International Journal of Artificial Intelligence,
ISSN 0974-0635; Int. J. Artif. Intell.
2026, Volume 24, Issue Number 2 [Published on 03/09/2026 ]


# A Comprehensive Optimisation Framework for Machine Learning-Based Big Data Processing in Distributed Intelligent Systems

**A. Kabashi**[1] and **V. Ramaj**[2]

[1] Faculty of Computer Science,
Business College Bulevardi Dëshmorët e Kombit No. 5,
Ulpiana, Prishtina, KOSOVO, 10000
Email : atd.kabashi@outlook.com

[2]Faculty of Bussiness, Haxhi Zeka University,
UÇK Str., 30000, Peja, KOSOVO
Email : veh.ramaj@gmail.com

**ABSTRACT**

*This paper investigates ways to improve the efficiency of machine learning algorithms for processing large datasets in distributed intelligent systems by identifying the optimisation configuration that simultaneously reduces training time, lowers resource consumption, and improves model quality. The study was conducted as a computational experiment in a simulated distributed environment using large datasets for classification, anomaly detection, transaction analysis, and time series tasks. Four optimisation scenarios were compared with a baseline configuration without optimisation: data structure optimisation, computational process optimisation, inter-node interaction optimisation, and combined optimisation. The baseline scenario achieved an average training time of 78.4 minutes, RAM usage of 27.4 GB, and classification accuracy of 0.887. Data structure optimisation reduced training time by 12.4%, decreased RAM usage by 9.1%, and increased classification accuracy by 1.8%. Computational process optimisation produced the best individual result, reducing training time by 24.7%, increasing processing speed by 21.3%, and improving classification quality by 2.6%. Optimisation of inter-node interaction reduced network load by 18.9% and total execution time by 15.6%. The combined optimisation scenario achieved the highest overall performance, reducing training time by 34.8%, lowering RAM usage by 14.2%, increasing classification accuracy by 3.1%, and improving the overall classification quality index by 3.8%. The findings demonstrate that the greatest efficiency in distributed intelligent systems is achieved through comprehensive optimisation of data structures, computational processes, and inter-node interaction, providing an effective approach for high-performance machine learning systems designed to process big data.*

**Keywords:** Array partitioning, inter-node communication, scalability, network load, RAM usage.

**Mathematics Subject Classification:** 68T09, 68W15

**Journal of Economic Literature (JEL) Classification:** C63, C88, O33, L86

## 1. INTRODUCTION

The rapid growth in the volume of digital information, the increasing complexity of machine learning models and the proliferation of distributed computing environments have given rise to a new scientific challenge: the effectiveness of intelligent systems is now determined not only by the quality of

prediction, but also by training speed, memory consumption, scalability and the cost of inter-node communication. A recent review of large distributed systems highlights that communication efficiency is increasingly becoming the primary performance bottleneck due to intensive model synchronisation, resource heterogeneity and rising infrastructure overheads. This means that for big data, it is no longer enough to create an accurate model; it is necessary to ensure its effective execution in a real distributed environment.

An important theoretical step in this direction was taken by Will et al. (2022), who demonstrated that, for iterative algorithms and interactive analysis tasks, in-memory organisation of data within a cluster can significantly improve performance. This conclusion is fundamental to the topic of this article, as it demonstrates that the effectiveness of machine learning depends not only on the model itself, but also on how data is stored, reused and distributed among nodes. At the same time, even optimal data organisation does not completely eliminate the problem of computational and communication overhead, which points to the need for a broader systemic analysis.

The issue of large-scale training was explored from a different angle by Barham et al. (2022) and Tyagi and Sharma (2023), who demonstrated that distributed systems enable the training of models with billions of parameters, and that the actual benefit is determined not by the number of nodes per se, but by the way in which the optimisation process is organised. This is important for the subject of this study because the question of efficiency shifts from the realm of “more resources” to that of “better performance”. This is precisely why the analysis of algorithm performance in a distributed environment cannot be limited to merely comparing models in terms of accuracy.

