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.

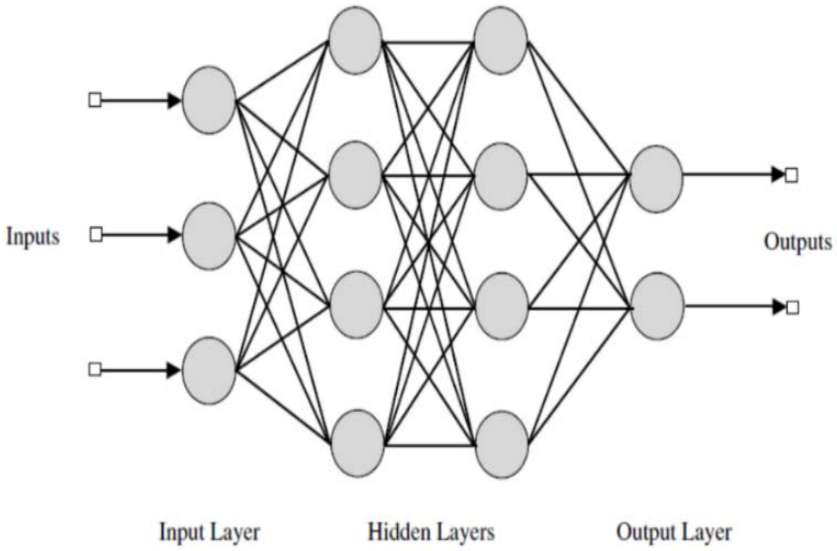

Fig. 1. Structure of a fully connected neural network

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.

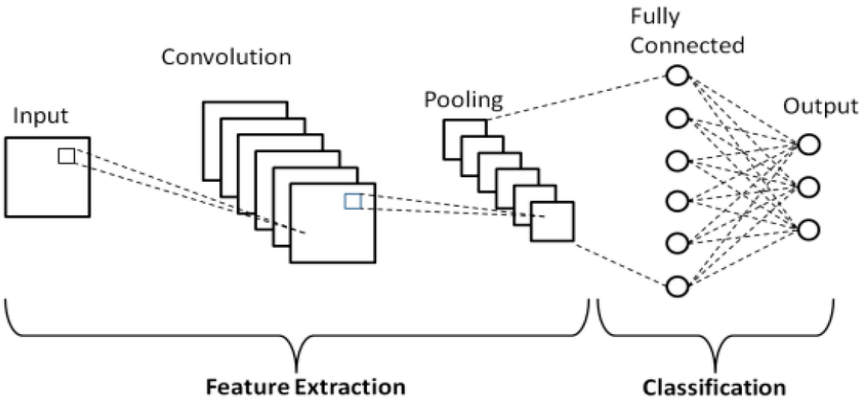

Fig. 2. Structure of a convolutional neural network

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.

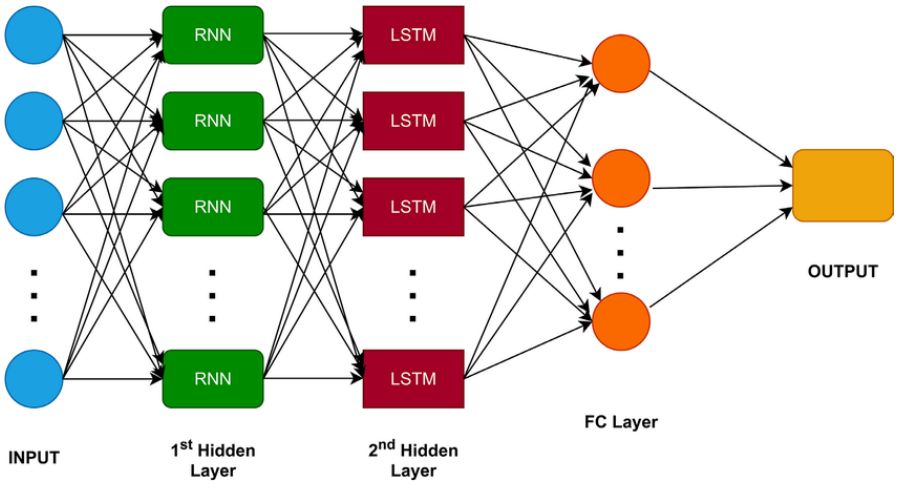

Fig. 3. Recurrent neural network structure

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.

