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).

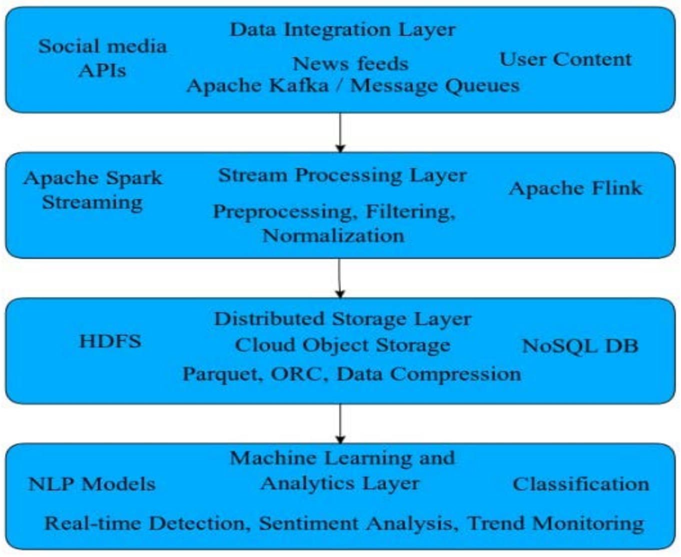

Data Integration Layer
Social media APIs
News feeds
User Content
Apache Kafka / Message Queues
Stream Processing Layer
Apache Spark Streaming
Apache Flink
Preprocessing, Filtering, Normalization
Distributed Storage Layer
HDFS
Cloud Object Storage
NoSQL DB
Parquet, ORC, Data Compression
Machine Learning and Analytics Layer
NLP Models
Classification
Real-time Detection, Sentiment Analysis, Trend Monitoring

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:

