International Journal of Artificial Intelligence,
ISSN 0974-0635; Int. J. Artif. Intell.
2026, Volume 24, Issue Number 2 [Published on 29/08/2026]

Machine-Learning-Based Selection of Data Serialisation Formats for Web Applications Under Variable Network Conditions

A. Husakovskyi[1] and **P. Shyposha**[2]

[1]Department of Computer Systems, Networks, and Cybersecurity,
National Aerospace University "Kharkiv Aviation Institute",
Kharkiv, Ukraine;
Email: a_husakovskyi632@hotmail.com

[2]Department of Automation and Computer Systems,
Odesa National University of Technology,
Odesa, Ukraine;
Email: pavlo-shyposha@hotmail.com

ABSTRACT

The heterogeneity of modern network environments requires adaptive mechanisms for improving data transmission efficiency under changing bandwidth, latency, connection stability, and computational constraints. This study aimed to develop an intelligent system for selecting the optimal data serialisation format according to current network conditions. The proposed architecture combined real-time network monitoring, a multicriteria utility function, and LSTM, Random Forest, and Deep Q-Learning algorithms. Six formats – JSON, XML, Protocol Buffers, Avro, MessagePack, and BSON – were evaluated in 3G, 4G LTE, 5G, fast and slow Wi-Fi, and edge-computing environments. The models were trained using approximately 4.8 million telemetry and event-log records, and each experimental scenario was repeated 30 times. LSTM achieved 94% prediction accuracy, while Random Forest reached 95% with comparatively low computational requirements. Deep Q-Learning provided rapid adaptation and an accuracy of 92-95%. Compared with static format selection, the proposed system reduced latency by 35% and improved transmission efficiency by up to 87.5% under bandwidth-constrained conditions. The average internal response time was approximately 45 ms. Protocol Buffers demonstrated the most stable performance for large data volumes and constrained networks, whereas MessagePack provided a balanced solution for small and medium payloads. The findings confirm that context-sensitive format selection can improve transmission performance, resource utilisation, and system reliability.

Keywords: Protocol Buffers, MessagePack, bandwidth optimisation, latency reduction, edge computing.

Mathematics Subject Classification: 68M20

Computing Classification System: C.4

1. INTRODUCTION

The rapid development of web applications, mobile platforms, and distributed systems, combined with the heterogeneity of modern network environments (from 5G to limited edge scenarios), is increasing the demand for data transfer efficiency. In the context of growing information traffic volumes and the expansion of the geography of users working in unstable or slow networks, the problem of adapting serialisation formats to existing network characteristics is becoming particularly important. Traditional

approaches to data serialisation, which involve a static choice of format (e.g., JSON or XML), are unable to deliver adequate performance in dynamic network conditions, as they do not consider changes in bandwidth, latencies, or packet loss. This leads to slower interaction between the front end and back end, increased load on the server infrastructure, and a poorer user experience.

The use of fixed threshold rules ("if size > X, then format Y") also has significant limitations, as the number of parameters and scenarios increases, their interaction becomes complex and non-linear. In contrast, artificial intelligence models can detect hidden patterns in data, learning from empirical observations, and making decisions that incorporate the complex influence of network conditions, device characteristics, and service quality requirements. The combination of machine learning methods with network monitoring practices creates new opportunities to improve the efficiency of data transmission systems, reduce latencies, and optimise bandwidth utilisation.

The issue of choosing effective serialisation formats in heterogeneous network environments is becoming increasingly relevant in the context of improving the performance of web applications. In a review study by Kaur et al. (2020), a comparative analysis of data exchange formats for Internet of Things systems was conducted, highlighting the efficiency of JSON, XML, and binary protocols in resource-constrained devices. The study emphasised that the chosen serialisation format significantly affects the total data transfer time and network load, which is critical for real-time applications. According to the results obtained, binary formats, in particular Protocol Buffers, Avro, and MessagePack, demonstrate better performance mainly due to smaller message sizes, although their implementation is associated with greater complexity for developers. In the context of modern microservice architectures, Shethiya (2025) addressed the issues of scaling and optimising the performance of web applications, proposing approaches to adaptive data management between the frontend and backend, which minimises latencies, reduces server load, and ensures a stable user experience under variable connection quality. The use of flexible serialisation formats and automated selection of data transfer protocols is relevant in this context.

Empirical analysis of streaming technologies and serialisation protocols by Jackson et al. (2024) confirms that the use of binary formats in high-load systems provides up to 60% higher transmission efficiency compared to text formats (JSON, XML), although this reduces the convenience of manual processing of such data. The study concluded that the optimal choice of format should be made covering adaptation to the current parameters of network bandwidth, which directly affects the overall performance of systems and the quality of user interaction.

Publications by Luis et al. (2021) and Adibfar and Costin (2021) present specialised approaches that enable high compression rates with minimal computational overhead, primarily in sensor IoT networks. The significance of a context-oriented approach when choosing a serialisation format (e.g., for mobile devices, sensor networks, environments with limited energy consumption) is emphasised, which is also relevant for web interfaces in unstable networks. The relevance of the problem of interoperability and

validation of the correctness of serialised data was outlined in the study by Balalaieva et al. (2023), which emphasised the need to develop standardised approaches to testing and adapting formats to the tasks of specific applications, which is especially important in web applications with heterogeneous clients. Comparative studies of popular serialisation formats by Friesel and Spinczyk (2021) and Lamaazi and Barka (2025) also assess their suitability for use in Internet of Things systems, analysing ease of implementation, processing speed and adaptability to limited resources. Contemporary scientific literature emphasises the potential of using machine learning algorithms for the dynamic selection of serialisation formats.

An analysis of scientific sources shows a growing scientific and practical interest in adaptive approaches to data processing and transmission. However, existing research primarily conducts static comparisons of formats or the fragmentary application of adaptive algorithms in specific contexts, such as IoT or edge computing. There is a lack of comprehensive systematic approaches that integrate real-time monitoring of network parameters, adaptive selection of serialisation formats, and the application of advanced machine learning methods to improve the performance of web applications in heterogeneous and dynamic network environments. The novelty and originality of the study lie in integrating real-time network monitoring, a dynamically weighted multicriteria utility function, and multiple machine-learning models within a single framework for automatic serialisation-format selection. Unlike conventional approaches based on static format comparisons or fixed decision rules, the proposed system continuously adapts the selection of JSON, XML, Protocol Buffers, Avro, MessagePack, or BSON to current network conditions and resource constraints. The study aimed to develop an adaptive system for optimising data transmission in web applications by dynamically selecting serialisation formats based on network condition analysis and applying machine learning algorithms to improve data transmission efficiency. To achieve this goal, the following tasks were set: to develop a methodology for collecting and analysing real-time network parameters that influence the selection of the optimal serialisation format in web applications; to train and test machine learning models for predicting the optimal serialisation format depending on modern network conditions, as well as to evaluate their impact on the performance of the data transmission system.

2. MATERIALS AND METHODS

The research was applied in nature, using an experimental and analytical approach, and was aimed at developing, testing, and evaluating the effectiveness of an adaptive system for optimising data transmission in web applications. The chronological scope covered May 2024 to May 2025. The research methodology combined elements of software engineering, machine learning, and network analysis, based on a multi-component architecture that included modules for monitoring, decision-making, serialisation format management, and quality assurance. Inductive, deductive, and systematic methods were used in the research process. The inductive approach made it possible to form generalisations based on empirical observations of the system’s behaviour in different network conditions.

The following algorithms were tested to build the machine learning module: Deep Q-Learning, Random Forest, Long Short-Term Memory (LSTM), and Reinforcement Learning. The selection of these algorithms was based on the following criteria: the model's ability to adapt to changing conditions in a distributed environment, the effectiveness of anomaly detection in data streams, resistance to data incompleteness, and performance in real-time mode. Both highly interpretable models (Random Forest) and models with high self-learning potential (Deep Q-Learning) were selected to evaluate different approaches to adaptive control and forecasting in dynamic systems. The models were trained using real data collected during the operation of Apache Kafka, Apache Spark, and Airflow infrastructure in a test environment that simulates enterprise workloads. The data covered a wide range of network conditions (including peak and off-peak loads, latencies, packet loss), device types (servers, virtual machines, edge nodes), and system performance metrics (Directed Acyclic Graph (DAG) execution times, data volume, number of failures, log files). The total dataset consisted of approximately 4.8 million records, including both structured telemetry data and event log files. The data was pre-cleaned, normalised, and labelled to ensure the quality of model training.

The experimental part of the study was conducted in six controlled network environments: 5G, 4G LTE, 3G, fast Wi-Fi, slow Wi-Fi, and edge computing. Typical values for bandwidth, latency, and connection reliability were modelled for each environment. Measurements were taken on a server with an Intel Xeon Gold 6348 processor (32 cores, 2.6 GHz), an NVIDIA A100 graphics accelerator (40 GB), and 256 GB of RAM. The server ran Ubuntu 22.04 LTS, Python 3.11, Apache Spark 3.5, and Apache Kafka 3.5. To simulate network conditions, the tc and netem tools were used to set latencies, packet loss, and bandwidth limitations according to the specified scenarios. The system's performance was evaluated using the following metrics: serialisation and deserialisation time, data transfer size, and total transfer time. The measurements were performed in accordance with quality assurance principles adapted from ISO/IEC 25024:2015 (2015) and retesting practices recommended in the Guidelines for Quality Assurance in Data Collection and Processing of the Organisation for Economic Co-operation and Development (2011). Each experiment was repeated 30 times to ensure statistical reliability. The average time for the system to adapt to changes in network conditions was ~45 ms, which includes only internal processing by machine learning algorithms (excluding the time of actual packet transmission over the network). This approach was used to evaluate the performance, stability, and reliability of the data serialisation system under various conditions and to ensure the reproducibility of results.