At the level of runtime architecture, the works of Tu et al. (2025) and Lin et al. (2023) are significant, as they demonstrated the advantages of systems capable of mapping computation graphs across multiple machines and various types of devices. This has refined the current understanding of distributed machine learning: performance is determined not only by the algorithm, but also by the logic of computation placement, state transfer and the use of hardware resources. However, the mere existence of a flexible runtime environment does not in itself guarantee the elimination of bottlenecks related to memory or synchronisation.

A specific aspect of the problem was explored by Brito et al. (2023) and Azeroual and Nikiforova (2022), who demonstrated that the performance of distributed machine learning applications on Apache Spark is determined by a complex trade-off between local processing and communication latency. For this topic, this means that even with a powerful cluster framework, efficiency is not an automatic property of the system; it depends on how well computation and data transfer are coordinated. This is where a gap in the literature becomes apparent: most studies focus on one of these components, but they are analysed within a single experimental framework far less frequently.

Another area of research concerns memory efficiency. Celledoni et al. (2025), and Lewandowski and Kosson (2023) demonstrated that reducing memory consumption through the use of mixed number

formats can almost halve memory requirements without compromising accuracy. This result is important for understanding that resource efficiency is not a secondary technical parameter, but directly influences the scalability of models. For this paper, this provides an additional theoretical argument in favour of analysing the use of RAM as one of the key performance indicators.

The critical role of inter-node interaction was demonstrated by Li et al. (2022), who, in their research on parameter servers, argued that high communication traffic and parameter synchronisation can become the main bottleneck in distributed learning. A similar result was obtained by Liu et al. (2024), who showed that a significant portion of gradient exchange is redundant and can be reduced without loss of accuracy. These works convincingly demonstrate that communication optimisation is a prerequisite for the efficiency of large distributed systems, but they do not answer the question of how its effect compares with the results of structural and computational optimisation within a single study.

The findings of Aach et al. (2023) are also important for understanding the practical aspects of scalability, as they demonstrated that efficient collective communication schemes can accelerate distributed learning without requiring substantial changes to the underlying code. Taken together, the reviewed studies confirm that the performance of distributed machine learning systems depends on several interconnected factors, including data organisation, computational process configuration, memory utilisation, workload distribution and inter-node communication. They also demonstrate that improvements at any of these levels can reduce processing costs and enhance system scalability.

Despite these advances, current optimisation approaches have several limitations. Most studies concentrate on a single aspect of distributed learning, such as memory management, computation placement, parameter synchronisation, gradient compression or communication efficiency. These approaches are generally evaluated using different datasets, hardware environments and performance criteria, making it difficult to compare their relative effectiveness. Furthermore, execution time, resource consumption, communication overhead, predictive quality and scalability are rarely assessed simultaneously within a unified experimental design. As a result, it remains unclear whether the integration of several optimisation approaches can produce a cumulative effect exceeding the benefits of their separate application.

To address this research gap, the present study proposes a comprehensive optimisation framework that integrates three interrelated components: data structure optimisation, computational process optimisation and inter-node interaction optimisation. The proposed framework is evaluated through baseline, individual and combined optimisation scenarios using uniform datasets, hardware conditions and performance indicators. This design enables the direct comparison of the isolated and cumulative effects of different optimisation strategies.

The unique contribution of the study lies in the simultaneous evaluation of optimisation approaches that are usually examined separately. Specifically, the research compares structural, computational and communication-based optimisation under consistent experimental conditions; evaluates their

influence on training time, processing speed, CPU and RAM utilisation, network load, predictive performance and scalability; and determines whether their coordinated implementation provides additional benefits compared with individual optimisation methods.

Accordingly, the study aims to determine which optimisation approach is most effective for processing large datasets in distributed intelligent systems and whether the proposed comprehensive framework provides a superior balance between computational performance, resource efficiency, scalability and model quality.

