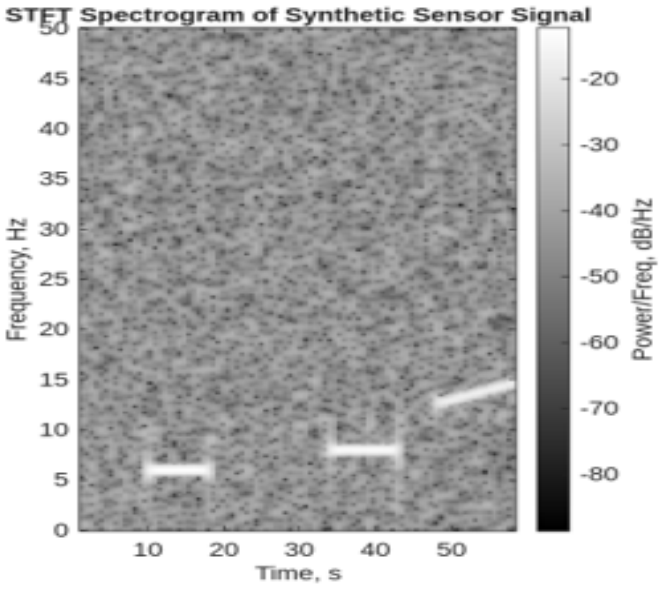


Рис. 3. Графік мінімальної ентропії

На рис. 4 зображено спектральну густину – амплітудно-частотну характеристику, побудовану на основі перетворення Фур'є для послідовності після фрактального кондиціонера. Вона використовується для виявлення прихованих періодичностей і оцінювання рівномірності енергетичного розподілу по частотах.

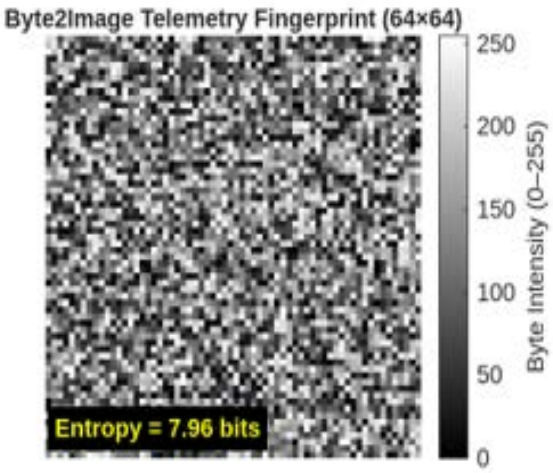


Рис. 4. Спектральна густина

Гістограма розподілу бітів (рис. 5) подана як стовпчиковий графік, що порівнює частоту нулів і одиниць у потоках до й після фрактального кондиціонера. Ідеальний результат визначається симетрією гістограм і мінімальною дисперсією між станами, що свідчить про коректну роботу ентропійного підсилення.

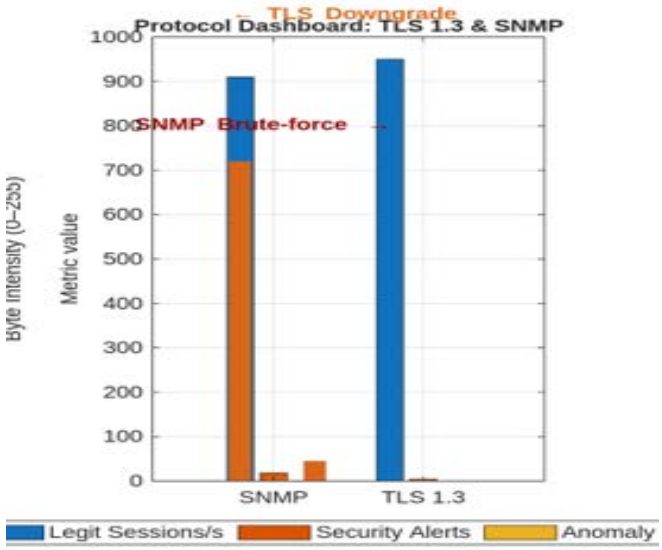


Рис. 5. Гістограма розподілу бітів

У табл. 3 подано узагальнено типові композиції KDF/AEAD для протоколів, практичні сценарії використання сформованого ключового матеріалу в криптографічних протоколах різного рівня.

Таблиця 3. *Композиції KDF та AEAD для протоколів*

Сценарій	KDF	AEAD	Примітки
TLS 1.3/QUIC	HKDF-SHA-3	ChaCha20-Poly1305	сумісність з RFC 8439 [7]
HSM/TPM seed	HKDF-SHAKE	AES-GCM	апаратний прискорювач
Паролі/PEM	Argon2id	XChaCha20-Poly1305	параметри t, m, p [8]

Табл. 3 містить поєднання функцій KDF (*Key Derivation Function*) та алгоритмів AEAD (*Authenticated Encryption with Associated Data*) для трьох основних типів застосувань:

• TLS 1.3/QUIC – сучасні протоколи безпечного з'єднання в мережах, де використовується HKDF-SHA-3 як функція похідного ключа і *ChaCha*20-*Poly*1305 як механізм автентифікованого шифрування;

• HSM/TPM seed – апаратні модулі безпеки, що потребують початкових параметрів для генерації та зберігання ключів. Тут використано HKDF-SHAKE з AES-GCM (блоковим шифром AES в режимі автентифікованого шифрування *Galois / Counter Mode*), який має апаратну підтримку;

• паролі/PEM-файли – сценарії, де ключ генерується з паролю, застосовується *Argon2id* як KDF і X*ChaCha*20-*Poly*1305 для стійкого шифрування.

Зведення демонструє, що фрактально підготовлений ключовий матеріал може бути використаний у будь-яких сучасних протоколах без втрати сумісності з міжнародними стандартами (RFC 5869, RFC 8439, FIPS 140-3) і водночас підвищує рівень надійності, відтворюваності та аудиторської прозорості криптографічних операцій.

Експериментальні дослідження підтвердили ефективність використання фрактальних відображень як ентропійного кондиціонера в криптографічних системах. Для мікроконтролерної платформи *Cortex*-M4F зафіксовано зростання середнього *p-value* з 0.43 до 0.52, а після додаткового криптографічного стискання – до 0.54, що свідчить про покращення рівномірності випадкових послідовностей. Мінімальна оцінка ентропії на біт підвищилась з 0.72 до 0.99, що відповідає практично ідеальному рівню стохастичності за стандартом NIST SP 800-90B.

Аналіз продемонстрував, що поліпшення не є результатом прямого нарощення ентропії, а зумовлено декореляцією, усуненням локальних залежностей і статистичним згладжуванням даних. Візуальні результати (карти *p-value* та спектральні діаграми) підтвердили наявність широких областей стабільних параметрів, де значення показників ентропії зберігаються на високому рівні, що вказує на потенційну придатність фрактальних моделей для практичних реалізацій у критичних системах.

Отже, експерименти продемонстрували, що фрактальне кондиціонування може суттєво підвищити ефективність і відтворюваність процесів генерації криптографічних ключів без збільшення апаратних витрат. Система зберігає сумісність із вимогами стандартів FIPS 140-3 та NIST SP 800-90A/B, забезпечуючи прозорий аудит і можливість сертифікаційної перевірки. Досягнуті результати створюють основу для подальшої оптимізації параметрів фрактальних ітерацій, інтеграції з DRBG-модулями й розроблення енергоефективних апаратних реалізацій.

6. Обговорення результатів дослідження

Результати експериментів демонструють, що фрактальні відображення можуть успішно виконувати функцію ентропійного кондиціонера між апаратним генератором істинно випадкових чисел (TRNG) та модулем формування ключів (KDF). На відміну від традиційних методів нормалізації, таких як схема фон Неймана чи універсальне гешування [6, 15, 16], фрактальні системи виявили здатність ефективно зменшувати кореляції між бітами, не порушуючи відтворюваність процесу.

Це узгоджується з висновками попередніх досліджень щодо застосування нелінійних динамічних систем для посилення випадковості [19–21]. Зіставлення результатів з наявними підходами свідчить, що запропонований фрактальний кондиціонер має переваги над хаос-орієнтованими генераторами, де основним недоліком залишається нестабільність параметрів і труднощі в сертифікації [20–23]. Долучення додаткового рівня детермінованого перетворення дає змогу не нарощувати ентропію безпосередньо (що суперечить інформаційній монотонності [10, 16]), а підвищує її ефективну оцінку внаслідок статистичного згладжування й декореляції даних.

Підвищення мінімальної ентропії з 0.72 до 0.99 підтверджує, що навіть у межах жорстких стандартів NIST SP 800-90B така модель може забезпечити значно кращі результати, ніж класичні апаратні генератори [1, 3, 4, 27]. Експериментальні результати підтверджують доцільність інтеграції фрактального підходу в сучасні криптографічні конвеєри. Сумісність з алгоритмами SHA-3, HKDF та *Argon2id* дає змогу впроваджувати модель у системи державних реєстрів, протоколи TLS 1.3/QUIC і модулі безпеки HSM чи TPM без зміни їх архітектур [5–8, 10].

Це відкриває перспективу створення адаптивних криптосервісів, де параметри фрактального перетворення можуть автоматично підлаштовуватись під зміну середовищних умов (температура, напруга, часові мітки) [19–22]. Водночас завдяки використанню контрольованих параметрів і журналювання процесів зберігається повна можливість аудиту, що відповідає вимогам FIPS 140-3 і профілів AIS 31 (*Application Notes and Interpretations of the Scheme*) [24, 25].

Попри досягнуті позитивні результати, дослідження має певні обмеження. По-перше, моделювання сенсорних даних і параметрів TRNG проводилося в лабораторних умовах, а отже, може не братися до уваги вплив зовнішніх електромагнітних збурень або температурних коливань у реальному середовищі [2, 11]. По-друге, частина фрактальних відображень за певних параметрів виявляє схильність до періодичних орбіт, що потребує введення фільтрів і додаткового контролю стабільності. По-третє, застосування метрик NIST 800-90B має консервативний характер і може занижувати реальну ефективну ентропію [1, 3].

Незважаючи на це, результати переконливо свідчать, що фрактальне кондиціонування є перспективним напрямом підвищення надійності генераторів випадкових чисел у системах критичної інфраструктури, де компрометація ключового матеріалу неприпустима [4, 10]. Порівняння з іншими роботами у сфері підвищення ентропії (зокрема з моделями на основі квантових процесів [6, 7] та мемристорних

генераторів [8, 9]) демонструє, що фрактальний підхід має менші вимоги до апаратних ресурсів і забезпечує кращу масштабованість. У перспективі інтеграція фрактальних кондиціонерів у модулі DRBG як *reseeding*-блоків або їх реалізація в FPGA/ASIC-платформах може створити новий клас гібридних генераторів, що поєднуватимуть переваги фізичної ентропії та контрольованої відтворюваності.

7. Висновки й напрями подальших досліджень

Запропоновано фрактальний ентропійний кондиціонер як керований нелінійний етап між апаратним генератором істинно випадкових чисел (TRNG) і криптографічним екстрактором. Модель не збільшує ентропію в строгому сенсі інформаційної теорії, проте забезпечує статистичне покращення якості випадкових потоків за допомогою декореляції та згладжування розподілу бітів, що підтверджено тестами NIST STS й оцінками мін-ентропії відповідно до методики SP 800-90B [1, 3, 6, 16].

Експериментальні результати на платформах *Cortex*-M4F та x86_64 підтвердили зростання середнього значення *p-value* з 0.43 до 0.54 і підвищення мінімальної ентропії з 0.72 до 0.99, що відповідає рівню високої криптографічної якості [3, 4, 10]. Основними перевагами розробленої моделі є низька вартість, програмна гнучкість, відтворюваність параметрів через односторонні відбитки, можливість захищеного журналювання та відповідність вимогам стандартів FIPS 140-3 і ISO/IEC 20543 [4, 27, 28].

Така архітектура забезпечує прозорість процесів генерації, аудит параметрів і сертифікаційну перевірку без розкриття секретної інформації, що є критично важливим для державних цифрових сервісів і систем критичної інфраструктури. Отримані результати свідчать про практичну придатність фрактального кондиціонера для використання в державних цифрових реєстрах і сервісах електронного урядування. Зокрема система може бути інтегрована в механізми генерації та ротації ключів у платформах типу "Дія", у засоби електронного підпису та в системи електронної ідентифікації (eID, *eSignature*), де потрібен високий рівень ентропії для створення приватних ключів [4, 10, 24].

У державних платіжних і фінансових шлюзах кондиціонер може забезпечити додаткову стійкість до компрометації ключів та імітаційних атак, а в апаратних модулях HSM, TPM та TEE – покращити контроль відтворюваності генерації [5–8]. Завдяки підтримці захищеного журналювання параметрів і часових міток фрактальний кондиціонер відповідає концепції *Zero Trust* і дає змогу створювати довірчі середовища для перевірки достовірності криптографічних процесів без необхідності доступу до внутрішніх станів TRNG [24, 25]. Таке рішення підвищує рівень інформаційної безпеки державних інформаційних систем, зберігаючи прозорість і керованість процесів.

Подальші дослідження мають бути спрямовані на формальну верифікацію змішувальних властивостей окремих систем ітераційних функцій (IFS) [19–22], інтеграцію фрактального кондиціонера з детермінованими генераторами DRBG як *reseeding*-блоків [2, 10], а також на розроблення апаратних реалізацій для енергообмежених платформ IoT та IIoT.

Особливу увагу необхідно приділити розширеній сертифікації модуля за профілями безпеки FIPS 140-3, AIS 31 і ISO/IEC 17825, що уможливить його практичне впровадження в національні криптосервіси.

Отже, результати дослідження підтверджують доцільність використання фрактальних методів для підсилення ентропійних властивостей генераторів випадкових чисел. Запропонований підхід є ефективним рішенням для побудови національної інфраструктури безпечних цифрових сервісів, здатних протидіяти гібридним кібератакам, забезпечувати довгострокову криптографічну надійність і відповідати міжнародним стандартам інформаційної безпеки [1, 3, 4, 6, 10, 24].

References

1. Herrero-Collantes M., Garcia-Escartin J. C. (2017), "Quantum random number generators", *Reviews of Modern Physics,* Vol. 89, 015004. DOI: https://doi.org/10.1103/RevModPhys.89.015004
2. Ma X., Yuan X., Cao Z., Qi B., Zhang Z. (2016), "Quantum random number generation", *Quantum Information*, Vol. 2, 16021. DOI: https://doi.org/10.1038/npjqi.2016.21
3. Meiser L. C., Koch J., Antkowiak P. L., Stark W. J., Heckel R., Grass R. N. (2020), "DNA synthesis for true random number generation", *Nature Communications*, Vol. 11, 5869. DOI: https://doi.org/10.1038/s41467-020-19757-y
4. Kim G., Lee J.-S., Lee D., et al. (2021), "Self-clocking fast and variation tolerant true random number generation using a NbO_x memristor", *Nature Communications*, Vol. 12, 2906. DOI: https://doi.org/10.1038/s41467-021-23184-y
5. Jiang H., Belkin D., Spector J., et al. (2017), "A novel true random number generator based on a stochastic diffusive memristor", *Nature Communications,* Vol. 8, 882. DOI: https://doi.org/10.1038/s41467-017-00869-x
6. Bruynsteen C., Witteveen J., Van den Berghe S., et al. (2023), "100-Gbit/s integrated quantum random number generator", *PRX Quantum*, Vol. 4, 010330. DOI: https://doi.org/10.1103/PRXQuantum.4.010330
7. Gras G., Stipčević M., Rarity J. G., et al. (2021), "Quantum entropy model of an integrated quantum-random number generator", *Physical Review Applied*, Vol. 15, 054048. DOI: https://doi.org/10.1103/PhysRevApplied.15.054048
8. Li X.-H., Zhang K.-Q., Ma X., et al. (2023), "True random number generator based on spin–orbit torque magnetic tunnel junctions", *Applied Physics Letters*, Vol. 123, 142403. DOI: https://doi.org/10.1063/5.0171768
9. Li X.-H., Zhang K.-Q., Ma X., et al. (2024), "Spin–orbit torque true random number generator with thermal stability", *Applied Physics Letters,* Vol. 124, 102409. DOI: https://doi.org/10.1063/5.0193558
10. Woo K.-S., Kim S.-I., Lee D., et al. (2024), "True random number generation using spin crossover in $LaCoO_3$", *Nature Communications,* Vol. 15, 49149. DOI: https://doi.org/10.1038/s41467-024-49149-5
11. Kim K.-S., Woo K.-S., Lee D., et al. (2021), "A high-speed true random number generator based on a Cu_xTe_{1-x} diffusive memristor", *Advanced Intelligent Systems,* Vol. 3, 2100062. DOI: https://doi.org/10.1002/aisy.202100062
12. Shrimpton T., Terashima R. S. (2015), "A provable-security analysis of Intel's Secure Key RNG. In: Oswald E., Fischlin M. (eds.) ", *Advances in Cryptology – EUROCRYPT 2015. Lecture Notes in*

Computer Science, Vol. 9056, Springer, P. 77–100. DOI: https://doi.org/10.1007/978-3-662-46800-5_4

13. Dodis Y., Pointcheval D., Ruhault S., Vergnaud D., Wichs D. (2013), "Security analysis of pseudo-random number generators with input: /dev/random is not robust", *Proceedings of the 2013 ACM Conference on Computer and Communications Security (CCS),* P. 647–658. DOI: https://doi.org/10.1145/2508859.2516653
14. Everspaugh A., Zhai Y., Jellinek R., Ristenpart T., Swift M. (2014), "Not-So-Random Numbers in Virtualized Linux and the Entropy Problem", *IEEE Symposium on Security and Privacy,* P. 559–574. DOI: https://doi.org/10.1109/SP.2014.42
15. Lubicz D., Fischer V. (2024), "Entropy computation for oscillator-based physical random number generators", *Journal of Cryptology*, Vol. 37. 13. DOI: https://doi.org/10.1007/s00145-024-09494-6
16. Mannalatha V., Mishra S., Pathak A. (2023), "A comprehensive review of quantum random number generators: concepts, classification and the origin of randomness", *Quantum Information Processing,* Vol. 22, 439. DOI: https://doi.org/10.1007/s11128-023-04175-y
17. Nannipieri P., Di Matteo S., Baldanzi L., et al. (2021), "True Random Number Generator Based on Fibonacci–Galois Ring Oscillators for FPGA", *Applied Sciences*, Vol. 11, 3330. DOI: https://doi.org/10.3390/app11083330
18. Matuszewski Ł., Poźniak K., Szczepaniak P. (2024), "Ring oscillators with additional phase detectors as a random source in a random number generator", *Entropy*, Vol. 27, 15. DOI: https://doi.org/10.3390/e27010015
19. Frustaci F., Spagnolo F., Perri S., Corsonello P. (2023), "A high-speed FPGA-based true random number generator using metastability with clock managers*", IEEE Transactions on Circuits and Systems II: Express Briefs,* Vol. 70, P. 756–760. DOI: https://doi.org/10.1109/TCSII.2022.3211278
20. Ni T., Peng Q., Bian J., et al. (2023), "Design of true random number generator based on multi-ring convergence oscillator using short pulse enhanced randomness", *IEEE Transactions on Circuits and Systems I: Regular Papers,* Vol. 70, P. 5074–5085. DOI: https://doi.org/10.1109/TCSI.2023.3287162
21. Della Sala R., Bellizia D., Scotti G. (2022), "A novel ultra-compact FPGA-compatible TRNG architecture exploiting latched ring oscillators", *IEEE Transactions on Circuits and Systems II: Express Briefs,* Vol. 69, P. 1672–1676. DOI: https://doi.org/10.1109/TCSII.2021.3121537
22. Park J., Kim B., Sim J.-Y. (2022), "A PVT-tolerant oscillation-collapse-based true random number generator with an odd number of inverter stages", *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol. 69, P. 4058–4062. DOI: https://doi.org/10.1109/TCSII.2022.3184950
23. Jin L., Yi M., Xiao Y., Sun L., Lu Y., Liang H. (2023), "A dynamically reconfigurable entropy source circuit for high-throughput true random number generator", *Microelectronics Journal,* Vol. 133, 105690. DOI: https://doi.org/10.1016/j.mejo.2023.105690
24. Cherkaoui A., Fischer V., Fesquet L., Aubert A. (2023), "A very high speed true random number generator with entropy assessment", *In: Cryptographic Hardware and Embedded Systems – CHES 2023, LNCS*, Vol. 8086. Springer, P. 179–196. DOI: https://doi.org/10.1007/978-3-642-40349-1_11
25. Bayon P., Bossuet L., Aubert A., Fischer V., Poucheret F., Robisson B., Maurine P. (2012), "Contactless electromagnetic active attack on ring oscillator-based true random number generator", *In: Constructive Side-Channel Analysis and Secure Design (COSADE 2012). LNCS*, Vol. 7275, Springer, P. 151–166. DOI: https://doi.org/10.1007/978-3-642-29912-4_12
26. Martínez A. C., Bulygin S., Pivoluska M., et al. (2018), "Advanced statistical testing of quantum random number generators", *EPJ Quantum Technology,* Vol. 5, 26. DOI: https://doi.org/10.1140/epjqt/s40507-018-0080-2
27. Haylock B., Thearle O., Assad S. M., et al. (2019), "Multiplexed quantum random number generation", *Quantum*, Vol. 3, 141. DOI: https://doi.org/10.22331/q-2019-05-13-141

28. Liu Y., Shalm L. K., et al. (2018), "High-speed device-independent quantum random number generation with entangled photons", *Physical Review Letters*, Vol. 120, 010503. DOI: https://doi.org/10.1103/PhysRevLett.120.010503
29. Bierhorst P., Knill E., Glancy S., et al. (2018), "Experimentally generated randomness certified by the impossibility of superluminal signals", *Nature*, Vol. 556, P. 223–226. DOI: https://doi.org/10.1038/s41586-018-0019-0
30. Cherkaoui A., Fischer V., Aubert A., Fesquet L. (2013), "A self-timed ring based true random number generator", *IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, P. 99–106. DOI: https://doi.org/10.1109/ASYNC.2013.28
31. Kim S.-I., Woo K.-S., Lee D., et al. (2024), "Cryptographic transistor (“cryptoristor”) for true random number generators", *Science Advances*, Vol. 10. DOI: https://doi.org/10.1126/sciadv.adk6042

Received (Надійшла) 19.09.2025
Accepted for publication (Прийнята до друку) 30.11.2025
Publication date (Дата публікації) 28.12.2025

Відомості про авторів / About the Authors

Прокопович-Ткаченко Дмитро Ігорович – кандидат технічних наук, доцент, Університет митної справи та фінансів, завідувач кафедри кібербезпеки та інформаційних технологій, Дніпро, Україна; e-mail: omega2417@gmail.com; ORCID ID: https://orcid.org/0000-0002-6590-3898

Рибальченко Людмила Володимирівна – кандидат економічних наук, доцент, Університет митної справи та фінансів, доцент кафедри кібербезпеки та інформаційних технологій, Дніпро, Україна; e-mail: luda_r@ukr.net; ORCID ID: https://orcid.org/0000-0003-0413-8296

Черкаський Олександр Валерійович – Університет митної справи та фінансів, незалежний дослідник у сфері кібербезпеки, Дніпро, Україна; e-mail: asherjoseph.c@gmail.com; ORCID ID: https://orcid.org/0009-0006-3105-5217

Черкаський Давид Олександрович – Університет митної справи та фінансів, незалежний дослідник у сфері кібербезпеки, Дніпро, Україна; e-mail: cherkaskyidavid12@gmail.com; ORCID ID: https://orcid.org/0009-0003-8516-6252

Зубченко Назар Станіславович – Університет митної справи та фінансів, незалежний дослідник кафедри кібербезпеки та інформаційних технологій, Дніпро, Україна; e-mail: nazik3110@gmail.com; ORCID ID: https://orcid.org/0009-0000-9973-7624

Prokopovych-Tkachenko Dmytro – PhD (Engineering Sciences), Associate Professor, University of Customs and Finance, Head at the Department of Cybersecurity and Information Technologies, Dnipro, Ukraine.

Rybalchenko Liudmyla – PhD (Economic Sciences), Associate Professor, University of Customs and Finance, Associate Professor at the Department of Cybersecurity and Information Technologies, Dnipro, Ukraine.

Cherkaskyi Oleksandr – University of Customs and Finance, Independent researcher in the field of cybersecurity, Dnipro, Ukraine.

Cherkaskyi David – University of Customs and Finance, Independent researcher in the field of cybersecurity, Dnipro, Ukraine.

Zubchenko Nazar – University of Customs and Finance, Independent Researcher, Department of Cybersecurity and Information Technologies, Dnipro, Ukraine.

ITERATIVE FRACTAL MAPPINGS AS A MEANS OF ENHANCING ENTROPY AND RELIABILITY OF CRYPTOGRAPHIC KEY GENERATION PROCESSES

The subject of the study is to improve the reliability of cryptographic key generation to ensure a reliable protection system. **The aim of the work** is to increase the effective entropy and reproducibility of cryptographic key material generation processes by using fractal mappings as entropy conditioners in the random number generation chain. Given the stated goal, the following **tasks** must be performed: conduct a comprehensive study of modern methods of generating random sequences and mechanisms for their conditioning; compare traditional normalization methods with fractal transformations; develop a model of fractal entropy amplification and determine its optimal parameters for different types of hardware platforms. **Methods used.** The article improves the method of forming cryptographic key materials based on the integration of physical entropy sources with fractal mathematical models. **Research results.** A structural and functional scheme for fractal key generation has been developed, in which a hardware true random number generator (TRNG) provides the initial entropy, while a fractal conditioner implements iterative mappings based on Mandelbrot and Julia sets and Iterated Function Systems (IFS). The paper demonstrates an approach to increase the stochasticity of the stream, reduce local correlation, and stabilize the minimum entropy without interfering with the basic generation mechanism. The proposed conditioning involves normalization, permutation mixing, and cryptographic compression using SHAKE128/256 (Secure Hash Algorithm Keccak Extendable) functions, which ensure uniform bit distribution and prepare material for HKDF (HMAC-based Key Derivation Function) or *Argon2id* (*Password-Based Key Derivation Function*). The system described in the article supports secure parameter logging, reproducibility control through timestamps and one-way hashes, and facilitates auditing and certification according to FIPS 140-3 (*Federal Information Processing Standard*) and NIST SP 800-90B (*National Institute of Standards and Technology*) standards. **Conclusions.** Experiments conducted on *Cortex*-M4F and x86_64 hardware platforms confirmed an increase in minimum entropy from 0.72 to 0.99 and an improvement in the average *p-value* in NIST STS tests from 0.43 to 0.54. This demonstrates the effectiveness of fractal conditioning as a way to enhance entropy properties without increasing hardware costs. Therefore, the technology developed within the framework of the study can be implemented in state digital services, electronic registries, electronic signature platforms, and hardware security modules to ensure high reliability, transparent auditing, and compliance with international cryptographic security standards.

Keywords: fractal cryptography; random number generation; entropy conditioning; key generation; cryptographic resistance.

Бібліографічні описи / Bibliographic descriptions

Прокопович-Ткаченко Д. І., Рибальченко Л. В., Черкаський О. В., Черкаський Д. О., Зубченко Н. С. Ітераційні фрактальні відображення як засіб посилення ентропії та надійності процесів формування криптографічних ключів. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 234–253. DOI: https://doi.org/10.30837/0135-1710.2025.187.234

Prokopovych-Tkachenko, D., Rybalchenko, L., Cherkaskyi, O., Cherkaskyi, D., Zubchenko, N. (2025), "Iterative fractal mappings as a means of enhancing entropy and reliability of cryptographic key generation processes", *Management Information Systems and Devises*, No. 4 (187), P. 234–253. DOI: https://doi.org/10.30837/0135-1710.2025.187.234

UDC 004.416 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.254

A. Chupryna, V. Repikhov

REFERENCE MODEL FOR PREVENTIVE SOFTWARE MAINTENANCE

Subject of the study. The study focuses on preventive software maintenance processes, particularly the formalization of mechanisms for monitoring, degradation-risk identification, and proactive decision-making within the software life cycle. **The purpose** of the research is to develop a reference model for preventive maintenance that enables systematic detection of deviations from reference operating modes, assessment of their temporal dynamics, and generation of well-grounded preventive actions prior to the emergence of failures. **Research tasks.** The tasks include: establishing a classification of metrics and features for characterizing software states; constructing a formal apparatus for preventive identification using an extended comparator identification method; defining the structural components of the model; and developing models for monitoring, risk analysis, maintenance-task derivation, and adaptive parameter adjustment. **Research methods.** The methodological basis comprises the extended comparator identification method, which provides comparative evaluation of current software states relative to reference modes and quantifies deviation trajectories. Feature aggregation, normalization, formal description of the functional state space, trend analysis in temporal windows, and adaptive feedback mechanisms are applied. **Results achieved.** A five-component reference model is developed, incorporating modules for state monitoring, preventive identification, task generation, action-effect evaluation, and adaptive tuning. The model supports early detection of latent degradation through analysis of deviation trajectories, integrates with CI/CD, DevOps and MLOps pipelines, and establishes a formalized decision-support mechanism for proactive maintenance. **Conclusions.** The proposed model constitutes a comprehensive formal framework for preventive software maintenance, aimed at early detection of potential failures and long-term system stability. Its adaptive nature ensures relevance under dynamic operational conditions, while comparator identification enhances interpretability and validity of maintenance decisions. The results provide a foundation for intelligent maintenance systems and future research on automated formation of reference states and ML-enhanced risk assessment.

Keywords: preventive maintenance; comparator identification; monitoring; technical debt; degradation prediction; software quality; DevOps; MLOps.

Introduction

Software (SW) requires continuous maintenance throughout its entire life cycle. The support and maintenance phase is the longest and most resource-intensive: according to estimates, it can account for up to 80–90 % of the total cost of a software product's life cycle [1]. Maintenance covers various activities, from defect correction (corrective maintenance) and adaptation to environmental changes (adaptive maintenance) to functionality expansion (perfective maintenance). Preventive maintenance is a separate category – proactive actions aimed at anticipating potential problems before they arise. According to the international standard ISO/IEC/IEEE 14764:2022 [2], preventive maintenance is defined as "modification of a software product after its delivery, aimed at correcting hidden defects before they manifest themselves in the operating system" [2]. This proactive approach is especially important for critical software systems that have high requirements for security and availability (e.g., medical systems) [3].

The relevance of developing preventive maintenance is driven by the need to improve the reliability and quality of software products amid their growing complexity and the multitude of factors affecting the effectiveness of maintenance [4]. In traditional practice, support is often provided reactively, i.e., only after failures or user complaints have occurred. Neglecting proactive measures leads to the accumulation of technical debt and increased long-term support costs [5, 6]. Preventive actions, on the other hand, make it possible to avoid costly emergency interventions in the future. As noted in [1], the cost of emergency fixes usually exceeds the resources required to prevent them. Therefore, the transition from responding to problems to proactively preventing them is a promising direction for the development of software engineering, designed to ensure high reliability and cost-effectiveness of software maintenance.

At the global level, it is emphasized that preventive SW maintenance should be carried out in a planned and systematic manner. To implement this approach within the life cycle, it is advisable to apply the reference model for preventive support (RMPS) – a specialized framework that integrates into the process of software development and operation. The reference model provides for continuous monitoring of the state of the software system and its components, evaluation of the metrics and indicators obtained, and adaptation of the support system based on this data. Based on established criteria, the model generates recommendations for preventive actions: it determines what needs to be improved or corrected and when it should be forwarded to the development team for implementation. This ensures that the necessary maintenance work is performed in a timely manner before potential problems turn into actual failures or incidents for users. The proposed preventive maintenance framework serves as a decision-making support tool for project management, allowing you to proactively manage the evolution of the software product and maintain its high quality.

Analysis of current publications on the issue of preventive maintenance

Currently, preventive maintenance is not yet widespread in the typical software development cycle. Many organizations still focus primarily on responding to already identified defects and user requests, while systematic prevention of potential problems is implemented in a fragmented manner. At the same time, scientific literature and standards note the significant potential of the preventive approach. Methods for predicting defects and "fatigue" in software code are being developed using data analysis and machine learning, and code "smells" and ways to reduce technical debt are being investigated. The prospects for implementing software maintenance models are linked to increasing the reliability and security of software products, reducing downtime and the number of critical failures, and optimizing support costs through the timely implementation of necessary updates.

International standards for the life cycle and quality of software systematically describe maintenance [2, 7]. The ISO/IEC/IEEE 14764 standard [2] defines four types of maintenance: corrective, adaptive, perfective, and preventive. Preventive maintenance is aimed at improving reliability, security, and facilitating further development and support of the system. The ISO/IEC/IEEE 14764 [2] standard defines maintenance as an integral part of the software life cycle and describes processes that can be applied to different classes of software products. ISO/IEC 25010 [7] sets out a model of SW quality and identifies eight characteristics, including

reliability, security, and maintainability. Preventive maintenance directly enhances these characteristics: it increases reliability by eliminating latent defects, improves maintainability (refactoring, updating documentation, test base), and strengthens security by promptly fixing vulnerabilities and keeping dependencies up to date. Thus, according to international standards, preventive maintenance is a formalized, planned activity that must be performed throughout the entire life cycle of a software product.