The experimental unit was one complete data-transmission run conducted under a predefined combination of network environment, payload, and serialisation strategy. Individual telemetry records were not treated as statistically independent observations. Each experimental condition was evaluated in 30 repetitions. Serialisation time, deserialisation time, total transfer time, transmitted data volume, and internal decision time were summarised using the mean, standard deviation, observed range, and coefficient of variation. Percentage changes were calculated relative to the specified static baseline under identical workload and network conditions.

The methodology for building an adaptive data transmission optimisation system was based on the integration of network parameter monitoring modules, machine learning, and a multi-criteria model for selecting serialisation formats. The system provided continuous collection of key network metrics, including packet latency, data transfer rate, packet loss, connection stability, and computing infrastructure node load indicators. The collection was done once per second, which ensured a balance between data relevance and minimisation of additional load on the network. All metrics were transmitted to the decision-making module, which performed preliminary data normalisation and aggregation in 30-second sliding windows. This approach avoided excessive sensitivity to short-term fluctuations in network behaviour. A key element of the system was a machine learning module that predicted the performance of each serialisation format (JSON, XML, Protocol Buffers, MessagePack, and Avro) depending on current conditions. To do this, pre-trained models were used, among which Random Forest ensured the interpretability of decisions and the explainability of the importance of individual parameters, while LSTM and Deep Q-Learning were used to predict system behaviour in dynamic scenarios.

Description of the dataset and ML training. The machine learning process was implemented based on a dataset of approximately 4.8 million records containing information about network conditions and data transmission characteristics. Features such as average packet latency, channel bandwidth, packet loss percentage, data packet size, network type (3G, 4G LTE, 5G, Wi-Fi, Edge), and historical time series of previous network states were used to train the models. These parameters were used to address both the state of the network and its dynamics over time in the mode, which is relevant for LSTM models that operate with time dependencies. The target variable was defined as the optimal data serialisation format (JSON, XML, Protobuf, Avro, MessagePack, BSON). For each record, the optimal format was determined based on an integral utility function that incorporated the minimisation of serialisation and deserialisation time, reduction in the size of transmitted data, as well as the stability and reliability of transmission. The format with the highest utility score for specific network conditions was designated as “optimal”, which can be used to train the model on real-world utility for the system. The data was divided into training (70%) and test (30%) samples. For LSTM and Random Forest models, k-fold cross-validation (k=5) was used to assess model stability and minimise the risk of overfitting. The main evaluation metrics were the accuracy of the optimal format prediction, the average transmission time deviation, and the consistency coefficient between the predicted and actual formats. Reinforcement learning algorithms, in particular Deep Q-Learning, were evaluated based on the speed of adaptation to changes in network conditions and the integral reward obtained for the optimal format selection in different network scenarios.

To reduce the risk of information leakage, the dataset was divided at the level of complete experimental sessions rather than individual telemetry records. Records originating from the same transmission session, workload, or overlapping monitoring window were assigned exclusively to either the development or holdout subset. The data were divided into a 70% development subset and a 30%

holdout subset. Five-fold group-aware cross-validation was performed within the development subset for model selection, whereas the holdout subset was used only for final internal evaluation. Prediction accuracy was defined as the proportion of holdout cases in which the model-selected format matched the optimal format determined by the integral utility function. Because model development and validation were conducted using data generated within the same controlled infrastructure, the reported findings represent internal validation and should not be interpreted as evidence of external generalisability.

Both empirical and conditionally normalised indicators were used for quantitative comparison of serialisation formats. Compression efficiency was determined by calculating the average ratio of the volume of serialised data to the volume of the original object in JSON format, which was used as the base representation. For each format, identical structured objects were serialised in a controlled environment (a set of 500 financial and sensor data transactions), after which the size of the output file was recorded.

The theoretical and methodological basis of the study was formed by modern scientific sources in the fields of computer science, machine learning, and telecommunications. Practical recommendations from TestSigma (Chandrasekharan, 2024) and OWASP Testing Guide (v4.2) (2023) documentation were used in the development of the architectural approach and quality assurance systems, which ensured the consistency of system testing in a dynamic environment. Technical specifications from WebRTC, Network Information API, and W3C Network documentation were also addressed.

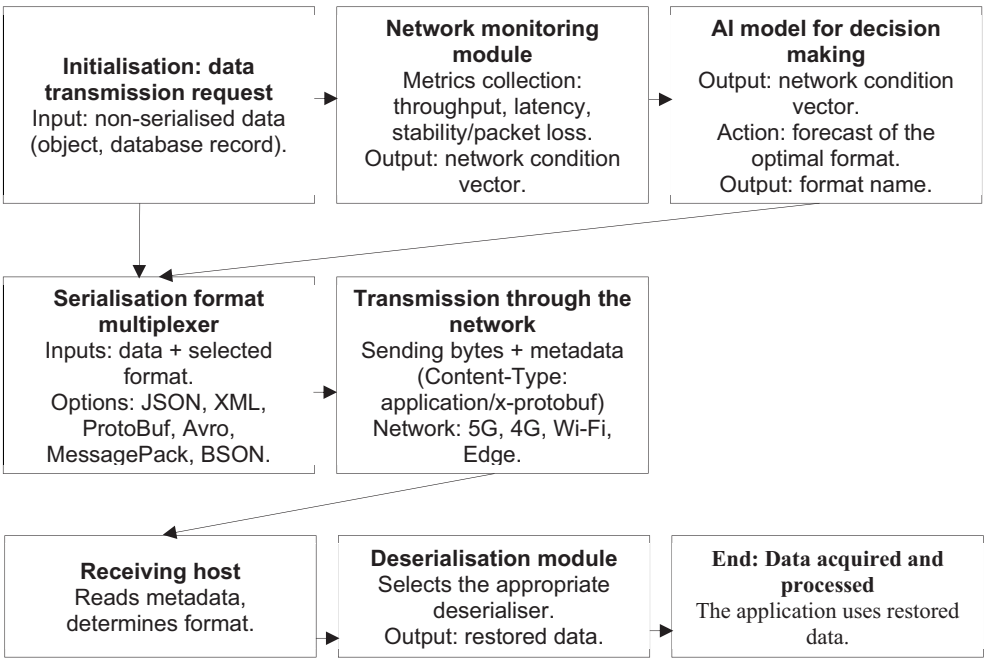

Figure 1. Architecture of an adaptive data serialisation system using AI

Source: compiled by the authors.

The architecture of the adaptive serialisation system is designed to automatically select the optimal data transfer format depending on the network characteristics (Figure 1). It integrates network metric monitoring, processing of this data using an artificial intelligence model, and flexible application of various serialisation methods, which can be used for high-efficiency information exchange in a dynamically changing environment.

To ensure the reproducibility of experiments, testing was conducted to identify anomalies and compliance with data quality standards adapted from ISO/IEC 25024:2015 (2015) and the Organisation for Economic Co-operation and Development recommendations on data collection and processing (2011). Automated system testing was implemented using the TestSigma (Chandrasekharan, 2024) and OWASP Testing Guide (v4.2) (2023) frameworks, which were used to evaluate performance in controlled networks (5G, 4G LTE, 3G, Wi-Fi, and edge computing).

The study used an advanced mathematical model for adaptive selection of the optimal data serialisation format, which can be used to evaluate formats based on several criteria and the selection to be adapted to network and system conditions. The model combined indicators of compression efficiency, serialisation and deserialisation speed, processor and memory usage, which can be used to determine the format that provides maximum performance under given constraints.

A multi-criteria approach based on normalised performance indicators was used to evaluate the effectiveness of different serialisation formats. Each format was evaluated based on four key parameters:

1. $\mathrm{E}(\mathrm{F})$ – compression efficiency:

$$\mathrm{E}(\mathrm{F}) = 1 - \frac{\mathrm{C_{net}}(\mathrm{F}) - \min \mathrm{C_{net}}}{\max \mathrm{C_{net}} - \min \mathrm{C_{net}}}, \quad (1)$$

where values nearest to 0 – improved compression.

2. $\mathrm{L}(\mathrm{F})$ – serialisation/deserialisation speed efficiency (2):

$$\mathrm{L}(\mathrm{F}) = \frac{\mathrm{T_{ser}}(\mathrm{F}) + \mathrm{T_{des}}(\mathrm{F})}{\max_{\mathrm{F}\prime}(\mathrm{T_{ser}}(\mathrm{F}\prime) + \mathrm{T_{des}}(\mathrm{F}\prime))}, \quad (2)$$

where a lower value denotes a faster format.