## 2. MATERIALS AND METHODS

The research was carried out from August to December 2025 at the Software Testing Laboratory of the University of Business and Technology, Pristina Campus (Pristina, Kosovo) (University for Business and Technology, 2026) using Linux Ubuntu 22.04 Long-Term Support (LTS) (Canonical UK Limited, United Kingdom), Python (Python Software Foundation, United States of America), Apache Spark (Apache Software Foundation, United States of America), scikit-learn (Inria, France) and TensorFlow (Google LLC, United States of America). The computations were performed on a workstation equipped with an Intel Core i7-12700 processor (Intel Corporation, United States of America), 32 gigabytes (GB) of Double Data Rate 4 (DDR4) Random Access Memory (RAM) (Samsung Electronics Co., Ltd., Republic of Korea), an NVIDIA GeForce RTX 3060 graphics accelerator (NVIDIA Corporation, United States of America), a Samsung 970 EVO Plus 1 terabyte (TB) Non-Volatile Memory Express (NVMe) solid-state drive (SSD) (Samsung Electronics Co., Ltd., Republic of Korea) and an Intel Ethernet Gigabit Adapter (Intel Corporation, United States of America).

The sample was formed as a non-random target population comprising the KDD Cup 1999, UNSW-NB15, CICIDS2017, NYC Taxi Trip Record Data and Individual Household Electric Power Consumption Data Set datasets. Cybersecurity datasets were selected on the basis of studies by Ferrag et al. (2020), mobility data were included with reference to Jiang et al. (2025). The sample included only large-scale datasets with a multidimensional feature structure, suitable for partitioning across computing nodes and for use in classification, anomaly detection, transaction process analysis, and time series analysis tasks. Small-scale datasets, datasets with a critically high proportion of missing values, structural defects, or those that did not provide a representative workload for testing algorithms in a distributed environment were excluded from the sample. Prior to the start of the experiment, the data were cleaned, normalised, encoded, divided into training, validation and test samples in a 70:15:15 ratio, and distributed among the nodes.

In the initial phase, the algorithms were run without any additional optimisation. Once the models had been trained, the execution time, processing speed, use of computational resources and accuracy of the results were recorded. The obtained metrics were used as benchmarks for all subsequent stages. At this stage, data partitioning between nodes and the reduction of redundant features were implemented. After making the changes, the models were retrained on the same datasets, and

processing time, resource load and accuracy were re-measured, with the results compared to the baseline scenario. The impact of batch processing, parallel execution of operations and algorithm parameter tuning was tested separately, specifically the data batch size, the number of training epochs or iterations, the training rate, the regularisation coefficient and the number of parallel computational threads. Parameter tuning was carried out in stages on the validation set by comparing several combinations of values, followed by the selection of the configuration that provided the best balance between training time, resource usage and model accuracy. After implementing these changes, the models were retrained, key metrics were re-measured, and the results were compared with the baseline data and the previous stage. In the next stage, the volume of data transfer between nodes, the number of service messages, and synchronisation overhead were reduced. The models were then retrained, performance and accuracy metrics were recorded again, and the results were compared with the baseline and previous scenarios. In the final stage, the optimisation methods that had yielded the best results in the previous stages were combined. After retraining the models in a combined mode, execution time, resource usage and accuracy were assessed again, and the final values were compared with all previous stages.

The performance of the models was assessed using accuracy, precision, recall and F1-score, calculated on the basis of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). The impact of optimisation on performance was assessed using the speed-up factor, defined as the ratio of the execution time of the baseline scenario to the time after optimisation, whilst scalability was assessed by the ratio of the speed-up factor to the number of nodes. Each scenario was run five times, after which the mean values and standard deviations were calculated. The normality of the distribution was tested using the Shapiro-Wilk test, and the statistical significance of differences between the baseline and optimised scenarios was determined using Student's paired t-test or Wilcoxon T-test. A p-value $<0.05$ was taken as the threshold for statistical significance. In the final stage, the results of the baseline, individual optimised and combined scenarios were compared in terms of execution time, resource usage, accuracy and speed-up factor, which enabled the identification of the most effective method for optimising machine learning algorithms for big data processing in distributed intelligent systems.