Despite the official recognition of its importance in the relevant standards, preventive maintenance receives significantly less attention in scientific literature compared to other forms of SW maintenance. An analysis of the literature over the past five years shows that preventive maintenance remains an under-researched area. For example, in a systematic review [6], only 7 of the 111 papers considered mentioned the benefits associated with preventive maintenance. Synthesis reviews (in particular [1]) emphasize that although support can account for up to 70–90 % of the SW life cycle, preventive actions remain secondary compared to defect correction or new feature implementation. A reactive support approach prevails over proactive intervention. This creates an obvious gap between the theoretical significance of preventive maintenance and its practical implementation.

Despite the general lack of research on the topic, some publications offer tools, models, or metrics directly related to preventive maintenance. Works [8–10] explored the possibility of predicting code changes (change-proneness) using software product metrics, commit history, and other factors. A vector of features is formed for each module from the history of changes: product metrics (complexity, size, "smells"), process metrics (code churn, edit frequency, defect commit density, age of last refactoring), social metrics (expertise and "ownership" of the module, number of authors, review duration). These features are used to train models that return the probability that a component will need to be changed or will produce a defect in the near future [8–10]. Empirical studies show the effectiveness of this approach for ranking risky components and planning preventive actions. Such predictive models allow identifying "vulnerable" areas of code for targeted refactoring or additional testing, which directly corresponds to the tasks of preventive support. Study [11] examines the use of machine learning algorithms to identify classes in which the presence or absence of "code smells" correlates with the probability of defects occurring. The results confirmed that removing such symptoms before real errors occur is an effective preventive approach. A number of studies (in particular [5, 12, 13]) substantiate the idea of systematic detection, registration, and repayment of technical debt through planned refactoring, library updates, etc.

Mobile applications create additional challenges for preventive support: frequent operating system updates, device diversity, high sensitivity to productivity and resources, the need for regular updates of dependencies and compliance with app store requirements. Works [14–16] emphasize the need for systematic planning of preventive releases, testing on OS beta versions, and monitoring user feedback as a source of signals for proactive improvements. Some authors suggest creating a maintenance calendar that takes into account the platform update cycle and user expectations [14].

Research on modern CI/CD and DevOps processes demonstrates the growing role of automated quality control tools that can signal the need for preventive intervention.

The maintenance of software systems with machine learning components is of particular scientific interest. In the context of ML systems, there is a similar increase in attention to monitoring data and model drift as triggers for proactive updates [3, 14, 15]. The first systematic review of the challenges of maintaining ML systems was published only in 2022 [16], which indicates the novelty and insufficient study of the topic. The main challenges are related to model degradation, the complexity of reproducing results, the lack of modularity, and the accumulation of technical debt in the form of uncleaned pipelines and unstructured data [17]. The authors [15] recommend the introduction of MLOps practices: versioning of models and data, continuous monitoring of model productivity, automated retraining, etc. Preventive maintenance in this context involves not only code maintenance, but also the quality and relevance of models, which quickly lose their effectiveness without proper support.

The analysis showed that preventive maintenance is recognized as an important but still under-researched topic in the scientific community. Existing approaches demonstrate the practical value of preventive actions but require further development, standardization, and validation. Thus, further study, formalization, and implementation of preventive maintenance models is a relevant and strategically expedient direction for software engineering.

Research objectives and tasks

The aim of this study is to improve the effectiveness of software maintenance by developing a reference model for preventive maintenance that provides early detection of degradation trends, formalized identification of risk states, and timely generation of preventive actions within the software system lifecycle.

The study has formed a comprehensive structure of a reference model of preventive maintenance, which includes components of monitoring, preventive identification, task definition, result evaluation, and adaptation. Mathematical models of the interaction of these components and mechanisms for linking them into a single cycle of proactive maintenance have also been determined. The practical applicability of the proposed model is separately substantiated, which creates the prerequisites for improving the reliability, security, and maintainability of software systems in dynamic operating conditions.

Materials and methods

The methodological basis for preventive software maintenance requires a clear understanding of the indicators that best reflect the potential risks of system degradation, as well as the mechanisms for interpreting them in order to make timely engineering decisions. This task is also complicated by the high dynamics of environmental changes (OS updates, device diversity, use of third-party service APIs), as well as pressure from user expectations for stability, performance, and data security. Preventive maintenance in such an environment should be based not only on fixing current defects, but also on analyzing changes in the behavioral, technical, and qualitative characteristics of the software system over time, as well as pressure from user expectations for stability, performance, and data security [4].

Preventive maintenance covers key groups of metrics that signal the need for intervention even before noticeable failures in SW operation occur. This study proposes the use of the

comparator identification method [18–20], which allows early signs of degradation to be detected through comparative analysis of metrics over controlled periods. This method is at the core of the analytical mechanism of the preventive maintenance framework, ensuring the systematization of risk signals, the formulation of decisions for the development team, and their integration into the existing SW life cycle.

Within the scope of this work, metrics are grouped into four complementary classes: 1) code and architectural metrics; 2) maintainability and technical debt metrics; 3) process evolution metrics (changes and defects); 4) field (operational) and security metrics. This classification is consistent with quality standards approaches [7], where maintainability, reliability, and security are considered key characteristics that must be maintained throughout the life cycle.

Code and architectural metrics assess structural complexity and design features that directly affect the risk of defects and the cost of changes. A classic measure is McCabe's structural complexity, which reflects the number of linearly independent paths in a flow control graph [10]:

$$CC = E - N + 2P,$$

where E is the number of edges, N is the number of nodes, P is the number of connectivity components (mostly $P = 1$) [16]. High values CC correlate with complicated verification and increased defectiveness. Typical thresholds that are appropriate to use can be defined as follows: 1–10 (low risk), 11–20 (medium), 21–50 (high), >50 (very high).

Additional indicators, such as Coupling Between Objects (CBO), Response for a Class (RFC), and Lack of Cohesion of Methods (LCOM), provide insight into the modularity, encapsulation, and "fragility" of the design [10, 11]. Low cohesion and excessive coupling are early signals for preventive refactoring. Practical collection of such metrics can be performed by static analyzers (e.g., SonarQube), which also identify "code smells", i.e., informal design symptoms (long methods, "divine classes", duplication) that are empirically associated with defectiveness and reduced maintainability. Eliminating such defects is a typical preventive measure [11].

Maintainability and technical debt metrics aggregate assessments from different complexity measurements into a single indicator that is convenient to observe over time [6, 11, 12]. A common example of this type of metric is the Maintainability Index (MI), a convenient integral indicator that combines Halstead Volume (information complexity), Cyclomatic Complexity (structural complexity), and LOC (code size) and provides a quick assessment of maintainability [11]. The following formula for determining MI is considered the most well-known:

$$MI = max\left(0{,}171 - 5{,}2\ln HV - 0{,}23 \times CC - 16{,}2\ln LOC\right) \times \frac{100}{171},$$

where MI is normalized in the range [0,100]. Gradations may differ between instruments, but often $MI < 49$ indicates the presence of problems and the need for code refactoring. The metric works well as a trend indicator: if it falls from release to release, this is a clear signal for preventive action (refactoring, simplification, decomposition).

Process evolution metrics reflect how the system changes and is released (activity in the repository, commit patterns, defect and support history), which is a source of early signals.

Such indicators include Code Churn, as well as the frequency of changes to individual modules, the density of "hot" commits, the ratio of defect and feature changes, etc. Churn is a key process metric that shows how often and how much a module changes over time. It naturally fits into the preventive loop as an indicator of fragility: increased churn correlates with a higher risk of defects, regressions, and subsequent maintenance costs, so it serves as a signal for proactive interventions (refactoring, additional tests, stabilization sprints) [9].

The relative intensity of code changes is calculated as follows:

$$ChurnRate = \frac{LOC_{added} + LOC_{modified} + LOC_{deleted}}{LOC_{total}}.$$

At the operational level, classic reliability metrics such as MTTF/MTBF (mean time between failures) and MTTR (mean time to repair) record the consequences of identified problems, and trends in these metrics show that preventive practices are working [19].

Field (or operational) and security metrics measure the "health" of the system in real-world use and its resistance to vulnerabilities. The former include the proportion of error-free sessions relative to the number of users, latency and throughput of critical transactions, stability of resource usage (CPU/memory), productivity degradation indicators under load, etc. They are collected by APM (Application Performance Monitoring) and observability tools (Sentry, Grafana) and form the basis for early detection of deterioration trends [3, 19].

The security dimension includes the number and severity of known vulnerabilities in dependencies on the CVSS (Common Vulnerability Scoring System) scale, the "age" of unpatched CVEs (Common Vulnerabilities and Exposures), and the proportion of components outside the support window, which makes it possible to prevent degradation and exploitation [15].

From a preventive perspective, it is important not only to track individual indicators, but also to interpret their dynamics and combinations. An increase in cyclomatic complexity, along with an increase in churn and a decrease in MI, supported by "code smells", makes a compelling case for targeted refactoring before field incidents occur.

Similarly, spikes in latency and deterioration in reliability metrics, combined with an increase in defective commits, are grounds for preventive fixes and expanded test coverage. In practice, metric collection is integrated into CI/CD as "quality gates", and their trend analysis directly feeds into the backlog of preventive tasks.

It is assumed that the results of observations can be characterized by an assessment of the system's state. Based on the software system's operating data, a system of features for each scenario and a set of output characteristics are formed, which can be used to compare different behavioral scenarios. In general, the input data can be of any type, and the model output is accepted as binary (0 or 1) – in fact, the model implements a certain predicate that checks the condition for the fulfillment of relations between input signals [19].

Such formalization through predicate logic ensures strict fulfillment of the specified conditions for all cases, making the method objective and rigorous. If necessary, the method is

extended to fuzzy data: instead of a rigid true/false, degrees of membership are allowed, which makes it possible to take into account the uncertainty of the output information [20].

Thanks to this, comparator identification has been applied not only to technical systems, but also to complex, weakly formalized objects – from the assessment of professional skills to the modeling of cognitive processes [18–20].

From a mathematical point of view, the comparator identification method has a rigorous theoretical basis. It provides an objective, highly accurate, and reliable description of the object model, which is not inferior in completeness to classical direct identification [19]. If the input and output relations are specified correctly, the solution to the comparator identification problem provides a model that is fully consistent with all observed facts (up to isomorphism in the model state space).

This means that within the limits of its formulation, the method is correct and capable of providing a complete mathematical description of the desired dependence. Moreover, its use formalizes subjective expert data: instead of intuitive "vague" features, a clearly defined function or rule is used for decision making.

Thus, based on the collection and analysis of features of software system operation scenarios, a system of input signals for a preventive maintenance model is formed. The use of the comparator identification method allows us to build strict and reasonable interdependencies between the received signals and the risks of system degradation through indirect identification. Based on the conclusions, SW maintenance tasks are formed.

Therefore, to implement continuous preventive SW maintenance, it is necessary to develop a framework (reference model) that will formalize the main components of the maintenance model and serve as a foundation for developing models for monitoring, identifying degradation risks, and defining maintenance tasks.

Results

The basis of the proposed preventive support model is the basic structure of software (SW) support, which includes four basic elements that correspond to the generally accepted cycle of proactive reliability assurance: monitoring, status assessment, action determination, and evaluation of the results of the actions taken.

This study proposes to extend the classical approach by adding a model tuning component that will ensure the adaptation of rules in a continuous cycle of observation and improvement of the software system for the purpose of preventive maintenance. Together, these components form a cyclical process of preventive support, similar to a monitoring and adaptation loop. It includes observation, analysis, action plan, and verification – all the elements necessary for continuous improvement of SW quality and reliability.

This approach fits in with the current trend of transition from passive support to active, intelligent support of software systems.

Components of the reference model of preventive support

The proposed reference model of preventive support thus consists of five main components: monitoring of status and usage scenarios, identification of degradation risks, determination of actions and formation of support tasks, evaluation of the results of actions, and adaptation (Fig. 1).

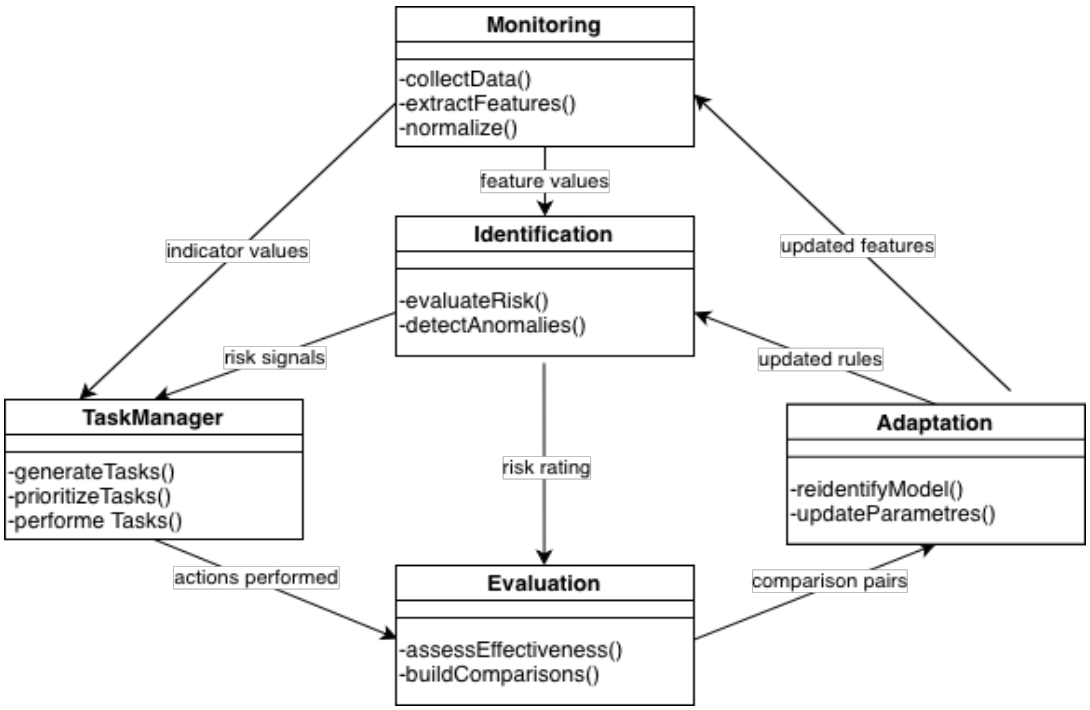


Fig. 1. Structure of the reference model for preventive SW maintenance

The monitoring process model is designed for continuous collection and tracking of metrics on the status of the application itself and the external conditions of its operation. For a mobile application with external sensors, such metrics may include, for example, application response delays, error or failure rates, abnormal sensor readings, the degree of deviation of input data from the training sample, device load indicators (CPU, memory, battery), etc. Regular monitoring allows you to accumulate historical data and record deviations in system behavior in real time.

The problem condition identification process model (preventive assessment) analyzes the collected data (using rules, statistics, or machine learning algorithms) to identify patterns and conditions that precede the occurrence of problem situations. In other words, the system attempts to predict potential failures or degradation based on detected anomalies, such as increased response time, memory leaks, sensor data distribution shifts, or decreased ML model prediction reliability. Such predictive analytics in SW is analogous to predictive maintenance in industry, where equipment failures are predicted based on data. Recent studies confirm the effectiveness of moving from purely reactive error correction to proactive prediction and prevention of failures using data analysis and AI methods.

The task identification process model is responsible for determining the actions that the development team needs to take. After identifying risky conditions, the system must determine what preventive measures or tasks should be performed to reduce the risk of a problem. In the context of agile development, this means forming specific tasks for the team (for example, including a fix for a potential memory leak in the next sprint, updating a machine learning model, improving sensor data processing, or adding monitoring of a specific parameter). In essence, this component carries out impact planning: based on the identified problem, the optimal actions are selected. In modern DevOps/AIOps practice, such decisions can be made automatically in the form of alerts to developers or even automatic remediation (e.g., service restart, model rollback). However, the automated conversion of analytical conclusions into development tasks is still in its infancy and often requires human involvement. Nevertheless, a formalized task model is important so that preventive measures do not remain at the level of signals but are implemented.

The process model for evaluating the results of actions is a feedback mechanism that assesses the effectiveness of the preventive measures taken. After completing the tasks (for example, releasing an update or reconfiguring the model), it is necessary to monitor the system again and compare the metrics before and after the intervention. This allows you to understand whether the probability of the predicted problem has decreased or whether the bottleneck has been corrected. In addition, such an analysis of results replenishes the knowledge base: it confirms or refutes the correctness of the identified conditions and the effectiveness of the measures. As a result, the results assessment model closes the classic cycle: successful cases strengthen confidence in the support system, and in case of failure, the monitoring metrics or identification rules are adjusted.

The adaptation process model is the final element of the preventive software maintenance cycle, which ensures its ability to self-improve based on feedback. Its purpose is to periodically review, refine, and update the rules, thresholds, machine learning models, and procedures used by the monitoring, preventive assessment, and task formation blocks. In particular, if a discrepancy is found between the predicted risks and the actual results of interventions, the adaptation module compares the expected and actual dynamics of metrics, assesses the degree of deviation, and determines which model parameters need to be updated. This allows for automatic adjustment of both the sensitivity of anomaly detection algorithms and the relevance of rules that initiate preventive actions.

The presence of an adaptive component is critical because the operating conditions of the mobile application, the nature of data from external sensors, and user behavior change over time, leading to the gradual degradation of static models. Thus, the adaptation block is a mechanism for "self-calibration" of the system: it closes the cycle, ensuring continuous improvement of preventive procedures and increasing the stability and reliability of the software in dynamic operating conditions.

Thus, the proposed five-component model forms a complete closed cycle of preventive support from monitoring and risk assessment to the formation of actions, evaluation of their effectiveness, and adaptive refinement of rules. This architecture ensures the software's ability to proactively self-improve, increasing its reliability in dynamic operating conditions.

Formalization of the set of software system states

Let $x \in \mathsf{R}^n$ is a is a vector of state characteristics of a software system running on a mobile device. Each component of the vector x describes a specific aspect of the system's operation, in particular, related to sensor data processing, machine learning model operation, user activity, network connection quality, device parameters, and inter-user interaction, etc. Let us denote the state of the software system at a given moment in time t as x_t. Thus, each vector $x_t \in \mathsf{R}^n$ fully characterizes the functional state of the software system over time. The vectors collected at different points in time form a set of observable states.

Let $X \subseteq \mathsf{R}^n$ – the space of all possible vectors x_t describing the software system over time. The set X is the basis for further analysis, including clustering of states, anomaly detection, formation of classes of equivalent operating modes, and construction of a system of preventive signals. Let us define the set of possible (attainable) functional states of the system as:

$$X^{\Omega} := \left\{ x_t \in X \middle| x_t \in \Omega \right\},$$

where $\Omega \subseteq \mathsf{R}$ – permissible interval or scale of values of the corresponding feature.

Therefore, vectors x_t are elements of observation streams:

$$D = \left\{ x_t \in X^{\Omega} \middle| t \in T \right\},$$

where T is a set of discrete observation points in time.

However, vectors x_t can be unstable, noisy, or interdependent. Therefore, for effective analysis of the SW state, we introduce a reflection:

$$\phi : X^{\Omega} \to Z,$$

where $Z \subseteq \mathsf{R}^m$ is a set of established (aggregated, normalized, discretized) features suitable for comparison, class construction, and use in the comparator identification method.

Let W be the size of the observation window (the number of steps back in time). Then, from the set of observations $X_t^W = \left\{ x_\tau \in X^{\Omega}, \tau \in \left[t - W + 1, t \right] \right\}$, i.e., from the time windows of observations, the aggregated features z_t are formed as:

$$z_t = \varphi\left(\{ x_\tau \}_{\tau=1}^{W} \right),$$

where φ – aggregation or transformation function: average, maximum, trend, autocorrelation, derivative, etc.

In general:

$$z_t = \Phi\left(X_t^W \right), z_t \in \mathsf{R}^m,$$

where $\Phi : \mathsf{R}^{W \times n} \to \mathsf{R}^m$ – feature construction function.

These aggregated features z_t, $t \in T$ are the basis for normalizing and comparing SW operating modes, constructing equivalence relations (in the comparator identification method), and determining functional classes. The set of all such vectors forms a space:

$$Z = \left\{ z_t \middle| t \in T \right\}.$$

Let us call those feature vectors $z^* \in Z$ that correspond to the normal (stable or functionally acceptable) mode of operation of the software system reference vectors. Let us denote the set of reference states as:

$$Z_{ref} = \left\{ z_i^* \in Z \middle| i = 1, \ldots, r \right\},$$

where r – number of known or recorded reference implementations.

Let us define the functional equivalence relation $\sim$ on the set Z if for two states $z_1, z_2 \in Z$:

$$z_1 \sim z_2 \Leftrightarrow \rho(z_1, z_2) \le \delta,$$

where $\rho(z_1, z_2)$ – selected similarity metric (e.g., Euclidean, Mahalanobis), $\delta > 0$ – acceptable deviation threshold.

Thus, each element $z \in Z$ either belongs to the class of one of the standards z_i^*, or initiates a new class.

The key idea of the proposed approach to preventive SW maintenance is not to attempt to identify numerical parameter values, but to determine classes of equivalent functional states based on comparisons with benchmarks. State identification is then carried out according to the following scheme:

Step 1. Calculate the distances $d_i = \rho(z_t, z_i^*), i = 1, \ldots, r$.

Step 2. If there is such i that $\mathrm{d}_i \le \delta$, then $z_t \sim z_i^*$ and z_t belongs to class i. Otherwise, the state of the software system at time t must be marked as abnormal or unknown.

This approach has certain limitations, i.e., the method is applicable when the system allows for the selection of representative reference implementations, features are aggregated, filtered from noise, and normalized, and a metric is available that preserves functional similarity rather than just geometric proximity.

The comparator identification method [18, 20] allows not only to record "failures", but also to classify states according to their functional proximity to known normal modes based on known feature values or to identify states with potential functional shifts (when the definedfeature vector does not match any reference).

Thus, this allows for preventive diagnostics without the need for explicit failures or "normal/abnormal" labels.

The complexity of preventive diagnostics lies in the fact that the state of the software system can be classified as normal based on formal characteristics, but the risk of degradation may increase. To eliminate uncertainty, it is proposed to extend the comparator identification method by identifying trends of deviation from reference states.

To do this, at each moment in time t, the SW state z_t is not only compared with the reference points, but also analyzed in terms of the dynamics of its approach or departure from the reference space Z_{ref} during the time interval $[t - W + 1, t]$.

For each moment in time t, the distance to the nearest reference point:

$$\mathrm{d}_t = \min_i \rho(z_t, z_i^*), \; z_i^* \in Z_{ref}.$$

Then the trajectory of distances over the time window W:

$$D_t^W = \{d_{t-W+1}, d_{t-W+2}, \ldots, \mathrm{d}_t\}.$$

This sequence reflects whether the system is approaching or moving away from the reference value.

To assess the trend or deviation trend, you can use the gradient assessment.

$$\Delta_t = \frac{\mathrm{d}_t - \mathrm{d}_{t-W}}{W},$$

or sliding average change

$$\bar{\Delta}_t = \frac{1}{W-1}\sum_{i=1}^{W-1}\left(d_{t-i+1} - d_{t-i}\right).$$

Thus, the identification of the SW state occurs according to the specified values of the aggregated features z_t, the distance to the nearest reference state d_t, and the assessment of the deviation trend from the reference states Δ_t. If the permissible threshold for deviation growth is exceeded or a trend toward deviation growth is identified, the preventive monitoring system generates a signal indicating an increased risk of software system degradation.

Thus, comparative identification is used to determine the trajectory of deviation from a set of benchmarks. The key object is not the SW state z_t, but the sequence of deviations from a set of reference states D_t^z. This model allows predictive signals to be issued regarding the growth of degradation risks and forms the basis for developing an action plan.

Formalization of a reference model for preventive software maintenance

In accordance with the above structure of the reference model for preventive SW maintenance, let us consider the following basic models that form the basis of the framework.

1. SW state monitoring provides data collection on SW functioning, reflecting the state of the application and the external environment, forming a stream of observations $D_t = \{x_{t1}, x_{t2}, \ldots, x_{tn}\}$. To do this, you need to specify a set of data sources K that form the raw data stream $s_t = \left\{s_t^{(k)}, k \in K\right\}$, the interval (frequency) of data collection τ, the operator for extracting SW state characteristics $\Psi(s_t)$, and the function $\Phi\left(X_t^W\right)$ for constructing features $z_t \in Z$. That is,

$$\mathrm{M}_{mon} : K, \tau, \Psi, \Phi, Z.$$

The basic monitoring model reflects defined procedures for regular collection of data on SW functioning and determination of aggregate indicators for identifying the state of the software system and risks of its degradation.

2. Preventive identification determines deviations and potentially dangerous SW states based on an extended method of comparator identification. It is proposed to consider the following classes of SW degradation risks $Q = \{q_1, q_2, q_3, q_4\}$, where q_1 determines stable operating mode, q_2 is the SW functioning state with a possible future risk (if a trend of deviation from

reference states is detected), q_3 combines states with a tendency to return to the standard and q_4 is a class that defines a critical situation (the deviation from the standard exceeds the threshold and there is a tendency to move away). Therefore

$$\mathsf{M}_{id} : z_t, D_t^W \to Q.$$

Thus, the basic preventive identification model generates an alarm signal for the preventive monitoring system based on the analysis of the SW status and trends approaching the reference modes.

3. The task identification component generates structured tasks for the development team and is an integral part of the decision support system. Task definition is based on a defined risk class q_i, anomaly identification, i.e., the identification of features that influenced the results of preventive identification, and the existing knowledge base of relevant precedents. Let there be a set of task templates $A = \{a_1, a_2, \ldots, a_h\}$, a given function $\mathsf{G}(z_t)$ for defining essential features Z_t, and a defined class q_i. Then the task identification model is

$$\mathsf{M}_{task} : Z_t, q_i \to A.$$

So, based on the model M_{task} tasks and priorities for their implementation are defined.

4. The basic model for evaluating the results of actions is designed to measure the effect of preventive actions taken in terms of reducing the risks of software system degradation. Let $M = \{M^{(u)}, u = \overline{1,U}\}$ – a set of metrics (evaluation criteria) for SW quality. Then it is possible to determine the integral effect of the actions a performed, for example, as

$$\varepsilon(a) = \sum_{u=1}^{U} w_u \Delta M^{(u)}(a).$$

In other words, evaluating the actions taken helps determine how the quality of the SW has changed and provides data for forming pairwise comparisons of states in terms of the obtained quality metric values, which collectively replenishes the knowledge base of precedents. That is,

$$\mathsf{M}_{eval} : A \to M, \varepsilon(a).$$

Therefore, M_{eval} evaluation of actions allows to form feedback between the planning of the development team's work on SW maintenance and the actual impact of the changes made on the stability and state of the SW.

5. The basic configuration model performs the function of adaptation and re-identification, i.e., updating the rules and parameters of the comparison model. Let there be a set of paired comparisons formed at a given moment in time t

$$C_t = \{(Z_i, Z_j, y_{ij})\},$$

where y_{ij} determines how the state of the software system has changed (approached or deviated from the reference). Let Θ determines the parameters of the comparator model used for preventive identification, then let us denote $f_\theta(Z_t)$ as an identification model that implements transformation M_{id} with parameters Θ.

Then the adaptation model implements an iterative cycle

$$\mathsf{M}_{adapt}:\Theta_{t+1}=\arg\min_{\Theta}\mathsf{L}\left(C_t,\Theta\right)+\alpha\Theta-\Theta_t^{\,2},$$

where L is the functional of the disturbance of the system of comparator equations, α is the stability parameter.

Thus, at the configuration stage, the preventive identification model is updated over time by accumulating historical data on the functioning of the software system, which ensures the ability of the preventive maintenance system to self-adapt.

Example of using a reference model for preventive maintenance

Let's consider a mobile application for patients with diabetes mellitus, which receives glucose readings from a Bluetooth glucometer, stores measurement history, builds short-term glucose level forecasts, and issues warnings. Preventive support is critically important for such an application: degradation of forecast accuracy or failure of sensor data synchronization without obvious application crashes directly affects medical risks.

During the operation of a software system that interacts with an external sensor device (glucometer), a stream of raw telemetry data is generated. Raw data is considered to be the direct results of the application's interaction with the hardware device and internal software components, which are recorded during the execution of individual read operations.

Each attempt to obtain or process sensor data is considered a discrete point in time, regardless of whether it was successful or ended in error. Thus, discrete observation moments correspond to events such as: initiating data reading from the glucometer; receiving a response from the device or recording a timeout; validating and parsing the received data; generating a glucose level forecast based on the input measurements. A set of raw data is recorded for each event:

$G(t)$ – actual glucose level;

$\hat{G}(t)$ – predicted value;

$s(t)$ – operation result (successful/error);

$\tau(t)$ – operation execution time.

Raw data is linked to events rather than to the state of the system, has different frequencies of occurrence, and may contain noise, so it cannot be used directly to assess the functional state of the system. To transition from raw data to formalized indicators, a set of current indicators is introduced, $r_i(t)$, which are calculated at the level of individual events and have a clear engineering meaning. Let's introduce three key operational metrics that the monitoring system collects daily:

$r_1(t)=G(t)-\hat{G}(t)$ – instantaneous glucose prediction error (absolute error in mmol/L between the actual glucose meter reading and the model prediction);

$r_2(t)=\begin{cases}0, if\ s(t)=fail\\ 1, if\ s(t)=ok\end{cases}$ – incorrect measurement indicator (0 or 1);

$r_3(t) = \tau(t)$ – Synchronization time with the glucometer (from screen opening to current value display, ms).

Monitoring operates in daily windows $W_k = \{t \mid t = 1, 2, \ldots\}$, where k is the day number and t is the event number in the monitoring window. That is, metric values are collected at discrete points in time throughout the day, forming observation streams. The number of successful reads is defined as $C_k = \sum_{t \in W_k} r_2(t)$, and the total number of attempts is $N_k = |W_k|$.

For each day, we determine the aggregated features:

$x_1(k) = \frac{1}{C_k} \sum_{t \in W_k} |r_1(t)|$ – mean error per day;

$x_2(k) = \frac{N_k - C_k}{N_k}$ – percentage of incorrect read attempts per day;

$x_3(k) = \frac{1}{N_k} \sum_{t \in W_k} r_3(t)$ – mean synchronization time.

Then the SW state vector can be denoted as $x(k) = (x_1(k), x_2(k), x_3(k))$. Thus, raw telemetry data is converted into current indicators $r_i(t)$ at the level of individual events. Next, at fixed time intervals W_k, these indicators are aggregated into a state feature vector $x(k)$ that describes the functional state of the software system per day (W_k interval).

Normalization is introduced to ensure correct combination of features. Normalization allows: bringing heterogeneous features to a dimensionless scale; assessing the degree of deviation of the current state from the normal mode; forming integral state indicators and applying formal identification rules. For a reference stable period (for example, the first 21 days after release), we form a set of reference states:

$$K_{ref} = \{1, 2, \ldots, 21\},\ Z_{ref} = \{x(k) \mid k \in K_{ref}\}.$$

For the reference period, we calculate the mean values and standard deviations:

$$\mu_i = \frac{1}{|K_{ref}|} \sum_{k \in K_{ref}} x_i(k),\ \sigma_i^2 = \frac{1}{|K_{ref}|} \sum_{k \in K_{ref}} (x_i(k) - \mu_i)^2,\ i = 1, 2, 3.$$

Then the normalized features are defined as:

$$z_i(k) = \frac{x_i(k) - \mu_i}{\sigma_i^2},\ i = 1, 2, 3.$$

Normalized state vector:

$$z(k) = (z_1(k), z_2(k), z_3(k)).$$

The obtained normalized vector $z(k)$ is used as input for further comparator identification and classification of the functional states of the software system. Next, the reference (baseline) mode of operation of the software system is determined. The set of indices K_{ref} can be expanded using all k that correspond to the period of stable operation. $Z_{ref} = \{x(k) \mid k \in K_{ref}\}$ is a reference set of normalized states. In the simplest case, the reference is represented by a single representative vector (the center of the reference mode).

$$z_{ref} = \frac{1}{\left|K_{ref}\right|} \sum_{k \in K_{ref}} z(k),$$

where $\sum_{k \in K_{ref}} z(k)$ means the component-wise sum of vectors, and z_{ref} is the centroid of the set of normalized states of the reference mode.

We define the deviation from the reference as the weighted Euclidean distance:

$$d(k) = \sqrt{w_1 \left(z_1(k) - z_{1,ref}\right)^2 + w_2 \left(z_2(k) - z_{2,ref}\right)^2 + w_3 \left(z_3(k) - z_{3,ref}\right)^2}, \; w_1 + w_2 + w_3 = 1,$$

where, for example, we give greater weight to medically critical errors in prognosis: $w_1 = 0,4$; $w_2 = 0,3$; $w_3 = 0,3$.

The local (component) deviation is determined separately for each feature:

$$\delta_i(k) = \left|z_i(k) - z_{i,ref}\right|, i = 1,2,3.$$

The vector $\delta(k)$ is used to explain the results (which metrics caused the growth $d(k)$ and to further form preventive tasks.

For preventive support, it is important not only to know the current value of the deviation, but also how it changes over time. To assess the deviation trend, we use a sliding window with a length of, for example, $L = 7$ days:

$$g(k) = \frac{d(k) - d(k - L + 1)}{L - 1}, k \geq L.$$

A pair of indicators $\left(d(k), g(k)\right)$ is sufficient statistics for further classification of states. To interpret the monitoring results and support decision-making, a set of discrete classes of the functional state of the software system is introduced:

R_0 – normal mode;

R_1 – deviation from the standard, but there is a tendency for the deviation to decrease;

R_2 – high-risk mode (preventive intervention is necessary);

R_3 – critical condition.