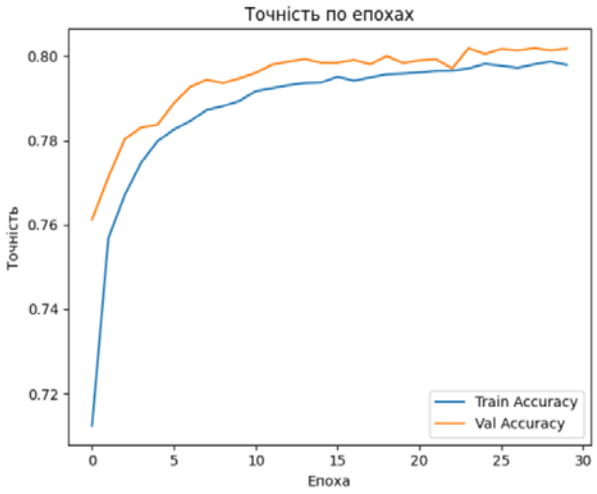

Fig. 4. Change in model accuracy on training and validation samples (Dense model)

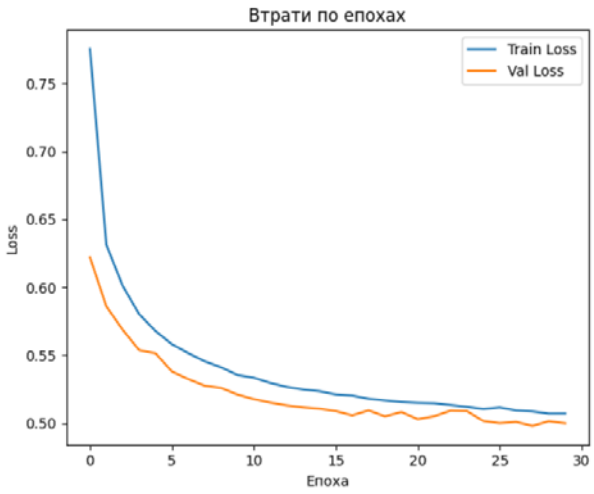

Fig. 5. Change in the loss function on the training and validation samples (Dense model)

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).