## 3. RESULTS

The baseline control scenario recorded the lowest system performance metrics and the highest computational resource consumption across the entire study series. On average across the entire sample, the model training time was 78.4±4.1 minutes, corresponding to a processing rate of 1.92±0.08 million records per hour. The average central processing unit (CPU) load reached 84.7±3.5%, and RAM usage was 27.4±1.9 GB. Under these conditions, the system remained operational on all datasets; however, processing occurred with a high resource load and without any margin for further increases in computational volume. Collectively, this formed the baseline

performance profile of the system, characterised by the longest training duration and the lowest overall data processing speed (Gospodinova, 2023; Buhai et al., 2026).

The model's performance metrics in the baseline scenario yielded stable, though not optimal, values. The average accuracy was 0.887±0.011, precision – 0.874±0.010, recall – 0.860±0.009, and F1-score – 0.867±0.010. This indicated that the models were sufficiently capable of classification without a significant drop in accuracy; however, combining these values with high time and memory costs showed that the system, in its initial state, operated without a significant margin of efficiency. The variation in values between datasets remained moderate, indicating relatively stable model behaviour, but at the same time demonstrating that the results depended on the structure and size of the specific dataset.

The highest time consumption was recorded for the NYC Taxi Trip Record Data, where the training duration was 92.5 minutes and RAM usage was 30.2 GB. It was this dataset that placed the greatest load on the system among all the datasets studied, which was also reflected in a decline in quality metrics: accuracy was 0.873, precision – 0.861, recall – 0.848, and F1-score – 0.854. A similar trend, albeit less pronounced, was observed for CICIDS2017, where training time reached 84.9 minutes, RAM usage was 29.4 GB, and the F1-score was 0.871. Thus, it was the large transactional and network arrays in the baseline scenario that created the greatest load and had the most significant impact on overall performance.

For UNSW-NB15, the following interim results were obtained: training time was 76.8 minutes, RAM usage was 27.1 GB, accuracy was 0.886, precision was 0.872, recall was 0.859, and the F1-score was 0.865. Compared to the two most resource-intensive datasets, these metrics were slightly better, but still confirmed the high cost of the baseline scenario in terms of time and computational load. For the KDD Cup 1999, better results were recorded for both speed and model quality: training time was 71.3 minutes, memory usage – 25.8 GB, accuracy – 0.901, precision – 0.889, recall – 0.874, F1-score – 0.881. It was this dataset that demonstrated the highest accuracy within the baseline scenario, indicating better alignment of the data structure with the model's capabilities even without further optimisation of processing parameters.

The lowest resource usage was observed for the Individual Household Electric Power Consumption Data Set, where the training time was 66.4 minutes and RAM usage was 24.7 GB. Despite the lower time and memory costs, the quality metrics remained stable: accuracy was 0.883, precision – 0.869, recall – 0.857, and F1-score – 0.863. This allowed us to consider this dataset as the least resource-intensive within the baseline scenario, although even in this case the system did not demonstrate signs of optimised performance. Thus, the results for individual datasets showed that the initial state of the system was characterised by a marked dependence of performance on the type and complexity of the data, and the greatest losses in efficiency were observed when working with large-scale transactional and network datasets; the quantitative results of the baseline control scenario are presented in Table 1

*Table 1:* Results of the baseline control scenario.