Let us introduce predicates for recognizing conditions related to degradation and risks of the software system:

$P_1(k) \Leftrightarrow d(k) \leq d_0$ determines proximity to the reference mode, d_0 – the threshold of proximity to the reference;

$P_2(k) \Leftrightarrow \left|g(k)\right| \leq g_0$ determines proximity to the reference during a sliding window, g_0 – the threshold of acceptable trend deviation;

$P_3(k) \Leftrightarrow g(k) < 0$ corresponds to a situation of decreasing deviation from the reference, i.e., there is a tendency to return to the reference;

$P_4(k) \Leftrightarrow g(k) > 0$ determines the tendency of deviation from the reference mode.

Then, the correspondence of the state to one of the classes is determined by the following conditions:

$$R_0(k) \Leftrightarrow \left(P_1(k) \wedge P_2(k)\right) = 1;$$

$$R_1(k) \Leftrightarrow \left(\neg P_1(k) \wedge P_3(k)\right) = 1;$$

$$R_2(k) \Leftrightarrow \left(P_1(k) \wedge P_4(k) \wedge \neg P_2(k)\right) = 1;$$

$$R_3(k) \Leftrightarrow \left(\neg P_1(k) \wedge P_4(k) \wedge \neg P_2(k)\right) = 1.$$

Analysis of experimental results for a 60-day period of operation allows us to clearly illustrate the operation of the proposed classification system. Figure 2 shows a graph of changes in forecast error values during the experiment.

At the initial interval of the experiment, the values $d(k)$ (see Fig. 3) do not exceed the threshold $d_0 = 1$, and the trend $g(k)$ fluctuates near zero, which corresponds to the fulfillment of conditions $\left(P_1(k) \wedge P_2(k)\right)$. This is consistent with the classification of the system into class R_0 and confirms a stable mode of operation, which is clearly visible in the graph.

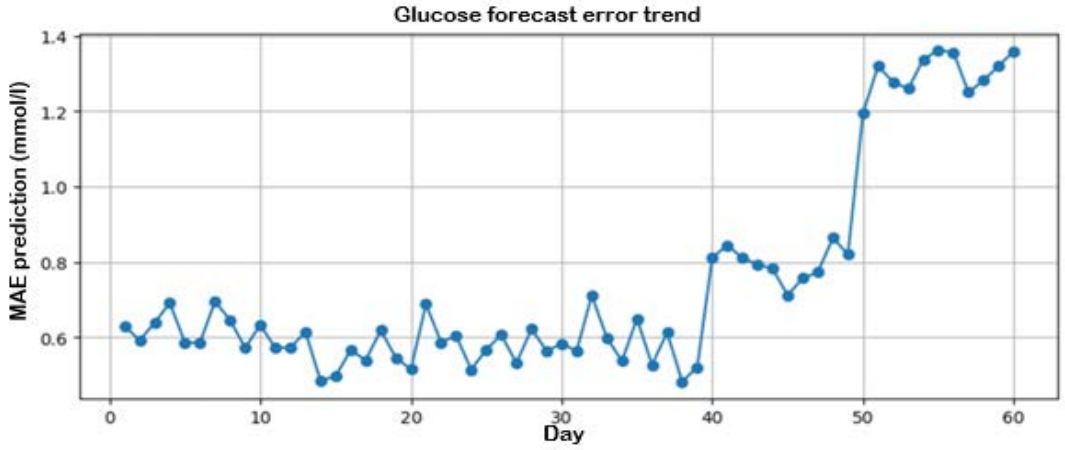


Fig. 2. $x_1(k)$ values during 60 days of observation

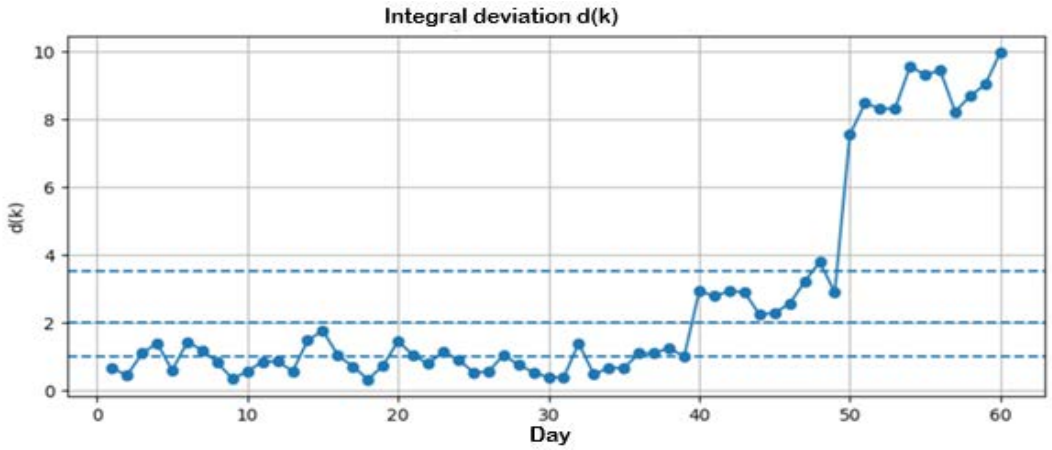


Fig. 3. Trend in deviation from the reference mode

With the onset of monotonous growth of the integral deviation (Fig. 3), it is possible to signal the formation of a pre-crisis state and the need for preventive action, which is determined by the transition to class R_2. Further simultaneous violation of the conditions of proximity to the standard and the permissible trend leads to the state being classified as class R_3, which corresponds to the critical state of the software system.

The proposed system of predicates and classes provides a formal link between numerical monitoring characteristics and engineering decisions. The transition to classes R_2 and R_3 is used as a trigger for the automatic generation of support tickets, in which the values $d(k)$, $g(k)$ and component deviations serve as justification for the necessary preventive or urgent actions.

The results of the classification of the functional state of the software system are not communicated to the development team in the form of formal classes or threshold values. Instead, they are transformed into support tickets (tasks for the development team) containing a clear description of the problem, quantitative observations, and recommended technical actions. An example of a ticket in the event of a transition to a class R_2 (high-risk mode) is shown in Figure 4.

Title:

Gradual deterioration in forecast quality and delays in synchronization with the glucometer

Description:

Over the past few days, there has been a steady increase in the average error of glucose level predictions. At the same time, the time it takes to receive data from the glucometer is increasing. Although the system continues to operate without any obvious malfunctions, there is a negative trend in key performance indicators.

Observations (aggregated per day):

- the average prediction error has increased by approximately 25-30% compared to the stable period;
- the average synchronization time with the glucometer has increased by 40-60 ms;
- there are isolated unsuccessful attempts to read data.

Recommended actions:

- check the logic of repeated reading attempts and timeouts;
- analyze whether the input data characteristics for the prediction module have changed;
- prepare a test update of the prediction model or its parameters if the trend continues.

Priority: medium

Deadline: planned, by the next release

Fig. 4. Example of a preventive ticket (early degradation)

The example considered demonstrates that the proposed reference model provides a coherent internal logic for monitoring the state of the software system based on aggregated operational metrics and their dynamics. The model formalizes the transition from raw data to integrated indicators of deviation and trends, allowing for the correct identification of both stable

operating modes and early signs of degradation. The use of formalized classification conditions ensures the reasonable formation of alerts without the need for rigidly fixed scenarios, and the analysis of component deviations provides structured information about the possible causes of deterioration in the quality of the system's operation and directions for its elimination. This allows the results of identification to be translated into engineering tasks that are understandable to the development team in the form of standard support tickets.

Thus, the model supports the automated formation of tasks for the development team based on quantitative observations and their trends, which significantly reduces the time between the first signs of degradation and the start of engineering intervention. This ensures the transition from reactive incident resolution to proactive software maintenance in real operating conditions.

Discussion

The comparator identification method [18, 20] is a special variant of solving identification problems when direct measurement of the parameter of interest is impossible. In classical identification, an attempt is made to determine the law of their connection based on the input and output signals of the object. However, in many cases, the output of the system (for example, the "failure susceptibility level") cannot be measured directly. For such situations, indirect identification methods are used, and the most convenient among them is the declared method of comparator identification.

The idea of using the comparator identification method in the task of preventive SW maintenance looks promising, since the very nature of the problem corresponds to the setting of indirect identification. It is not possible to directly measure the "probability of future failure" or the "level of problematicity" of the system state, but we can compare different states using indirect reliability indicators. Despite its scientific validity, the practical feasibility of this method in complex software systems should be critically evaluated.

First, the comparator identification method essentially provides a binary conclusion. Additional mechanisms are needed for a fine gradation of risks.

Second, the key step – forming a system of equations – requires the availability or accumulation of representative data on various SW operating situations.

Third, software and its operating environment change over time (updates, new features, changing loads), so the model will have to be regularly updated as new data becomes available. This leads to the need to modify the classical method.

In engineering practice, statistical methods and machine learning are more often used for SW support [9, 15], which usually requires a well-established infrastructure for collecting and processing large amounts of data. The comparator approach, on the contrary, can introduce an element of strict justification and interpretability. Its mathematical correctness and ability to work with fuzzy information provide significant advantages for solving the complex task of preventive maintenance, where uncertainty and multi-criteria are the main characteristics. If the method is adapted, it can become part of intelligent maintenance systems, increasing the objectivity of decisions on preventive measures.

In general, the idea of applying the comparator identification method is in line with the global trend towards proactive, data-driven software maintenance and appears promising, although the effective application of predictive analytics also requires a culture that is ready to accept the model's recommendations.

The results show that the proposed reference model of preventive support can become a fundamental basis for the transition from reactive software support to systematic proactive quality management. Unlike traditional approaches, which focus primarily on fixing already identified defects, the developed model uses a formalized system of aggregated features, deviation trends, and comparator assessment, which allows identifying degradation risks without the presence of obvious failures.

This approach provides early warning of potential problems and creates a basis for automated task generation, which is especially important in highly dynamic environments such as mobile applications or systems with machine learning components and smart manufacturing complexes [21].

It is important to emphasize that the proposed model forms not only a mechanism for analyzing the state of the software system, but also a closed cycle of its evolution through adaptive updating of parameters and identification rules. This allows the maintenance system to respond to changes in the external environment, SW updates, the emergence of new usage modes, and the dynamics of machine learning models. The adaptive component ensures the stability of the approach in the long term, preventing accuracy degradation in conditions of data drift, seasonal load, or changes in user activity profiles.

At the same time, the results highlight a number of important challenges. The effectiveness of the method largely depends on the quality of the set of reference states and the adequacy of the selected distance metrics. The presence of insufficiently representative benchmarks or noisy features can lead to incorrect signals, which is critical in systems where the frequency of false alarms must be minimized.

Another aspect that requires further research is the integration of the model into real CI/CD and DevOps processes: determining the frequency of analysis, limitations on computing resources, communication with the task backlog, and the balance between automated and manual developer solutions.

Overall, the results obtained indicate the promise of combining comparator identification with the principles of preventive maintenance. This approach provides a formally justified, interpreted, and at the same time practically applicable mechanism for detecting hidden degradation trends, which enhances the overall ability of software systems to self-identify and proactively improve.

Further research directions may be related to the automatic formation of reference classes, the expansion of the model by means of machine learning, the verification of effectiveness on different classes of systems, and the development of prototypes for integration into the support infrastructure.

Conclusions

The paper proposes a reference model for preventive software maintenance that covers the full cycle of proactive software system management, from monitoring and detecting degradation trends to forming maintenance tasks, evaluating results, and adaptively updating identification rules. Unlike traditional reactive approaches, the model provides early detection of potentially dangerous changes in software behavior and allows intervention before critical failures occur.

The analytical mechanism of the model is based on an extended comparator identification method, which provides a formalized comparison of current states with a set of reference modes and the identification of deviation trends. This approach makes it possible to detect hidden degradation processes even in the absence of obvious incidents, which significantly increases the effectiveness of preventive support.

The adaptive component of the model ensures that identification parameters and rules remain relevant in dynamic operating conditions, particularly when there are changes in load, user behavior, or data drift in systems with machine learning components.

The formal models formulated for each element of the reference structure (monitoring, preventive identification, task definition, result evaluation, and adaptation) create a comprehensive framework for preventive support. It can be integrated into modern CI/CD, DevOps, and MLOps processes, serving as a mechanism for improving the reliability, security, and maintainability of software systems.

The results obtained open up prospects for further research in the areas of automatic reference state formation, the use of machine learning methods to refine risk class boundaries, the development of tools for integration with the support infrastructure, and experimental validation of the model on different types of software products.

The proposed model can become the basis for the creation of intelligent preventive maintenance systems that ensure the stable and predictable functioning of software complexes in the long term.

References

1. Masrat, A., Makki, M. A., Gawde, H. (2021), "Software maintenance models and processes: An overview", *SSRN Electronic Journal*. DOI: https://doi.org/10.2139/ssrn.3838444
2. ISO/IEC/IEEE 14764:2022. Software engineering – Software life cycle processes – Maintenance. – Geneva: International Organization for Standardization.
   URL: https://www.iso.org/standard/80710.html (дата звернення: 10.10.2025).
3. Young, Z., Steele, R. (2022), "Empirical evaluation of performance degradation of machine learning-based predictive models – A case study in healthcare information systems", *International Journal of Information Management Data Insights,* Vol. 2, Iss. 1., Art. 100070.
   DOI: https://doi.org/10.1016/j.jjimei.2022.100070

4. Ibrahim, K. S. K., Mansor, Z., Yahaya, J., Deraman, A. (2025), "Software maintenance assessment: An analysis of determination factors", *Journal of Mathematical Sciences and Informatics*, Vol. 5, No. 1. DOI: https://doi.org/10.46754/jmsi.2025.06.006
5. Kleinwaks, H., Gärtner, M., Reich, J., Woehrle, G. (2023), "Technical debt in systems engineering - A systematic literature review", *Systems Engineering*, Vol. 26, Iss. 6, P. 741–760. https://doi.org/DOI: 10.1002/sys.21681
6. Lenarduzzi, V., Besker, T., Taibi, D., Martini, A., Arcelli Fontana, F. (2021), "A systematic literature review on technical debt prioritization: Strategies, processes, factors, and tools", *Journal of Systems and Software*, Vol. 171, Art. 110827. DOI: https://doi.org/10.1016/j.jss.2020.110827
7. ISO/IEC 25010:2023. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Product quality model. – Geneva: International Organization for Standardization. URL: https://www.iso.org/standard/78176.html (дата звернення: 12.10.2025).
8. Abdu, A., Abdo, H. A., Ullah, I., Khan, J., Gu, Y. H., Algabri, R. (2025), "Deep multi-metrics learning for mobile app defect prediction using code and process metrics", *Scientific Reports*, Vol. 15, Iss. 1, Art. 38620. DOI: https://doi.org/10.1038/s41598-025-22566-2
9. Dai, H., Xi, J., Dai, H.-L. (2024), "Improving effort-aware just-in-time defect prediction with weighted code churn and multi-objective slime mold algorithm", *Heliyon,* Vol. 10, Iss. 18, e37360. DOI: https://doi.org/10.1016/j.heliyon.2024.e37360
10. Rebro, D. A., Rossi, B., Chren, S. (2023), "Source code metrics for software defects prediction", *Proc. 38th ACM/SIGAPP Symposium on Applied Computing*, P. 1668–1677. DOI: https://doi.org/10.48550/arXiv.2301.08022
11. Heričko, T., Šumak, B. (2023), "Exploring Maintainability Index variants for software maintainability measurement in object-oriented systems", *Applied Sciences*, Vol. 13, Iss. 5, Art. 2972. DOI: https://doi.org/10.3390/app13052972
12. Moulla, D. K., Mnkandla, E., Oumarou, H., Fehlmann, T. (2023), "Technical debt measurement: An exploratory literature review", *Proc. Int. Workshop on Software Measurement and Metrics (IWSM)*, P. 27–38.
13. Saraiva, J. D., Neto, J. G., Kulesza, U., Freitas, G., Rebouças, R., Coelho, R. (2021), "Technical debt tools: A systematic mapping study", *Proc. 23rd International Conference on Enterprise Information Systems (ICEIS)*, P. 280–303. DOI: https://doi.org/10.5220/0010459100880098
14. Dreyfus, P.-A., Pélissier, A., Psarommatis, F., Kiritsis, D. (2022), "Data-based model maintenance in the era of Industry 4.0: A methodology", *Journal of Manufacturing Systems*, Vol. 63, P. 304–316. https://doi.org/DOI: 10.1016/j.jmsy.2022.03.015.
15. Eken, B., Pallewatta, S., Tran, N. K., Tosun, A., Babar, M. A. (2025), "A multivocal review of MLOps practices, challenges and open issues", *ACM Computing Surveys*, Vol. 58, Iss. 2, P. 1–35. DOI: https://doi.org/10.1145/3747346.
16. Paleyes, A., Urma, R.-G., Lawrence, N. D. (2022), "Challenges in deploying machine learning: A survey of case studies", *ACM Computing Surveys*, Vol. 55, Iss. 6, Art. 114. DOI: https://doi.org/10.1145/3533378
17. Indykov, V., Strüber, D., Lwakatare, L. E., Felderer, M. (2025), "Architectural tactics to achieve quality attributes of machine-learning-enabled systems: A systematic literature review", *Journal of Systems and Software*. DOI: https://doi.org/10.1016/j.jss.2025.112373
18. Cherednichenko, O., Vovk, M., Sharonova, N., Vorzhevitina, A. (2025), "Comparator-based identification of food edibility from natural language description", *AICS-CoLInS 2025, CEUR Workshop Proceedings*, Vol. 4015, P. 185–199. DOI: https://doi.org/10.31110/COLINS/2025-3/014

19. Sharonova, N., Doroshenko, A., Cherednichenko, O. (2018), "Issues of fact-based information analysis", *Proc. 2nd International Conference on Computational Linguistics and Intelligent Systems (COLINS), CEUR Workshop Proceedings*, Vol. 2136, P. 11–19.
20. Sharonova, N., Doroshenko, A., Cherednichenko, O. (2018), "Towards the ontology-based approach for factual information matching", *Proc. 7th International Scientific and Technical Conference "Information Systems and Technologies" (IST-2018),* P. 230–233.
21. Xu, J., Kovatsch, M., Mattern, D., Mazza, F., Harasic, M., Paschke, A., Lucia, S. (2022), "A review on AI for smart manufacturing: Deep learning challenges and solutions", *Applied Sciences,* Vol. 12, Iss. 16, Art. 8239. DOI: https://doi.org/10.3390/app12168239

Received (Надійшла) 24.11.2025
Accepted for publication (Прийнята до друку) 07.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Chupryna Anastasiya – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Software Engineering, Kharkiv, Ukraine; e-mail: anastasiya.chupryna@nure.ua; ORCID ID: https://orcid.org/0000-0003-0394-9900

Repikhov Vadym – Kharkiv National University of Radio Electronics, PhD Student, Department of Software Engineering, Kharkiv, Ukraine; e-mail: vadym.repikhov@nure.ua; ORCID ID:https://orcid.org/0000-0002-1274-4205

Чуприна Анастасія Сергіївна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри програмної інженерії, Харків, Україна.

Репіхов Вадим Миколайович – Харківський національний університет радіоелектроніки, аспірант кафедри програмної інженерії, Харків, Україна.

ОПОРНА МОДЕЛЬ ПРЕВЕНТИВНОГО СУПРОВОДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Предметом роботи є процеси превентивного супроводження програмного забезпечення, зокрема формалізація механізмів моніторингу, ідентифікації ризиків деградації та побудови рішень щодо проактивного втручання в життєвий цикл програмних систем. **Мета дослідження** – розробити опорну модель превентивного супроводження, яка забезпечує систематичне виявлення відхилень від еталонних режимів роботи SW, оцінювання тенденцій їх зміни й генерацію обґрунтованих превентивних дій до появи критичних відмов. З огляду на окреслену мету необхідно було виконати такі **завдання:** запропонувати класифікацію метрик і ознак для опису функціонального стану SW; побудувати формальний апарат превентивної ідентифікації на основі розширеного методу компараторної ідентифікації; визначити структурні компоненти опорної моделі; розробити моделі моніторингу, аналізу ідентифікованих ризиків, керування задачами супроводження й

адаптивного оновлення параметрів моделі. **Методи дослідження.** Методологічну основу становить розширений метод компараторної ідентифікації, який дає змогу порівнювати поточні стани SW з еталонними режимами та кількісно оцінювати динаміку їх відхилення. Застосовано агрегування та нормалізацію ознак, формалізацію множини функціональних станів, аналіз часових вікон і трендових властивостей, а також механізми зворотного зв'язку й адаптивної корекції параметрів. **Досягнуті результати.** Сформовано п'ятикомпонентну опорну модель, що містить моделі: моніторингу стану SW; превентивної ідентифікації ризиків деградації; формування задач супроводження; оцінювання результативності превентивних дій; адаптації параметрів. Розроблена модель забезпечує виявлення латентних процесів деградації за допомогою аналізу траєкторій відхилень, інтегрується в сучасні процеси *CI/CD*, *DevOps* і *MLOps* та створює формалізований механізм підтримки прийняття рішень щодо вчасного супроводження. **Висновки.** Запропонована модель формує цілісний формальний фреймворк превентивного супроводження, спрямований на раннє виявлення потенційних відмов і підтримку довгострокової стабільності програмних систем. Адаптивний характер моделі забезпечує її актуальність у динамічних умовах експлуатації, а застосування компараторної ідентифікації підвищує інтерпретованість і обґрунтованість прийнятих рішень. Отримані результати створюють підґрунтя для розроблення інтелектуальних систем супроводження й подальших досліджень з автоматизації формування еталонних станів і розширення моделі засобами машинного навчання.

Ключові слова: превентивне супроводження; компараторна ідентифікація; моніторинг; технічний борг; прогнозування деградації; якість ПЗ; *DevOps*; *MLOps*.

Bibliographic descriptions / Бібліографічні описи

Chupryna, A., Repikhov, V. (2025), "Reference model for preventive software maintenance", *Management Information Systems and Devises*, No. 4 (187), P. 254–277.
DOI: https://doi.org/10.30837/0135-1710.2025.187.254

Чуприна А. С., Репіхов В. М. Опорна модель превентивного супроводження програмного забезпечення. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 254–277. DOI: https://doi.org/10.30837/0135-1710.2025.187.254

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004.056.53:004.032.26 : 004.65 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.278

S. Shtangey, L. Melnikova, A. Marchuk, O. Linnyk, YE. Hrebeniuk

ADAPTATION OF MULTICLASS CLASSIFICATION ALGORITHMS FOR IDENTIFYING NETWORK-ATTACK TYPES IN ROUTING PROTOCOLS USING STRUCTURED DATABASES

The subject of the article is the adaptation of multi-class classification algorithms for detecting types of network attacks in routing protocols. **The purpose of the study** is to develop and experimentally test the effectiveness of various deep learning architectures (*Dense*, CNN, LSTM) in the task of multi-class classification of network attacks, as well as to evaluate the feasibility of combining them into an ensemble to improve the accuracy and stability of classification. To achieve the stated goal**, the** following **tasks** must be performed**:** analyze the types of network attacks characteristic of the routing level and their features; to evaluate the advantages and disadvantages of traditional and modern attack detection methods, in particular signature, anomaly, and hybrid systems; to investigate deep learning architectures (*Dense*, CNN, LSTM) for their suitability for network traffic classification; to implement individual models and combine them into an ensemble using a voting mechanism. **The following methods were used:** deep neural networks of various types, ensemble learning (*bagging, stacking, voting*), and analysis of imbalanced data. To test the effectiveness of the models, the UNSW-NB15 dataset was used, which contains realistic examples of normal and abnormal traffic. The experiments were conducted using modern machine learning libraries, as well as regularization and normalization to prevent overfitting. **Research results.** Three neural network architectures were implemented and tested. The dense model confirmed stable results on aggregated features, CNN effectively identified local patterns even in the presence of noise, and LSTM ensured the detection of long-term dependencies in sequential data. The ensemble of models demonstrated higher classification accuracy compared to individual architectures, reduced the number of false positives, and increased generalizability. **Conclusions.** Adapting and combining different deep learning architectures can significantly improve the quality of multi-class classification of network attacks. The ensemble approach provides resistance to data imbalance and increases the accuracy of detecting complex attacks. The results confirm the feasibility of using ensembles in cybersecurity tasks and open up prospects for further research, in particular the integration of models into real-time systems and the extension of analysis to other types of network threats.

Keywords: network attacks; routing protocols; deep learning; dense neural networks; ensemble learning; intrusion detection; cybersecurity.

1. Introduction

In today's digital environment, information security is critically important for both organizations and individual users. With the growth of network traffic, the number of malicious actions is also increasing, in particular attacks at the routing level, which can lead to data integrity violations, loss of communication between legitimate nodes, or theft of confidential information [1]. Attacks that change the routing logic by redirecting packets through malicious nodes or blocking communication are particularly dangerous. The most common types of such attacks include DoS, Spoofing, Man-in-the-Middle, Routing Table Poisoning, as well as complex combined attacks such as Worms, Backdoors, and Shellcode [2].

Despite the availability of modern protection tools, detecting attacks at the routing level remains a difficult task. This is due to the short duration of attack signs, their similarity to

legitimate activity, and the high variability of behavior patterns. The lack of effective methods for detecting such threats can lead to serious theoretical and practical consequences, ranging from disruption of critical infrastructure to data loss and financial damage. Therefore, there is a need to create adaptive systems capable of self-learning and effective classification of new types of attacks.

One promising approach to solving this problem is the use of deep learning methods. This article examines the effectiveness of an ensemble of neural networks – dense, convolutional (CNN), and recurrent (LSTM) – for multi-class classification of network attacks based on the UNSW-NB15 dataset. Combining the advantages of different architectures allows for increased classification accuracy and robustness in the presence of imbalanced data, as well as the preservation of results in databases.

2. Analysis of current scientific publications and formulation of the research problem

Scientific publications present a wide range of approaches to the detection and classification of network attacks [3]. Signature-based IDS systems demonstrate high accuracy in detecting known threat patterns, but their significant drawback is their inability to respond to new, unknown types of attacks [4]. Anomaly-based IDS, on the other hand, are capable of detecting unusual behavior, but have a high number of false positives, which complicates their practical application [5]. These limitations stimulate the development of hybrid solutions, in particular with the use of machine learning and deep learning methods.

Deep learning methods, as noted by LeCun, Bengio, and Hinton, allow for the automatic extraction of features from high-dimensional data without manual intervention, which is particularly relevant for the classification of complex behavior patterns [6].

However, most studies focus on individual neural network architectures without considering their combined effectiveness. For example, feed-forward (dense) models work well with aggregated features but are unable to take time dependencies into account [7].

CNNs effectively extract local patterns even in noisy conditions, but do not always cope with sequential information [8]. LSTM models preserve temporal context, but have high computational complexity [9].

Special attention in the literature is paid to the problem of data imbalance, which negatively affects the quality of classification. Ensemble methods, as shown in Zhou's work, allow combining the strengths of different models, increasing generalizability and accuracy even on complex multi-class sets [10].

However, most existing solutions are not adapted to the specifics of the routing layer, where it is important to process both tabular and sequential data. To test the effectiveness of different architectures, the UNSW-NB15 dataset was selected, which covers a wide range of attacks, including Fuzzers, DoS, Shellcode, Reconnaissance, Worms, and others, and is relevant for multi-class classification tasks in the field of cybersecurity [11, 12].

In summary, recent scientific publications confirm the potential of deep learning in detecting network attacks, but at the same time reveal a number of limitations: narrow specialization of models, insufficient adaptation to the routing level, and ignoring data imbalance.

This creates a need to develop a comprehensive approach that combines the advantages of different architectures and takes into account the specifics of traffic.

The research problem lies in the need to create an adaptive architecture for classifying network attacks at the routing level that can effectively work with imbalanced multi-class data, combining the strengths of different types of neural networks within an ensemble approach.

3. Purpose and objectives of the study

The purpose of the study is to develop and experimentally investigate an ensemble deep learning architecture that combines dense, convolutional (CNN), and recurrent (LSTM) neural networks to improve the accuracy and robustness of network attack classification at the routing level.

The main objectives of the study are

– To analyze the types of network attacks characteristic of the routing level and their features.

– To evaluate the advantages and disadvantages of traditional and modern attack detection methods, including signature, anomaly, and hybrid systems.

– Analyze deep learning architectures (Dense, CNN, LSTM) in terms of their suitability for network traffic classification.

– Implement individual models and combine them into an ensemble using a voting mechanism.

4. Materials and methods of research

Methods for detecting attacks in computer networks have historically been divided into two main categories: signature-based systems and anomaly-based systems. Signature-based IDS work by comparing traffic with known attack patterns. Such systems are highly accurate in detecting known threats, but are unable to respond to new or modified attacks.

Anomaly-based IDS are based on studying normal system behavior and detecting deviations. These methods are capable of detecting unknown threats, but often have a high number of false positives [13].

Due to these limitations, hybrid methods that combine the advantages of both approaches, as well as machine learning-based methods, are becoming increasingly popular.

Among the classic ML algorithms used in attack detection, the following can be highlighted: Decision Trees – interpretable and fast, but prone to overfitting; Support Vector Machines (SVM) – effective for classification with a clear separation boundary, but difficult to scale; K-Nearest Neighbors (k-NN) – simple to implement, but slow with large amounts of data; Random Forest, XGBoost – ensemble models that demonstrate high accuracy in classification tasks [14].

Despite the effectiveness of these methods, they often require manual feature construction and do not always cope with high-dimensional or unstructured data. This has led to the need for deep learning, which can automatically extract relevant features without prior processing.

Deep learning is a subset of machine learning that allows you to build multi-level models for automatically extracting relevant features from raw data. Due to their ability to handle high-dimensional, complex, and fuzzy data, deep learning models are widely used in cybersecurity, particularly in detecting network attacks.

Below are three types of neural networks that are most commonly used in attack detection.

Dense or Feedforward Neural Networks. These are the simplest and most common architectures. The network consists of a sequence of dense layers, where each neuron is connected to all neurons in the next layer.

The structure of a dense neural network is shown in Fig. 1.

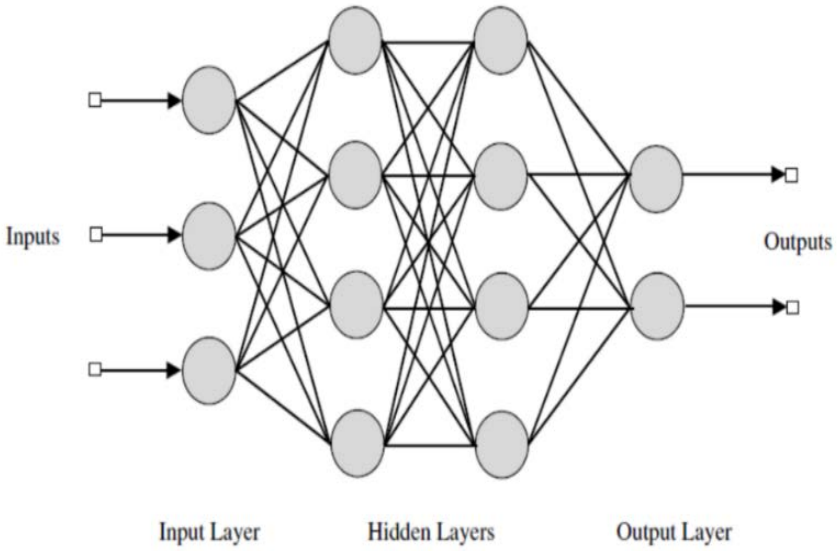


Fig. 1. Structure of a fully connected neural network

The main advantage of these models is their versatility. They can work with any tabular data set, including network traffic.

Dense models learn quickly and are easy to implement, but they have a number of limitations: susceptibility to overfitting, limited ability to capture spatial or temporal dependencies, and dependence on high-quality preprocessing of data. To combat this, regularization (Dropout, L1/L2), normalization, and early stopping are used.

Convolutional neural networks (CNN). Although CNNs are mainly used for image processing, they have also shown high efficiency in working with tabular and sequential data. The main idea is to detect local patterns through convolution. The network structure is shown in Fig. 2.

In the context of network traffic, CNN can, for example, detect attack patterns or recurring patterns in feature sequences.

CNNs use filters that "slide" over the data, automatically extracting key characteristics. This allows the model to be noise-resistant and reduce dimensionality through pooling.

Another advantage is the reduction in the number of parameters compared to dense models. For network traffic, one-dimensional convolutions (Conv1D) are typically used.

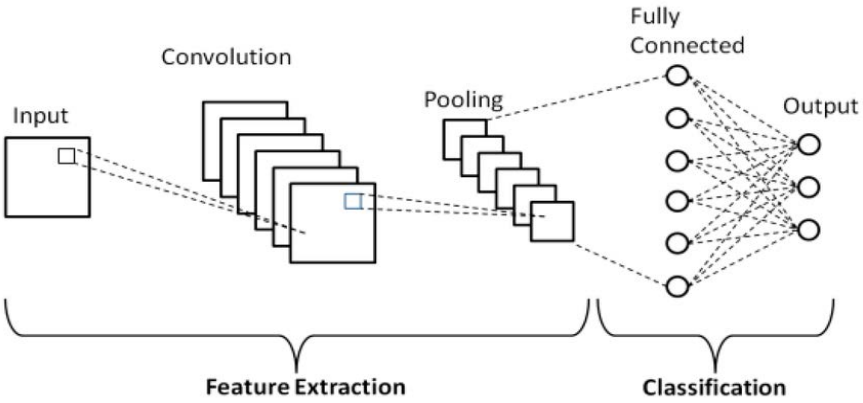


Fig. 2. Structure of a convolutional neural network

Recurrent neural networks (RNN), in particular LSTM. RNNs are designed to process sequential data. However, classic RNNs have a problem with losing long-term dependencies. This is solved by the LSTM (Long Short-Term Memory) architecture, which stores information for a long time using a special structure – a memory cell. The network architecture is shown in Fig. 3.