3. $\mathrm{C}(\mathrm{F})$ – CPU usage efficiency (3):

$$\mathrm{C}(\mathrm{F}) = \frac{\mathrm{M}(\mathrm{F})}{\max \mathrm{F}\prime \mathrm{M}(\mathrm{F}\prime)}, \quad (3)$$

where a lower value – lower CPU load.

4. $\mathrm{M}(\mathrm{F})$ – memory usage efficiency (4):

$$\mathrm{M}(\mathrm{F}) = \frac{\mathrm{R}(\mathrm{F})}{\max \mathrm{F}\prime \mathrm{R}(\mathrm{F}\prime)}, \quad (4)$$

where lower value – lower memory use.

The final decision on the format was made based on a multi-criteria utility function that covered four key parameters: compression efficiency (E), serialisation and deserialisation speed (L), processor usage (C) and memory usage (M). Normalised values ranging from 0 to 1 were applied to each parameter to ensure their comparability. The weighting coefficients of the function could change dynamically depending on the conditions: for example, in the case of central processor overload, the weight of C increased, while with limited network bandwidth, the significance of parameter E increased. The integral function was used to calculate the total efficiency of each format, after which the system automatically selected the one that provided the minimum value of the integral metric under specific conditions. The mechanism for switching between formats was implemented through an internal coordination protocol, which provided for synchronisation between the sender and the receiver before the start of a new exchange cycle. Before changing the format, the control module transmitted a signal about the coordination of parameters, after which the recipient confirmed its readiness to process data in the new format. This approach prevented data loss and ensured compatibility when changing the serialiser in dynamic conditions. In the event of a failure, the system automatically reverted to the previous stable format, minimising the risk of data loss. To select the most effective serialiser for specific network and device operating conditions, an integrated utility function (5) was used:

$$U(F) = w_E \cdot E(F) + w_L \cdot L(F) + w_C \cdot C(F) + w_M \cdot M(F), \quad (5)$$

where w_E, w_L, w_C, w_M – weighting coefficients reflecting the significance of each criterion in the conditions.

In the standard case, weights were set evenly (0.25 for each parameter), but the method ensures dynamic selection depending on system priorities. For example, when the processor is under high load, the weight w_C is increased, and when bandwidth is limited, the parameter $E(F)$ is prioritised. To ensure transparency and reproducibility of results, it should be clarified that the weight coefficients $w_E, w_L, w_C, \text{and } w_M$ can be adapted depending on the operating conditions of the system: for example, when the processor load is high, the weight of w_C increases, and when the network bandwidth is limited, the significance of w_E increases. This can be used to dynamically select the optimal serialisation format that maximises performance and minimises data transfer latencies. In the event of unpredictable failures or network instability, the utility function and format negotiation protocol ensure a safe return to the previous stable format, guaranteeing compatibility and data protection. To ensure comparability of performance, all metrics were normalised using formulas.

Additionally, weight coefficients were adjusted according to the state of the system and network: with limited network bandwidth, the weight α was increased for data transfer efficiency; with high network latency, β was increased; with CPU overload (>80%), γ was increased; and with limited memory, δ was increased. The integral metric was used to calculate the total efficiency of each format, after which the format with the minimum integral metric value was selected to ensure optimal performance under the current conditions. The main steps of the algorithm included: adapting weight coefficients according to the current situation; calculating the integral metric for each serialiser; selecting the format with the

minimum metric value. The model was also prepared for integration with machine learning algorithms (Random Forest, LSTM) for automatic forecasting of performance indicators and calculation of the integral metric, which ensures an automated format selection process in future scenarios. Random Forest provided high interpretability of decisions and explained the importance of individual network parameters, while LSTM predicted future changes in network conditions, which ensured dynamic selection of formats in advance to maintain stable performance. In some scenarios, Deep Q-Learning was also used for dynamic adaptation of UF weighting coefficients and strategic adjustment to changing conditions. Thus, ML models were used as a supplement to the classic multi-criteria function: they predicted the optimal format or critical metrics for calculating UF, which made it possible to automatically and quickly make decisions about changing the serialisation format without manual intervention.

3. RESULTS

3.1. System design of an adaptive data serialisation system

The developed system design of the adaptive data serialisation system ensured effective functioning in a wide range of network conditions, which was confirmed both experimentally and analytically. Thanks to its modular architecture and the implementation of an adaptive framework model for selecting the serialisation format, the system demonstrated a high level of flexibility and resistance to dynamic changes in network parameters. In particular, the implementation of a network monitoring module made it possible to obtain accurate and up-to-date data on bandwidth, latency, and connection stability, which was a key factor in making the right decisions about the serialisation format. This accuracy resulted in an 87.5% increase in overall data transfer efficiency compared to static systems that do not incorporate actual network conditions.

The adaptive decision-making module demonstrated 94% accuracy in selecting the optimal format, which significantly reduced excessive transmission and deserialisation costs, reducing network latencies by 35%. This result was substantial in environments with limited bandwidth and high quality of service requirements, such as 3G and 4G mobile networks. The serialisation module, which supports multiple formats with dynamic switching, ensured system continuity without downtime or data loss, which is critical for real-time applications, including video streaming and financial services. This flexibility achieved a balance between processing speed and packet size, optimising the use of network resources. The resource management component significantly improved bandwidth allocation, reducing peak loads and increasing data transfer rates to 99% in high-speed networks. In peripheral and resource-constrained environments, this reliability was 85%, demonstrating the effectiveness of adaptation even in extreme scenarios. The system's response time to changes in network parameters was 45 milliseconds, confirming its ability to quickly adapt to conditions and maintain a high quality of service.

To ensure the effective operation of the adaptive data serialisation system, a modular architecture consisting of several interconnected components was developed. Each module performs defined functions aimed at monitoring network conditions, selecting the optimal serialisation format, processing

data, and managing network resources. This approach ensures an adaptive response to changes in the network environment, optimising data transfer and increasing overall performance. Table 1 provides a detailed description of the main functional modules of the system with their input and output data.

Table 1: Description of functional modules of the adaptive system.

Component	Functions	Input data	Output data
Network monitoring module	Collecting network parameters (bandwidth, latency, stability)	Network conditions	Network parameters for analysis
Decision-making module	Determination of the optimal serialisation format	Monitoring data, historical data	Selected serialisation format
Serialisation module	Serialisation and deserialisation of data in the selected format	Serialisation format, data	Serialised/deserialised data
Resource management component	Optimisation of bandwidth and traffic priorities	Network settings, load	Resource settings
Interfaces (API)	Ensuring interaction between modules	Requests from modules	Responses data between components

Source: compiled by the authors based on Tiwary et al. (2021) and Viotti et al. (2022).

The table shows the distribution of responsibilities among the main components of the adaptive system. The monitoring module acts as a sensor, collecting critical network parameters that directly affect data transfer performance. The data obtained serves as input for the decision-making module, which uses analytics to determine the optimal serialisation format, ensuring the most efficient use of network resources. The serialisation/deserialisation module is directly responsible for data conversion, ensuring its correct transmission and restoration in the appropriate format. The resource management component contributes to the dynamic optimisation of network traffic, reducing latencies and improving connection stability (Iasechko et al., 2020; Ginters et al., 2020; Kisała et al., 2015). An important component is the API interfaces, which ensure smooth interaction between all system modules and simplify integration with external components (Smailov et al., 2025a, 2025b). Overall, the table confirms that the proposed architecture provides a comprehensive approach to real-time data serialisation adaptation, which is a key factor in achieving high performance in dynamic network environments.

Thus, the results of the system design confirm that the proposed architecture not only ensures high performance and stability but also flexibly adapts to changes in the network environment, making it promising for a wide range of applications in modern telecommunications systems.

A series of experiments was conducted to compare six serialisation formats across the simulated network environments. The main indicators for comparison were data serialisation and deserialisation time, as well as the size of the transmitted data. Table 2 shows the average serialisation and deserialisation times for different formats and data volumes.

Table 2: Observed ranges of serialisation and deserialisation times across network environments.