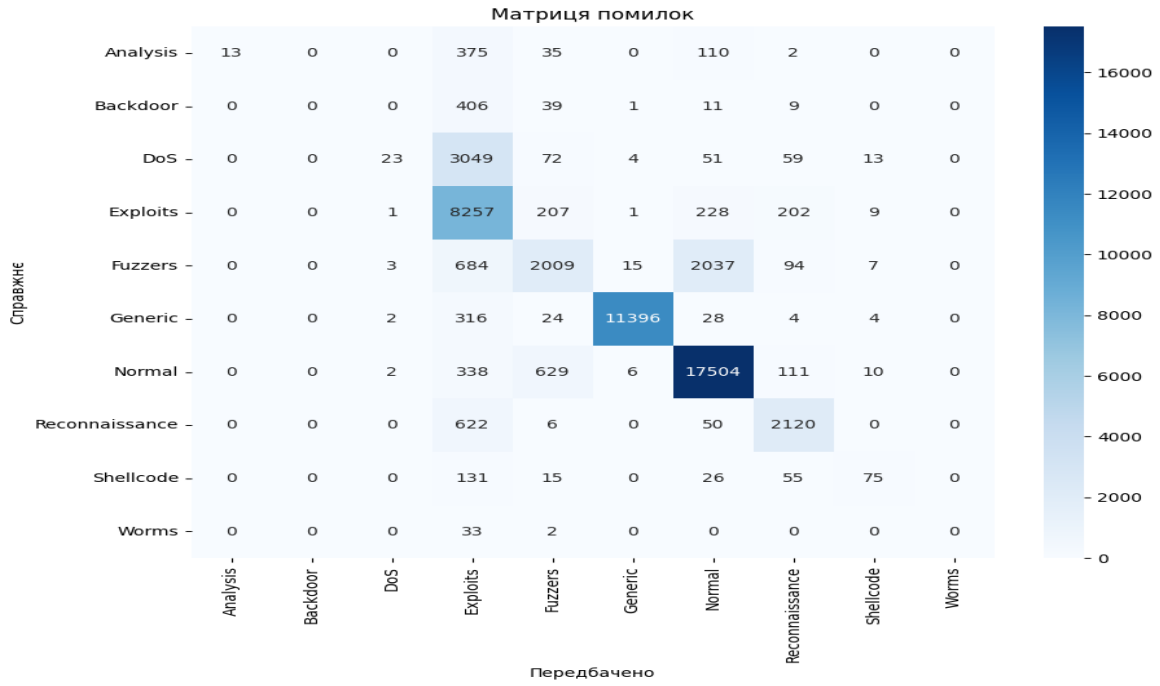

Fig. 6. Classification error matrix for the Dense model

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.

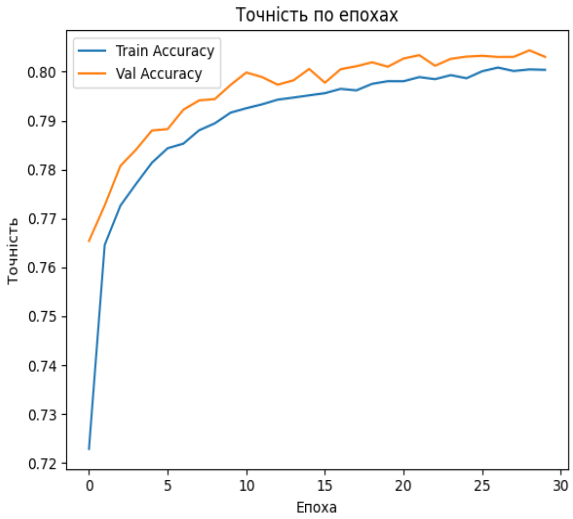

Fig. 7. Accuracy on training and validation samples (LSTM model)

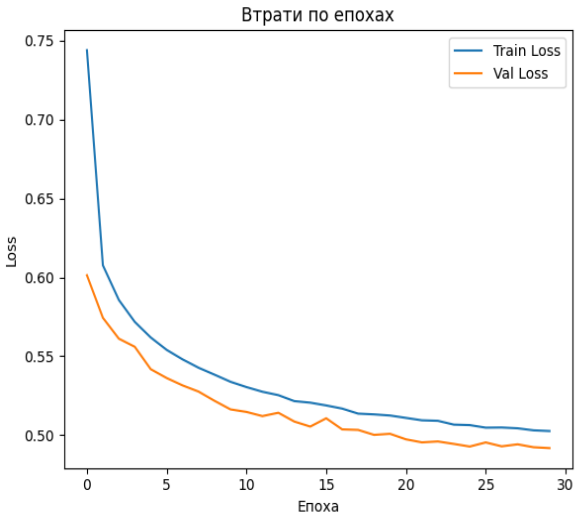

Fig. 8. Losses on training and validation samples (LSTM model)

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.