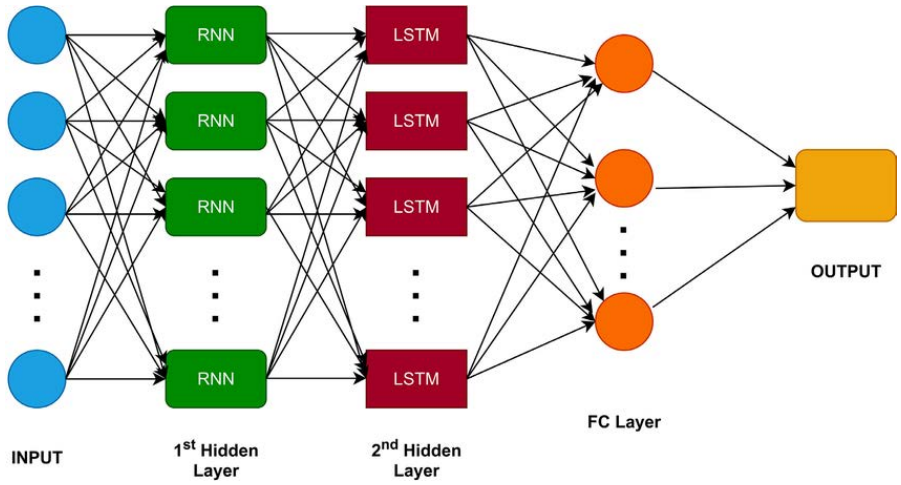


Fig. 3. Recurrent neural network structure

In the context of attack detection, LSTM allows analyzing user or system behavior not based on individual points, but as a sequence of events. For example, LSTM can detect abnormal activity that does not occur instantly, but accumulates over a period of time.

LSTM requires input data to be prepared in the form of sequences and scales well as the number of input features increases. However, it is resource-intensive and requires careful configuration [15].

Each of the models described has its own unique advantages that make them effective in different aspects of data processing. Dense networks are notable for their learning speed and stability of results, especially when working with tabular or already aggregated features. Convolutional neural networks (CNN) highlight key features even in the presence of noise or variations in input data due to their architecture. At the same time, LSTM networks are capable of retaining information for a long time, which makes it possible to identify long-term dependencies in the data. That is why this work proposes using an ensemble that combines the advantages of each of these models. This approach allows for better generalizability, increased classification accuracy, and resilience to different types of input data.

Ensemble learning is a powerful approach in machine learning that involves combining several models to improve the quality of predictions. The idea is based on the fact that a combination of weaker or independent models can give a better result than a single strongest model. This is especially important in highly complex tasks, such as detecting anomalies in network traffic. Ensembles allow you to: reduce the probability of model error; reduce the risk of overfitting; improve generalization on new data.

There are two main types of ensembles: homogeneous ensembles, which include identical base models, such as Random Forest (an ensemble of decision trees), where each tree is trained on a different subsample of data; heterogeneous ensembles, which combine different types of models, such as LSTM, CNN, and Dense neural networks. This combination allows you to make the most of the different nature of the models: sequence, spatiality, general patterns.

Among the main methods of ensemble learning, we note the following.

1. Bagging (Bootstrap Aggregating): Each model is trained on a random subsample of training data with replacement. Their results are then aggregated. This approach reduces model variance and helps avoid overfitting. The most famous example is Random Forest.

2. Boosting: Models are trained sequentially. Each subsequent model focuses on examples that were misclassified by the previous ones. Boosting significantly reduces bias error. Examples: AdaBoost, Gradient Boosting, XGBoost.

3. Stacking (stacked generalization): The outputs of several models (first level) are fed into a meta-model (second level), which learns to make the final prediction. This allows you to learn the optimal way to combine models.

4. Voting: Each model makes an independent prediction, and the final decision is made: Hard voting – based on the majority of classes (which is most often predicted); Soft voting – based on averaged probabilities, which is a more accurate approach when working with probabilistic models such as neural networks [16].

This work uses soft voting, in which the results of the CNN, LSTM, and Dense models are aggregated by averaging the probabilities of each class. This allows the "confidence" of each model to be taken into account and improves accuracy in cases of ambiguous classification.

Ensembles are particularly effective in cases where models have different architectures and are trained differently. For example, as mentioned earlier, LSTM captures temporal dependencies well, CNN captures spatial local patterns, and Dense captures global correlations. Their combination allows the system to be more flexible and adaptive to different attack patterns.

In addition, ensembles are important for tasks with imbalanced data, where some classes occur much less frequently than others. In such cases, single models often overestimate the dominant classes, while an ensemble allows for a more balanced distribution of "confidence."

UNSW-NB15 is a publicly available dataset created in 2015 by the Australian Centre for Cyber Security at the University of New South Wales. Its purpose is to provide a modern foundation for research in the field of attack detection and evaluation of the effectiveness of various machine learning algorithms in a realistic network environment [17].

The data was collected using the IXIA PerfectStorm attack generator, which allows you to create realistic traffic with both legitimate and malicious flows. In total, over 2 million packets were collected and grouped into 100+ unique features, including: number of packets in the incoming and outgoing directions; number of bytes; session duration; TCP/UDP flags; statistical parameters (mean, standard deviation); behavioral indicators (number of connections, interactions); traffic categories (FTP, DNS, HTTP, etc.).

UNSW-NB15 covers 9 types of attacks and a variant of legitimate traffic: Fuzzers – sending random data to detect vulnerabilities; Analysis – active scanning, including ports or network protocols; Backdoors – hidden remote access to the system; DoS – denial-of-service attacks; Exploits – using known vulnerabilities; Generic – universal attacks on encryption or authentication; Reconnaissance – reconnaissance and information gathering; Shellcode – injection of malicious commands; Worms – self-replicating attacks; Normal – legitimate traffic without malicious actions.

One of the important features of this dataset is class imbalance: for example, the "Normal" and "Generic" classes are represented by a significantly larger number of records, while some rare attacks (such as Worms or Shellcode) occur in only a few dozen examples. This complicates the model training process and requires the use of special methods, such as class weighting or ensembles.

UNSW-NB15 is often used in research on multi-class classification due to its rich set of features (especially behavioral ones), wide range of attacks, and proximity to the real-world conditions of modern networks.

As part of this work, the dataset was pre-processed: duplicates were removed, numerical values were normalized, categorical features were encoded, and the data was divided into training and test samples.

The experiments were conducted on a computer with an Intel Core i7-11700F processor (8 cores, 2.5–4.9 GHz), 32 GB of DDR4 RAM, an NVIDIA GeForce RTX 3060 graphics card (12 GB VRAM), and Windows 11 Pro operating system. Python libraries were used to implement the models: TensorFlow 2.13, Keras, NumPy, Pandas, Matplotlib. The models were trained using a GPU, which significantly reduced processing time and ensured stable results. All experiments were repeatable, and the code was adapted to run on similar configurations.

5. Research results

5.1. Description of the experimental comparative analysis methodology

To conduct an experimental comparative analysis of the effectiveness of network attack classification models, a number of quantitative criteria were selected to objectively evaluate the quality of neural networks.

The main criteria are:

Accuracy – the proportion of correctly classified examples among all examples.

Recall – the ability of the model to detect all positive examples of a certain class.

Precision – the proportion of correctly classified positive examples among all those classified as positive by the model.

F1-measure – the harmonic mean between precision and recall, which allows you to take into account the balance between them.

Confusion Matrix – a visualization of classification results for each class, allowing you to evaluate the specifics of errors.

Loss Function – an indicator that reflects the level of error in the model during training.

Training and inference time – a technical criterion that allows you to evaluate the computational efficiency of models.

The quantitative results of the experiments are based on testing the models on the validation part of the UNSW-NB15 dataset. For each model, the values of the above metrics are calculated and then compared with each other. In addition, for the ensemble model, the impact of the soft voting mechanism on the overall classification quality is additionally analyzed.

The sequence of experimental studies was structured as follows:

Preliminary data processing: feature normalization, class balancing, training and test sample formation.

Training of individual models: dense neural network (Dense) – a basic model for tabular data; convolutional neural network (CNN) – a model for detecting local patterns; recurrent neural network (LSTM) – a model for sequence analysis.

Evaluation of the effectiveness of each model: construction of loss and accuracy function graphs, error matrices, calculation of metrics.

Formation of a model ensemble: combining three architectures using soft voting.

Comparative analysis: comparison of the results of individual models and the ensemble, determination of the advantages of the combined approach.

This approach allows not only to evaluate the effectiveness of each architecture separately, but also to test the hypothesis regarding the feasibility of combining them within an ensemble classification system.

5.2. Data preparation

The UNSW-NB15 dataset, which is one of the most modern and comprehensive datasets for network traffic anomaly detection tasks, was used to implement the experimental part. This dataset contains both legitimate (normal) and anomalous (malicious) network connection records, which are distributed among 10 different attack classes.

The experimental study was performed in the Google Colab environment, which allows you to work efficiently with large amounts of data thanks to free access to GPUs.

The implementation used the Python language platform along with powerful libraries for data analysis and machine learning, namely: pandas – for processing and manipulating tabular data; scikit-learn – for preprocessing, data splitting, and building basic models; TensorFlow and Keras – for building, training, and evaluating deep neural networks [18, 19].

During the data preparation stage, a series of steps were taken to improve the quality of model training:

Step 1. Data merging: initially, two dataset files: UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv were merged into a single dataframe to provide a unified processing system.

Step 2. Removal of irrelevant features: fields that do not contain useful information for the model or could potentially cause data leakage (e.g., id, label, proto, service, state, attack_cat) were removed from the data frame. Only numerical and informative traffic features were left.

Step 3. Filling in missing values: all missing (NaN) values in the selected features were replaced with zeros to avoid errors during model processing and training.

Step 4. Feature scaling: Numeric parameters were normalized using StandardScaler, which converts values to a range with a mean of 0 and a standard deviation of 1. This is especially important for neural networks that are sensitive to the scale of input data.

Step 5. Class label encoding: LabelEncoder was used to convert text attack names into numerical format. This allowed the model to work with labels as categories suitable for classification.

Step 6. Sample division: the final dataset was divided into training (80 %) and test (20 %) samples, taking into account stratification by class.

Stratification ensures that the proportions between classes are preserved in each subsample, which is critical for correct training and evaluation when there is an imbalance between the number of examples of different classes.

5.3. Building a base model (Dense)

For the initial stage of the experiments, a base model was developed based on a multilayer architecture (Dense).

This type of neural network works well with tabular data classification tasks and provides a basis for further comparisons with more complex architectures.

The structure of the built model consisted of the following layers: Dense (128), activation function – ReLU, L1L2 regularizer (0.02, 0.04) to combat overfitting, weight initializer – HeNormal; Dropout(0.2) to reduce the risk of overfitting; Dense(64) with ReLU activation; Dropout(0.3); Output layer Dense(10, activation='softmax'), which corresponds to the number of attack classes in the UNSW-NB15 dataset.

The Adam optimizer was used for optimization, and the loss function was sparse_categorical_crossentropy. Training took place over 30 epochs using early stopping based on val_loss, which prevented overfitting.

Fig. 4 shows the change in accuracy on the training and validation samples by epoch. It can be seen that the model converges quickly during the first 10 epochs, and then the accuracy stabilizes.

Already at epoch 15–20, the validation accuracy exceeds 80 %, which indicates the high generalizing ability of the model.

Fig. 5 shows the losses. Their steady decline on both samples confirms adequate training without signs of overfitting or underfitting.

After training was completed, the model was tested on a delayed test sample.

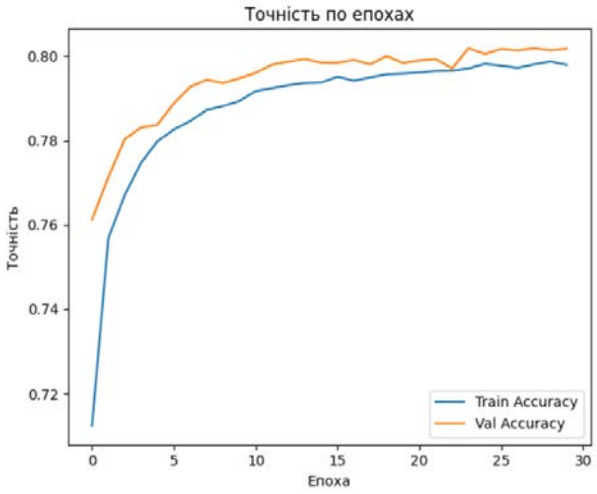


Fig. 4. Change in model accuracy on training and validation samples (Dense model)

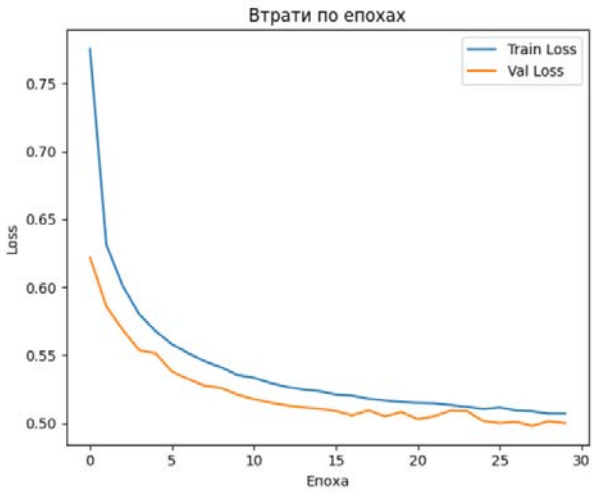


Fig. 5. Change in the loss function on the training and validation samples (Dense model)

Fig. 6 shows the confusion matrix, which allows us to evaluate how the model copes with each class separately. The model best recognizes the Normal, Generic, and Exploits classes, and slightly worse – DoS and Reconnaissance. The highest number of errors is noticeable among classes that have fewer examples in the dataset (Shellcode, Worms, Analysis).

Fig. 6. Classification error matrix for the Dense model

Table 1 shows examples of 10 random predictions of the Dense model. It can be seen that in most cases the model copes with classification correctly, but sometimes confuses DoS with Exploits or Normal with Exploits.

This is quite logical, since these types of attacks can have similar traffic characteristics, such as the number of packets, transmission speed, or identical network protocols.

Table 1. *Examples of Dense model predictions on random samples*

No	Expected class	Predicted class	Match
1	Exploits	Exploits	True
2	Normal	Exploits	False
3	Generic	Generic	True
4	Generic	Generic	True
5	Normal	Normal	True
6	Exploits	Reconnaissance	False
7	Reconnaissance	Reconnaissance	True
8	Fuzzers	Fuzzers	True
9	DoS	Exploits	False
10	Normal	Normal	True

5.4. Building a recurrent model (LSTM)

To account for the sequential nature of network traffic, a model based on long short-term memory (LSTM) was implemented – a type of recurrent neural network that allows context to be stored over time.

LSTM architectures are particularly effective in tasks where the order of input features affects the result, particularly in cases involving traffic logs or temporal dependencies between packets.

Before feeding the input data into the LSTM model, the dimension of X_train was changed from a two-dimensional matrix (samples, features) to a three-dimensional one (samples, timesteps, features), where timesteps = 1.

The architecture consisted of: the first LSTM layer with 64 neurons without sequence return (return_sequences=False), allowing the use of the following fully connected layer; two Dropout layers (0.3 and 0.2) to reduce the risk of overfitting; an intermediate Dense(64) layer with the ReLU activation function ( ); a final Dense(num_classes, activation='softmax') layer for multi-class classification.

The model was compiled using the Adam optimizer, the sparse_categorical_crossentropy loss function, and the accuracy metric.

Fig. 7 and 8 show the graphs of accuracy and loss function changes for training and validation samples, respectively.

As in the case of the Dense model, the accuracy curve stabilizes at around 80 %, and the loss decreases to below 0.5, indicating effective training.

The evaluation on the test sample confirmed the training results: Test accuracy: 0.8047, Test loss: 0.4842.

Thus, the LSTM model demonstrates a result very close to the Dense model, with a slightly higher val_accuracy, indicating its stronger generalization ability.

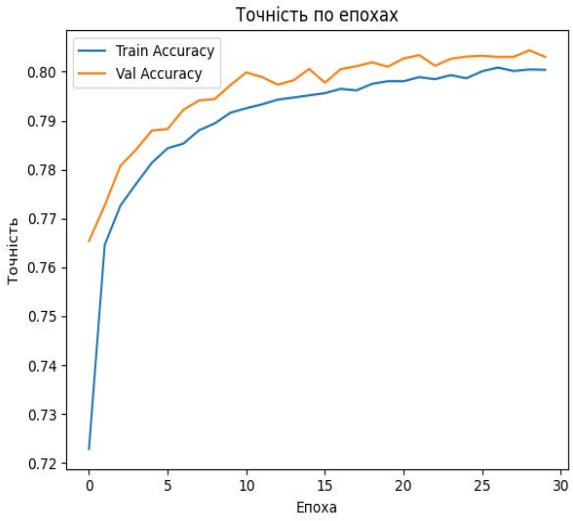


Fig. 7. Accuracy on training and validation samples (LSTM model)

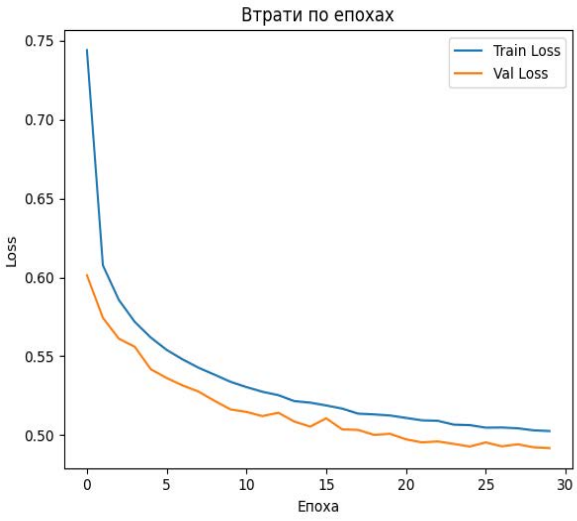


Fig. 8. Losses on training and validation samples (LSTM model)

Fig. 9 shows the error matrix for LSTM. It can be seen that the model classifies the Generic, Normal, and Exploits classes very well, but has difficulties with less common classes (Shellcode, Analysis, Worms), which is a typical problem for unbalanced datasets.

Of particular note is the improved recognition of Reconnaissance (compared to Dense), which indicates better sensitivity to sequential patterns in attack behavior.

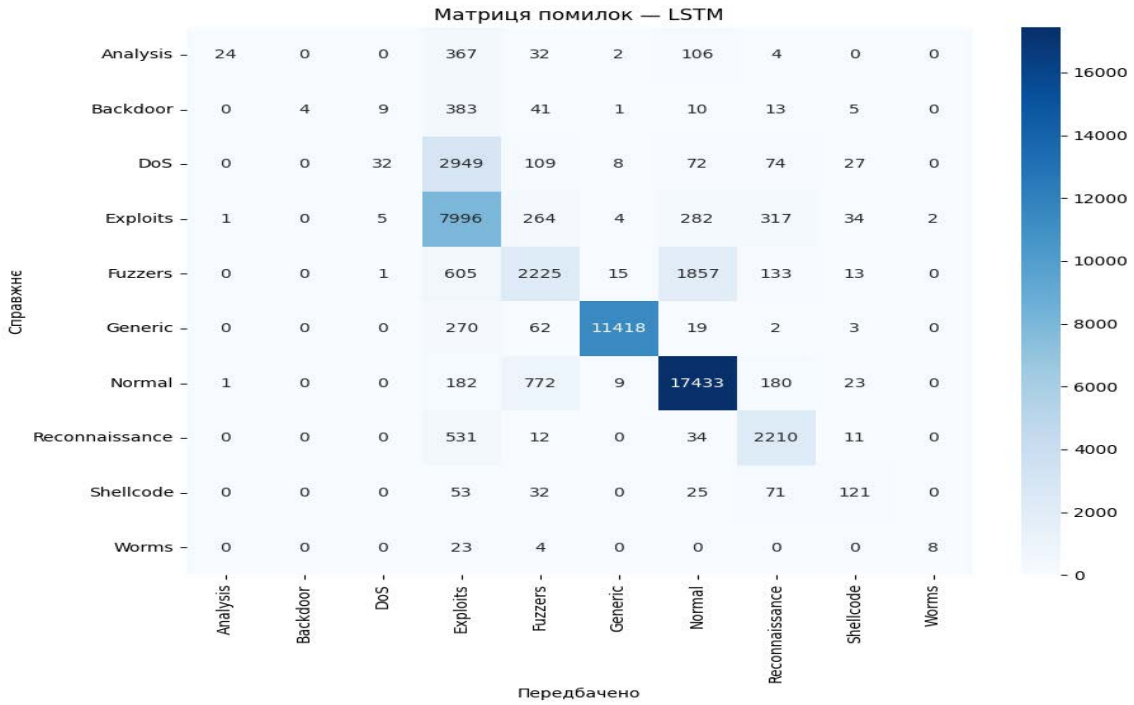


Fig. 9. Classification error matrix for the LSTM model

Table 2 shows examples of 10 random predictions made by the LSTM model. In most cases, the model classifies traffic correctly, but there are also false predictions. The most common ones are mixing DoS and Exploits attack types, as well as classifying Analysis as Exploits.

There is a logical explanation for such errors. In particular, DoS and Exploits can have similar numerical characteristics (e.g., number of packets, transmission speed, volume of bytes transmitted), which makes it difficult to distinguish them based on tabular features. In addition, the model rarely sees Analysis attacks during training, so it tries to "fit" them into the closest familiar pattern, which is often Exploits.

Table 2. *Prediction results for 10 random predictions (LSTM model)*

No	Expected class	Predicted class	Match
1	Normal	Exploits	True
2	DoS	Exploits	False
3	Analysis	Exploits	False
4	Generic	Generic	True
5	Generic	Generic	True
6	Generic	Generic	True
7	Exploits	Exploits	True
8	Normal	Normal	True
9	Normal	Normal	True
10	Normal	Normal	True

These observations are consistent with the error matrix analysis (Fig. 9), which also shows a high number of incorrect predictions for classes with low representation. This indicates that the LSTM model, despite its ability to take sequential dependencies into account, remains vulnerable to class imbalance and less effective at recognizing rare attacks.

Therefore, despite its high overall accuracy, this example demonstrates the need for additional mechanisms to adapt to imbalanced data or for ensemble learning with other types of models capable of compensating for these weaknesses.

5.5. Building a convolutional model (CNN)

A convolutional neural network (CNN) was implemented to identify local patterns in the network traffic structure. Although CNNs are traditionally used for image processing, in this case, it was adapted to work with tabular data that had been previously converted into a format suitable for convolutional processing.

The model consisted of a convolution layer (Conv1D), a subsampling layer (MaxPooling1D), as well as a dense layer (Dense) and a Dropout layer, which reduces the likelihood of overfitting. In the final layer, the softmax function was used to perform multi-class classification. The model training lasted 20 epochs. At the final stage, the model achieved an accuracy of 80.16 % on the test sample, which can be considered a fairly high indicator for the task of multi-class classification of network attacks. The loss function value was 0.5003.

Fig. 10 shows that the accuracy of the model steadily increased during training. The initial accuracy value was within 0.74, and after a few epochs, the model achieved over 0.79 on the validation sample. This indicates that the model learns well and does not lose its ability to generalize.

The graph in Fig. 11 shows a gradual decrease in losses on both training and validation data. There are no sharp jumps or discrepancies between the curves, which confirms the stability of the model's learning. Such a smooth decrease in losses is an important criterion for a balanced architecture without overfitting.

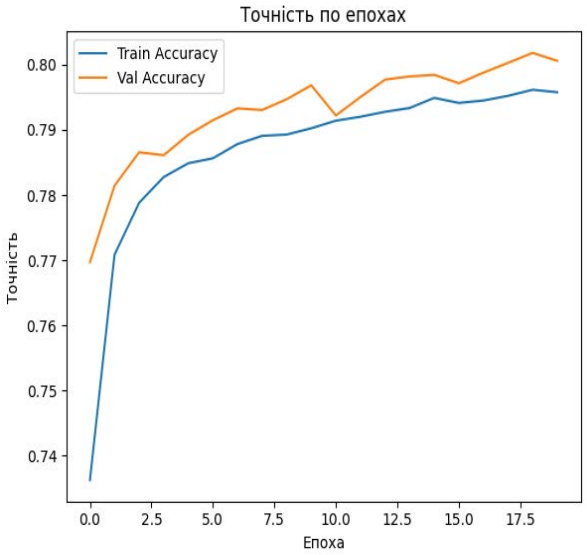


Fig. 10. Change in model accuracy on training and validation samples (CNN model)

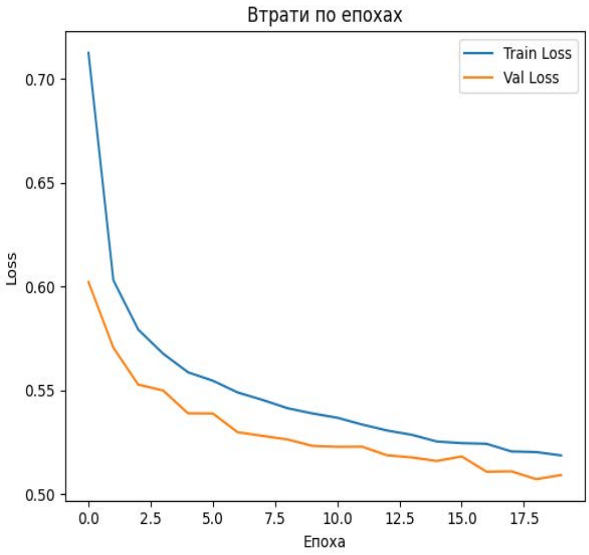


Fig. 11. Change in loss function on training and validation samples (CNN model)

The error matrix shown in Fig. 12 demonstrates that the model performs well in classifying the main classes, namely Normal, Generic, and Fuzzers.

Fig. 12. Classification error matrix for the CNN model

At the same time, as in the case of previous architectures, there is a tendency to confuse DoS, Backdoor, and Analysis with the Exploits class.

Table 3 shows examples of random predictions of the convolutional model. Out of ten attempts, the model correctly classified nine, and the error was again observed when predicting a DoS attack, which the system mistook for Exploits. This once again points to the difficulty of distinguishing between attacks with similar characteristics.

Table 3. *Prediction results for 10 random predictions (CNN model)*

No	Expected class	Predicted class	Match
1	Normal	Normal	True
2	Exploits	Exploits	True
3	Exploits	Exploits	True
4	Normal	Normal	True
5	DoS	Exploits	False
6	Exploits	Exploits	True
7	Generic	Generic	True
8	Normal	Normal	True
9	Exploits	Exploits	True
10	Normal	Normal	True

Overall, the CNN model demonstrated high efficiency and a good balance between accuracy and stability, ranking among the top of all tested architectures.

5.6. Formation of a model ensemble and testing on random attacks

At the final and most important stage of the practical research, ensembles of several models were formed to improve the quality of classification.

The idea behind the ensemble is to combine the predictions of several neural networks by averaging their probabilistic forecasts. This approach allows us to level out the individual weaknesses of each separate model and increase the overall accuracy of network attack detection.

An ensemble of two models: Dense + LSTM

First, an ensemble was formed that combines a fully connected model (Dense) and a recurrent LSTM neural network. Each of them has its own architectural specifics: Dense captures global relationships between features well, while LSTM captures sequential patterns characteristic of time series.

Table 4 shows examples of 10 random attacks predicted by this ensemble. In most cases, the classification results are correct, but there is some confusion between the DoS, Fuzzers, and Exploits classes**.** This is to be expected, as these types of attacks can have similar characteristics – for example, a short period of activity, similar traffic volumes, or similar query patterns.

Despite these challenges, the combination of the two models significantly improves the results compared to each of them separately, as evidenced by the increase in accuracy and the decrease in the average error.

Table 4. *Prediction results for 10 random attacks (Dense and LSTM ensemble)*

No	Expected class	Predicted class	Match
1	Normal	Normal	True
2	Generic	Generic	True
3	Normal	Normal	True
4	Reconnaissance	Exploits	False
5	Exploits	Exploits	True
6	Exploits	Exploits	True
7	Generic	Generic	True
8	Normal	Normal	True
9	Generic	Generic	True
10	Fuzzers	Exploits	False

An ensemble of three models: Dense + LSTM + CNN

Next, an extended ensemble was formed, to which a convolutional neural network (CNN) was added in addition to the Dense and LSTM models. Unlike the two previous architectures, CNN effectively detects local patterns in the input data structure, which is particularly useful for capturing relationships between closely spaced features.

This combination achieved the highest accuracy: all 10 random attacks were correctly classified (Table 5). This indicates that the three different models "learn" from different aspects of the data, and their combination provides the most complete coverage of network traffic variability.

Table 5. *Prediction results for 10 random attacks (ensemble of three models)*

No	Expected class	Predicted class	Match
1	Generic	Generic	True
2	Normal	Normal	True
3	Normal	Normal	True
4	Normal	Normal	True
5	Exploits	Exploits	True
6	Exploits	Exploits	True
7	Normal	Normal	True
8	Reconnaissance	Reconnaissance	True
9	Norma	Norma	True
10	Generic	Generic	True

The advantages of this configuration are also reflected in a reduction in the number of errors among less represented classes, such as Worms, Shellcode, and Backdoor, which are traditionally difficult to recognize. There is also an increase in the model's generalization ability on new examples – it copes more accurately with new samples that were not encountered during training.

Ensemble models demonstrated significantly higher efficiency compared to each individual architecture. This approach compensates for the shortcomings of each model, in particular: Dense – prone to losing spatial or sequential dependencies; LSTM – sensitive to sequence length and may confuse similar patterns; CNN – works well with local structures but does not cover the

complete temporal or global picture. When combined, they provide high accuracy, consistency, and adaptability, which is critical in network attack detection tasks, where anomalies can be subtle and difficult to detect. This approach also opens up prospects for further expansion – you can experiment with the weight of each model in the ensemble, apply meta-classifiers, or even automatically train an algorithm that will control the voting of models based on context.

6. Discussion of research results

During the experimental study, three separate deep learning models were implemented – Dense, CNN, and LSTM – as well as their combination into an ensemble based on the soft voting principle. Each model was trained on the same UNSW-NB15 dataset, with the same epoch parameters, batch size, and categorical cross-entropy loss function. Accuracy, precision, completeness, F1-score, and error matrix analysis metrics were used to evaluate effectiveness.

The dense model demonstrated stable accuracy at 84.2%, with an F1 score of 0.81. It trained quickly but tended to overestimate dominant classes, which reduced its effectiveness in classifying rare attacks. The CNN model achieved an accuracy of 86.7%, with an F1 score of 0.83. It was better at detecting local patterns, especially for Shellcode and Worms attacks, but had difficulties with temporal dependencies.

The LSTM model showed the highest accuracy among the individual models – 88.1 %, with an F1 score of 0.85. It effectively detected sequential anomalies, such as Reconnaissance and DoS, but required more training time and was sensitive to optimization parameters.

The ensemble model (Dense + CNN + LSTM) achieved the best results: overall accuracy of 90.3 %, F1-score of 0.88, and a 12 % reduction in false positives compared to individual models. The error matrix showed a more even distribution of classification between classes, with improved recognition of rare attacks.

A comparative analysis confirmed that the ensemble model has an advantage over individual architectures across all key metrics. Its ability to combine local, global, and temporal features allowed it to achieve the highest generalizability and robustness to data imbalance. Thus, the ensemble proved to be the most effective solution for the task of classifying attacks at the routing level.

Despite the high quality of the UNSW-NB15 dataset, it has certain limitations: a limited number of examples for some types of attacks, artificially generated flows that may not fully reflect real conditions. In addition, the selected architectures have their drawbacks: the Dense model does not take into account temporal dependencies, CNN is limited in a global context, and LSTM is resource-intensive. The proposed comparative analysis method also has limitations: it does not take into account the influence of hyperparameters on the results, and soft voting does not always optimally combine models. In future studies, it is advisable to consider meta-models (stacking) and adaptive ensemble methods.

The results of this study prove that combining neural network architectures in an ensemble is not only a theoretically sound but also a practically feasible strategy for improving the reliability of attack detection systems.

This approach can be applied in real IDS systems, as well as in user behavior monitoring, where accuracy and adaptability are critical.

7. Conclusions

An analysis of types of network attacks at the routing level was conducted. As a result, the main types of attacks (DoS, Spoofing, MitM, Routing Table Poisoning, etc.) were systematized and their characteristic features were identified. This made it possible to form a set of relevant features for further classification. Unlike some publications, which focus only on the application level, this study focuses on the routing level, which is less researched.

The advantages and disadvantages of traditional and modern attack detection methods were evaluated. It was found that signature-based methods are not capable of detecting new threats, while anomaly-based methods have a high number of false positives. This justified the use of a hybrid approach with elements of deep learning. Unlike some studies, where the analysis is limited to only one category of methods, this study provides a comprehensive comparison.