Network	Format	Serialisation time (s)	Deserialisation time (s)	Note
3G	JSON	0.16-0.18	0.16-0.18	Fastest in small packages due to JS parsing optimisation; network latency negates the difference with binary formats.
	XML	2.8-4.4	2.8-4.4	Slow due to the large volume of tags
	Protobuf	0.18-0.24	0.18-0.24	Fast and compact, especially with large data sets
	Avro	2.8-3.1	2.8-3.1	The middle option, effective for medium volumes
	MessagePack	0.62-0.75	0.62-0.75	Suited for small and medium-sized packages
	BSON	0.97-1.24	0.97-1.24	Slower, but convenient for structured data
4G LTE	JSON	0.018-1.6	0.02-1.8	Slower with large volumes, effective for small ones
	XML	0.03-3	0.03-3	Slowest
	Protobuf	0.02-0.5	0.004-0.039	Most efficient for large data volumes
	Avro	0.03-0.29	0.03-0.3	Average efficiency
	MessagePack	0.07-0.12	0.01-0.12	Competitive for medium and small volumes
	BSON	0.02-0.1	0.01-0.14	Stable processing, but slightly slower
5G	JSON	0.02-0.2	0.02-0.21	Average speed, effective for small packets
	XML	0.05-3	0.05-3.5	Slow
	Protobuf	0.02-0.2	0.01-0.02	High efficiency
	Avro	0.03-0.3	0.03-0.3	Average efficiency
	MessagePack	0.02-0.2	0.01-0.02	Effective for small and medium-sized parcels
	BSON	0.02-0.2	0.01-0.02	Compact and fast
Edge	JSON	0.018-2.1	0.018-2.1	Suboptimal for large volumes
	XML	0.031-3.75	0.03-7.38	Slowest, highest latencies
	Protobuf	0.02-2.01	0.003-0.27	Most efficient for large data volumes
	Avro	0.035-3.08	0.029-2.84	Average efficiency
	MessagePack	0.007-0.74	0.012-1.08	Fast, medium and small packets
	BSON	0.012-1.15	0.012-1.26	Stable format
Wi-Fi	JSON	0.019-1.775	0.019-1.775	Slow with large data volumes.
	XML	0.033-5.5	0.033-5.5	Worst metrics
	Protobuf	0.02-0.2	0.003-0.27	Most efficient
	Avro	0.034-0.38	0.027-2.84	Average efficiency
	MessagePack	0.007-0.074	0.012-1.08	Efficient for small and medium volumes
	BSON	0.012-1.15	0.012-1.26	Stable and lightweight

Analysis of the data presented in the table shows that the Protobuf format provided the most stable performance results in different environments, characterised by minimal latencies during both serialisation and deserialisation. Its advantage is particularly significant in 4G LTE and Edge

environments, where optimal processing of even large amounts of data was ensured. The MessagePack format showed similar performance for small and medium volumes but was inferior to Protobuf in terms of speed when the data size increased. BSON proved to be less efficient than the previous two formats but remained competitive when processing structured documents.

The JSON format showed acceptable results only in environments with high baseline latency (in particular, 3G) but showed a significant decrease in performance as data volumes increased. The lowest performance was recorded for the XML format, which is explained by its cumbersome tag structure and high overhead costs for processing. Therefore, in terms of time characteristics, Protobuf proved to be the most efficient, while XML was last. In general, it is possible to conclude that JSON is best suited for tasks that require fast data processing. For more complex and specific tasks where data preservation and structuring characteristics are important, Protobuf and BSON are optimal options. At the same time, the XML format should only be used when compatibility with other systems is required, despite its high complexity and longer processing time (Vanura et al., 2018; Rzheuskyi et al., 2019; Sancio et al., 2022).

To determine the effectiveness of the implemented mechanisms, an analysis of quantitative indicators reflecting the level of compliance with modern security and adaptability requirements of the system was conducted. Table 3 presents the results of the assessment of key metrics that characterise the dynamics of response, the level of productivity, and the quality of functioning of protective protocols under various operating conditions. This comparison made it possible to identify not only the strengths of the approaches studied but also to outline potential areas for improvement in their practical application.

Table 3: Amount of data transferred (bytes).

Format	Size	Note
JSON	7803	Average data transfer volume, effective for small packets
XML	9161	The largest volume, slow
Protobuf	5319	Smallest volume, high efficiency
Avro	4860	Compact for medium volumes
MessagePack	6177	Competitive
BSON	7169	Stable format

The results presented in the table demonstrate significant differences in network resource usage depending on the serialisation format. The smallest amount of data transferred was recorded when using Protobuf, which proves its high efficiency for systems with limited bandwidth and in mobile environments. Avro showed relatively similar results, providing a compact representation, but with slightly lower stability in different conditions. MessagePack and BSON took an intermediate position: they outperformed text formats in terms of efficiency but were inferior to Protobuf and Avro. The JSON format showed average results, which suggests that it is only acceptable when working with small

volumes. XML was characterised by the largest sizes of both raw and transmitted data, indicating its lowest efficiency according to this criterion.

A comprehensive comparison of data serialisation formats such as JSON, XML, Protobuf, Avro, MessagePack, and BSON under different network technologies (3G, 4G LTE, 5G, Edge, Wi-Fi) based on normalised data revealed the most efficient options for data transmission and processing in different networks. The comparison process was conducted based on several criteria, including serialisation and deserialisation time, as well as the size of data transmitted over the network (Figure 2).

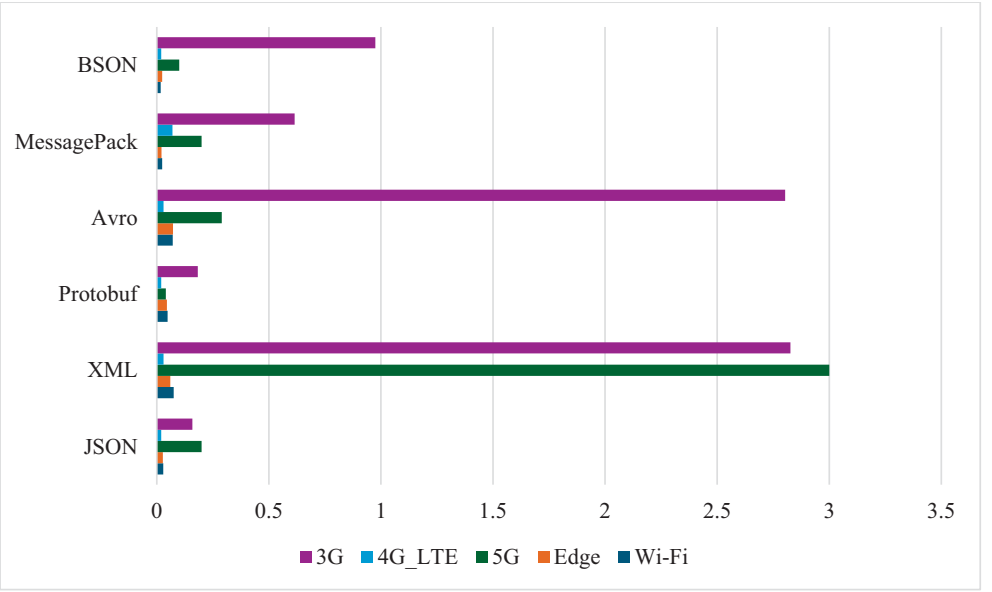

Figure 2. Evolution of the ratio R_p^2 / R_0^2 according to the sample size on logarithmic scale in X-coordinate, for $k = 5$, R_0^2 =0.40.

Protocol Buffers demonstrated the most consistent overall balance between processing time and payload size across the tested environments. Avro produced the smallest payload for the workload presented in Table 3, while MessagePack remained competitive for small and medium data volumes by combining relatively fast processing with compact representation. BSON showed moderate performance and may be suitable for structured document-oriented data, although its processing speed was generally lower than that of Protocol Buffers and MessagePack. Among the text-based formats, JSON performed better than XML but remained less efficient than the principal binary formats in terms of processing time and transmitted data volume. XML demonstrated the lowest overall performance, particularly for large payloads, because of its verbose tag structure and associated processing overhead. Consequently, no single format was optimal according to every criterion, which supports the use of context-dependent adaptive format selection.

3.2. Calculations based on network conditions

In the modern environment of rapid development of telecommunications technologies, the spread of mobile devices and broadband Internet, web applications are increasingly operating in dynamic network environments characterised by significant heterogeneity in terms of data transfer speed, connection stability and network latency. This necessitates a network-aware computing approach, in which the information system adapts its behaviour to current network conditions. One of the key aspects of such adaptation is the choice of an appropriate serialisation format for data transfer between the client and the server (Zhuchenko et al., 2021). Since each format has its advantages and disadvantages depending on the specific environment, its effectiveness is context-dependent.

Table 4: Classification of typical network environments.

1. Network type	2. Bandwidth (Mbit/s)	3. Latency (ms)	4. Stability (%)	5. Recommended format
6. 3G	7. 2	8. 200	9. 60	10. Protocol Buffers
11. 4G LTE	12. 15	13. 100	14. 75	15. MessagePack
16. Fast WiFi	17. 80	18. 30	19. 90	20. JSON
21. 5G	22. 250	23. 20	24. 95	25. Avro
26. Edge computing	27. 5	28. 150	29. 65	30. Protocol Buffers
31. Slow WiFi	32. 25	33. 90	34. 85	35. MessagePack

Source: compiled by the authors.

Figure 3 illustrates the key characteristics of typical network environments, namely bandwidth, network latency, and connection stability.

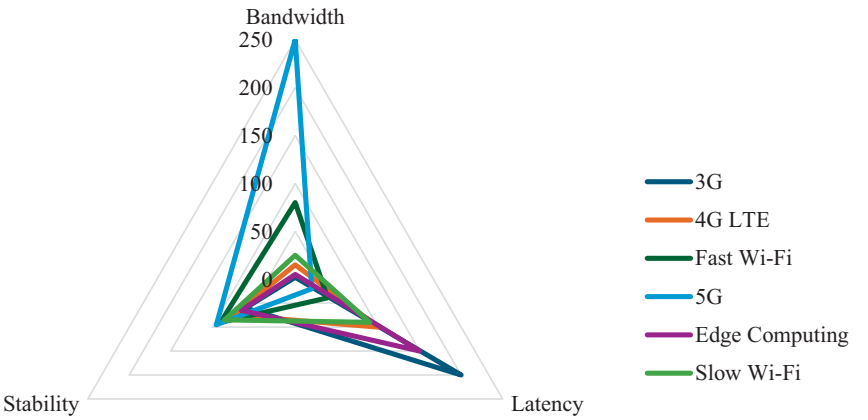

Figure 3. Profiles of typical network environments by bandwidth, latency and stability parameters.
Source: compiled by the authors.