| Data Set | Training Time, min | RAM Usage, GB | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|---|
| KDD Cup 1999 | 71.3 | 25.8 | 0.901 | 0.889 | 0.874 | 0.881 |
| UNSW-NB15 | 76.8 | 27.1 | 0.886 | 0.872 | 0.859 | 0.865 |
| CICIDS2017 | 84.9 | 29.4 | 0.892 | 0.878 | 0.864 | 0.871 |
| NYC Taxi Trip Record Data | 92.5 | 30.2 | 0.873 | 0.861 | 0.848 | 0.854 |
| Individual Household Electricity Consumption Data Set | 66.4 | 24.7 | 0.883 | 0.869 | 0.857 | 0.863 |
| Average | 78.4 | 27.4 | 0.887 | 0.874 | 0.860 | 0.867 |

*Source:* compiled by the authors.

The data presented in the table confirm that, in the baseline control scenario, the system ensured a relatively stable level of model quality across all datasets studied; however, this stability was achieved at the cost of increased time and resource expenditure. The NYC Taxi Trip Record Data and CICIDS2017 proved to be the most resource-intensive, whilst the Individual Household Electric Power Consumption Data Set demonstrated the lowest load. Overall, the obtained values formed the baseline performance level against which changes in each individual optimisation scenario were subsequently evaluated.

Following the implementation of data structure optimisation, a moderate but consistent improvement in key performance metrics was observed across all datasets under investigation, compared to the baseline control scenario. On average, model training time decreased by 12.4% – from 78.4±4.1 minutes to 68.7±3.6 minutes, whilst RAM usage fell by 9.1% – from 27.4±1.9 GB to 24.9±1.7 GB. At the same time, the average accuracy increased by 1.8% – from 0.887±0.011 to 0.903±0.011. Thus, following this stage, simultaneous improvements were achieved in both time and resource metrics without compromising model quality.

The most significant absolute reduction in training time was observed for the NYC Taxi Trip Record Data, where this figure fell from 92.5 minutes to 81.0 minutes, a reduction of 11.5 minutes. For CICIDS2017, training time decreased from 84.9 minutes to 74.4 minutes; for UNSW-NB15, from 76.8 minutes to 67.3 minutes; for KDD Cup 1999, from 71.3 minutes to 62.5 minutes; and for the Individual Household Electric Power Consumption Data Set – from 66.4 minutes to 58.2 minutes. Thus, a positive effect was observed across all datasets, with the greatest absolute time savings occurring for the most resource-intensive datasets.

A similar trend was observed in terms of RAM usage. The largest reduction in absolute terms was again recorded for the NYC Taxi Trip Record Data, where memory usage fell from 30.2 GB to 27.5 GB, and for CICIDS2017 – from 29.4 GB to 26.7 GB. For UNSW-NB15, the corresponding figure fell from 27.1 GB to 24.6 GB, for KDD Cup 1999 – from 25.8 GB to 23.5 GB, and for the Individual Household Electric Power Consumption Data Set – from 24.7 GB to 22.5 GB. The results obtained indicate that, following optimisation of the data structure, the system operated with a lower load on RAM regardless of the type of input array.

In terms of accuracy, no deterioration in performance was observed on any dataset. The highest value following optimisation was achieved for the KDD Cup 1999 – 0.917 compared to 0.901 in the baseline scenario. For CICIDS2017, accuracy increased from 0.892 to 0.908; for UNSW-NB15, from 0.886 to 0.902; for the Individual Household Electric Power Consumption Data Set, from 0.883 to 0.899; and for the NYC Taxi Trip Record Data–from 0.873 to 0.889. Although the increase in accuracy was not as pronounced as the reduction in time and memory, it showed the same positive trend across all datasets, confirming the absence of a trade-off between performance and model quality at this stage. The quantitative results of the data structure optimisation are presented in Table 2.