Deep learning architectures (Dense, CNN, LSTM) were analyzed for the task of network traffic classification. It was found that each of the models has specific advantages: Dense – speed and simplicity, CNN – detection of local patterns, LSTM – sequence processing.

This became the basis for the formation of the ensemble. Compared to analogues, where usually only one architecture is used, a combined approach is proposed.

Individual models and their ensemble were implemented and tested. The following results were achieved: accuracy of Dense – 84.2 %, CNN – 86.7 %, LSTM – 88.1 %.

The ensemble model outperformed all individual architectures, achieving an accuracy of 90.3 % and an F1-score of 0.88. This confirms the effectiveness of combining models within soft voting.

A comparative analysis of model effectiveness was conducted. It was found that the ensemble provides the best balance between accuracy, completeness, and resistance to data imbalance. Compared to similar studies that use only basic metrics, this study additionally considers error matrices, loss functions, and inference time.

The limitations of the study were identified. The main drawbacks are the limited representativeness of the UNSW-NB15 dataset in terms of real traffic, as well as the sensitivity of LSTM to training parameters.

The soft voting technique does not always provide an optimal combination of models, which opens up prospects for further research using meta-models (stacking) and attention mechanisms.

References

1. Stallings, W. (2017), "Network security essentials: applications and standards", *Pearson, 4th edition*, 432 p.
2. Obaid, H. S., Abeed, E. H. (2020), "DoS and DDoS Attacks at OSI Layers", *International Journal of Multidisciplinary Research and Publications (IJMRAP),* Vol. 2 (8), P. 1–9. DOI: https://doi.org/10.5281/zenodo.3610833
3. Butun, I., Österberg, P., Song, H. (2019), "Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures", *IEEE Communications Surveys & Tutorials*, Vol. 22, P. 616-644.
4. Mell, K., Scarfone, K. (2007), "Guide to Intrusion Detection and Prevention Systems", *NIST*.

5. Melnikova, L., Linnyk, E., Pastushenko, I. (2024), "Assessment of internet providers in Ukraine: a multi-criterion decision-making model", International Scientific and Technical Conference "*Information and Communication Technologies and Cybersecurity" (ICTC-2024)*, P. 111–114. Available: https://ice.nure.ua/wp-content/uploads/2024/12/22_Melnikova-L.I.-Linnyk-O.V.-Pastushenko-I.Iu._Str.111-114.pdf
6. LeCun, Y., Bengio, Y., Hinton, G. (2015), "Deep Learning", *Nature*.
7. Mishra, B., Sahu, S. (2020), "Evaluation of DNN models in cybersecurity", *IEEE Access*.
8. Binbusayyis, A. (2024), "Reinforcing Network Security: Network Attack Detection Using Random Grove Blend in Weighted MLP Layers", *Mathematics*, Vol. 12 (11), 1720. DOI: https://doi.org/10.3390/math12111720
9. Abuagoub, A. (2024), "Security Concerns with IoT Routing: A Review of Attacks, Countermeasures, and Future Prospects", *Advances in Internet of Things*, Vol. 14, P. 67–98. DOI: https://doi.org/10.4236/ait.2024.144005
10. Lin, Z., Shi, Y., Xue, Z. (2022), "Idsgan: Generative adversarial networks for attack generation against intrusion detection", *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Chengdu, China, May 16–19, Springer*, P. 79–91.
11. Abbasi, M., Florez, S., Shahraki, A., Taherkordi, A., Prieto, J., Corchado Rodríguez, J. (2025), "Class Imbalance in Network Traffic Classification: An Adaptive Weight Ensemble-of-Ensemble Learning Method", *IEEE Access*, P. 1-1. DOI: https://doi.org/10.1109/ACCESS.2025.3538170
12. Alshamrani, A. et al. (2019), "A Survey on IDS for Cloud Environments", *Journal of Network and Computer Applications*.
13. Alamleh, H., Estremera, L., Arnob, S. S., AlQahtani, A. A. S. (2025), "Advanced Persistent Threats and Wireless Local Area Network Security: An In-Depth Exploration of Attack Surfaces and Mitigation Techniques", *Journal of Cybersecurity and Privacy*, Vol. 5 (2), 27. DOI: https://doi.org/10.3390/jcp5020027
14. Goyal, P. (2025), "The Role of Databases in Cybersecurity and Threat Detection: Advancements through Spanner Graph Technology", *Journal of Computer Science and Technology Studies*, Vol. 7, P. 544–557. DOI: https://doi.org/10.32996/jcsts.2025.7.5.61
15. Thwaini, M. H. (2022), "Anomaly detection in network traffic using machine learning for early threat detection", *Data and Metadata*, Vol. 1, P. 34–34. DOI: https://doi.org/10.56294/dm202272
16. Zhou, Z.-H. (2012), "Ensemble Methods: Foundations and Algorithms", *Microsoft Research Ltd*, 232 p.
17. UNSW-NB15 (2025), "Dataset". Available: https://www.kaggle.com/datasets/dhoogla/unswnb15
18. Google Colaboratory (2025), "Platform". Available: https://colab.google/
19. Shtangey, S., Melnikova, L., Hrebeniuk, Y. (2025), "Adaptation of Multiclass Classification Methods for Identifying Network Attack Types in Routing Protocols Using Structured Databases", *Zenodo, Aug. 1*. DOI: https://doi.org/10.5281/zenodo.16684887

Received (Надійшла) 19.08.2025
Accepted for publication (Прийнята до друку) 01.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Shtangey Svitlana – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Infocommunication Engineering V. V. Popovsky, Kharkiv, Ukraine; e-mail: Svitlana.shtanhei@nure.ua; ORCID ID: https://orcid.org/0000-0002-9200-3959

Melnikova Lubov – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Infocommunication Engineering V. V. Popovsky, Kharkiv, Ukraine; e-mail: liubov.melnikova@nure.ua; ORCID ID: https://orcid.org/0000-0003-0439-7108

Marchuk Artem – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor at the Department of Infocommunication Engineering V. V. Popovsky, Kharkiv, Ukraine; e-mail: artem.marchuk@nure.ua; ORCID ID: https://orcid.org/0000-0002-2720-3954

Linnyk Olena – PhD (Engineering Sciences), Associate Professor, National Technical University "Dnipro Polytechnic", Associate Professor at the Department of Mechanical and Biomedical Engineering, Dnipro, Ukraine; e-mail: linnyk.ol.o@nmu.one; ORCID ID: https://orcid.org/0000-0002-4906-3796

Hrebeniuk Yelizaveta – Kharkiv National University of Radio Electronics, Higher Education Applicant, Faculty of Infocommunications, Kharkiv, Ukraine; e-mail: yelyzaveta.hrebeniuk@nure.ua; ORCID ID: https://orcid.org/0009-0002-4648-3485

Штангей Світлана Вікторівна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри інфокомунікаційної інженерії ім. В. В. Поповського, Харків, Україна.

Мельнікова Любов Іванівна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри інфокомунікаційної інженерії ім. В. В. Поповського, Харків, Україна.

Марчук Артем Володимирович – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри інфокомунікаційної інженерії ім. В. В. Поповського, Харків, Україна.

Лінник Олена Вячеславівна – кандидат технічних наук, доцент, Національний технічний університет "Дніпровська політехніка", доцент кафедри машинобудування та біомедичної інженерії, Дніпро, Україна.

Гребенюк Єлизавета Андріївна – Харківський національний університет радіоелектроніки, здобувач вищої освіти, факультет інфокомунікацій, Харків, Україна.

АДАПТАЦІЯ АЛГОРИТМІВ БАГАТОКЛАСОВОЇ КЛАСИФІКАЦІЇ ДЛЯ ВИЯВЛЕННЯ ТИПУ МЕРЕЖЕВОЇ АТАКИ В МАРШРУТИЗАЦІЙНИХ ПРОТОКОЛАХ З ВИКОРИСТАННЯМ СТРУКТУРОВАНИХ БАЗ ДАНИХ

Предметом статті є адаптація алгоритмів багатокласової класифікації для виявлення типів мережевих атак у маршрутизаційних протоколах. **Мета дослідження** – розробити й експериментально перевірити ефективність різних архітектур глибокого навчання (*Dense*, CNN, LSTM) у задачі багатокласової класифікації мережевих атак, а також оцінити доцільність їх об’єднання в ансамбль для підвищення точності та стійкості класифікації. Для досягнення окресленої мети необхідно виконати такі **завдання:** проаналізувати типи мережевих атак, властивих для маршрутизаційного рівня, та їх ознаки; оцінити переваги й недоліки традиційних і сучасних методів виявлення атак, зокрема сигнатурні, аномальні та гібридні системи; дослідити архітектури глибокого навчання (*Dense*, CNN, LSTM) щодо їх придатності до класифікації мережевого трафіку; реалізувати окремі моделі та їх об’єднання в ансамбль із

застосуванням *voting*-механізму. **Використано такі методи:** глибокі нейронні мережі різних типів, ансамблеве навчання (*bagging, stacking, voting*), а також аналіз дисбалансованих даних. Для перевірки ефективності моделей застосовано датасет UNSW-NB15, що містить реалістичні приклади нормального й аномального трафіку. Експерименти проведено із застосуванням сучасних бібліотек машинного навчання, а також регуляризації та нормалізації для запобігання перенавчанню. **Результати дослідження.** Реалізовано й протестовано три архітектури нейронних мереж. *Dense*-модель підтвердила стабільні результати на агрегованих ознаках, CNN ефективно виділяла локальні патерни навіть за наявності шуму, а LSTM забезпечила виявлення довгострокових залежностей у послідовних даних. Ансамбль моделей продемонстрував вищу точність класифікації порівняно з окремими архітектурами, зменшив кількість хибнопозитивних результатів і підвищив узагальнюваність. **Висновки.** Адаптація та поєднання різних архітектур глибокого навчання дають змогу суттєво покращити якість багатокласової класифікації мережевих атак. Ансамблевий підхід забезпечує стійкість до дисбалансу даних і підвищує точність виявлення складних атак. Досягнуті результати підтверджують доцільність використання ансамблів у завданнях кібербезпеки та відкривають перспективи для подальших досліджень, зокрема інтеграції моделей у системи реального часу й розширення аналізу на інші типи мережевих загроз.

Ключові слова: мережеві атаки; протоколи маршрутизації; глибоке навчання; щільні нейронні мережі; ансамблеве навчання; виявлення вторгнень; кібербезпека.

Bibliographic descriptions / Бібліографічні описи

Shtangey, S., Melnikova, L., Marchuk, A., Linnyk, O., Hrebeniuk, Y. (2025), "Adaptation of multiclass classification algorithms for identifying network-attack types in routing protocols using structured databases", *Management Information Systems and Devises*, No. 4 (187), P. 278–298. DOI: https://doi.org/10.30837/0135-1710.2025.187.278

Штангей С. В., Мельнікова Л. І., Марчук А. В., Лінник О. В., Гребенюк Є. А. Адаптація алгоритмів багатокласової класифікації для виявлення типу мережевої атаки в маршрутизаційних протоколах з використанням структурованих баз даних. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 278–298. DOI: https://doi.org/10.30837/0135-1710.2025.187.278

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 621.396 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.299

D. Kovalchuk, O. Zhyla

MEASUREMENT GEOMETRY, SIGNAL MODELS AND ALGORITHMS FOR RADIO IMAGE RECOVERY IN SYNTHESIZED APERTURE RADARS USING CONTINUOUS LFM SIGNALS

This article analyzes the methods of forming radar images on the surface formed by SAR with a continuous LFM probing signal. It is of interest to determine the main algorithmic operations performed on the "raw" data after their registration in the receivers. The geometry of measurements, the sounding signal and the features of the formation of "raw" data that will determine further processing will be considered. To compare the quality of work of different algorithms, a simulation model of the formation of radar images in RSA with the processing of continuous LFM signals has been developed. **The aim of the work** is to create a universal geometric basis for building effective measurement schemes and signal processing algorithms in radio engineering systems. **The research tasks** include: 1) formalization of the problem of determining coordinates based on the results of direction-finding measurements; 2) construction of a mathematical model of the mutual location of objects in three-dimensional space; 3) determination of the influence of geometric factors on the measurement accuracy; 4) analysis of single and multiple observation options. **The results obtained** allow establishing an analytical relationship between the measurement parameters and the configuration of the spatial scene, which ensures an increase in the accuracy of coordinate determination. **Field of application**: the results can be used to increase the efficiency of navigation, reconnaissance and monitoring systems, as well as in the tasks of tracking moving objects and building situational awareness systems.

Keywords: measurement geometry; direction finding; coordinate determination; three-dimensional model; radio measurement system.

Relevance

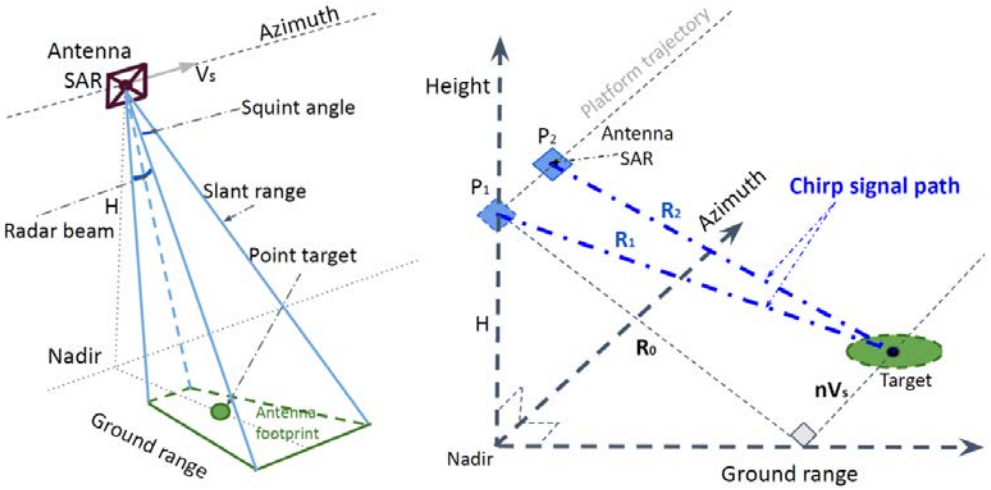
In today's environment of active development of unmanned aerial vehicles, monitoring and reconnaissance systems, there is a growing need for accurate determination of the coordinates and trajectories of radio emission sources. Reliable positioning is critical to ensuring the autonomy and effectiveness of such systems. Most measurement tasks require consideration of the spatial geometry of the scene, but existing approaches often simplify the model to two dimensions, which limits accuracy. Building a three-dimensional geometric model allows for adequate consideration of the influence of the location of receivers and signal sources. This, in turn, improves data processing algorithms and reduces coordinate determination errors.

The analysis of the influence of space configuration on measurement results during multiple observations is particularly relevant. Understanding geometric constraints allows for the rational placement of sensors and optimization of observation trajectories. The creation of a universal spatial model is an important prerequisite for the construction of high-precision radio measurement systems.

Therefore, the study of measurement geometry is of great importance for the development of modern radio engineering and navigation technologies.

Measurement geometry in the formation of radar images in the SAR from an aircraft

To illustrate the principle of forming an "artificial" aperture, we will use the geometry shown in Fig. 1 [1].



a b

Fig. 1. Geometry of surface scanning by radar implementing the antenna aperture synthesis algorithm: *a* – spatial position of the beam of a non-synthesized antenna; *b* – distance traveled by the signal during the movement of the carrier (taken from [1])

The idea of forming a synthesized aperture is the same for both pulsed and continuous wave radars. First, electromagnetic waves are emitted from an onboard antenna mounted on a moving platform (see Fig. 1 Antenna SAR) in the direction of the surface within the non-synthesized directional pattern, over a wide range of angles. The platform moves in a straight line at a speed V_s along a coordinate conventionally called Azimuth. The coordinate is called Azimuth because the synthesis of the antenna reduces the width of the directional pattern at the azimuth angle in a spherical coordinate system.

Thus, traditionally in the literature, SAR resolution or image characteristics are associated with the conventional azimuth coordinate. At the same time, all radar images are converted to Cartesian surface coordinates with units of measurement in meters. Another coordinate that plays an important role in measurements is the ground range (in Fig. 1 Ground range).

Ground range is converted from slant range (Fig. 1 Slant range) within the beam width of the physical antenna's directional pattern. The beam width is perpendicular to the flight path (Fig. 1 Radar beam) and determines the width of the viewing range along the ground range coordinate. In practice, the wider the viewing range, the faster the radio image of a given surface area is formed.

The size of the radio image pixel along the ground range coordinate is determined by the radar's resolution along the slant range. At oblique range, the resolution determines how many areas can be observed separately along the entire signal propagation path. Each area has

a constant value at oblique range, but when converted to ground range, the sizes of the areas change according to the formula

$$\Delta R_{Ground\ range} = \frac{\Delta R_{Slant\ range}}{\sin\theta}, \quad (1)$$

where $\Delta R_{Slant\ range}$ – resolution of the radio system in terms of slant range, θ – angle of observation of each point on the surface, $\Delta R_{Ground\ range}$ – resolution of the radio system in terms of ground range. The geometry of recalculating the areas of separate observation of objects on the surface is shown in Fig. 2.

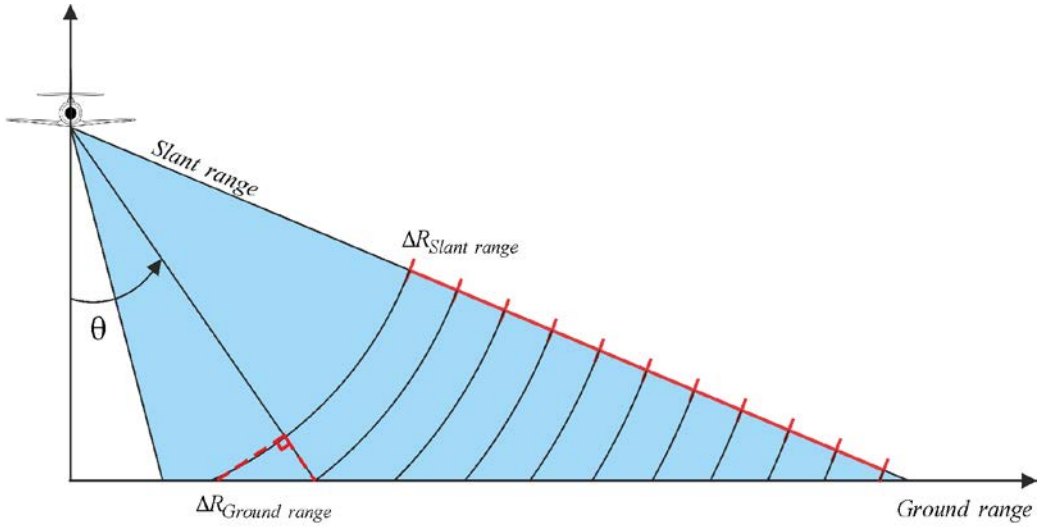


Fig. 2. Geometric transformations of resolution at slant range into resolution at ground range

The resolution in ground range for all types of probing signals is inversely proportional to their spectral width, therefore, along this coordinate, the main attention is paid to signal processing in time and within a certain modulation period [2–4]. Resolution along the other coordinate, azimuth, is determined by spatially-temporally coherent processing of complex envelope reflected signals accumulated during motion [2–4].

Fig. 1, *b* shows the process of signal emission from point P_1 and its reception at point P_2. The observation of the object (in Fig. 1, *a* Point target, in Fig. 1, *b* Target) occurs over a certain period of time, while the directional pattern, having a certain width along the flight path (in Fig. 1, *a* Squint angle), irradiates it.

The observation time at a fixed carrier speed or the interval of spatial accumulation of reflected signals determines the size of the synthesized aperture. The azimuth resolution is inversely proportional to the size of the synthesized aperture in the azimuth coordinate [2–4]. The azimuth resolution in angular coordinates at a known height and elevation angle or at a known slant range is converted into the resolution in spatial coordinates on the surface.

Thus, we have considered the geometry of measurements in SAR and determined the main values that affect the resolution of radar images.

A probing signal with linear frequency modulation looks like this [5]:

$$\dot{s}_t(t) = \exp\left\{ j\left(2\pi f_0 t + \pi k_r t^2\right)\right\}, \tag{2}$$

where $f_0 = c/\lambda$ is the carrier frequency of the probing signal, c is the speed of radio wave propagation, λ is the wavelength, $\pi k_r t^2$ is the quadratic phase shift inherent in the LFM signal, $k_r = \Delta f / t_p$ the rate of frequency change by an amount Df over the modulation period t_p.

In expression (2), a dot is placed above the probe signal designation, emphasizing that the signal model is presented in complex form. This approach is a mathematical simplification of all subsequent calculations. The physical signal emitted by the antenna cannot be complex and is described by the harmonic signal model as follows:

$$s_t(t) = A\cos\left(2\pi f_0 t + \pi k_r t^2\right), \tag{3}$$

where A is the amplitude of the harmonic oscillation. The following mathematical operations can be performed on expression (3):

$$\begin{aligned} s_t(t) &= A\cos\left(2\pi f_0 t + \pi k_r t^2\right) = \\ &= A\,\mathrm{Re}\left\{\exp\left(2\pi f_0 t + \pi k_r t^2\right)\right\}. \end{aligned}, \tag{4}$$

Assuming that the amplitude of the probing signal is equal to 1, in further calculations we move on to complex signals, omitting the operator $\mathrm{Re}\{\cdot\}$ and denoting $s_t(t)$ with a dot above. The received signal reflected from the surface has the form [5]

$$\dot{s}_r(t) = \exp\left\{ j\left(2\pi f_0 \left(t-\Delta t\right) + \pi k_r \left(t-\Delta t\right)^2\right)\right\}, \tag{5}$$

where $\Delta t = \left(R_1(n) + R_2(n)\right)/c$ – is the signal delay time during propagation, distances $R_1(n)$ and $R_2(n)$ are shown in Fig. 1, *b*. The variable n in distances is referred to in the literature as slow time, azimuth time, or azimuth coordinate of the radar image.

As a result of the carrier's movement, the received signal will have a Doppler frequency shift, which changes the initial carrier frequency by an amount $\alpha = c^2/\left(c^2 - V_s^2\right)$. Taking into account the formula for range [1]:

$$R_1(n) = \sqrt{R_0^2 + \left(V_s n\right)^2}, \tag{6}$$

we obtain the delay time

$$\Delta t = 2\alpha\left(R_1(n)/c + \left(V_s/c\right)^2 n\right). \tag{7}$$

In (6), R_0 – the range to the surface point at an angle of 90 degrees to the flight path.

According to the classical theory of optimal signal processing [7, 8], received signals must be processed in a matched filter or undergo correlation processing in a correlation integral. In radars with continuous LFM signal processing, the impulse response of the matched filter repeats the probing signal. Physical devices for implementing such processing include

a linear receiver path with a set of filters (input circuits, high-frequency filter), mixer, intermediate frequency filter, mixer for transfer to the zero-frequency region, low-frequency filter, low-frequency amplifier [8–10]. At the output of the matched filter after multiplication (5) and (2), we obtain

$$\begin{aligned} \dot{s}_{dc}(t) &= \dot{s}_t(t)\dot{s}_r^*(t) = \\ &= \exp\left\{j\left(2\pi f_0 t + \pi k_r t^2\right)\right\}\exp\left\{-j\begin{pmatrix} 2\pi f_0(t-\Delta t)+ \\ +\pi k_r(t-\Delta t)^2 \end{pmatrix}\right\} = \\ &= \exp\left\{j\left(2\pi f_0 \Delta t + 2\pi k_r t\Delta t - \pi k_r \Delta t^2\right)\right\}, \end{aligned} \tag{8}$$

where $(\cdot)^*$ – complex conjugation sign.

The resulting expression (8) can be discretized and further processed by a computer. This expression can also be used to test various algorithms for restoring radar images and focusing radio systems. Let us consider these algorithms in more detail.

Algorithms for restoring radar images in SAR with continuous LFM signal processing

Based on the analysis of the literature and formulas (1)–(8) presented, it can be stated that the radar image is represented in the coordinates of "fast" time t along the ground range and "slow" time n along the azimuth [1]. However, this representation is conditional, since in essence it only shows that spatial-temporal signal processing is necessary to form a radar image.

In the literature, such processing of received signals is also called two-dimensional and, in general, can be performed in four variations: 1) processing in time by ground range and processing in time by azimuth, 2) processing in the spectrum by ground range and processing in time by azimuth, 3) processing in time by ground range and processing in spectrum by azimuth, 4) spectral processing of signals by ground range and azimuth.

These four approaches and modifications of each of the time or spectrum processing methods have led to the development of a large number of algorithms for forming radio images based on the results of receiving so-called “raw” data in expression (8). Let us consider the main ones.

Omega-K algorithm (ωKA)

This algorithm accepts the assumption that $\Delta t^2 \to 0$ and phase shift $\pi k_r \Delta t^2$ does not carry useful information [5, 11]. In this case, the expression for "raw" data will look like this:

$$\dot{s}_0(t,n) = \exp\left\{j\left(2\pi f_0 \Delta t + 2\pi k_r t\Delta t\right)\right\}. \tag{9}$$

The first mathematical operation on (9) is the Fourier transform by coordinate t:

$$\dot{S}(f_t, n) = \int_{-t_0/2}^{t_0/2} \dot{s}_0(t,n)\cdot\exp\left(-j2\pi f_t t\right)dt, \tag{10}$$

where t_0 – observation time of received signals.

Next, we determine the Fourier transform by azimuth

$$\dot{S}(f_t, f_n) = \int_{-\infty}^{\infty} \dot{S}(f_t, n) \cdot \exp(-j2\pi f_n n)\, dn. \quad (11)$$

After calculating the Fourier transform in two coordinates, we obtain "raw" data for further processing in the spectral plane:

$$\dot{S}(f_t, f_n) = t_0 \mathrm{sinc}\left(\pi t_0 \left(f_t - k_r \Delta t\right)\right) \exp\left(j\varphi\left(f_t, f_n\right)\right), \quad (12)$$

where $\varphi(f_t, f_n) = \frac{4\pi\alpha R_0}{c}\sqrt{\left(f_0 + f_t\right)^2 - \left(cf_n/\left(2\alpha V_s\right)\right)^2}$ – frequencies by "fast" time coordinate t, f_n – frequencies by the "slow" time coordinate n.

The algorithm for processing "raw" data (12) according to the Omega-K algorithm is shown in Fig. 3.

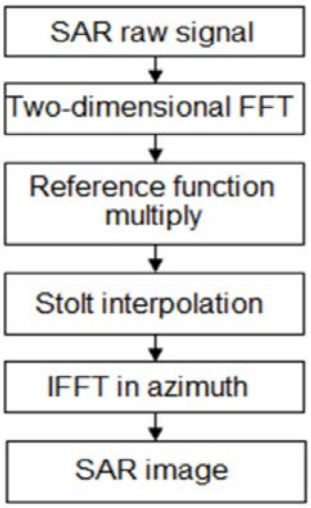


Fig. 3. Omega-K algorithm for processing "raw" data in SAR with continuous LFM signals (taken from [5])

After converting the received signals into spectral form, the resulting expression (12) is multiplied by the reference function

$$\dot{H}_{ref}(f_t, f_n) = \exp\left(-j\varphi_{ref}(f_t, f_n)\right), \quad (13)$$

where

$$\varphi_{ref}(f_t, f_n) = 4\pi\alpha R_{ref} \times \times\sqrt{\left(f_0 + f_t\right)^2 - \left(cf_n/\left(2\alpha V_s\right)\right)^2}\Big/c. \quad (14)$$

After multiplication, the phase will look like this:

$$\varphi_{RFM}(f_t, f_n) = 4\pi\alpha\left(R_0 - R_{ref}\right)\times \times\sqrt{\left(f_0 + f_t\right)^2 - \left(cf_n/\left(2\alpha V_s\right)\right)^2}\Big/c, \quad (15)$$

The next mathematical operation consists in interpolating data using the Stolt method [12–14].

This interpolation is performed in connection with the "migration" of the range to a separate point on the surface during the straight-line motion of the carrier.

The effect of range migration is shown in Fig. 4.

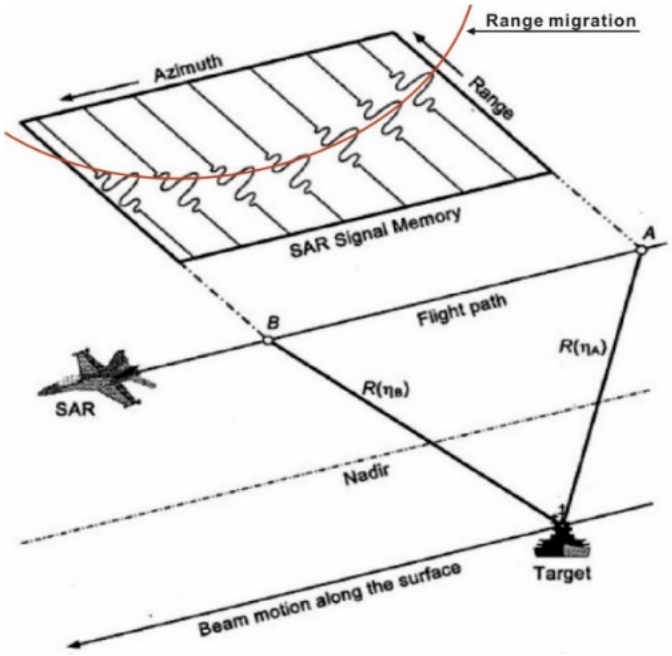


Fig. 4. Range migration to a point target in SAR ([1])

The range migration effect also manifests itself in the spectrum of received signals and leads to defocusing of radar images. The principle of data interpolation using the Stolt method is demonstrated in Fig. 5.

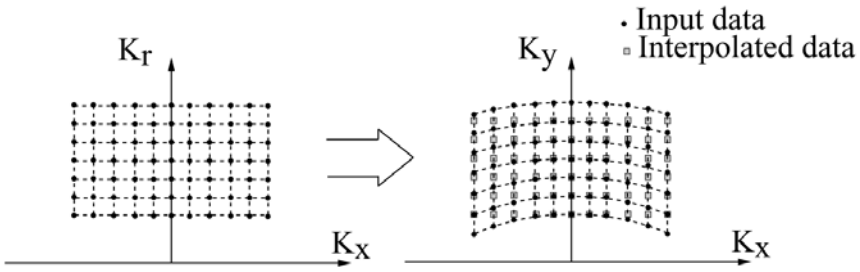


Fig. 5. Data interpolation using the Stolt method (taken from [15])

The essence of the presented interpolation is to replace the law of variation f_t from linear to nonlinear in such a way that the following equality occurs in expression (15):

$$\sqrt{\left(f_0+f_t\right)^2-\left(cf_n/\left(2\alpha V_s\right)\right)^2}=f_0+f_t'. \qquad (16)$$

In other words, it is necessary to replace f_t with a new variable f_t' to compensate for the "migration" of the distance along the coordinate f_t. Many works [1–10] are devoted to the issue

of interpolation using the Stolt method, most of which conclude that the new variable should be calculated as follows:

$$f_t' = \sqrt{\left(f_0 + f_t\right)^2 - \left(\frac{cf_n}{2\alpha V_s}\right)^2} - f_0\sqrt{1 - \left(\frac{cf_n}{2\alpha V_s f_0}\right)^2}\,. \tag{17}$$

The phase after interpolation will look like this

$$\varphi_{RFM}\left(f_t, f_n\right) = 4\pi\alpha\left(R_0 - R_{ref}\right)\cdot\left(f_0 + f_t'\right)/c\,. \tag{18}$$

The last algorithmic operation consists in calculating the inverse Fourier transform by azimuth. The result is a radar image of the surface formed by SAR with processing of continuous LFM signals. It should be noted that when implementing the Omega-K algorithm in pulse SAR, at the last stage it is necessary to calculate a two-dimensional Fourier transform in azimuth and range. A feature of processing in SAR with continuous LFM signals is the availability of information about the surface characteristics in each spectral component.

Modified Omega-K algorithm (ωKA-M)

The block diagram of the ωKA-M algorithm [16] is shown in Fig. 6.

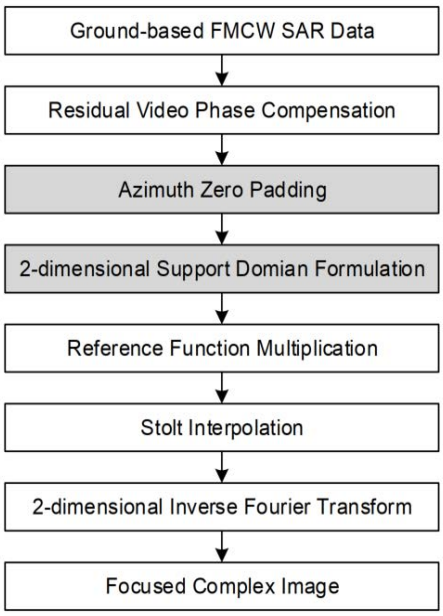


Fig. 6. Modified Omega-K algorithm for processing raw data in SAR with continuous LFM signals ([16])

Unlike the previous algorithm, instead of assuming that the phase component $\pi k_r \Delta t^2$ is insignificant, in this algorithm, the first operation is to compensate for it by calculating the forward Fourier transform in range, multiplying by the function

$$\dot{S}_{RVP}\left(f_t\right) = \exp\left(-j\left(\pi f^2 / k_r\right)\right), \tag{19}$$

and calculating the inverse Fourier transform in range.