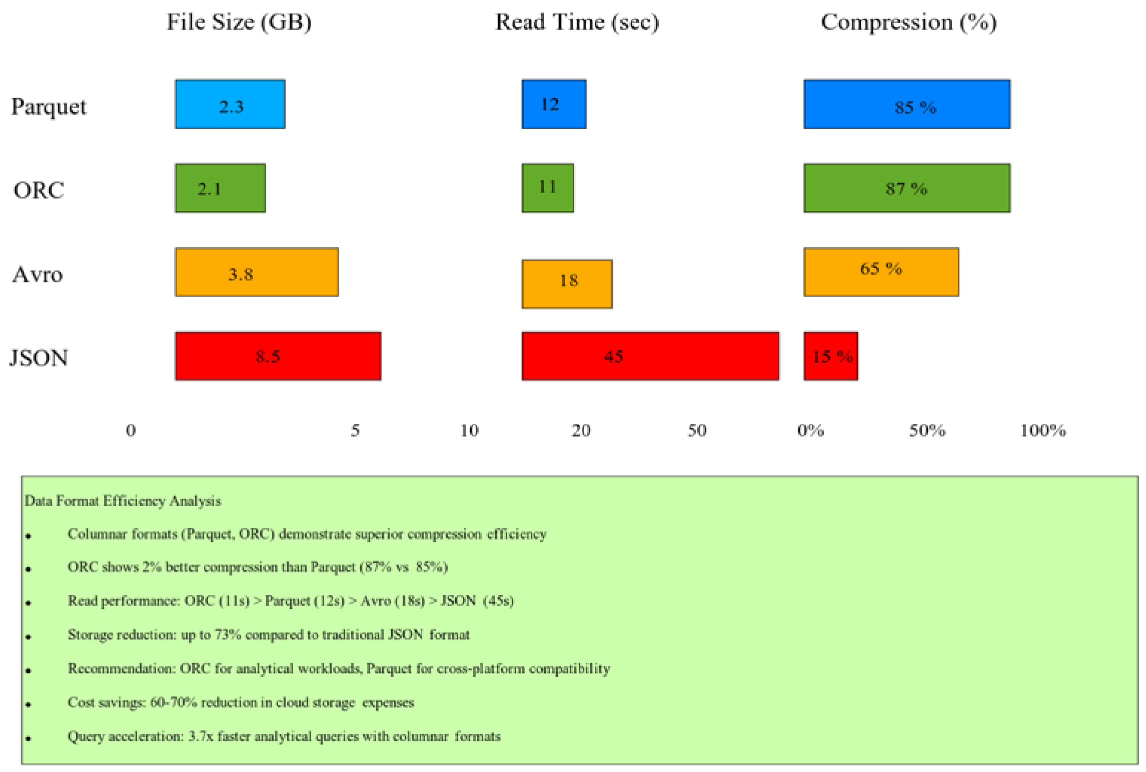
$$SE = \frac{V_k}{Vz} \times 100\%,$$

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.



Data Format Efficiency Analysis

- Columnar formats (Parquet, ORC) demonstrate superior compression efficiency
- ORC shows 2% better compression than Parquet (87% vs 85%)
- Read performance: ORC (11s) > Parquet (12s) > Avro (18s) > JSON (45s)
- Storage reduction: up to 73% compared to traditional JSON format
- Recommendation: ORC for analytical workloads, Parquet for cross-platform compatibility
- Cost savings: 60-70% reduction in cloud storage expenses
- Query acceleration: 3.7x faster analytical queries with columnar formats

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.

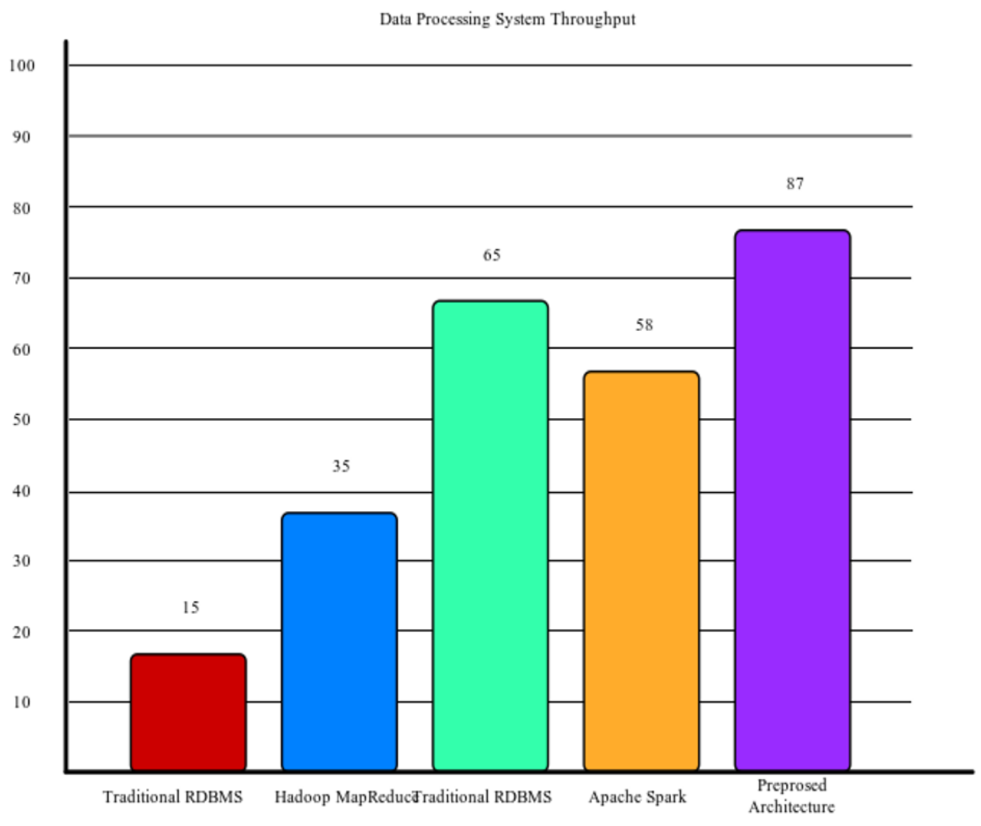


Fig. 3. Efficiency of different data storage formats

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