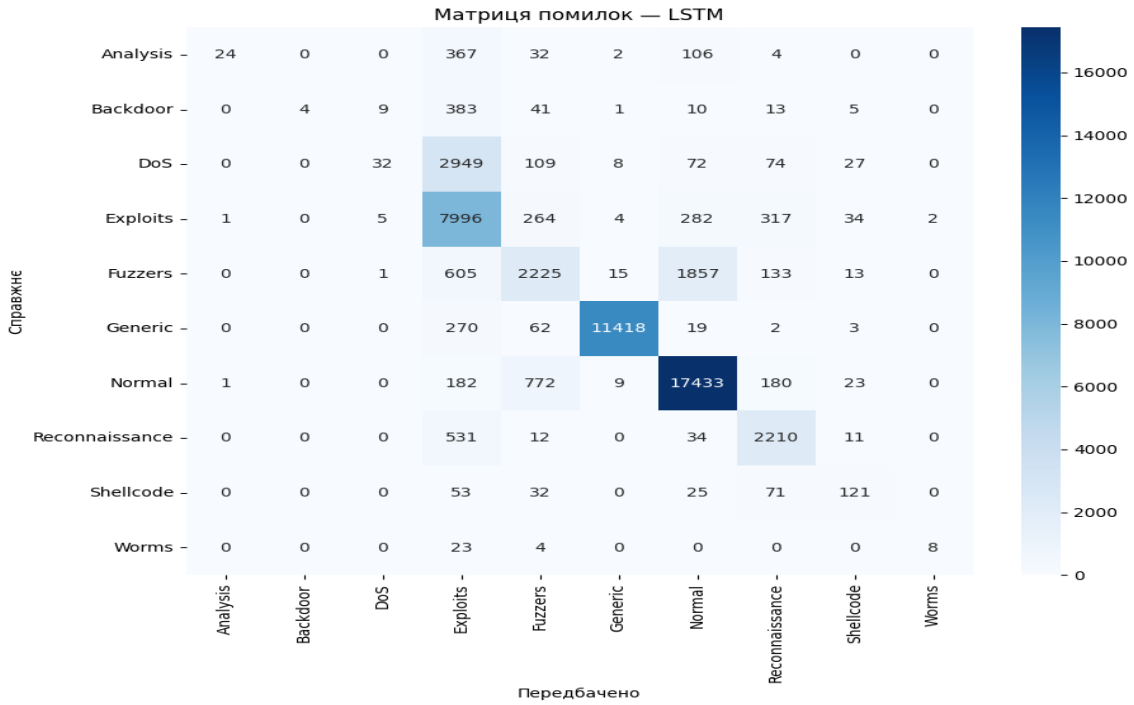

Fig. 9. Classification error matrix for the LSTM model

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.

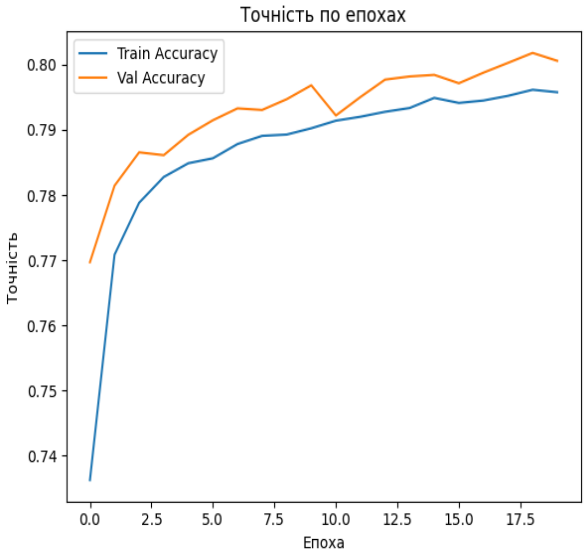

Fig. 10. Change in model accuracy on training and validation samples (CNN model)