The next operation in the block diagram in Fig. 6 is to supplement the "raw" data with zeros along the azimuth coordinate. This method was developed for a ground station with aperture synthesis moving on rails. The research in this work is devoted to the formation of radio images from an aircraft, where data is constantly fed into the processor during movement and, in this case, there is no need to supplement with zeros.

The main difference between the modified algorithm is the formation of two-dimensional data at the stage that the authors [16] called 2-dimensional Support Domain Formulation. To explain the essence of this operation, let us write down another representation of "raw" data

$$\dot{s}_0(t,x) = \exp\left\{ j\left(2\pi f_0 \frac{2R(x)}{c} + 2\pi k_r t \frac{2R(x)}{c} \right) \right\} \times \\ \times \exp\left\{ j\left(4\pi k_r / c^2\right) R^2(x) \right\}, \tag{20}$$

where x the analogue of "slow" time n, is just represented in more physical quantities – surface coordinates, $x = V_s \Delta t + x_n$, $x_n = V_s n T_p$ – discrete positions of the SAR carrier, n – discretization period number, T_p – modulation period of the probing LFM signal.

For further processing, a variable $k_r t = f_t$, frequency is introduced for the range coordinate, and the range is written as follows

$$R(x) = \sqrt{R_0^2 + (x_0 - x)^2}\,. \tag{21}$$

Substituting the expressions given in (20) and performing RVP compensation, we obtain the result of the first stage of two-dimensional data formation

$$\dot{S}(f_t, x_n) = \exp\left\{ j\left((4\pi/c)\cdot(f_0 + f_t) \times \right.\right. \\ \left.\left. \times \sqrt{R_0^2 + \left(x_n - x_0 + V_s f_t / k_r\right)^2} \right)\right\}. \tag{22}$$

The next step is to calculate the Fourier transform by azimuth.

$$\dot{S}(f_t, x_n, f_x) = \int_{-\infty}^{\infty} \dot{S}(f_t, x_n) \cdot \exp\{-j 2\pi f_x x_n\}\, dx_n\,, \tag{23}$$

where f_x – frequency by azimuth coordinate. The phase in expression (23) has the following form:

$$\Phi(f_t, x_n, f_x) = -(4\pi/c)\cdot(f_0 + f_t) \times \\ \times \sqrt{R_0^2 + \left(x - x_0 + V_s f_t / k_r\right)^2} - 2\pi f_x x_n. \tag{24}$$

Using the principle of the stationary phase

$$\frac{d\Phi(f_t, x_n, f_x)}{dx_n} = 0\,,$$

we obtain an expression for the azimuth coordinates

$$x_n = -\frac{c R_0 f_x}{\sqrt{4(f_0 + f_t)^2 - c^2 f_x^2}} + x_0 - \frac{V_s f_t}{k_r}. \tag{25}$$

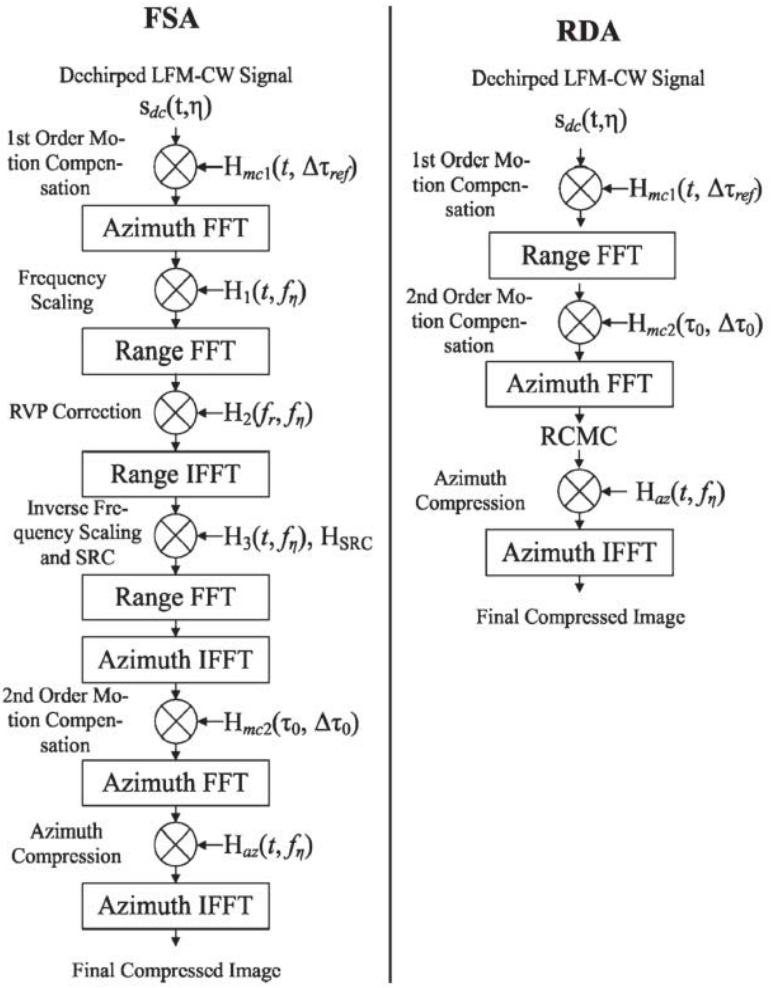
Substituting (25) into (24), we obtain a two-dimensional representation of data in the spectrum for further processing using already known algorithmic operations

$$\dot{S}\left(f_t, f_x\right) = \exp\left\{\begin{array}{c} -\dfrac{4\pi R_0}{c}\sqrt{\left(f_0 + f_t\right)^2 + c^2 f_x^2 / 4} - \\ -2\pi f_x x_0 + \left(2\pi V_s / k_r\right) \cdot f_t f_x \end{array}\right\}. \quad (26)$$

Frequency Scaling Algorithm (FSA or CSA) and Range-Doppler algorithm (RDA)

Complete information on the implementation of the FSA algorithm is presented in [17–19], and the block diagram is shown in Fig. 7, *a*. The RDA algorithm is described in detail in [17, 20, 21] and demonstrated in Fig. 7, *b*. From the analysis of the main operations, it follows that FSA is supplemented with new RDA operations.

In this case, let us consider in detail all the main operations using the frequency scaling method. First of all, it should be noted that the "raw" data is represented as $s(t,\eta)$ and fully corresponds to expression (9), only instead of the variable for "slow" time n, η is used.



a b

Fig. 7. Methods for processing "raw" data in SAR with continuous LFM signals: *a* – frequency scaling method; *b* – Range-Doppler method (taken from [17])

The first operation on the received signals is to compensate for the carrier's own motion, which consists of multiplying the received signals by the function

$$H_{mc1}\left(t,\Delta\tau_{ref}\right)=\exp\left\{-j\left(2\pi f_0\Delta\tau_{ref}+\right.\right.$$
$$\left.\left.+2\pi k_r t\Delta\tau_{ref}-\pi k_r\left(2\eta\Delta\tau_{ref}-\tau_{ref}^2\right)\right)\right\}, \quad (27)$$

where $\Delta\tau_{ref}=2\Delta R_{ref}/c$, $\Delta R_{ref}=R_{actual}-R_{ideal}$, R_{actual} – sloped range from the carrier to the point target, taking into account known deviations of the platform from the ideal trajectory; R_{ideal} – the ideal trajectory of the platform for the same position of the point target.

The next stage of processing according to the FSA and RDA algorithms is the calculation of the Fourier transform by azimuth and frequency scaling of data together with the elimination of Doppler frequency shift, which consists in multiplying the preliminary result by the function

$$H_1\left(t,f_\eta\right)=$$
$$=\exp\left\{-j\left[2\pi f_\eta t+\pi k_r t^2\left(1-D\left(f_\eta,V_s\right)\right)\right]\right\}, \quad (28)$$

where $D\left(f_\eta,V_s\right)=\sqrt{1-\lambda^2 f_\eta^2/\left(4V_s^2\right)}$ – coefficient determining the degree of range migration.

After frequency scaling, a Fourier transform is performed in the range and in the frequency domain, and the constant phase with a quadratic ramp is compensated for in the range, as was done in the ωKA-M algorithm.

Compensation for the excess phase is performed by multiplying the data by the function

$$H_2\left(f_r,f_\eta\right)=\exp\left\{-j\left(\pi f_r^2/\left(k_r D\left(f_\eta,V_s\right)\right)\right)\right\}. \quad (29)$$

After performing the inverse Fourier transform in range, inverse frequency scaling is performed by multiplying by the function

$$H_3\left(t,f_\eta\right)=$$
$$=\exp\left\{-j\pi k_r t^2\left[D^2\left(f_\eta,V_s\right)-D\left(f_\eta,V_s\right)\right]\right\}. \quad (30)$$

At this stage, secondary range correction and phase shift compensation for range are also introduced, as described in detail in [18].

Next, the signal is sequentially transformed into the spectral plane by range, and an inverse Fourier transform is applied by azimuth. In the resulting two-dimensional signal, range migration is secondarily compensated by multiplying the data by the function

$$H_{mc2}\left(\tau_0,\Delta\tau_0\right)=\exp\left\{-j\left(2\pi f_0\Delta\tau_0+\right.\right.$$
$$+2\pi k_r\tau_0\Delta\tau_0-\pi k_r\Delta\tau_0^2+2\pi f_0\Delta\tau_{ref}-.$$
$$\left.\left.-2\pi k_r\tau_{ref}\Delta\tau_{ref}+\pi k_r\Delta\tau_{ref}^2\right)\right\}, \quad (31)$$

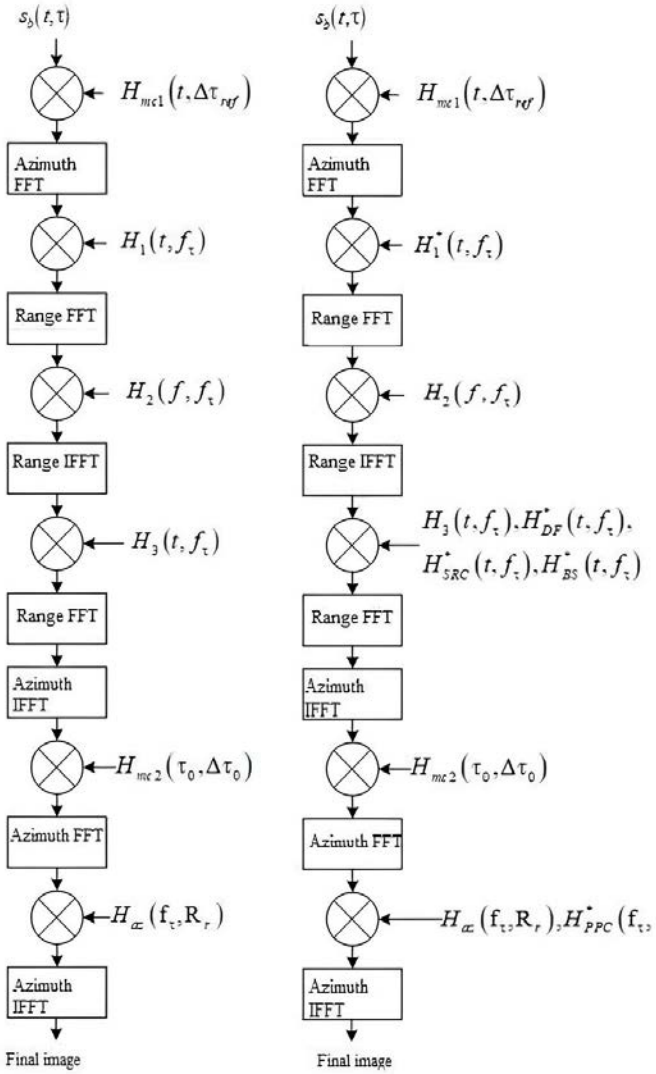
where $\tau_0=2R_0/c$, $\Delta\tau_0=2\Delta R_0/c$, $\tau_{ref}=2R_{ref}/c$, $\Delta\tau_{ref}=2\Delta R_{ref}/c$.

The following operations are known from previous methods: calculation of Fourier transform by azimuth, data compression by azimuth, calculation of inverse Fourier transform by azimuth.

The difference between these operations lies only in their names; their essence remains the same. Azimuth data compression is the same operation as multiplication by a reference function in the ωKA and ωKA-M algorithms.

Modified frequency scaling method (FSA-M)

This algorithm is an extension of the FSA algorithm [22, 23], which is shown in Fig. 8



a b

Fig. 8. Methods for processing "raw" data in SAR with continuous LFM signals: *a* – frequency scaling method; *b* – modified frequency scaling method

If we disregard the forward and inverse Fourier transforms in range and azimuth and the carrier motion compensation operations, the key differences between the algorithms are as follows.

The frequency scaling function in the modified algorithm takes the form, instead of (28),

$$H_1^*\left(t,f_\eta\right)=\exp\left\{-j\pi k_r t^2\left(1-D\left(f_\eta,V_s\right)\right)\right\}. \quad (32)$$

In the modified algorithm, after the inverse Fourier transform, the data is multiplied by three more functions that perform:

1) Doppler shift correction

$$H_{DF}^*\left(t,f_\eta\right)=\exp\left\{-j2\pi f_\eta D\left(f_\eta,V_s\right)t\right\}, \quad (33)$$

2) second-order range compression

$$H_{SRC}^*\left(t,f_\eta\right)=\exp\left\{\begin{array}{c}-j\dfrac{2\pi R_{ref}k_r^2\lambda}{c^2}\dfrac{D^2\left(f_\eta,V_s\right)-1}{D^3\left(f_\eta,V_s\right)}\times\\ \times\left(D\left(f_\eta,V_s\right)t-2R_{ref}/c\right)^2\end{array}\right\}\times$$

$$\times\exp\left\{\begin{array}{c}-j\dfrac{2\pi R_{ref}k_r^3\lambda^3}{c^3}\dfrac{D^2\left(f_\eta,V_s\right)-1}{D^5\left(f_\eta,V_s\right)}\times\\ \times\left(D\left(f_\eta,V_s\right)t-2R_{ref}/c\right)^3\end{array}\right\}, \quad (34)$$

3) group phase shift compensation

$$H_{BS}^*\left(t,f_\eta\right)=$$

$$=\exp\left\{\begin{array}{c}-j\dfrac{4\pi k_r}{c}R_{ref}\left(\dfrac{1}{D(f_\eta,V_s)}-1\right)\times\\ \times\left(D\left(f_\eta,V_s\right)t-2R_{ref}/c\right)\end{array}\right\}. \quad (35)$$

At the final stage, when compressing data by azimuth, multiplication by the phase preservation function is additionally introduced.

$$H_{PPC}^*\left(f_r,f_\eta\right)=\exp\left\{j\frac{4\pi R_{ref}}{c}\frac{f_r}{D\left(f_\eta,V_s\right)}\right\}. \quad (36)$$

The presented modified algorithm is similar to the existing FSA, but has the advantage of taking into account the compensation of carrier motion during emission in signal processing.

The classic FSA was borrowed from the algorithms of pulse SAR, where the emission time is significantly less than the observation time of the reflected signals. In SAR with continuous LFM signal processing, the radiation time is significant and the carrier motion must be taken into account in this case.

Conclusions

The article considers a generalized approach to constructing spatial models that describe the process of measuring the coordinates of radio emission sources. Geometric schemes are proposed that allow formalizing the mutual location of the source and receiver in three-dimensional space. Functional dependencies between measurement parameters and scene configuration are established. It is shown that geometric factors significantly affect the accuracy

of coordinate determination. Particular attention is paid to multiple observation options that reduce ambiguity and increase the reliability of results. The analysis confirms the feasibility of using three-dimensional models in direction finding and navigation tasks. The results obtained can serve as a basis for the development of new signal processing algorithms in radio measurement systems.

The proposed approach opens up opportunities for improving sensor placement and trajectory planning. The research has both theoretical and practical significance. Further work may be directed toward the practical implementation of simulation models and testing their effectiveness in real-world conditions.

References

1. Jancco-Chara, J., Palomino-Quispe, F., Coaquira-Castillo, R. J., Herrera-Levano, J. C., Florez, R. (2024), "Doppler Factor in the Omega-k Algorithm for Pulsed and Continuous Wave Synthetic Aperture Radar Raw Data Processing", *Applied Sciences*, Vol. 14, P. 320. DOI: https://doi.org/10.3390/app14010320
2. Ulaby, F. T., Moore, R. K., Fung, A. K. (1986), *Microwave Remote Sensing: Active and Passive*, Vol. II, ARTECH House, Norwood, Massachusetts, P. 583–595.
3. Curlander, J. C., McDonough, R. N. (1991), *Synthetic Aperture Radar: Systems and Signal Processing*, John Wiley & Sons, New York.
4. Elachi, C. (1988), *Spaceborne Radar Remote Sensing: Applications and Techniques*, IEEE Press, New York.
5. Chara, J. J., Palomino-Quispe, F., Coaquira-Castillo, R. J., Clemente-Arenas, M. (2020), "Omega-k Algorithm Implementation for Linear Frequency Modulated-Continuous Wave SAR Signal Processing", *IEEE INTERCON 2020*, Lima, Peru, P. 1–4. DOI: https://doi.org/10.1109/INTERCON50315.2020.9220195
6. Volosyuk, V., Zhyla, S. (2022), "Statistical Theory of Optimal Functionally Deterministic Signals Processing in Multichannel Aerospace Imaging Radar Systems", *Computation*, Vol. 10, P. 213. DOI: https://doi.org/10.3390/computation10120213
7. Volosyuk, V., Zhyla, S. (2022), "Statistical Theory of Optimal Stochastic Signals Processing in Multichannel Aerospace Imaging Radar Systems", *Computation*, Vol. 10, P. 224. DOI: https://doi.org/10.3390/computation10120224
8. Stringham, C., Long, D. G., Wicks, B., Ramsey, G. (2011), "Digital Receiver Design for an Offset IF LFM-CW SAR", *IEEE RadarCon 2011*, Kansas City, USA, P. 960–964. DOI: 10.1109/RADAR.2011.5960678
9. Aguasca, A., Acevo-Herrera, R., Broquetas, A., Mallorqui, J. J., Fabregas, X. (2013), "ARBRES: Light-Weight CW/FM SAR Sensors for Small UAVs", *Sensors*, Vol. 13, P. 3204–3216. DOI: https://doi.org/10.3390/s130303204
10. Mencia-Oliva, B., Grajal, J., Yeste-Ojeda, O. A., Rubio-Cidre, G., Badolato, A. (2013), "Low-Cost CW-LFM Radar Sensor at 100 GHz", *IEEE Transactions on Microwave Theory and Techniques*, Vol. 61, No. 2, P. 986–998. DOI: https://doi.org/10.1109/TMTT.2012.2235457
11. Cumming, I. G., Neo, Y. L., Wong, F. H. (2003), "Interpretations of the Omega-K Algorithm and Comparisons with Other Algorithms", *IGARSS 2003*, Toulouse, France, P. 1455–1458. DOI: https://doi.org/10.1109/IGARSS.2003.1294142

12. Stolt, R. H. (1978), "Migration by Transform", *Geophysics*, Vol. 43, No. 1, P. 23–48.
13. Chun, J. H., Jacowitz, C. A. (1981), "Fundamentals of Frequency Domain Migration", *Geophysics*, Vol. 46, P. 717–733.
14. Cafforio, C., Prati, C., Rocca, F. (1991), "SAR Data Focusing Using Seismic Migration Techniques", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 2, P. 194–207.
15. Subiza, B., Gimeno-Nieves, E., Lopez-Sanchez, J. M., Fortuny-Guasch, J. (2003), "An Approach to SAR Imaging by Means of Non-Uniform FFTs", *IGARSS 2003*, Toulouse, France, Vol. 6, P. 4089–4091. DOI: https://doi.org/10.1109/IGARSS.2003.1295371
16. Guo, S., Dong, X. (2016), "Modified Omega-K Algorithm for Ground-Based FMCW SAR Imaging", *ICSP 2016*, Chengdu, China, P. 1647–1650. DOI: https://doi.org/10.1109/ICSP.2016.7878107
17. Zaugg, E. C., Long, D. G. (2008), "Theory and Application of Motion Compensation for LFM-CW SAR", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 46, No. 10, P. 2990–2998. DOI: https://doi.org/10.1109/TGRS.2008.921958
18. Mittermayer, J., Moreira, A., Loffeld, O. (1999), "Spotlight SAR Data Processing Using the Frequency Scaling Algorithm", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, No. 5, P. 2198–2214.
19. Mittermayer, J. (2002), "The Frequency Scaling Algorithm and Interferometric Spotlight SAR Processing", *Aerospace Science and Technology*, Vol. 6, No. 2, P. 147–158. DOI: https://doi.org/10.1016/S1270-9638(02)01149-5
20. Guo, Y., Wang, P., Men, Z., Chen, J., Zhou, X., He, T., Cui, L. (2023), "A Modified Range Doppler Algorithm for High-Squint SAR Data Imaging", *Remote Sensing*, Vol. 15, P. 4200. DOI: https://doi.org/10.3390/rs15174200
21. Raney, R. K., Runge, H., Bamler, R., Cumming, I. G., Wong, F. H. (1994), "Precision SAR Processing Using Chirp Scaling", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 32, No. 4, P. 786–799. DOI: https://doi.org/10.1109/36.298008
22. Jiang, Z., Huang, F., Wan, J., Cheng, Z. (2007), "Modified Frequency Scaling Algorithm for FMCW SAR Data Processing", *Chinese Journal of Aeronautics*, Vol. 20, No. 4, P. 339–345. DOI: https://doi.org/10.1016/S1000-9361(07)60053-3
23. Zheng, J., Cheng, J.-Y., Chen, C.-H. (2008), "A Modified Frequency Scaling Algorithm for Missile-Borne SAR Imaging", *IITA 2008*, Shanghai, China, P. 337–341. DOI: https://doi.org/10.1109/IITA.2008.315

Received (Надійшла) 19.08.2025
Accepted for publication (Прийнята до друку) 07.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Kovalchuk Danyil – PhD, State Scientific and Research Institute of Cybersecurity Technologies and Information Protection, Lead Engineer, Kyiv, Ukraine; e-mail: d.kovalchuk@ws.cip.gov.ua; ORCID ID: https://orcid.org/0009-0007-6847-6610

Zhyla Olha – PhD (Physical and Mathematical Sciences), Kharkiv National University of Radio Electronics, Associate Professor at the Department of Higher Mathematics, Kharkiv, Ukraine; e-mail: olha.kuryzheva@nure.ua; ORCID ID: https://orcid.org/0000-0002-6888-8953

Ковальчук Даниїл Іванович – PhD, Державний науково-дослідний інститут технологій кібербезпеки та захисту інформації, провідний інженер науково-дослідного центру, Київ, Україна.

Жила Ольга Володимирівна – кандидат фізико-математичних наук, Харківський національний університет радіоелектроніки, доцент кафедри вищої математики, Харків, Україна.

ГЕОМЕТРІЯ ВИМІРЮВАНЬ, МОДЕЛІ СИГНАЛІВ І АЛГОРИТМИ ВІДНОВЛЕННЯ РАДІОЗОБРАЖЕНЬ В РАДАРАХ ІЗ СИНТЕЗОВАНОЮ АПЕРТУРОЮ, ЩО ВИКОРИСТОВУЮТЬ БЕЗПЕРЕРВНІ ЛЧМ-СИГНАЛИ

У статті проаналізовано методи формування радіолокаційних зображень поверхні, що отримані за допомогою радара із синтезованою апертурою (РСА) з безперервним зондувальним ЛЧМ-сигналом. Становить інтерес визначення основних алгоритмічних операцій, що виконуються над "сирими" даними після їх реєстрації в приймачах. У роботі розглянуто геометрію вимірювань, зондувальний сигнал і особливості формування "сирих" даних, що визначатимуть подальше оброблення. Для порівняння якості роботи різних алгоритмів запропоновано імітаційну модель формування радіолокаційних зображень в РСА з обробленням безперервних ЛЧМ-сигналів. **Мета дослідження** – створити універсальну геометричну основу для побудови ефективних вимірювальних схем і алгоритмів оброблення сигналів у радіотехнічних системах. **Завдання роботи** передбачають: 1) формалізацію задачі визначення координат за результатами пеленгаційних вимірювань; 2) побудову математичної моделі взаємного розташування об’єктів у тривимірному просторі; 3) визначення впливу геометричних факторів на точність вимірювання; 4) аналіз варіантів одноразового й багаторазового спостереження. **Досягнуті результати** дають змогу встановити аналітичний зв’язок між параметрами вимірювання та конфігурацією просторової сцени, що забезпечує підвищення точності координатного визначення. **Перспективи застосування**: результати можуть бути використані для підвищення ефективності навігаційних, розвідувальних і моніторингових систем, а також у задачах супроводу рухомих об’єктів і побудови систем ситуаційної обізнаності.

Ключові слова: геометрія вимірювань; пеленгація; координатне визначення; тривимірна модель; радіовимірювальна система.

Bibliographic descriptions / Бібліографічні описи

Kovalchuk, D., Zhyla, O. (2025), "Measurement geometry, signal models and agorithms for radio image recovery in synthesized aperture radars using continuous LFM signals", *Management Information Systems and Devises*, No. 4 (187), P. 299–314. DOI: https://doi.org/10.30837/0135-1710.2025.187.299

Ковальчук Д. А., Жила О. В. Геометрія вимірювань, моделі сигналів і алгоритми відновлення радіозображень в радарах із синтезованою апертурою, що використовують безперервні ЛЧМ-сигнали. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 299–314. DOI: https://doi.org/10.30837/0135-1710.2025.187.299

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 528.88 + 515.127.1 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.315

R. Pashchenko, M. Mariushko

SELECTION OF "WINDOW" SIZES WHEN CALCULATING FRACTAL DIMENSIONS OF AGRICULTURAL LAND SPACE IMAGES

Abstract. The quality of crop condition assessment can be influenced by the parameters of the method used to determine fractal dimensions, in particular the size of the "window". **The subject** of the study is to assess the impact of the size of the "window" on the values of fractal dimensions of satellite images. **The object** of the study is satellite images of agricultural crops at different stages of vegetation taken by the Sentinel-2 satellite. **The aim** is to assess the impact of the "window" size on the values of fractal dimensions of satellite images of agricultural land and to select the "window" size for studying the condition of agricultural crops. The following **results** were obtained. The influence of the size of the "window" involved in the construction of the field of fractal dimensions on the minimum, maximum, and average values of fractal dimensions in satellite images of agricultural land was investigated. It was found that throughout the entire growing season for a field sown with corn, as the size of the "window" increases, the minimum fractal dimensions increase, while the average and maximum fractal dimensions decrease. At the same time, in the first half of the growing season, the differences in minimum fractal dimensions with small "window" sizes (up to 24 × 24 pixels) are the largest compared to the differences in maximum and average fractal dimensions. **Conclusions.** The conducted research allows us to develop recommendations for choosing the size of the "window" and the type of fractal dimension for analyzing satellite images of agricultural land. Thus, to assess the condition of agricultural crops, it is advisable to use the minimum fractal dimensions found in images that have the greatest differences at different stages of vegetation. To ensure a compromise between the speed of image processing and the quality of crop condition assessment, it is advisable to choose small "sliding window" sizes (up to 16×16 pixels).

Keywords: crop condition assessment; satellite images; fractal dimension; selection of "window" sizes.

Introduction

Remote sensing (RS) data are currently widely used to assess the condition of agricultural land [1]. The dynamics of agricultural work require agricultural producers to quickly assess the results and quality of their work. These tasks are solved by the widespread use of space-based RSE tools, which allow for regular monitoring of agricultural land [2]. The speed of obtaining RSI data from large areas (with a frequency of 1 to 8 days) and the clarity of the results during their processing (with spatial resolution from 250 to 10 meters) allows specialists to quickly respond to changes in the condition of agricultural land [3]. Recently, there has been a trend towards increasing access to remote sensing data. Users can obtain satellite images of the areas under study from the Terra, Aqua, Landsat 8, and Sentinel-2 remote sensing satellites free of charge (they are freely available) via the Internet. For example, there is an up-to-date and constantly updated database of satellite images from the Sentinel-2 satellite [4], which allows them to be used for various tasks, including assessing the condition of agricultural land.

Many methods for assessing the condition of agricultural land and crops are based on the use of the normalized difference vegetation index (NDVI). The paper [5] presents the results of studies of changes in the NDVI index of multi-temporal satellite images of agricultural land. It should be

noted that data from the near-infrared and red channels of Earth observation satellites are used to calculate the NDVI index, which complicates the acquisition and processing of the source image.

To increase the informativeness of satellite images when monitoring various surfaces, in particular the Earth's surface, methods of fractal image analysis are used [6, 7]. The possibility of applying fractal analysis of Sentinel-2 satellite images to assess the condition of agricultural crops at different stages of vegetation is shown in [8, 9]. In [8], fields sown with corn are analyzed, and in [9], fields sown with buckwheat, wheat, sunflower, and barley are analyzed. In this case, space images in one wavelength range are used. However, these works do not consider the influence of the "window" size on the values of fractal dimensions in satellite images of agricultural land. In this regard, it is advisable to consider the possibility of selecting the size of the "window" to be used when calculating the fractal dimensions of satellite images of agricultural land.

The aim of the article: to evaluate the influence of the "window" size on the values of fractal dimensions of satellite images of agricultural land and the choice of "window" size for studying the state of agricultural crops.

Characteristics of agricultural land satellite images

To study the effect of the "window" size on the fractal dimensions of satellite images of agricultural land, we will use Sentinel-2 satellite images, which are freely available on the Internet [4]. During the fractal analysis, satellite images in the near-infrared range (Sentinel-2 satellite channel b8) will be used. Channel b8, with a wavelength of 832 nm, allows us to obtain satellite images that capture the reflection of solar radiation from vegetation.

For further assessment of the condition of agricultural land and crops, a database of satellite images of the land of the Vilkhuvatskaya village council of the Chutivsky district of the Poltava region was created, most of which is cultivated by the private agricultural enterprise "Druzhba" [8]. The satellite images were obtained throughout 2018, and the images have a spatial resolution of 10 meters. A field sown with corn was selected for further study.

Fig. 1 shows elements of images of a field sown with corn, cut out from the main (large) satellite images, as of:

04.06.2018 (a), 14.06.2018 (b); 16.06.2018 (c), 21.06.2018 (d); 26.06.2018 (e); 29.06.2018 (f); 29.07.2018 (g); 05.08.2018 (h); 08.08.2018 (i); 10.08.2018 (j); 18.08.2018 (k); 23.08.2018 (l); 25.08.2018 (m).

Visual analysis of the elements of the satellite images in Fig. 1 showed that they differ in the degree of gray color (gray gradation), but the differences are not significant. Thus, in the initial phases of vegetation (early June to June 26, 2018), darker shades of gray are observed, as shown in Fig. 1, *a–d*. At the end of June (June 29, 2018), in July (July 29, 2018), and at the beginning of August (until August 18, 2018), lighter shades of gray gradations are observed (see Fig. 1, *e–j*), and at the end of August (in the final stages of vegetation), the shades of gray become darker again (see Fig. 1, *k–m*).

It should also be noted that the satellite images reveal some features of the structure of the field under study, which is planted with corn. Thus, all satellite images show the results of field

cultivation in the form of stripes with slight differences in gray gradations. This slight layering spreads from the upper left corners to the lower right corners of all images.

In addition, a small light area is observed in the lower left corner of the satellite images, which is especially evident in the final stages of vegetation (see Fig. 1, *i*–*m*). Most likely, this anomalous area is due to the negative impact of weather conditions.