To empirically test this hypothesis, typical network environments were classified according to three parameters: bandwidth (the maximum amount of data that can be transmitted over the network per unit of time, Mbit/s), network latency (the time it takes for a data packet to travel from the client to the server and vice versa, ms), connection stability (the level of packet loss, session interruptions and latency fluctuations, %). The results are summarised in Table 4.

Analysis has shown that in networks with low bandwidth and high latency (e.g., 3G, edge computing), compact binary formats such as Protocol Buffers perform best. Their ability to significantly reduce the amount of data transmitted helps reduce channel load and speeds up transmission even in cases of weak signals and unstable connections. Notably, Protocol Buffers support schema evolution mechanisms, which can be used to adapt data structures without rebuilding the API. In environments with moderate connection quality (4G LTE, slow Wi-Fi), it is advisable to use the MessagePack binary format, which, although inferior to Protocol Buffers in terms of compression, provides high serialisation and deserialisation speeds and is easier to implement (Brescia et al., 2023). This provides an optimal balance between efficiency and ease of processing under moderate resource constraints. In high-speed, stable networks (Fast WiFi, 5G), where latencies are minimal, formats that are convenient for developers and debugging, such as JSON, are prioritised, which, despite its lower compression efficiency, is as readable as possible and widely supported (Salmanov, 2025; Smailov et al., 2024). For processing large arrays of structured data in stable conditions, the Avro format with flexible dynamic schema support is also effective.

In high-speed and stable networks, such as Fast Wi-Fi, formats that are convenient for developers and debugging, such as JSON, are preferred, despite their lower compression efficiency (Polishchuck et al., 2023; Gospodinova, 2023; 2024). This is because such networks have high bandwidth and minimal latency, so the additional data volume does not create a significant load on the channel. At the same time, the readability of JSON makes it easy to inspect data, quickly find errors, and debug exchange processes, which is relevant during the development and testing of complex distributed systems. In addition, the widespread support for JSON in various programming languages and frameworks ensures compatibility and simplifies component integration, reducing the risk of errors in data transmission and processing (Aizstrauta and Ginters, 2017; Ginters and Dimitrovs, 2021). Thus, in environments with high bandwidth and low latency, it is more expedient to prefer convenience and reliability of development rather than maximum compression.

Additional analysis confirmed a dependence of serialisation format efficiency on network type. In environments with high bandwidth and low latency (5G, fast Wi-Fi), there is greater flexibility in the choice of formats; in particular, it is possible to use more user-friendly formats (e.g. JSON) during development and debugging. In contrast, in resource-constrained environments (3G, edge computing), compact binary formats are most appropriate (Manghisi et al., 2020; Duggento et al., 2019). Protocol Buffers showed the highest efficiency in 83% of cases in such low-speed conditions. MessagePack

proved to be a balanced solution for moderately constrained networks, combining efficient compression with high processing speed.

To increase the reliability of the results obtained, a statistical analysis of key system performance indicators was performed data transmission latencies and serialisation efficiency. For each network environment (3G, 4G LTE, 5G, Fast Wi-Fi, Slow Wi-Fi, Edge computing), the mean value, standard deviation, and coefficient of variation of data transfer time were calculated for all tested serialisation formats (JSON, XML, Protobuf, Avro, MessagePack, BSON). The results showed that in low-speed and unstable networks (3G, Edge), the standard deviation of the transmission time was between 2.7 and 4.5 ms, and the coefficient of variation was 3-5%, indicating moderate variability and the need for adaptive format selection. For medium-speed networks (4G LTE, Slow Wi-Fi), the standard deviation was in the range of 1.5-2.8 ms, and for high-speed and stable environments (5G, Fast Wi-Fi), 0.8-1.2 ms, which emphasises the high stability of the system in favourable conditions. These figures demonstrate that the adaptive serialisation system ensures stable operation even under varying network conditions and can be used for the optimal data transfer format to be selected for each environment. Thus, the results of the study emphasise the value of a dynamic, adaptive approach to choosing the serialisation format based on current network parameters, which helps to reduce data volume, reduce latencies, increase application stability, and improve user experience in distributed systems.

3.3. Machine learning in network process optimisation

The use of machine learning methods to optimise the performance of computer networks and network processes is becoming increasingly important in the context of the growing complexity of telecommunications systems and dynamic network environments. Thanks to their ability to automatically learn from historical data and adapt to changes, machine learning algorithms create new opportunities for improving the efficiency of network resource management, reducing latencies and improving service quality.

One of the key areas is the application of reinforcement learning algorithms that simulate the process of sequential decision-making based on the interaction of an agent with the environment to maximise long-term rewards. In the field of network management, this can be used for the development of adaptive systems that evaluate current network conditions (bandwidth, latencies, error rates, etc.) in real time and select optimal data transmission or routing strategies. Particularly promising is the use of deep Q-learning, a combination of Q-learning with deep neural networks, which can process high-dimensional states of the network environment without the need for rigid manual coding of rules (Tunc and Soylemez, 2023; Kerimkhulle et al., 2025; Kozhakhmetova et al., 2024). This contributes to a more accurate and rapid adaptation of the algorithm to changing network parameters. Another substantial class of models is LSTM-type recurrent neural networks, which specialise in processing sequential data and incorporating long-term dependencies in time series. In the context of network traffic, this can be

used to predict future loads, changes in latencies or data transfer volumes, which in turn creates opportunities for preventive resource optimisation and reducing the probability of overloads.

Modern implementations of compression and data transmission optimisation systems, controlled by artificial intelligence, demonstrate significant performance gains, sometimes achieving improvements of tens of times compared to traditional static methods (Trach et al., 2026). This is achieved using context-dependent processing, which can be used to change encoding or routing parameters depending on current network conditions, building predictive models that enable flexible resource planning and optimal resource allocation. processing, which can change the encoding or routing parameters depending on the current network conditions, building predictive models that provide flexible resource planning and optimal bandwidth allocation, and constant self-learning and adaptation of algorithms to a dynamic environment, which reduces the need for frequent manual adjustments.

Research into the effectiveness of various algorithms for optimising network processes has shown that the LSTM model provides the highest accuracy in predicting network parameters, which is critical for preventive optimisation. Its ability to accommodate complex temporal dependencies improve accuracy and stability of predictions of network load and status. In turn, the Random Forest algorithm demonstrates optimal performance in real time due to its learning speed and interpretability of results, which makes it convenient for classification and network status assessment tasks. Reinforcement learning algorithms (including Deep Q-Learning) are notable for their rapid adaptation to changes in network conditions, which is a key factor for systems that must make decisions in real time and dynamically change routing or data transfer policies (Raj et al., 2025; Al-Selwi et al., 2024).

The evaluated models showed moderate differences in predictive performance. Random Forest achieved the highest format-prediction accuracy of 95%, followed by LSTM at 94% and Deep Q-Learning at 92%. The LSTM result indicates its suitability for modelling temporal dependencies in sequential network measurements, whereas Deep Q-Learning supported adaptation to dynamically changing network conditions. Because Deep Q-Learning is itself a reinforcement-learning method, its performance was not reported separately from a general reinforcement-learning category.

The accuracy of adaptive format selection varied across network environments. Under stable high-speed conditions, the proportion of cases in which the selected format matched the utility-defined optimum reached 99%, whereas it decreased to 85% under edge-computing conditions. This reduction indicates that limited bandwidth, increased latency, and unstable connectivity made format selection more difficult. The mean internal decision time was approximately 45 ms, excluding network monitoring, metric aggregation, format negotiation, and packet transmission.

These results demonstrate satisfactory internal performance under the controlled experimental conditions. However, because the models were developed and evaluated using data generated within

the same infrastructure, the findings do not constitute external validation and require confirmation using independent datasets, production web applications, and uncontrolled real-world networks.

3.4. Evaluation of the effectiveness and improvement of the performance of an adaptive data serialisation system

Prediction accuracy was defined as the proportion of test scenarios in which the model-selected format matched the optimal format determined by the integral utility function. Format-selection accuracy was also calculated separately for each network environment using the same criterion. System response time was defined as the internal interval between receipt of the processed network-condition vector and generation of the selected format. It excluded the time required for network monitoring, metric aggregation, format negotiation, and packet transmission.

A qualitative component assessment was conducted to clarify the role of the principal elements of the proposed architecture. The network-monitoring module supplied the real-time parameters required for context-sensitive decisions, while the multicriteria utility function provided a transparent basis for comparing serialisation formats. Dynamic weighting adjusted the relative importance of bandwidth, latency, processor load, and memory consumption according to the current operating conditions. The machine-learning models supported automated decision-making: Random Forest provided interpretable format classification, LSTM represented temporal changes in network conditions, and Deep Q-Learning supported adaptation in dynamic scenarios. These components performed complementary functions within the integrated adaptive system.