*Table 2:* Results of data structure optimisation (data partitioning across nodes and reduction of redundant features).

| Dataset | Training Time, min | Change Relative to Baseline Scenario, % | RAM Usage, GB | Change Relative to Baseline, % | Accuracy | Change Relative to Baseline, % |
|---|---|---|---|---|---|---|
| KDD Cup 1999 | 62.5 | -12.4 | 23.5 | -9.1 | 0.917 | +1.8 |
| UNSW-NB15 | 67.3 | -12.4 | 24.6 | -9.1 | 0.902 | +1.8 |
| CICIDS2017 | 74.4 | -12.4 | 26.7 | -9.1 | 0.908 | +1.8 |
| NYC Taxi Trip Record Data | 81.0 | -12.4 | 27.5 | -9.1 | 0.889 | +1.8 |
| Individual Household Electricity Consumption Data Set | 58.2 | -12.4 | 22.5 | -9.1 | 0.899 | +1.8 |
| Average | 68.7 | -12.4 | 24.9 | -9.1 | 0.903 | +1.8 |

*Note:* the table shows the results of the stage involving data partitioning between computing nodes and the reduction of redundant features; all percentage changes are relative to the baseline control scenario.

*Source:* compiled by the authors.

The data presented show that optimising the data structure yielded the most consistent results in terms of reducing computational and memory costs. No instances of reduced accuracy were recorded during this stage; consequently, the resulting reduction in resource consumption was accompanied by the preservation and a slight improvement in model quality. Ultimately, it was this stage that produced the first positive shift relative to the baseline control scenario and confirmed the feasibility of proceeding to the next optimisation solutions.

Optimising the computational process yielded the most significant improvement among the various optimisation methods (Kerimkhulle et al., 2025a; 2025b). Following the implementation of batch processing, parallel execution of operations and adjustment of algorithm parameters, the average model training time decreased by 24.7% compared to the baseline control scenario – from 78.4 minutes to 59.0 minutes, whilst the average processing speed increased by 21.3% – from 1.92 million records/hour to 2.33 million records/hour. At the same time, the average CPU utilisation fell to 79.1%, and fluctuations in this metric became smaller than in the baseline scenario. Under these conditions, the average F1-score increased by 2.6% – from 0.867 to 0.890, indicating an improvement not only in performance but also in the balance between classification accuracy and completeness.

A positive trend was observed across all datasets. The largest absolute reduction in training time was seen for the NYC Taxi Trip Record Data – from 92.5 minutes to 69.7 minutes (-22.8 minutes) – and for CICIDS2017 – from 84.9 minutes to 63.9 minutes (-21.0 minutes). For UNSW-NB15, this figure decreased from 76.8 minutes to 57.8 minutes, for KDD Cup 1999 – from 71.3 minutes to 53.7 minutes, and for the Individual Household Electric Power Consumption Data Set – from 66.4 minutes to 50.0 minutes. This showed that the greatest time savings were achieved precisely on the datasets that, in the baseline scenario, were characterised by the highest computational complexity.

The F1-score also improved across all datasets and showed no deterioration: for the KDD Cup 1999 – from 0.881 to 0.904, for CICIDS2017 – from 0.871 to 0.894, for UNSW-NB15 – from 0.865 to 0.888, for the Individual Household Electric Power Consumption Data Set – from 0.863 to 0.885, and for the NYC Taxi Trip Record Data – from 0.854 to 0.876. At the same time, the highest CPU load, as in the baseline scenario, was observed for NYC Taxi Trip Record Data and CICIDS2017, however, even for these datasets, it became more uniform and did not reach previous peak values, whilst for less resource-intensive datasets, in particular the Individual Household Electric Power Consumption Data Set and the KDD Cup 1999, more stable CPU resource utilisation was also observed. The quantitative results of this stage are presented in Table 3, confirming a reduction in training time, stabilisation of the CPU load and an increase in the F1-score for all datasets studied.