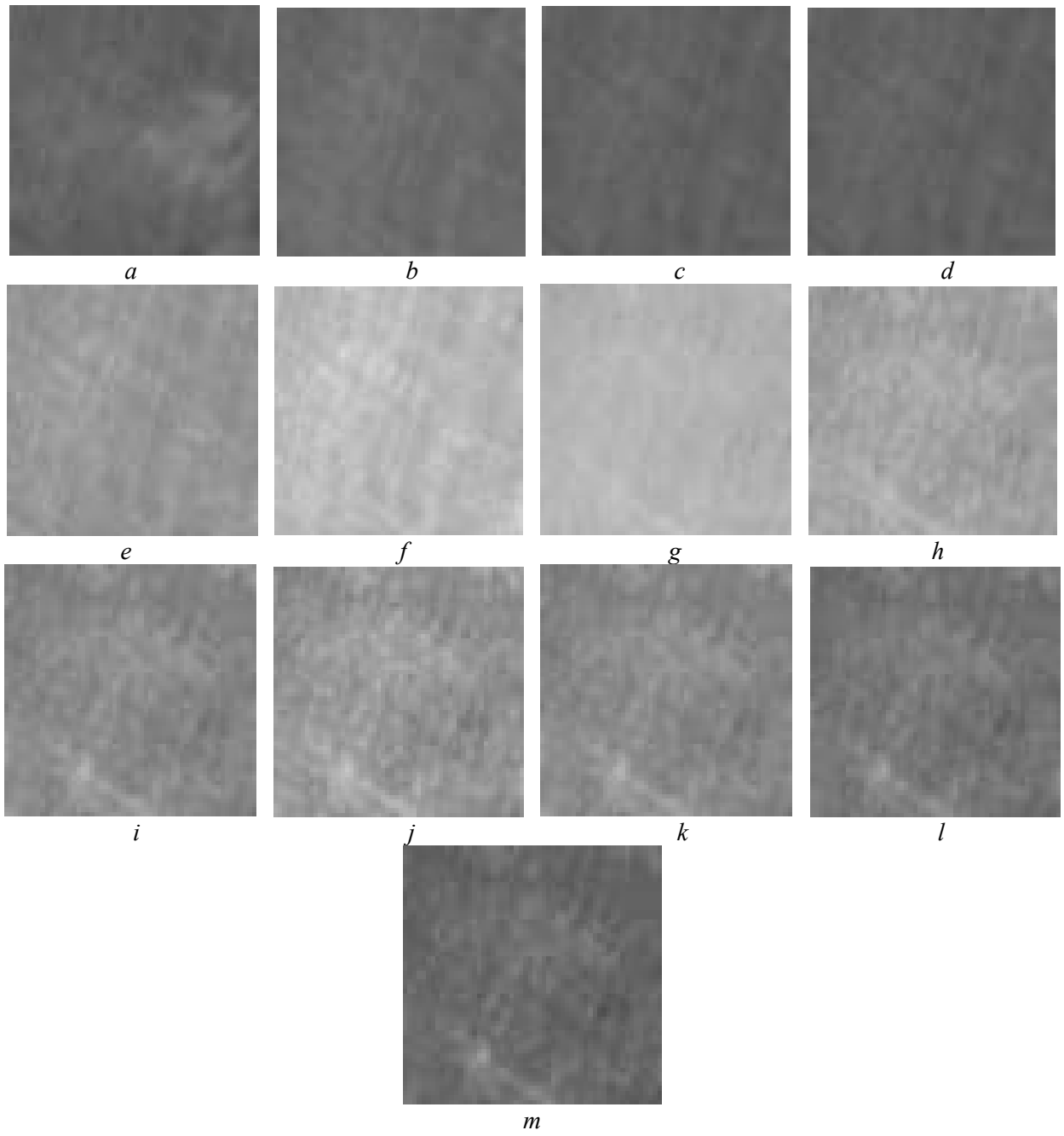


Fig. 1. Elements of space images (channel b8) of corn fields as of: June 4, 2018 (*a*), June 14, 2018 (*b*); 16.06.2018 (*c*), 21.06.2018 (*d*); 26.06.2018 (*e*); 29.06.2018 (*f*); 29.07.2018 (*g*); 05.08.2018 (*h*); 08.08.2018 (*i*); 10.08.2018 (*j*); 18.08.2018 (*k*); 23.08.2018 (*l*); 25.08.2018 (*m*)

The satellite images also show that the presence of clouds causes a change in gray gradations (a light shade appears in Fig. 1, a), and in the presence of cloud shadows, a dark shade appears.

Thus, using only visual analysis of satellite images, it is only possible to qualitatively assess changes in the condition of agricultural crops, evaluate the uniformity of crops, and the progress of field work. However, it is practically impossible to quantitatively assess the condition of agricultural crops at different stages of vegetation based on the results of visual analysis.

Let us process the satellite images of a field sown with corn, discussed above, using fractal analysis and consider the influence of the "window" size on the values of fractal dimensions.

Selecting the size of the "window" when calculating the fractal dimensions of agricultural land satellite images

The information carrier in the Sentinel-2 satellite image (channel b8) is light waves in the near-infrared range, formed by the reflection of solar radiation depending on the characteristics of the underlying surface. The light wave carries information about the object due to the fact that its various parameters change depending on the nature of the points on the underlying surface (relief). The result is a heterogeneous matrix of values of the intensity of solar radiation reflection from the Earth's surface. The size of the matrix is equal to the size of the satellite image.

During the fractal analysis of space (digital) images, the fractal dimensions are first calculated (the order of their determination is described in [10]), and then the field of fractal dimensions is constructed (the process of its construction is described in [7]).

There are many methods for calculating fractal dimensions, but in practice, the covering method [10] is most often used for analyzing digital images. This method provides the calculation of the Hausdorff–Bezikovich dimension, which is described by the expression:

$$D = \lim_{\varepsilon \to 0} \frac{\log N(\varepsilon)}{\log(1/\varepsilon)}, \qquad (1)$$

where ε – the length of the cube's side (the maximum length is equal to the size of the "window") covering the three-dimensional representation of the digital image; $N(\varepsilon)$ – the number of cubes covering the three-dimensional representation of the image.

As can be seen from expression (1), it is not possible to directly calculate the fractal dimension of a space image using this formula. The fractal dimension can be determined as follows.

At the beginning of the application of the covering method, a cube is created with a side length equal to ε pixels, and this cube covers the three-dimensional representation of the space image (set I). Next, the number of cubes with a side length of ε pixels covering the entire space image (set I) is calculated $N(\varepsilon)$. Then, $N(\varepsilon)$ is determined for several other values of the cube side ε. Such values of the cube sides can be, for example, $\varepsilon_2 = \varepsilon_1/2$, $\varepsilon_3 = \varepsilon_1/4$, $\varepsilon_4 = \varepsilon_1/8$, etc. In this way, the set of values $N(\varepsilon)$ is determined. Depending on the structure of the space image, the number of values may vary. At the final stage, a graph of the dependence of log $N(\varepsilon)$ on log $1/\varepsilon$ is constructed and its linear approximation is performed. The least squares method is most often used for approximation, which allows the calculation process to be automated.

After constructing the approximated straight line, the tangent of its slope angle is determined. This value will be the fractal dimension D of the space image.

To construct the field of fractal dimensions, the space image is scanned using a "window" of size $n \times m$ pixels, which is moved in steps of s (the "window" is "sliding" if $s = 1$, and "jumping" if $s > 1$) [7]. At each step of the "window" movement, the numerical value of the fractal dimension is calculated and recorded in the matrix $\boldsymbol{D}$. This matrix is called the "field of fractal dimensions". In this work, fractal dimension fields are constructed for elements of space images with dimensions of 56×56 pixels, and the dimensions of the "sliding window" $n \times m$ vary.

In [9], the state of agricultural crops was assessed using the minimum, maximum, and average values of fractal dimensions present in the satellite image, and it was determined that they are the most informative, but no comparative analysis was performed.

Let us consider the influence of the “window” size on the value of these fractal dimensions in satellite images of a field sown with corn. The numerical values of the minimum, maximum, and average values of fractal dimensions for different “window” sizes are given in Table 1.

Table 1. *Minimum, maximum, and average values of fractal dimensions of space images of a field sown with corn at different "window" sizes*

Date	Size of the "window" in pixels								
	4 × 4			8 × 8			16 × 16		
	D_{min}	D_{max}	D_{av}	D_{min}	D_{max}	D_{av}	D_{min}	D_{max}	D_{av}
04.06.18	2,882	2,991	2,964	2,884	2,984	2,961	2,906	2,976	2,954
14.06.18	2,923	2,993	2,970	2,939	2,989	2,968	2,951	2,981	2,967
16.06.18	2,927	2,993	2,968	2,942	2,985	2,966	2,947	2,980	2,966
21.06.18	2,927	2,995	2,971	2,935	2,987	2,969	2,940	2,982	2,967
26.06.18	2,936	2,993	2,974	2,944	2,987	2,973	2,947	2,983	2,970
29.06.18	2,932	2,994	2,972	2,938	2,990	2,971	2,942	2,984	2,970
29.07.18	2,950	2,995	2,981	2,960	2,990	2,979	2,967	2,987	2,979
05.08.18	2,913	2,992	2,967	2,931	2,994	2,965	2,940	2,976	2,965
08.08.18	2,904	2,991	2,964	2,925	2,985	2,962	2,936	2,978	2,962
10.08.18	2,875	2,985	2,948	2,893	2,972	2,946	2,903	2,967	2,944
18.08.18	2,860	2,991	2,953	2,882	2,978	2,949	2,900	2,967	2,946
23.08.18	2,844	2,986	2,948	2,860	2,976	2,943	2,881	2,964	2,937
25.08.18	2,814	2,986	2,944	2,837	2,978	2,938	2,863	2,961	2,929
	24 × 24			32 × 32			48 × 48		
04.06.18	2,920	2,972	2,947	2,932	2,948	2,940	2,942	2,943	2,943
14.06.18	2,959	2,975	2,966	2,960	2,973	2,965	2,964	2,966	2,965
16.06.18	2,958	2,977	2,966	2,959	2,972	2,964	2,964	2,965	2,965
21.06.18	2,956	2,979	2,966	2,957	2,975	2,963	2,961	2,962	2,962
26.06.18	2,958	2,980	2,969	2,958	2,978	2,966	2,966	2,968	2,967
29.06.18	2,956	2,983	2,969	2,962	2,977	2,967	2,966	2,968	2,967
29.07.18	2,966	2,986	2,980	2,969	2,984	2,980	2,974	2,977	2,975
05.08.18	2,947	2,976	2,967	2,951	2,975	2,966	2,955	2,957	2,956
08.08.18	2,944	2,973	2,963	2,948	2,970	2,962	2,952	2,955	2,953
10.08.18	2,913	2,959	2,942	2,920	2,953	2,939	2,927	2,930	2,928
18.08.18	2,911	2,961	2,946	2,919	2,959	2,944	2,926	2,929	2,927
23.08.18	2,894	2,957	2,935	2,902	2,948	2,931	2,911	2,914	2,912
25.08.18	2,878	2,954	2,926	2,888	2,941	2,922	2,898	2,901	2,899

In [8], it was shown that throughout the entire vegetation period, the average values of fractal dimensions first increase and then decrease to minimum values. The same pattern of change is observed for NDVI indices. Table 1 shows that the minimum and maximum fractal dimensions undergo the same changes.

As can be seen from the data in Table 1, throughout the entire vegetation period for a field sown with corn, as the size of the "window" increases, the minimum fractal dimensions increase, while the average and maximum fractal dimensions decrease. At the same time, in the initial and final phases of vegetation, the increase (decrease) in fractal dimensions is greater than in the middle phase of vegetation.

For example, the minimum fractal dimensions increase by an average of 0.010 until June 16, 2018, by 0.005 from June 16, 2018, to July 29, 2018, and by 0.020 after July 29, 2018. For average and maximum fractal dimensions, approximately the same decrease in values is observed for the specified dates.

Table 1 also shows that the maximum fractal dimensions for small "window" sizes remain virtually unchanged (change in the third decimal place for "window" of 4 × 4 and 8 × 8 pixels) throughout the entire vegetation period, for larger "window" sizes, differences in maximum fractal dimensions are observed in the second decimal place. The opposite trend is observed for minimum fractal dimensions. For small window sizes, the values of the minimum fractal dimensions change in the second decimal place throughout the entire vegetation period, and for large window sizes, they change in the third decimal place.

Changes in average fractal dimensions depend to a lesser extent on the size of the "window" throughout the entire vegetation period, and differences for all "window" sizes are manifested in the second decimal place. At the same time, for small sizes of space images, for example, such as 56 × 56 pixels, it is advisable to select a "window" with a size of 4 × 4 pixels and a movement step equal to one, i.e., to select a "sliding window". Choosing a "window" of this size also allows you to more confidently identify anomalous areas in the images.

Let's consider how much the minimum, maximum, and average fractal dimensions change when the "window" size increases. Fig. 2 shows graphs of changes in these fractal dimensions between two dates: between 04.06.2018 and 29.07.2018 (*a*), between 29.07.2018 and 25.08.2018 (*b*), i.e., in the first and second half of the growing season.

In Fig. 2, the dimensions of the "window" are plotted on the x-axis, so $w = 4 = 4 \times 4$, $w = 8 = 8 \times 8$, $w = 16 = 16 \times 16$, etc., and the y-axis shows the difference between the minimum (dashed line), maximum (dotted line), and average (solid line) fractal dimensions (ΔD).

As can be seen in Fig. 2, *a*, in the first half of the growing season, the differences between the minimum fractal dimensions (dashed line) are the largest compared to the differences between the maximum (dotted line) and average (solid line) fractal dimensions. With larger "window" sizes, these differences in fractal dimensions do not differ, i.e., they have the same value.

It should be noted that as the "window" size increases, the differences in minimum fractal dimensions decrease, while the differences in maximum and average dimensions increase.

Fig. 2, *b* shows that in the second half of the growing season, the nature of changes in the minimum (dashed line), maximum (dotted line), and average (solid line) fractal dimensions is practically the same as in the first half of the growing season.

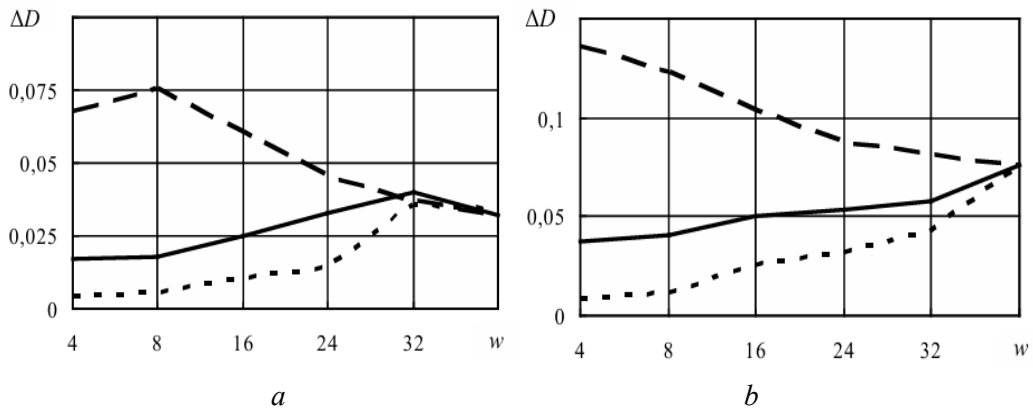


Fig. 2. Dependence of the differences between the minimum, maximum, and average fractal dimensions in a space image of a field sown with kukuaudza on the size of the "window": between June 4, 2018, and July 29, 2018 (*a*), between July 29, 2018, and August 25, 2018 (*b*)

A slight difference is that these differences in fractal dimensions become the same at a "window" size of 48 × 48 pixels.

Thus, the studies conducted allow us to develop recommendations for selecting the size of the "window" and the type of fractal dimension for analyzing satellite images of agricultural land. So, to assess the condition of agricultural crops, it is advisable to use the minimum fractal dimensions found in images that have the greatest differences at different stages of vegetation. It is advisable to select window sizes up to 24 × 24 pixels.

It should also be noted that the use of large windows leads to an increase in the calculation time of fractal dimensions within the window, which in turn leads to an increase in the time required to construct the fractal dimension field. Therefore, to ensure a compromise between the speed of image processing and the quality of agricultural land assessment, it is advisable to choose small "sliding window" sizes (up to 16×16 pixels).

Conclusions

The dynamics of agricultural work requires agricultural producers to quickly assess their results and the quality of their work. These tasks are solved through the widespread use of RS space tools, which allow for regular monitoring of agricultural land.

To increase the informativeness of satellite images during the monitoring of various surfaces, in particular the Earth's surface, methods of fractal image analysis are used, which involve the construction of their fractal dimension fields. In this case, satellite images in a single wavelength range are used.

The influence of the size of the "window" involved in the construction of the field of fractal dimensions on the minimum, maximum, and average values of fractal dimensions in satellite

images of agricultural land has been studied. It has been established that throughout the entire vegetation period for a field sown with corn, as the size of the "window" increases, the minimum fractal dimensions increase, while the average and maximum fractal dimensions decrease. At the same time, in the first half of the growing season, the differences in minimum fractal dimensions with small "window" sizes (up to 24 × 24 pixels) are the largest compared to the differences in maximum and average fractal dimensions.

The studies conducted allow us to develop recommendations for selecting the size of the "window" and the type of fractal dimension for analyzing satellite images of agricultural land. Thus, to assess the condition of agricultural crops, it is advisable to use the minimum fractal dimensions found in images that have the greatest differences at different stages of vegetation. To ensure a compromise between the speed of image processing and the quality of crop condition assessment, it is advisable to choose small "sliding window" sizes (up to 16×16 pixels).

In further research, it is advisable to evaluate the possibility of using fractal analysis to determine the size of field areas where agricultural work is carried out using "windows" of different sizes.

References

1. Bégué, A., Arvor, D., Bellon, B., Betbeder, J. et al. (2018), "Remote Sensing and Cropping Practices: A Review", *Remote Sensing*, Vol. 10, Iss. 1, P. 1–32. DOI: https://doi.org/10.3390/rs10010099
2. Tararik, O. G., Syrotenko, O. V., Ilienko, T. V., Kuchma, T. L. (2019), "Agroecological Satellite Monitoring", *Kyiv: Agrarian Science*, 204 p.
3. "Drones and satellites: monitoring crop conditions throughout the season" [Electronic resource]. – Access mode: https://www.smartfarming.ua/drony-i-suputnyky-monitorynh-stanu-posiviv-vprodovzh-sezonu/ – 03.02.2025.
4. "Copernicus Europe's eyes on Earth, Sentinel-2. Copernicus Europe's eyes on Earth" [Electronic resource]. – Access mode: https://www.copernicus.eu/en/about-copernicus/infrastructure/discover-our-satellites – 04.02.2025.
5. Maryushko, M. V., Pashchenko, R. E., Koblyuk, N. S. (2019), "Monitoring of agricultural crops using SENTINEL-2 satellite images", *Radioelectronic and Computer Systems*, No. 1 (89), P. 99–108. DOI: https://doi.org/10.32620/reks.2019.1.11
6. Feder, J. (1988), "Fractals", *New York: Springer US*, 263 p. DOI: https://doi.org/10.1007/978-1-4899-2124-6
7. Dolya, G. N., Ivanov, V. K., Kuchuk, G. A., Pashchenko, R. E. et al. (2006), "Fractal analysis of processes, structures, and signals" / Ed. by R.E. Pashchenko, *Kharkiv: NEO Ekoperspektiva*, 348 p.
8. Maryushko, M. V., Pashchenko, R. E. (2020), "Fractal analysis of SENTINEL-2 space images for monitoring agricultural crops", *Radioelectronic and Computer Systems*, No. 4 (96), P. 34–47. DOI: https://doi.org/10.32620/reks.2020.4.03
9. Pashchenko, R. E., Maryushko, M. V. (2023), "Assessment of the condition of various agricultural crops using fractal analysis", *Modern Information Systems*, Vol. 7, No. 3, P. 81–88. DOI: https://doi.org/10.20998/2522-9052.2023.3.12
10. Crownover, R. M. (1995), "Introduction to Fractals and Chaos", *London: Jones and Bartlett Publishers, Inc*., 352 p.

Received (Надійшла) 19.08.2025
Accepted for publication (Прийнята до друку) 07.11.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Pashchenko Ruslan – Doctor of Sciences (Engineering), Professor, O. Ya. Usikov Institute for Radiophysics and Electronics of the National Academy of Sciences of Ukraine, Senior research scientist at the Department Remote Sensing of the Earth, Kharkiv, Ukraine; e-mail: r.paschenko@i.ua; ORCID ID: https://orcid.org/0000-0001-6218-0324

Mariushko Maksim – National Aerospace University "Kharkiv Aviation Institute", Teaching Assistant at the Department of Geo-information Technologies and Space Monitoring of the Earth; Kharkiv, Ukraine; e-mail: max.maryushko@gmail.com; ORCID ID: https://orcid.org/0000-0003-3743-8535

Пащенко Руслан Едуардович – доктор технічних наук, професор, Інститут радіофізики та електроніки ім. О. Я. Усикова НАН України, старший науковий співробітник відділу дистанційного зондування Землі, Харків, Україна.

Марюшко Максим В'ячеславович – Національний аерокосмічний університет "Харківський авіаційний інститут", асистент кафедри геоінформаційних технологій та космічного моніторингу Землі, Харків, Україна.

ВИБІР РОЗМІРІВ "ВІКНА" В ПРОЦЕСІ РОЗРАХУНКУ ФРАКТАЛЬНИХ РОЗМІРНОСТЕЙ КОСМІЧНИХ ЗНІМКІВ СІЛЬСЬКОГОСПОДАРСЬКИХ ЗЕМЕЛЬ

На якість оцінювання стану сільськогосподарських культур можуть впливати параметри методу визначення фрактальних розмірностей, зокрема розмірів "вікна". **Предмет дослідження** – оцінювання впливу розмірів "вікна" на величини фрактальних розмірностей космічних знімків. **Об'єктом дослідження** є космічні знімки супутника Sentinel-2 сільськогосподарських культур на різних фазах вегетації. **Мета роботи –** оцінити вплив розмірів "вікна" на величини фрактальних розмірностей космічних знімків сільськогосподарських земель і вибору розмірів "вікна" для аналізу стану сільськогосподарських культур. **Досягнуті результати**. Досліджено вплив розмірів "вікна", яке бере участь у побудові поля фрактальних розмірностей, на величини мінімальних, максимальних і середніх значень фрактальних розмірностей, що є на космічних знімках сільськогосподарських земель. Установлено, що впродовж усього процесу вегетації для поля, засіяного кукурудзою, внаслідок збільшення розмірів "вікна" мінімальні фрактальні розмірності збільшуються, а середні й максимальні – зменшуються. Крім того, у першій половині періоду вегетації різниці мінімальних фрактальних розмірностей за незначних розмірів "вікна" (до 24 × 24 пікселів) найбільші, якщо порівнювати з різницями максимальних і середніх фрактальних розмірностей. **Висновки**. Результати дослідження дають змогу розробити рекомендації щодо вибору розмірів "вікна" й типу фрактальної розмірності для аналізу космічних знімків сільськогосподарських земель. Так, для оцінювання стану сільськогосподарських культур доцільно використовувати мінімальні фрактальні розмірності, що є на знімках, які мають найбільші розбіжності на різних етапах вегетації. Для забезпечення компромісу між швидкістю оброблення знімків та якістю оцінювання стану сільськогосподарських культур доречно обирати невеликі розміри "ковзного вікна" (до 16×16 пікселів).

Ключові слова: оцінювання стану сільськогосподарських культур; космічні знімки; фрактальна розмірність; вибір розмірів "вікна".

Bibliographic descriptions / Бібліографічні описи

Pashchenko, R., Mariushko, M. (2025), "Selection of "window" sizes when calculating fractal dimensions of agricultural land space images", *Management Information Systems and Devises*, No. 4 (187), P. 315–323. DOI: https://doi.org/10.30837/0135-1710.2025.187.315

Пащенко Р. Е., Марюшко М. В. Вибір розмірів "вікна" в процесі розрахунку фрактальних розмірностей космічних знімків сільськогосподарських земель. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 315–323. DOI: https://doi.org/10.30837/0135-1710.2025.187.315

Автоматизовані системи управління та прилади автоматики. 2025. № 4 (187)

UDC 004. 6 **DOI:** https://doi.org/10.30837/0135-1710.2025.187.324

V. Sitnikov, V. Liashchenko

METHODS FOR EFFICIENT BIG DATA STORAGE AND PROCESSING IN DISINFORMATION DETECTION TASKS

The subject of the study is methods and tools that facilitate highly productive, scalable, and reliable data analysis in cloud environments for real-time disinformation detection. The purpose of the article is to investigate and evaluate methods for improving the efficiency of storing and processing large volumes of text, multimedia information, and social network data in cloud infrastructures using real-time disinformation detection. **Objectives:** to evaluate distributed storage architectures and columnar data formats for their efficient organization; to evaluate compression strategies and caching mechanisms to reduce I/O overhead; to analyze scalable processing frameworks for streaming and batch workloads; Measure system performance, scalability, and cost-effectiveness in real-time disinformation detection scenarios. **Methods:** Comparative analysis of storage formats (Parquet, ORC, Avro, JSON), compression algorithms (Snappy, Zstandard), and distributed processing frameworks (Apache Spark, Apache Flink); Performance evaluation involves measuring throughput, analyzing latency, and evaluating costs using cloud infrastructure with multi-level storage and stream data pipelines. **Results achieved.** The impact of distributed storage architectures, columnar data formats, compression strategies, caching mechanisms, and scalable processing frameworks on system performance, scalability, and cost-effectiveness has been evaluated. These approaches demonstrated significant improvements in throughput and reliability in large-scale streaming and batch processing scenarios, especially when detecting misinformation in real time. The results proved that the system can quickly adapt to data load peaks, maintain high detection accuracy, and reduce operating costs. **Conclusions**. Improving the efficiency of big data storage and processing in cloud platforms comes from integrating columnar formats with compression and pushdown capabilities, combined with stream-oriented distributed computing. A layered architecture covering data ingestion, streaming processing, distributed storage, and analytics reduces I/O costs and increases end-to-end throughput for real-time scenarios. The evaluation demonstrates a throughput improvement of approximately one-third compared to baseline systems, making the approach well-suited for latency-sensitive workloads such as disinformation detection. Format-aware optimizations, low-latency streaming, and adaptive capacity management are key drivers of performance in modern cloud data platforms.

Keywords: big data; cloud computing; distributed systems; data analysis; storage optimization; machine learning; scalability; disinformation; fake news.

Introduction

The explosive growth in digital information volumes has created a serious problem for modern information systems in cloud environments. This problem stems from the four Vs of big data – volume, velocity, variety, and veracity [1] – which require new architectural solutions that go beyond traditional database systems. At the same time, cloud computing offers a scalable, elastic infrastructure [1–3], but its principles of resource pooling often contradict the resource sharing model inherent in many big data frameworks. Modern cloud environments must adapt to the continuous influx of heterogeneous data – from structured records to social media and multimedia streams – which requires real-time analysis [4, 5] and high-performance processing. The context of disinformation detection exacerbates these requirements: vast amounts of text and multimedia data must be quickly processed, analyzed, and classified to detect and mitigate misleading

narratives – a task that goes far beyond the capabilities of outdated monolithic processing systems. Moreover, the interplay between data complexity, real-time requirements, architectural mismatches, and threats such as disinformation, as well as privacy, legal, and geopolitical issues, makes the development of adaptive, cost-effective, and elastic data platforms more urgent than ever [1, 3, 6].

Analysis of recent studies. Current research in the field of big data cloud analytics emphasizes the importance of integrating scalable data storage systems and efficient processing pipelines. The studies emphasize the need to align big data architectures with cloud models such as distributed storage, in-memory computing, and streaming analytics [4–6]. Significant attention is paid to optimizing data storage formats. Ivanov and Pergolesi [7] demonstrated the advantages of ORC and Parquet columnar formats for SQL queries in Hadoop environments, while Zeng et al. [8] conducted an empirical evaluation of various columnar formats, finding significant differences in performance depending on the type of workload. Nelluri and Saldanha [9] proposed a strategy for choosing between ORC, Parquet, Avro, and Iceberg formats based on specific application requirements. In the context of streaming data processing, Sedghani et al. [10] evaluated distributed streaming frameworks for IoT applications in smart cities, identifying critical performance factors for real-time systems. However, research on data lifecycle management in the cloud remains limited: Sami et al. [6] considered methodologies for data storage and placement in the Cloud-Big Data ecosystem, but did not cover dynamic optimization of multi-level storage. Regarding the detection of misinformation, Pervaiz et al. [11] and Choraś et al. [12] explored the use of artificial intelligence and machine learning to detect fake news, while Ahmad et al. [13] proposed ensemble machine learning methods for this task. Liu and Wu [14] developed early detection methods through classification of propagation paths, and Abbas et al. [15] conducted a comprehensive review of deep learning algorithms for detecting fake news. Yang and Yao [16] summarized deep learning theories and models for this problem. However, the existing literature often considers these strategies separately, without a unified framework that integrates storage optimization, compression, caching, and adaptive provisioning of computational resources tailored to real-time misinformation detection. Although general reviews describe the capabilities of cloud infrastructure [3, 4] for big data tasks, they typically do not focus on the specific operational requirements and time sensitivity of disinformation analysis. Furthermore, the integration of optimized storage formats [7–9] with stream processing [10] for specific disinformation detection tasks [11–16] has not been sufficiently explored. This knowledge gap limits the ability to develop truly sensitive and cost-effective systems for this high-stakes application.

Research objective. To investigate and evaluate methods for improving the efficiency of storing and processing large volumes of text, multimedia, and social network data in cloud infrastructures, using real-time disinformation detection as an application use case. To achieve this goal, a comprehensive approach is proposed that combines distributed storage models, columnar formats, compression mechanisms, caching, and scalable processing frameworks to improve the performance, reliability, and cost-effectiveness of cloud-based big data analytics [11, 12].

Main part

Cloud environments impose specific characteristics on the storage and processing of big data that differ significantly from traditional on-premises solutions. These characteristics determine architectural decisions and affect the overall efficiency of the system. A key advantage of cloud platforms is the ability to dynamically scale resources according to the current load. Unlike the fixed infrastructure of traditional data centers, cloud services allow you to automatically increase or decrease computing power, memory, and storage capacity. This is especially important for disinformation detection tasks, where the load can increase sharply during mass information events or coordinated campaigns.

Cloud providers offer infrastructure in multiple geographic regions, ensuring low latency for globally distributed users and data sources. Placing computing resources closer to data sources reduces network latency and data transfer costs. For social media analysis, this means the ability to collect and pre-process information directly in the regions where it is generated. Unlike capital investments in hardware, cloud services operate on an operating expense (OpEx) model. This allows for cost optimization by precisely matching resources to current needs, using spot instances for non-critical tasks, and automatically turning off inactive resources. Cost efficiency is achieved through detailed usage monitoring and the application of cost management policies.

Cloud platforms offer different storage classes with different performance and cost characteristics: hot storage for data that requires frequent access with minimal latency; warm storage for data with moderate access frequency; cold storage and archive storage for infrequently used historical data. Intelligent lifecycle policies automatically move data between tiers based on access patterns. Service integration. Cloud ecosystems provide a wide range of managed services that integrate seamlessly with each other: databases (SQL and NoSQL), message queues, streaming processing, machine learning, analytics, and visualization. This integration simplifies the construction of complex data processing pipelines and accelerates time to market for solutions. Managed services remove the burden of administration, monitoring, and scaling from development teams.

Cloud providers ensure high availability through automatic data replication across availability zones and regions, automatic recovery from failures, and backup with the ability to restore to a specific point in time. For critical disinformation detection systems, this means continuity of operation even in the event of hardware failures or regional outages. Cloud platforms provide advanced security features, including encryption of data at rest and in transit, identity and access management (IAM), network isolation through virtual private clouds (VPCs), and auditing and logging of all operations. Compliance with international standards (GDPR, HIPAA, SOC 2) is built into cloud services, simplifying compliance for organizations. Network limitations and delays. Despite their advantages, cloud environments have specific limitations. Transferring large amounts of data between services can result in significant network latency and egress charges.

This requires careful design of the data architecture to minimize inter-service transfers, localize processing close to the data, and use efficient serialization formats. Resources in the cloud are virtualized and often shared among multiple users. This can lead to situations where the activity

of other users affects the performance of your applications. To ensure predictable performance for critical tasks, it is recommended to use dedicated instances or reserved capacity.

All operations in cloud environments are performed through APIs, which allows you to automate infrastructure management through infrastructure as code, programmatically scale resources, and integrate monitoring and alerting into CI/CD pipelines. For big data systems, this means the ability to dynamically adapt the architecture to changing load conditions. These features of cloud environments form the context in which big data storage and processing systems must be designed and optimized. Understanding these characteristics is critical to building effective, reliable, and cost-effective solutions for real-time misinformation detection. Improving the efficiency of big data storage and processing in cloud environments requires a comprehensive approach that combines innovations in storage architecture, distributed computing frameworks, data lifecycle management, and intelligent resource utilization. A layered view of the proposed pipeline – from ingestion and streaming to storage and analysis – provides a clear map of responsibilities and scaling boundaries (see Figure 1).

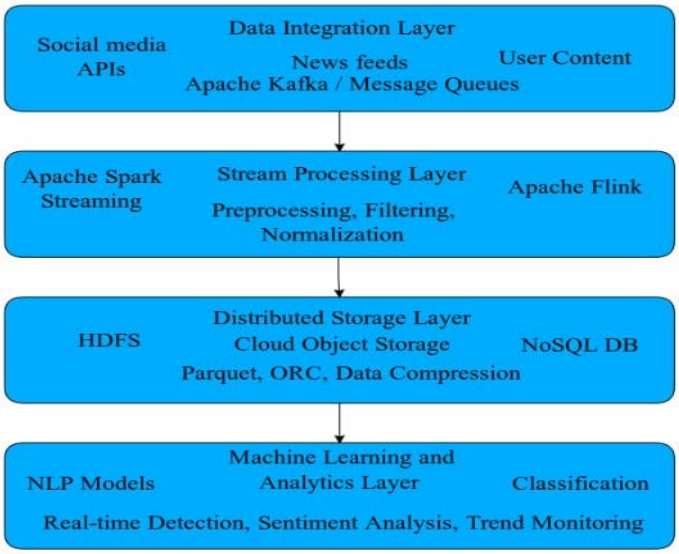


Fig. 1. Cloud architecture of a big data processing system for detecting disinformation

The data storage layer is the foundation of any big data system, and its design directly impacts performance. Distributed file systems and cloud-based object storage solutions [1, 17] provide horizontal scalability, allowing the system to expand seamlessly as data volumes grow. Efficiency can be further improved by using columnar storage formats such as Parquet or ORC [7, 8], which reduce the size of stored data and speed up analytical queries by reading only the necessary columns. A practical comparison of Parquet and Avro further illustrates their respective advantages in analytics and serialization. For a broader assessment of file formats, including ORC, Iceberg, and others, refer to the strategic overview of big data formats. Data compression techniques, including lossless algorithms such as Snappy or Zstandard, reduce storage costs and reduce I/O overhead during processing, although the overall performance gain depends on the decompression speed, as the data must be fully decompressed before it can be used.