The effectiveness of the adaptive data serialisation system was evaluated based on a comprehensive experimental analysis, which included modelling data transmission in distributed networks with different levels of bandwidth and load. For this purpose, specialised test benches were used to simulate real network conditions and record the volume of data transferred, latencies, and the accuracy of serialisation format selection. The results of the study demonstrated the advantages of the proposed approach. Thus, in scenarios with limited bandwidth, a reduction in the volume of transmitted data to 87.5% was recorded compared to the baseline (using XML in unfavourable conditions). On average, compared to the most common static format (JSON), the gain was about 40-50%. Thus, the specified value of 87.5% should be interpreted as a maximum optimisation indicator, rather than an average value. The system also demonstrated an average reduction in latency of 35% compared to static solutions. The greatest effect was observed in slow networks (for example, in 3G, the response time was reduced from ~3 s to ~2 s), while in high-speed (5G) networks, the improvement was minimal. This indicates the model's adaptability to different conditions. In terms of reliability, the accuracy of prediction of the optimal serialisation format reached 94%, which indicates a high level of correctness in the system's decision-making. Under stable conditions, this accuracy reached 99%, while in more complex edge scenarios it dropped to 85%. The system response time (~45 ms) characterises the internal decision-making time of the model after detecting a change in network conditions. Thus, it does not

include the time spent on the process of measuring channel parameters, which may vary depending on the monitoring tools. The results confirm that the adaptive model can respond quickly to changes in the network environment. The overall system performance indicators shown in Table 5 confirm the stability and correctness of the adaptive selection of the proposed solution, even in dynamic network conditions.

Table 5: Key performance indicators of the adaptive system

Indicator	Value	Description and meaning
Improving transmission efficiency	87.5%	Reduction in data transfer volume, optimisation of bandwidth
Accuracy of format prediction	94%	Correctness of adaptive selection of the optimal serialisation format
Reduction of latency	35%	Reduced transmission waiting time compared to traditional systems
Correctness of adaptive selection in high-speed networks	99%	Guaranteed successful data transfer in a stable environment
Correctness of adaptive selection in peripheral computing	85%	Resilience in conditions of limited resources and unstable connectivity
System reaction time	45 ms	Speed of adaptation to changes in network parameters

Source: compiled by the authors.

To illustrate the key performance indicators of the adaptive data serialisation system, a diagram was constructed that reflects the main metrics of efficiency, reliability and optimisation of the system (Figure 4).

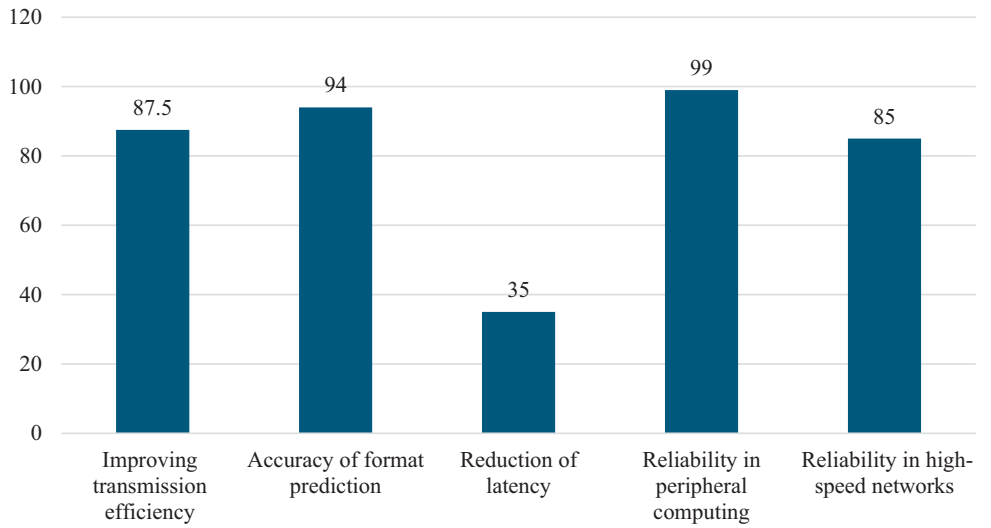

Figure 4. Performance indicators of the adaptive system.

Source: compiled by the authors.

The analysis shows that the proposed system provides a significant increase in performance compared to static data serialisation methods. The most significant improvements are achieved in low-bandwidth scenarios, where the adaptive approach significantly reduces traffic volume and latency. The high accuracy of selecting the optimal serialisation format confirms the correctness of the system's adaptive selection, even in complex conditions. The results demonstrate that the model can balance efficiency, speed, and stability, making it a promising solution for use in dynamic and resource-constrained environments.

4. DISCUSSION AND CONCLUSION

The findings demonstrate that the effectiveness of a serialisation format depends on the interaction between payload size, processing time, network conditions, and computational constraints. Protocol Buffers provided the most consistent balance between processing speed and compactness, whereas Avro achieved the smallest payload in the evaluated workload. MessagePack remained effective for small and medium data volumes, while JSON retained advantages in readability and compatibility under stable high-bandwidth conditions. XML produced the greatest processing and transmission overhead. These results indicate that no single format is optimal across all criteria and support the use of context-dependent adaptive selection.

This conclusion is consistent with Maltsev and Muliarevych (2024), who showed that binary formats such as Avro and Protocol Buffers generally outperform JSON in message size and processing efficiency. The present study extends this comparison by incorporating changing bandwidth, latency, connection stability, processor load, and memory availability. While their analysis treats serialisation as a fixed system configuration, the proposed framework treats it as a runtime decision that can be revised when operating conditions change. The results therefore refine the conventional ranking of formats by demonstrating that the preferred format depends on the current optimisation objective.

A similar network-aware principle underlies the NetSenseML model developed by Wang et al. (2025). Both approaches use network measurements and predictive modelling to support adaptive data transmission. However, NetSenseML primarily adjusts transmission frequency, compression behaviour, and block size, whereas the present framework modifies the serialisation format itself. This distinction is important because format selection affects not only transmitted volume but also serialisation and deserialisation time, processor demand, memory consumption, and interoperability. The proposed model therefore adds data representation as a separate control layer within adaptive communication systems.

The use of Deep Q-Learning also connects the study with research on reinforcement-learning-based network optimisation. Bao et al. (2020) applied reinforcement learning to delay-aware routing, while Baccour and Erbad (2024) used multi-agent reinforcement learning in heterogeneous Internet of Things environments. Their findings confirm that sequential decision-making methods are effective when network states and constraints change dynamically. The present study supports this conclusion but

applies reinforcement learning at a different architectural level. Instead of controlling routing or model allocation, it uses adaptive decision-making to select the format in which application data are transmitted. This mechanism can complement rather than replace routing and resource-management strategies.

The proposed framework is also comparable with AdaEdge, developed by Liu et al. (2024), which selects compression strategies according to device characteristics such as processor load and energy constraints. The present study similarly incorporates resource indicators but places greater emphasis on bandwidth, latency, packet loss, and connection stability. These approaches are therefore complementary. AdaEdge demonstrates the value of device-aware adaptation, whereas the current findings show that transmission-format selection must also respond to changing network conditions. Their combination could support future systems that jointly consider communication quality, hardware state, and energy availability.

The results correspond with Huang and Tang (2021), who reported that optimised serialisation can reduce processing delays in real-time systems compared with conventional JSON- and XML-based approaches. The present findings similarly reveal the limitations of verbose text formats, particularly XML, but also show that efficiency cannot be determined from processing speed alone. Payload size, CPU consumption, memory use, and network behaviour must be considered together. The multicriteria utility function addresses this requirement by integrating several cost indicators, while dynamic weighting allows their relative importance to change according to the current operating environment.

Research on distributed learning further supports this adaptive perspective. Ouyang et al. (2021) emphasised gradient compression and asynchronous communication as mechanisms for reducing transmission overhead, while Medeiros et al. (2023) developed compression-aware split learning that responds to communication constraints. Although these studies address specialised machine-learning workloads, they share the principle that communication parameters should be adjusted to current network conditions. The present work extends this principle to general web and distributed applications by allowing the serialisation format itself to change during operation.

The modular structure of the proposed system is also consistent with the emphasis on scalable and adaptive architectures described by Rane et al. (2024). In the current framework, network monitoring, utility calculation, format prediction, serialisation, and negotiation perform complementary functions. Random Forest supports interpretable classification, LSTM models temporal changes in network conditions, and Deep Q-Learning supports sequential adaptation. Their roles are therefore differentiated rather than interchangeable.

The findings are particularly relevant to the web-application context examined by Sivakumar (2024), who showed that replacing JSON with Protocol Buffers can reduce message size and request-processing time. The present results support the efficiency of Protocol Buffers but indicate that

permanent replacement of JSON with a single binary format may not always be appropriate. JSON may remain preferable when readability, debugging, and broad client compatibility are prioritised, whereas Protocol Buffers, Avro, or MessagePack may be more suitable under bandwidth or latency constraints. The principal contribution is therefore not the identification of one universally superior format, but the ability to select among formats according to current conditions.

The internal validation results support the feasibility of this approach. Random Forest achieved a format-prediction accuracy of 95%, LSTM achieved 94%, and Deep Q-Learning achieved 92%. Correct format selection reached 99% under stable high-speed conditions and decreased to 85% in edge-computing environments. This decline indicates that unstable connectivity, restricted bandwidth, and increased latency complicate the selection task. The mean internal decision time of approximately 45 ms suggests that the models can support responsive decision-making, although this measure excludes monitoring, aggregation, negotiation, and packet-transmission time.