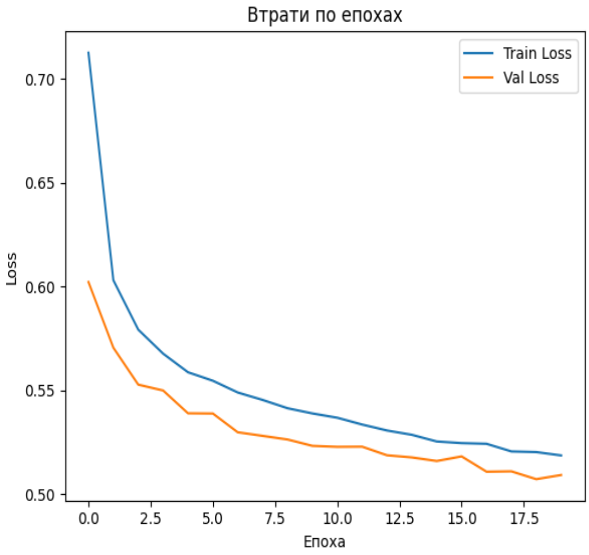

Fig. 11. Change in loss function on training and validation samples (CNN model)

The error matrix shown in Fig. 12 demonstrates that the model performs well in classifying the main classes, namely Normal, Generic, and Fuzzers.

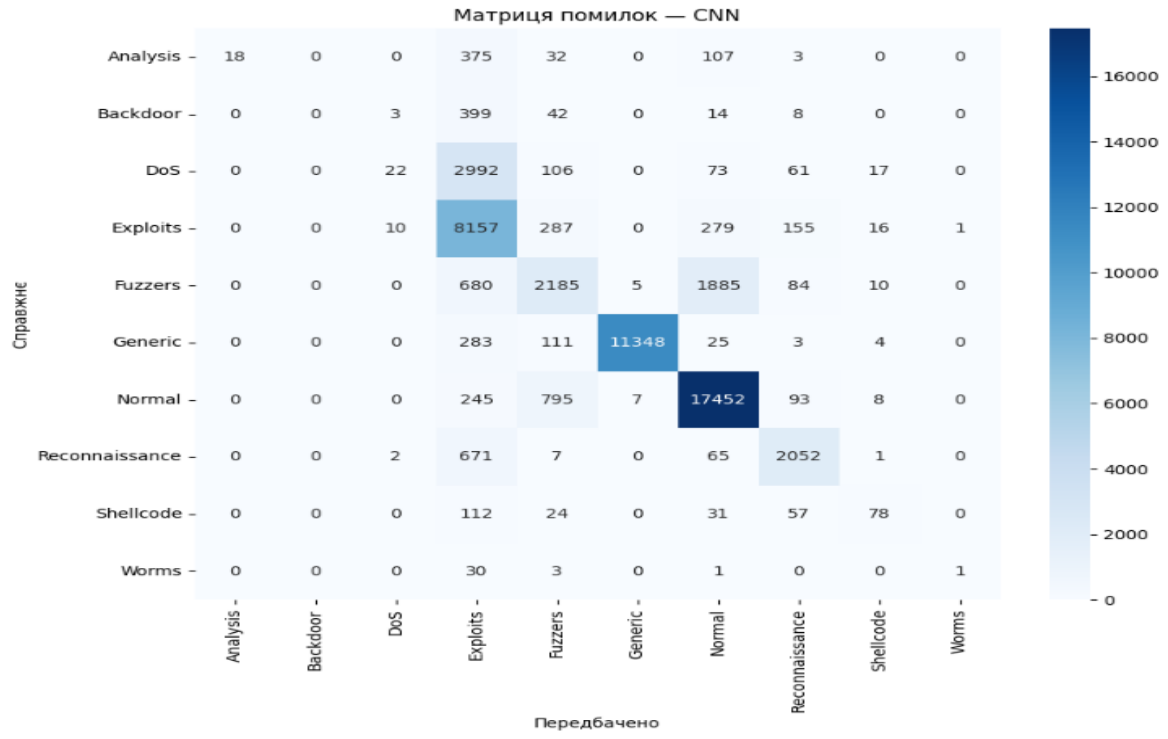

Fig. 12. Classification error matrix for the CNN model

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