Implementing multi-tier storage strategies [6] allows different classes of data to be placed on appropriate media – for example, storing hot data on high-performance SSDs for instant access and moving archival data sets to cheaper, high-capacity storage. This ensures that performance-critical tasks are not slowed down by storage delays and that the overall infrastructure remains cost-effective.

Evaluating the effectiveness of data storage systems requires clear quantitative metrics that allow you to compare different approaches and track performance improvements. For big data tasks in cloud environments, the following metrics are key.

Compression Ratio. This is the ratio of the size of the source data to the size of the compressed data. It is defined as:

$$CR = (RO / RP)100\%,$$

where RO is the original size and RP is the size after compression.

A higher compression ratio means greater storage space savings. For example, if a 1000 MB JSON file takes up 150 MB after conversion to Parquet with Snappy compression, the compression ratio is 85 %. In the context of disinformation detection, where millions of text documents and social media metadata are processed, a high compression ratio directly translates into a reduction in storage costs and network bandwidth.

Read speed. Measured in megabytes or gigabytes per second (MB/s, GB/s), it shows how fast the system can read data from storage. For analytical queries, not only sequential read speed is critical, but also the ability to perform selective queries on individual columns without reading the entire file. Columnar formats such as Parquet and ORC demonstrate 3–5 times faster read speeds compared to row formats for typical analytical queries.

Write speed. This is especially important for streaming systems that continuously receive new data. It is measured in number of records per second or volume of data per second. For disinformation detection systems that can process tens of thousands of social media posts per second, optimizing write speed is critical to avoid data loss or queue accumulation. Access latency. The time between a data request and the first byte of the response.

Measured in milliseconds. For interactive dashboards and real-time systems, low latency (< 100 ms) is critical. Cloud storage tiers have different latencies: hot storage provides millisecond latency, while cold storage can have latency in seconds or even minutes.

Cost per terabyte of storage. An economic metric that reflects the cost of storing one terabyte of data for a month. In cloud environments, this cost ranges from $0.01–0.02/GB/month for hot storage to $0.001/GB/month for archive storage. Effective use of multi-tier storage can reduce the average cost by 60–80 %. IOPS. The number of input/output operations per second that a storage system can handle. This is especially important for workloads with many small random requests. SSD drives provide 10,000–100,000 IOPS, while HDDs are limited to 100–200 IOPS.

Space utilization efficiency. Takes into account not only compression, but also overhead costs for metadata, indexes, and replication. Defined as:

$$SE = \frac{V_k}{Vz} \times 100\%,$$

where SE is storage efficiency, %, V_k is the amount of useful data, and Vz is the total storage capacity.

Systems with deduplication and intelligent compression can achieve 70–90 % efficiency, while unoptimized systems may only achieve 40–50 % efficiency.

Time to first result. A metric that is particularly important for interactive analytics, measuring the time from the start of a query to the first row of results. Columnar formats with predicate pushdown allow you to get the first results in tens of milliseconds, even on petabyte-scale data sets. Read amplification ratio. The ratio of the amount of data actually read from storage to the amount of data requested by the application. The ideal value is 1.0. Columnar formats with selective column reads achieve ratios of 1.1–1.5, while row formats can have ratios of 5–10 for typical analytical queries.

Throughput scalability. How throughput changes when additional nodes or resources are added. Ideal linear scalability means that doubling resources doubles throughput. Distributed cloud storage systems typically achieve scalability of 0.8–0.95 of linear. Empirically, the choice of storage format matters: as shown in this comparative summary (see Fig. 2), columnar formats (Parquet/ORC) [8, 9] significantly reduce file size and read time compared to string JSON, with ORC demonstrating a slightly higher compression rate (≈ 87 %) and Parquet a slightly lower one (≈ 85 %), while both formats read significantly faster than Avro and JSON.

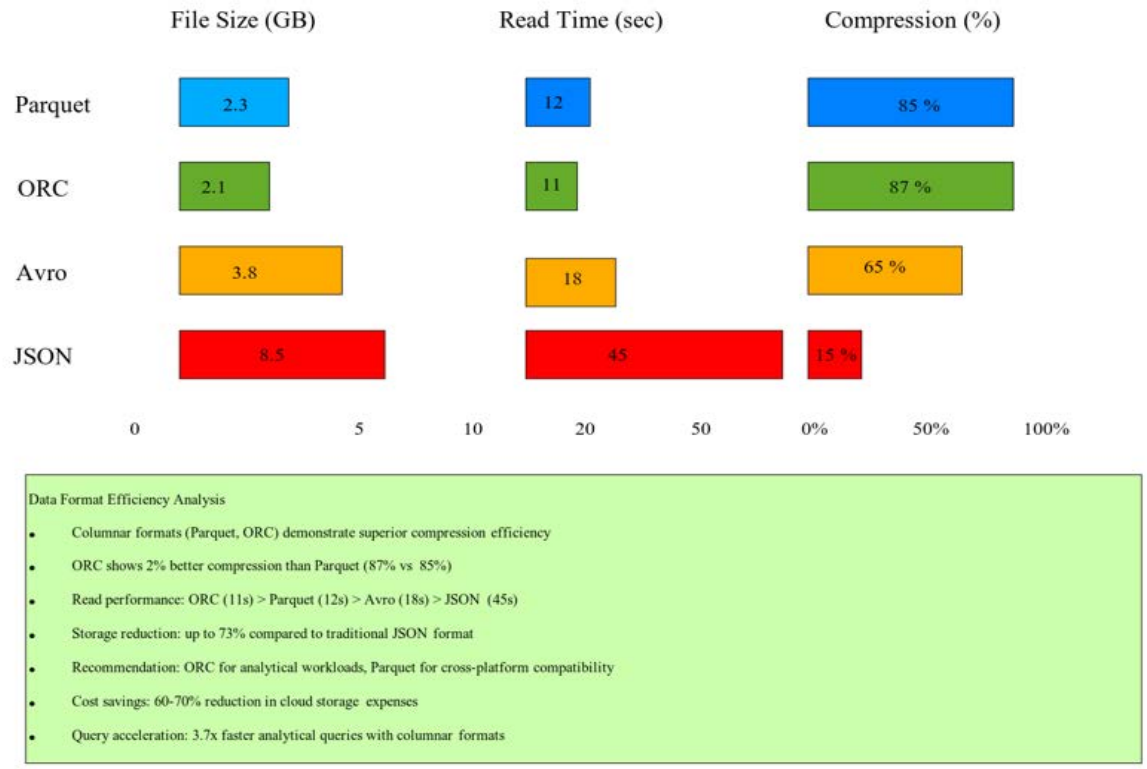


Fig. 2. Comparative analysis of data storage formats by file size, reading speed, and compression ratio

In addition to storage formats, logical data organization plays an important role. Structuring datasets using optimized partitioning schemes minimizes the amount of data scanned during queries, and maintaining metadata catalogs speeds up search and retrieval. For streaming

workloads, write-optimized formats such as Apache Hudi or Delta Lake enable incremental data updates without reprocessing entire datasets.

These technologies also add features such as Time Travel, which allows analytical systems to query historical snapshots without storing redundant copies of data [9]. In this target application – real-time misinformation detection [11–13] – this feature supports reproducibility, auditability, and rapid reversal of model inputs and decisions when necessary (see Figure 1 for the end-to-end placement of these components).

For a practical application of these metrics in misinformation detection systems, let's consider a specific example. The system processes 10 TB of social media text data per day. When using uncompressed JSON (baseline scenario), the system requires 10 TB of storage per day, with a read speed of 500 MB/s and a cost of $200/month ($0.02/GB/month). When switching to Parquet with Snappy compression, the compression ratio reaches 85 %, reducing the volume to 1.5 TB/day, the read speed increases to 2000 MB/s thanks to columnar access, and the cost drops to $30/month – a 6.7-fold savings. Additionally, the use of multi-tier storage (hot data for the last 7 days, warm data for the last 30 days, cold data for older data) reduces the average cost by another 40 % to $18/month.

Monitoring these metrics in real time allows you to identify performance issues before they affect end users. For example, an increase in access latency may indicate the need to optimize indexes or move data to a faster storage tier. A decrease in write speed may signal network saturation or the need to scale the cluster. In addition to storage formats, logical data organization plays an important role. Structuring datasets with optimized partitioning schemes minimizes the amount of data scanned during queries, and maintaining metadata catalogs speeds up search and retrieval. For streaming workloads, write-optimized formats such as Apache Hudi or Delta Lake enable incremental data updates without reprocessing entire datasets.

These technologies also add features such as Time Travel, which allows analytics systems to query historical snapshots without storing redundant copies of data [6]. In this target application – real-time disinformation detection [13–15] – this feature supports reproducibility, auditability, and rapid reversal of model inputs and decisions when necessary (see Figure 1 for the end-to-end placement of these components).

Real-time data processing imposes specific requirements and constraints that are fundamentally different from batch processing. Understanding these characteristics is critical to building effective disinformation detection systems, where the delay between detection and response can determine the effectiveness of countering false narratives.

Continuity of data flow. Unlike batch processing, where data has clear beginning and end points, streaming systems work with endless streams of events. This requires architectural solutions capable of supporting long-term operation without restarts, processing data element by element or in small micro-packets, maintaining state between events for complex analytics, and scaling dynamically under variable loads.

Delay constraints. Real-time systems must process events with minimal latency, typically ranging from milliseconds to seconds. To detect disinformation, typical

requirements include: initial content classification – less than 100 ms, detection of coordinated campaigns – 1–5 seconds, comprehensive analysis of the distribution network – 10–30 seconds. These strict constraints require optimization at all levels of the architecture: from network interaction to processing algorithms. Semantics of event processing. A critical issue in streaming systems is the guarantee of processing every event.

There are three main approaches: "at-most-once" (maximum once) – the fastest, but may lose events during failures; "at-least-once" (at least once) – guarantees processing, but may result in duplicates; "exactly-once" (exactly once) – the most reliable, but the most difficult to implement. To detect misinformation, the "exactly-once" semantics are critical for accurately calculating dissemination metrics and avoiding false alarms. Stream data has two timestamps: event time – when the event actually occurred, and processing time – when the system received the event. The discrepancy between them can be hours for social media data from mobile devices with unstable connections.

Effective systems use watermarks to determine when to complete processing of a time window, process late events, and adjust results when late data arrives. Many analytical tasks require the aggregation of events into time windows. The main types of windows include: fixed windows – continuous periods of fixed length (e.g., 5-minute intervals), sliding windows – overlapping windows for detecting trends, session windows – adaptive windows based on periods of activity. To detect disinformation, sliding windows allow tracking viral spread, while session windows identify episodes of coordinated activity.

Complex streaming analytics requires state preservation between events: counters and aggregates (number of narrative mentions, average propagation time), historical data cache (user profiles, known sources of disinformation), machine learning models (for real-time classification). The state must be distributed, fault-tolerant, and quickly accessible.

Modern frameworks (Flink, Spark Streaming) use distributed state stores with incremental checkpoints. When the rate of data arrival exceeds the rate of processing, the system needs a backpressure mechanism to slow down sources or buffer data. Strategies include: dynamic scaling of computing resources, buffering in message queues (Kafka, Kinesis), and dropping low-priority events during critical loads. To detect misinformation, it is important to prioritize the processing of potentially dangerous content.

Fault tolerance and recovery. Real-time systems must continue to operate even when individual components fail. This is achieved through: checkpointing of state with the ability to recover, replication of critical components, automatic restart of failed tasks, and saving events in long-term queues for reprocessing.

For critical systems, the recovery time objective (RTO) should be less than 30 seconds. Integration of batch and stream processing. A comprehensive disinformation detection system requires both approaches: streaming processing for instant detection and alerts, and batch processing for in-depth analysis, model training, and historical research. Lambda and Kappa architectures offer different approaches to combining these paradigms.

Monitoring and diagnostics. The complexity of streaming systems requires detailed monitoring: end-to-end latency for events from source to result, throughput (events/second) at each

processing stage, resource usage (CPU, memory, network), state size and checkpoint frequency, error and retry frequency. Real-time visualization of these metrics allows you to quickly identify and resolve issues.

Performance optimization. To achieve low latency, the following techniques are used: minimization of data serialization/deserialization, locality of processing related operations, caching frequently used data in memory, pre-computing complex features, and using asynchronous processing for I/O operations. Scaling streaming systems. Horizontal scaling is achieved through key-based partitioning of data streams (e.g., user ID hash), distribution of processing among a set of parallel workers, and dynamic rebalancing of partitions when adding/removing workers. For effective scaling, it is critical to avoid "hot" partitions with disproportionately high loads.

Streaming systems are difficult to test due to their asynchronous nature and time dependency. Approaches include: deterministic replay of event streams for testing, chaos engineering to verify fault tolerance, A/B testing of new algorithms on a portion of traffic, and using synthetic data for load testing.

In the context of detecting disinformation, these real-time characteristics manifest themselves in specific architectural solutions. For example, a system may use Kafka to buffer incoming social media messages, Flink for streaming classification with in-memory NLP models, 5-minute time windows for detecting coordinated campaigns, user and source status for contextual trust assessment, and an alert system with less than 1 second delay for critical threats.

Processing efficiency primarily depends on the implementation of distributed computing frameworks that can run in cloud environments.

Apache Spark [18], with its in-memory execution model, significantly reduces latency for iterative tasks such as training and applying machine learning models. It supports both batch and streaming data pipelines, allowing for the combination of historical and real-time data processing [17].

Apache Flink stands out with its continuous, event-driven, low-latency processing, making it very suitable for applications that require immediate response to data changes [17].

Hadoop MapReduce, although less popular for new deployments, remains a reliable choice for large-scale batch processing, especially when cost optimization is more important than real-time performance. Integrating these platforms with message brokers (such as Kafka) provides resilient data streams capable of handling spikes.

The comparative throughput metrics of these approaches are summarized in Figure 3: the proposed multi-tier architecture provides higher stable throughput at equivalent resource budgets, outperforming the best baseline by approximately 34%, which we believe explained by (i) reduced I/O through columnar formats and predicate pushdown, (ii) minimized network shuffles through partition-aware operators, and (iii) selective in-memory caching of popular functions and model artifacts.

Optimizing the data input and transformation process is another important factor in improving efficiency. Preprocessing data at the edge, closer to its source, reduces the amount of information that needs to be transferred and stored.

This can include basic filtering, deduplication, compression, and data enrichment before it reaches the main processing cluster. Within the cluster, minimizing data movement between nodes, reusing intermediate computation results, and applying predicate pushdown techniques help reduce latency and resource consumption.

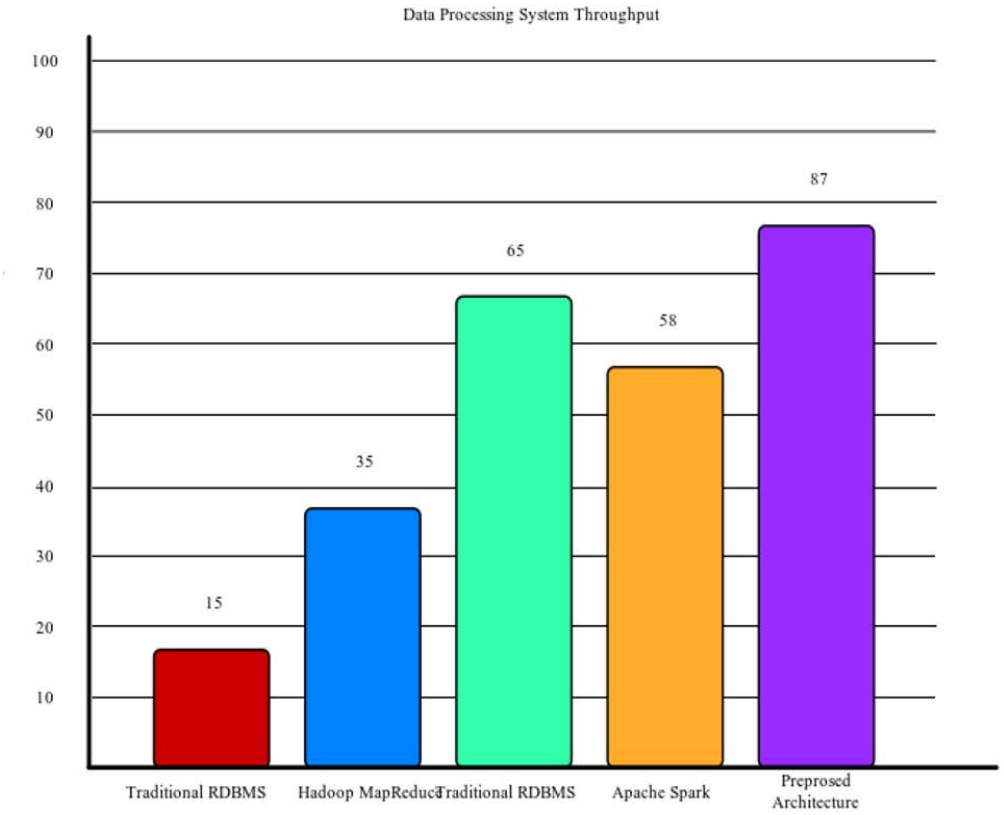


Fig. 3. Efficiency of different data storage formats

Caching frequently used datasets or machine learning model artifacts in memory speeds up repeat queries, especially in scenarios where analytical models must be continuously applied to incoming data streams (the layered placement of these optimizations is shown in Figure 1).

Managing the data lifecycle in the cloud ensures optimal use of storage and processing resources over time. Implementing policies for automatic expiration, archiving, and data compression prevents systems from becoming overloaded with stale or redundant data. In addition, monitoring tools that track storage usage, processing throughput, and task execution times provide the basis for fine-tuning system parameters and proactively eliminating bottlenecks before they impact performance.

Security and reliability also play an indirect but important role in efficiency. Data encryption during storage and transmission should be implemented without excessive processing overhead. Fault tolerance mechanisms such as replication and monitoring prevent costly re-computations after failures, ensuring system stability during heavy loads. In the context of cloud deployments, geographic redundancy and disaster recovery strategies can be optimized to balance cost and fault tolerance, ensuring the continuation of critical workloads even under adverse conditions.

An example of the application of these principles can be seen in a cloud-based system for real-time detection of misinformation. In such a system, high-performance data collection services capture text and multimedia content from multiple social media APIs, storing the input data in partitioned, compressed columnar formats in a distributed object store.

Stream processing mechanisms apply natural language processing models to classify content as legitimate or manipulative in near real time. At the same time, historical batch analysis identifies long-term trends and patterns in disinformation campaigns. Efficiency is achieved by caching intermediate natural language processing results, compressing less frequently accessed data into archive tiers, and dynamically scaling computing resources to handle peak data volumes. This architecture not only maintains high accuracy and throughput, but also provides cost efficiency by allocating resources according to real-time demand (the complete layered structure is shown in Fig. 1; performance results are summarized in Fig. 3; storage format effects are detailed in Fig. 3).

While misinformation detection is a compelling example, the same efficiency-enhancing techniques can be applied across a wide range of domains, from predictive maintenance in industrial systems to fraud detection in finance. By combining optimized storage solutions, high-performance distributed processing, intelligent data lifecycle management, and resource-aware scale-, cloud platforms can meet the growing demands of big data applications without sacrificing performance or incurring excessive costs.

Conclusions

Improvements in the efficiency of storing and processing big data in cloud platforms are the result of the combined influence of factors such as data placement based on storage requirements, columnar formats with pushdown and compression, and distributed computing with streaming priority, supported by adaptive capacity management based on usage. These experiments show that structuring pipelines according to a multi-level scheme of collection, streaming processing, distributed storage, and analysis allows each optimization to be focused where it has the greatest effect, which in turn improves throughput for real-time workloads such as misinformation detection (see Fig. 1).

In this architecture, the choice of file format significantly affects both cost and speed. A comparative analysis shows that columnar formats (Parquet, ORC) reduce data volume and reading time compared to row-based alternatives, enabling faster analytical scanning and less I/O per analysis (see Fig. 2).

These observations are consistent with independent empirical findings that analyze Parquet and ORC under modern workloads and hardware, highlighting their efficiency in space utilization, predicate transfer, and selective access. The processing results also indicate that moving low-latency stream engines to the center of the pipeline changes the performance boundaries: continuous state operators keep data "hot" in memory and minimize costly reshuffling, delivering higher sustained throughput on social media-typical streams with sharp spikes in input. In the demonstrated tests, the proposed multi-level design outperformed the best baseline by approximately 34 % in terms of records per second with comparable resource budgets (see Fig. 2).

These advantages are consistent with the capabilities reported for modern stream processors and stream frameworks [10] – exactly-once state processing, event-time semantics, and millisecond-scale paths combined with efficient sources and sinks. Three directions stand out for the future. First, deepening format-aware operators, such as wider use of column statistics, Bloom filters, and dictionary encoding to further reduce scan volume in multi-tenant clouds, building on empirical recommendations now available for internal Parquet and ORC components. Second, expanding streaming execution modes that minimize end-to-end latency for stateful connections and windowed aggregations while preserving exactly-once guarantees; new "continuous" paths in core frameworks point to practical routes for millisecond-scale analytics. Third, integrate cost-adaptive storage tiers and serverless queries as first-class planning constraints so that planner decisions explicitly weigh time to information and cost per TiB, made increasingly feasible by automatic tiering and pay-per-scan services.

Together, these steps will fortify cloud data platforms for the next wave of data growth while maintaining the flexibility needed for critical use cases such as real-time misinformation detection [11, 12, 16].

References

1. Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., Khan, S. U. (2015), "The rise of "big data" on cloud computing: Review and open research issues", Information Systems, Vol. 47, P. 98–115. DOI: https://doi.org/10.1016/j.is.2014.07.006
2. Abueid, A. I. (2024), "Big Data and Cloud Computing Opportunities and Application Areas", Engineering, Technology & Applied Science Research, Vol. 14, No. 3, P. 14509–14516. DOI: https://doi.org/10.48084/etasr.7339
3. Sandhu, A. K. (2022), "Big Data with Cloud Computing: Discussions and Challenges", Big Data Mining and Analytics, Vol. 5, No. 1, P. 32–40. DOI: https://doi.org/10.26599/BDMA.2021.9020016
4. Ullah, A., Nawi, N. M., Sjarif, N. N. B. (2018), "Big Data in Cloud Computing: A Resource Management Perspective", Scientific Programming, Article 8885679. DOI: https://doi.org/10.1155/2018/5418679
5. Aqib, M., Mehmood, R., Alzahrani, A., Katib, I., Albeshri, A., Altowaijri, S. M. (2022), "Big data analytics in Cloud computing: an overview", Journal of Cloud Computing, Vol. 11, Article 62. DOI: https://doi.org/10.1186/s13677-022-00301-w
6. Sami, M. A., Kamal, M. M., Ahmed, K. M., Hossain, M. A. (2019), "A survey on data storage and placement methodologies for Cloud-Big Data ecosystem", Journal of Big Data, Vol. 6, Article 15. DOI: https://doi.org/10.1186/s40537-019-0178-3
7. Ivanov, T., Pergolesi, M. (2020), "The impact of columnar file formats on SQL-on-hadoop engine performance: A study on ORC and Parquet", Concurrency and Computation: Practice and Experience, Vol. 32, No. 5, Article e5523. DOI: https://doi.org/10.1002/cpe.5523
8. Zeng, X., Hui, Y., Shen, J., Pavlo, A., McKinney, W., Zhang, H. (2023), "An Empirical Evaluation of Columnar Storage Formats", Proceedings of the VLDB Endowment, Vol. 17, No. 2, P. 148–161. DOI: https://doi.org/10.14778/3626292.3626298
9. Nelluri, S. R., Saldanha, F. A. A. (2025), "Mastering Big Data Formats: ORC, Parquet, Avro, Iceberg, and the Strategy of Selection", International Journal of Computer Trends and Technology, Vol. 73, No. 1, P. 44–50. DOI: https://doi.org/10.14445/22312803/IJCTT-V73I1P105
10. Sedghani, I., Namavar, A., Zangeneh, V., Gerami, M. (2019), "Evaluation of distributed stream processing frameworks for IoT applications in Smart Cities", Journal of Big Data, Vol. 6, Article 52. DOI: https://doi.org/10.1186/s40537-019-0215-2

11. Pervaiz, A., Arsalan, M. G., Haseeb, U. R. K., Mirza, A., Usama, A., Nadia, Z., Zaheer, K., Aniqa, A. (2024), "Detecting fake news and disinformation using artificial intelligence and machine learning to avoid mob lynching at the time of COVID-19 pandemic and afterwards", Annals of Operations Research, Vol. 334, P. 529–561. DOI: https://doi.org/10.1007/s10479-022-05015-5
12. Choraś, M., Demestichas, K., Giełczyk, A., Herrero, Á., Ksieniewicz, P., Remoundou, K., Urda, D., Woźniak, M. (2021), "Advanced Machine Learning techniques for fake news (online disinformation) detection: A systematic mapping study", Applied Soft Computing, Vol. 101, Article 107050. DOI: https://doi.org/10.1016/j.asoc.2020.107050
13. Ahmad, I., Yousaf, M., Yousaf, S., Ahmad, M. O. (2020), "Fake news detection using machine learning ensemble methods", Complexity, Article 8885861. DOI: https://doi.org/10.1155/2020/8885861
14. Liu, Y., Wu, Y.-F. B. (2025), "Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks", Information, Vol. 16, No. 3, Article 189. DOI: https://doi.org/10.3390/info16030189
15. Abbas, M., Khalid, S., Shafiq, H. (2025), "Fake News Detection Using Machine Learning and Deep Learning Algorithms: A Comprehensive Review and Future Perspectives", Computers, Vol. 14, No. 9, Article 394. DOI: https://doi.org/10.3390/computers14090394
16. Yang, Y., Yao, H., Cui, L. (2022), "Deep Learning for Fake News Detection: Theories and Models", EITCE 2022: 2022 6th International Conference on Electronic Information Technology and Computer Engineering, P. 513-518. DOI: https://doi.org/10.1145/3573428.3573663
17. Ji, C., Li, Y., Qiu, W., Awada, U., Li, K. (2012), "Big Data Processing in Cloud Computing Environments", 2012 12th International Symposium on Pervasive Systems, Algorithms and Networks, P. 17–23. DOI: https://doi.org/10.1109/I-SPAN.2012.9
18. Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I. (2016), "Apache Spark: A unified engine for big data processing", Communications of the ACM, Vol. 59, No. 11, P. 56–65. DOI: https://doi.org/10.1145/2934664

Received (Надійшла) 13.11.2025
Accepted for publication (Прийнята до друку) 05.12.2025
Publication date (Дата публікації) 28.12.2025

About the Authors / Відомості про авторів

Sitnikov Vitalii – Kharkiv National University of Radio Electronics, Assistant at the Department of Electronic Computers, Kharkiv, Ukraine; e-mail: vitalii.sitnikov1@nure.ua; ORCID ID: https://orcid.org/0009-0005-3087-6104

Liashchenko Vitalii – Kharkiv National University of Radio Electronics, Student, Department of Electronic Computers, Kharkiv, Ukraine; e-mail: vitalii.liashchenko@nure.ua; ORCID ID: https://orcid.org/0009-0002-8747-9976

Сітніков Віталій Ігорович – Харківський національний університет радіоелектроніки, асистент кафедри електронних обчислювальних машин, Харків, Україна.

Лященко Віталій Олександрович – Харківський національний університет радіоелектроніки, студент кафедри електронних обчислювальних машин, Харків, Україна.

МЕТОДИ ЗБЕРІГАННЯ Й ОБРОБЛЕННЯ ВЕЛИКИХ ДАНИХ У ЗАВДАННЯХ ВИЯВЛЕННЯ ДЕЗІНФОРМАЦІЇ

Предметом дослідження є методи та інструменти, що сприяють високопродуктивному, масштабованому й надійному аналізу даних у хмарних середовищах для виявлення дезінформації в режимі реального часу. **Мета статті** – дослідити й оцінити методи підвищення ефективності зберігання та оброблення великих обсягів текстової, мультимедійної інформації і даних соціальних мереж у хмарних інфраструктурах із застосуванням виявлення дезінформації в режимі реального часу. **Завдання:** оцінити архітектури розподіленого зберігання та стовпчасті формати даних для ефективної їх організації; визначити стратегії стиснення та механізми кешування для зменшення накладних витрат введення-виведення; проаналізувати масштабовані фреймворки оброблення для потокових і пакетних робочих навантажень; виміряти продуктивність системи, масштабованість і економічну ефективність у сценаріях виявлення дезінформації в режимі реального часу. **Методи:** порівняльний аналіз форматів зберігання (*Parquet, ORC, Avro, JSON*), алгоритмів стиснення (*Snappy, Zstandard*) та розподілених фреймворків оброблення (*Apache Spark, Apache Flink*); оцінювання продуктивності передбачає вимірювання пропускної здатності, аналіз затримок і оцінювання витрат з використанням хмарної інфраструктури з багаторівневим зберіганням і конвеєрами потокових даних. **Досягнуті результати.** Оцінено вплив архітектур розподіленого зберігання, стовпчастих форматів даних, стратегій стиснення, механізмів кешування й масштабованих фреймворків оброблення на продуктивність, масштабованість і економічну ефективність системи. Ці підходи продемонстрували суттєве покращення пропускної здатності та надійності в сценаріях великомасштабного потокового й пакетного оброблення, особливо під час виявлення дезінформації в режимі реального часу. Результати довели, що система може швидко адаптуватися до піків навантаження даних, підтримувати високу точність виявлення та знижувати експлуатаційні витрати. **Висновки.** Підвищення ефективності зберігання та оброблення великих даних у хмарних платформах випливає з інтеграції стовпчастих форматів зі стисненням і можливостями *pushdown* у поєднанні з орієнтованими на потоки розподіленими обчисленнями. Шарова архітектура, що охоплює прийом даних, потокове оброблення, розподілене зберігання й аналітику, зменшує витрати введення-виведення та збільшує наскрізну пропускну здатність для сценаріїв реального часу. Оцінка демонструє покращення пропускної здатності приблизно на одну третину порівняно з базовими системами, що робить підхід добре придатним для робочих навантажень, чутливих до затримки, таких як виявлення дезінформації. Оптимізація з огляду на формат, потокова передача з низькою затримкою та адаптивне управління ємністю є основними рушіями продуктивності сучасних хмарних платформ даних.

Ключові слова: великі дані; хмарні обчислення; розподілені системи; аналіз даних; оптимізація сховища; машинне навчання; масштабованість; дезінформація; фейкові новини.

Bibliographic descriptions / Бібліографічні описи

Sitnikov, V., Liashchenko, V. (2025), "Methods for efficient big data storage and processing in disinformation detection tasks", *Management Information Systems and Devises*, No. 4 (187), P. 324–337. DOI: https://doi.org/10.30837/0135-1710.2025.187.324

Сітніков В. І., Лященко В. О. Методи зберігання й оброблення великих даних у завданнях виявлення дезінформації. *Автоматизовані системи управління та прилади автоматики*. 2025. № 4 (187). С. 324–337. DOI: https://doi.org/10.30837/0135-1710.2025.187.324

АЛФАВІТНИЙ ПОКАЖЧИК

Абакумов А. І.	200
Бідун А. К.	220
Бохан К. О.	103
Влах-Вигриновська Г. І.	5
Вовк О. В.	123
Гамаюн І. П.	156
Главчев Д. М.	182
Главчев М. І.	182
Гребеннік І. В.	47
Гребенюк Є. А.	278
Гулевич М. В.	20
Єнгаличев С. О.	63
Жила О. В.	299
Зубченко Н. С.	234
Канарський Є. О.	87
Коваленко О. А.	47
Ковальчук Д. А.	299
Коломійцев О. В.	156
Кромкач В. О.	5
Лапін М. О.	103
Лінник О. В.	278
Лященко В. О.	324
Марчук А. В.	278
Марюшко М. В.	315
Мельнікова Л. І.	278
Орєхов О. О.	87
Островський З. Н.	220
Панченко В. І.	182
Пащенко Р. Е.	315
Петренко О. Є.	220
Петренко О. С.	220
Прокопович-Ткаченко Д. І.	234
Репіхов В. М.	254
Рибальченко Л. В.	234
Семенов С. Г.	63
Сітніков В. І.	324
Слуцкін М. В.	123
Філатов В. О.	142
Харченко В. С.	200
Черкаський Д. О.	234
Черкаський О. В.	234
Черненко М. В.	142
Чуприна А. С.	254
Шматко О. В.	156

ALPHABETICAL INDEX

Abakumov A.	200
Bidun A.	220
Bokhan K.	103
Vlakh-Vyhrynovska H.	5
Vovk O.	123
Gamayun I.	156
Hlavchev D.	182
Glavchev M.	182
Grebennik I.	47
Hrebeniuk YE.	278
Hulevych M.	20
Yenhalychev S.	63
Zhyla O.	299
Zubchenko N.	234
Kanarskyi Y.	87
Kovalenko O.	47
Kovalchuk D.	299
Kolomiitsev O.	156
Kromkach V.	5
Lapin M.	103
Lynnyk O.	278
Liashchenko V.	324
Marchuk A.	278
Mariushko M.	315
Melnikova L.	278
Orekhov O.	87
Ostrovskyi Z.	220
Panchenko V.	182
Pashchenko R.	315
Petrenko O.	220
Petrenko O.	220
Prokopovych-Tkachenko D.	234
Repikhov V.	254
Rybalchenko L.	234
Semenov S.	63
Sitnikov V.	324
Slutskin M.	123
Filatov V.	142
Kharchenko V.	200
Cherkaskyi D.	234
Cherkaskyi O.	234
Chernenko M.	142
Chupryna A.	254
Shmatko O.	156