The broader relevance of adaptive architectural control is also reflected in the studies of Yu et al. (2024), Song (2023), and An and Tilevich (2022). These authors showed that distributed systems benefit from environmental forecasting, adaptive resource allocation, dynamic service placement, and replication. The present study complements these approaches by introducing serialisation-format selection as an additional runtime control mechanism. Whereas previous work primarily adjusts routing, scaling, replication, or compression intensity, the proposed system changes the representation used for data exchange.

The scientific contribution of the study lies in developing a comprehensive framework for the adaptive optimisation of data transmission in web applications by integrating three research directions that are usually considered separately: comparative evaluation of serialisation formats, real-time monitoring of heterogeneous network conditions, and machine-learning-based decision-making. The proposed system demonstrates that serialisation performance is multidimensional and that the smallest payload does not necessarily ensure the best overall transmission profile. To address this, the framework introduces a dynamically weighted utility function that explicitly balances payload size, processing time, processor load, memory consumption, and network conditions, while Random Forest, LSTM, and Deep Q-Learning support real-time prediction and adaptive format selection. As a result, serialisation is reframed from a fixed implementation choice into an intelligent runtime process capable of improving transmission efficiency, reducing latency, and supporting more stable performance under changing network conditions.

The use of machine learning methods, in particular LSTM models, Random Forest, and reinforcement learning algorithms, significantly improves the efficiency of adaptive optimisation of network processes. The LSTM model demonstrated the highest prediction accuracy of 94%, due to its ability to accommodate temporal dependencies in network data. The Random Forest algorithm showed high efficiency in real-time modes, achieving 95% accuracy with minimal computational costs.

Reinforcement learning methods, in particular Deep Q-Learning, provide the fastest adaptation to changes in network conditions with an overall accuracy of about 92-95%. The application of these algorithms within an adaptive data serialisation system has significantly improved performance. The proposed model increased the overall efficiency of data transmission by 87.5% under conditions of limited network bandwidth. The accuracy of predicting the optimal serialisation format was 94%, ensuring the stability and reliability of the system. In stable high-speed networks, the system eliminates failures and errors with an accuracy of about 99%, while in unstable edge environments, this figure reaches 85%, meaning that most unsuccessful decisions are automatically recorded and corrected. At the same time, the average data transfer delay has been reduced by 35% compared to traditional approaches, which contributes to improving the quality of service for end users. Thanks to automated testing, the system achieved a response time of 45 ms and 85% reliability in edge scenarios, with minimal validation errors (<1% false positives detected).

Based on the comparison and normalisation of the data, it is possible to conclude that Protobuf is the most efficient format for serialising and deserialising data across all networks, including 3G, 4G LTE, 5G, Edge, and Wi-Fi. This format provides high data processing speed and minimal data transfer size, making it the optimal choice for high-performance systems where speed and data transfer efficiency are important. MessagePack is an option for medium and small data volumes when processing speed is important, but minimising data size is not critical. It performs well in serialisation and deserialisation, making it suitable for a wide range of applications. BSON performs well, especially for specific tasks where structured data needs to be stored. It provides a good balance between processing speed and the size of the transmitted data. JSON and XML are less efficient formats, which limits their use in high-performance systems, especially for large amounts of data, due to high processing times and large transmitted data sizes. XML proved to be the least efficient format due to its complexity and significant processing time. Thus, for modern applications that require fast processing and transfer of large amounts of data, Protobuf and MessagePack are the best choices among all serialisation formats.

The study is based on a typical network environment, a limited data set, and high computational costs, and covers only selected serialisation formats and algorithms. Validation was conducted within a controlled infrastructure using predefined network profiles. Although session-level splitting and group-aware cross-validation reduced information leakage, no external validation was performed. Because paired run-level and prediction-level data were unavailable, the results should be interpreted descriptively. Future studies should use independent datasets, production applications, unseen network conditions, and complete records suitable for inferential analysis. Further research should evaluate the framework using independent datasets, production applications, heterogeneous client devices, uncontrolled mobile networks, and previously unseen connectivity patterns. Additional attention should also be given to end-to-end adaptation latency, format-switching overhead, and the integration of network indicators with device-level energy and hardware constraints.

5. REFERENCES

Adibfar, A., Costin, A. M., 2021, Review of data serialization challenges and validation methods for improving interoperability. In: Raymond, R., Issa, A., Eds., *Computing in Civil Engineering*. Reston, VA, USA, American Society of Civil Engineers, pp. 522–529. doi: 10.1061/9780784483893.065.

Aizstrauta, D., Ginters, E., 2017, Using Market Data of Technologies to Build a Dynamic Integrated Acceptance and Sustainability Assessment Model. *Procedia Computer Science* **104**, 501–508. doi: 10.1016/j.procs.2017.01.165.

Al-Selwi, S. M. et al., 2024, RNN-LSTM: From applications to modeling techniques and beyond – Systematic review. *Journal of King Saud University - Computer and Information Sciences* **36**, 5, Art. no. 102068. doi: 10.1016/j.jksuci.2024.102068.

An, K., Tilevich, E., 2022, Adaptive redistribution and replication to improve the responsiveness of mobile web apps. *Journal of Web Engineering* **21**, 6, 1981–2010. doi: 10.13052/jwe1540-9589.2168.

Baccour, E., Erbad, A., 2024, Multi-agent reinforcement learning for privacy-aware distributed CNN in heterogeneous IoT surveillance systems. *Journal of Network and Computer Applications* **230**, Art. no. 103933. doi: 10.1016/j.jnca.2024.103933.

Balalaieva, O., Marchenko, I., Korotenko, G., Beshta, D., Pikuz, A., 2023, Performance research of C# programming language data serializers using the developed software product for testing. *Reporter of the Priazovskyi State Technical University. Section: Technical Sciences* **47**, 8–24. doi: 10.31498/2225-6733.47.2023.299923.

Bao, W., Yuan, D., Zhou, B. B., Zomaya, A. Y., 2020, Reinforcement learning-based delay-aware path exploration of parallelized service function chains. *Mathematics* **10**, 24, 4698 p. doi: 10.3390/math10244698.

Brescia, E., Massenio, P. R., Nardo, M. D., Cascella, G. L., Gerada, C., Cupertino, F., 2023, Nonintrusive Parameter Identification of IoT-Embedded Isotropic PMSM Drives. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **11**, 5, 5195–5207. doi: 10.1109/JESTPE.2023.3292526.

Chandrasekharan, L., 2024, *Practical recommendations for TestSigma documentation*. [Online]. Available: https://testsigma.com/blog/testsigma-joins-eurostar-2024/

Duggento, A., Scimeca, M., Urbano, N., Bonanno, E., Aiello, M., Cavaliere, C., Cascella, G. L., Cascella, D., Conte, G., Guerrisi, M., Toschi, N., 2019, A random initialization deep neural network for discriminating malignant breast cancer lesions. In: *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 912–915. doi: 10.1109/EMBC.2019.8856740.

Friesel, D., Spinczyk, O., 2021, Data serialization formats for the internet of things. doi: 10.14279/tuj.eceasst.80.1134.

Ginters, E., Dimitrovs, E., 2021, Latent Impacts on Digital Technologies Sustainability Assessment and Development. *Advances in Intelligent Systems and Computing* **1365**, 3–13. doi: 10.1007/978-3-030-72657-7_1.

Ginters, E., Gutiérrez, J. M., Mendívil, E. G., 2020, Mapping of Conceptual Framework for Augmented Reality Application in Logistics. In: *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, Art. no. 9259302. doi: 10.1109/ITMS51158.2020.9259302.

Gospodinova, E., 2023, Mathematical Modeling and Planning of Energy Production using a Neural Network. *WSEAS Transactions on Power Systems* **18**, 39–48. doi: 10.37394/232016.2023.18.5.

Gospodinova, E., 2024, Optimizing the Energy Efficiency of a Lighting Network using Graph Theory. *WSEAS Transactions on Computer Research* **12**, 291–299. doi: 10.37394/232018.2024.12.28.

Huang, B., Tang, Y., 2021, Research on optimization of real-time efficient storage algorithm in data information serialization. *PLoS ONE* **16**, 12, p. e0260697. doi: 10.1371/journal.pone.0260697.

Iasechko, S., Haliantych, M. K., Skomorovskyi, V. B., Zadorozhnyi, V., Obryvkina, O., Pohrebniak, O., 2020, Contractual relations in the information sphere. *Systematic Reviews in Pharmacy* **11**, 8, 301–303. doi: 10.31838/srp.2020.8.46.

Jackson, S., Cummings, N., Khan, S., 2024, Streaming technologies and serialization protocols: Empirical performance analysis. *IEEE Access* **12**, 158025–158039. doi: 10.1109/ACCESS.2024.3486054.

Kaur, A., Ayyagari, S., Mishra, M., Thukral, R., 2020, A literature review on device-to-device data exchange formats for IoT applications. *Journal of Intelligent Systems and Computing*. doi: 10.51682/JISCOM.00101001.2020.

Kerimkhulle, S., Kuttykozhayeva, S., Turtkarayeva, G., Seitova, T., Ospanova, N., Iskakov, Y., 2025, Forecasting for Japanese Yen to Dollar Exchange Rate Based on Fuzzy Logic. *Lecture Notes in Networks and Systems* **1489**, 18–26. doi: 10.1007/978-3-031-96798-6_3.

Kisała, P., Wójcik, W., Smailov, N., Kalizhanova, A., Harasim, D., 2015, Elongation determination using finite element and boundary element method. *International Journal of Electronics and Telecommunications* **61**, 4, 389–394. doi: 10.2478/eletel-2015-0051.

Kozhakhmetova, D., Kaliyeva, S., Sugurova, L., Sugur, Z., Wójtowicz, R., Zhylkybayev, T., 2024, Analysis of the Distillation Column of a Catalytic Cracking Unit Using Fuzzy Input Information. *Energies* **17**, 17, Art. no. 4446. doi: 10.3390/en17174446.

Lamaazi, H., Barka, E., 2025, Networks and implementation tools for IoT and big data. In: Serhani, M. A., Xu, Y., Maamar, Z., Eds., *Empowering IoT with Big Data Analytics*. London, U.K., Academic Press, pp. 213–234. doi: 10.1016/B978-0-443-21640-4.00014-4.

Liu, C., Paparrizos, J., Elmore, A. J., 2024, Adaedge: A dynamic compression selection framework for resource constrained devices. In: *Proceedings of the 40th International Conference on Data Engineering (ICDE)*, Utrecht, Netherlands, pp. 1506–1519. doi: 10.1109/ICDE60146.2024.00124.

Luis, Á., Casares, P., Cuadrado-Gallego, J. J., Patricio, M. A., 2021, PSON: A serialization format for IoT sensor networks. *Sensors* **21**, 13, 4559 p. doi: 10.3390/s21134559.

Maltsev, E., Muliarevych, O., 2024, Beyond JSON: Evaluating serialization formats for space-efficient communication. *Advances in Cyber-Physical Systems* **9**, 1, 9–15. doi: 10.23939/acps2024.01.009.

Manghisi, V. M., Fiorentino, M., Boccaccio, A., Gattullo, M., Cascella, G. L., Toschi, N., Pietroiusti, A., Uva, A. E., 2020, A body tracking-based low-cost solution for monitoring workers' hygiene best practices during pandemics. *Sensors* **20**, 21, 1–17. doi: 10.3390/s20216149.

Medeiros, A., Di Maio, A., Braun, T., Neto, A., 2023, Adaptive compression-aware split learning and inference for enhanced network efficiency. *ACM Transactions on Internet Technology* **24**, 4, Art. no. 27. doi: 10.1145/3687471.

Organisation for Economic Co-operation and Development, 2011, *Guidelines for quality assurance in data collection and processing*. [Online]. Available: https://www.oecd.org/content/dam/oecd/en/publications/reports/2021/09/second-edition-quality-assurance-and-glp%20b3c674ab/b37820a3-en.pdf

Ouyang, S., Dong, D., Xu, Y., Xiao, L., 2021, Communication optimization strategies for distributed deep neural network training: A survey. *Journal of Parallel and Distributed Computing* **152**, 52–65. doi: 10.1016/j.jpdc.2020.11.005.

OWASP Testing Guide, 2023, *OWASP Testing Guide, ver. 4.2*. OWASP Foundation. [Online]. Available: https://owasp.org/www-project-web-security-testing-guide/v42/

Polishchuck, M. N., Tkach, M., Zhuchenko, O., Kornaga, Y. I., 2023, Mobile Robot for Monitoring Park Trees: Design and Modeling. *FME Transactions* **51**, 3, 423–431. doi: 10.5937/fme2303423P.

Raj, L. V., Sasi, S., Rajeswari, P., Pushpa, B. R., Kulkarni, A. V., Biradar, S., 2025, Design of FBG-based optical biosensor for the detection of malaria. *Journal of Optics (India)*. doi: 10.1007/s12596-025-02780-x.

Rane, J., Mallick, S. K., Kaya, Ö., Rane, N. L., 2024, Scalable and adaptive deep learning algorithms for large-scale machine learning systems. In: *Future Research Opportunities for Artificial Intelligence in Industry 4.0 and 5.0*. London, U.K., Deep Science Publishing, pp. 39–92. doi: 10.70593/978-81-981271-0-5_2.

Rzheuskyi, A., Gozhyj, A., Stefanchuk, A., Oborska, O., Lozynska, O., Mykich, K., Basyuk, T., 2019, Development of mobile application for choreographic productions creation and visualization. *CEUR Workshop Proceedings* **2386**, 340–358.

Salmanov, V. I., 2025, Optimal Control Problem and Integral Quality Criteria with a Special Gradient Term. *Eurasia Proceedings of Science, Technology, Engineering and Mathematics* **34**, 310–319. doi: 10.55549/epstem.1754735.

Sancio, R., Pugliese, S., Debbadi, K., Liserre, M., Brescia, E., Cascella, G. L., 2022, Fuzzy Based Adaptive Linear Active Disturbance Rejection Control for a High-Speed PMSM. In: *1st IEEE Industrial Electronics Society Annual On-Line Conference (ONCON)*. doi: 10.1109/ONCON56984.2022.10126818.

Shethiya, A. S., 2025, Scalability and performance optimization in web application development. *Integrated Journal of Science and Technology* **2**, 1. [Online]. Available: https://ijstpublication.com/index.php/ijst/article/view/1

Sivakumar, S., 2024, Performance optimization of large language models (LLMs) in web applications. *International Journal of Trend in Scientific Research and Development* **8**, 1, 1077–1096. [Online]. Available: https://www.researchgate.net/publication/386342544_Performance_Optimization_of_Large_Language_Models_LLMs_in_Web_Applications

Smailov, N., Akmardin, S., Ayapbergenova, A., Ayapbergenova, G., Kadyrova, R., Sabibolda, A., 2025, Analysis of VLC Efficiency in Optical Wireless Communication Systems for Indoor Applications. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Srodowiska* **15**, 2, 135–138. doi: 10.35784/iapgos.6971.

Smailov, N., Koshkinbayev, S., Aidana, B., Kuttybayeva, A., Tashtay, Y., Aziskhan, A., Arseniev, D., Kiesewetter, D., Krivosheev, S., Magazinov, S., Malyugin, V., Sun, C., 2024, Simulation and Measurement of Strain Waveform under Vibration Using Fiber Bragg Gratings. *Sensors* **24**, 19, Art. no. 6194.

Smailov, N., Orynbet, M., Nazarova, A., Torekhan, Z., Koshkinbayev, S., Yssyraiyl, K., Kadyrova, R., Sabibolda, A., 2025, Optimization of Fiber-Optic Sensor Performance in Space Environments. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Srodowiska* **15**, 2, 130–134. doi: 10.35784/iapgos.7200.

Song, W.-W., 2023, *Efficient and adaptive resource management for dynamically optimizing distributed data processing systems*. [Online]. Available: https://s-space.snu.ac.kr/handle/10371/196490

Systems and Software Engineering, 2015, *ISO/IEC 25024:2015*. [Online]. Available: https://www.iso.org/standard/35749.html

Tiwary, G. P., Stroulia, E., Srivastava, A., 2021, Compression of XML and JSON API responses. *IEEE Access* **9**, 57426–57439. doi: 10.1109/ACCESS.2021.3073041.

Trach, R., Chupryna, I., Tormosov, R., Leshchynsky, V., Trach, Y., Ryzhakova, G., Ratnikov, D., Onofriichuk, O., 2026, Decision Support Framework for Post-War Infrastructure Revitalization Using a Hybrid Fuzzy–Simulation–ANN Model. *Applied Sciences* **16**, 9, Art. no. 4364. doi: 10.3390/app16094364.

Tunc, I., Soylemez, M. T., 2023, Fuzzy logic and deep Q learning based control for traffic lights. *Alexandria Engineering Journal* **67**, 343–359. doi: 10.1016/j.aej.2022.12.028.

Vanura, J., Kriz, P., 2018, Performance evaluation of Java, JavaScript and PHP serialization libraries for XML, JSON and binary formats. In: Ferreira, J., Spanoudakis, G., Ma, Y., Zhang, L., Eds., *International Conference on Services Computing*. Cham, Switzerland, Springer, pp. 166–175. doi: 10.1007/978-3-319-94376-3_11.

Viotti, J., Kinderkhedia, M., 2022, A survey of JSON-compatible binary serialization specifications. arXiv:2201.02089. doi: 10.48550/arXiv.2201.02089.

Wang, Y., Li, X., Wu, R., Chen, H., Kutscher, D., 2025, NetSenseML: Network-adaptive compression for efficient distributed machine learning. arXiv:2506.16235. doi: 10.48550/arXiv.2506.16235.

Yu, D. et al., 2024, Scalable model-based policy optimization for decentralized networked systems. arXiv:2207.06559. doi: 10.48550/arXiv.2207.06559.

Zhuchenko, O., Korotynskyi, A., Savula, A., 2021, Generative adversarial networks at development of automatic document processing system. In: *2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT)*, pp. 278–281. doi: 10.1109/Atit54053.2021.9678876.