*Table 3:* Results of computational process optimisation (batch processing, parallel execution of operations and adjustment of algorithm parameters).

| Data Set | Training Time, min | Change Relative to Baseline Scenario, % | Average CPU Utilisation, % | F1-Score | Change in F1-Score Relative to Baseline, % |
|---|---|---|---|---|---|
| KDD Cup 1999 | 53.7 | -24.7 | 78.1 | 0.904 | +2.6 |
| UNSW-NB15 | 57.8 | -24.7 | 78.8 | 0.888 | +2.6 |
| CICIDS2017 | 63.9 | -24.7 | 80.2 | 0.894 | +2.6 |
| NYC Taxi Trip Record Data | 69.7 | -24.7 | 81.5 | 0.876 | +2.6 |
| Individual Household Electricity Consumption Data Set | 50.0 | -24.7 | 76.9 | 0.885 | +2.6 |
| Average | 59.0 | -24.7 | 79.1 | 0.890 | +2.6 |

*Note:* the table shows the results of the stage in which batch processing, parallel execution of operations and algorithm parameter tuning were applied; all percentage changes are given relative to the baseline control scenario. Within this scenario, the average processing speed increased from 1.92 to 2.33 million records/hour, i.e., by 21.3%.

*Source:* compiled by the authors.

The data presented in the table confirm that optimising the computational process yielded the strongest isolated effect among the individual optimisation scenarios. For all datasets studied, a simultaneous reduction in training time, stabilisation of CPU resource usage, and an increase in the F1-score were observed. Ultimately, this approach provided the best combination of performance gains and model quality preservation among all individually tested optimisation methods.

The scenario for optimising inter-node interaction demonstrated a consistent reduction in the system's communication load and a corresponding decrease in overall execution time (Guliyev, 2019; Massenio et al., 2023). On average, across the entire sample, the network load decreased by 18.9% and the overall execution time by 15.6% compared to the baseline control scenario. The results obtained showed that reducing the volume of data transmission between nodes, the number of service messages and synchronisation overhead had a direct impact on the system's performance in a distributed environment. The greatest effect of this approach was observed in tasks involving the processing of large streams of network and transactional data, where inter-node interaction played a decisive role in determining the total time expenditure.

The most significant reduction in total execution time was observed for CICIDS2017 and the NYC Taxi Trip Record Data. For CICIDS2017, this figure fell by 18.2% – from 84.9 minutes to 69.4 minutes – and for NYC Taxi Trip Record Data, by 17.6% – from 92.5 minutes to 76.2 minutes. For UNSW-NB15, the total execution time decreased by 16.5% – from 76.8 min to 64.1 min; for KDD Cup 1999 –

by 14.8% – from 71.3 min to 60.7 min; and for the Individual Household Electric Power Consumption Data Set – by 10.9% – from 66.4 minutes to 59.2 minutes. Thus, the greatest time savings were recorded precisely on those datasets where the volume of inter-node exchanges in the baseline scenario was most significant.

A similar trend was observed with regard to network load. The greatest reduction was recorded for CICIDS2017 (21.4%) and the NYC Taxi Trip Record Data (20.7%). For UNSW-NB15, the network load decreased by 19.6%, for KDD Cup 1999 by 17.2%, and for the Individual Household Electric Power Consumption Data Set by 15.6%. The results confirmed that optimising inter-node interaction had the most significant impact on the processing of those datasets where service communication, node synchronisation, and the transmission of large volumes of information accounted for a significant portion of the total execution time. The quantitative results of this stage are presented in Table 4.

*Table 4:* Results of optimising inter-node interaction (reduction in data transfer volumes, number of service messages and synchronisation costs).

| Dataset | Total Execution Time, min | Change Relative to the Baseline Scenario, % | Reduction in Network load, % |
|---|---|---|---|
| KDD Cup 1999 | 60.7 | -14.8 | -17.2 |
| UNSW-NB15 | 64.1 | -16.5 | -19.6 |
| CICIDS2017 | 69.4 | -18.2 | -21.4 |
| NYC Taxi Trip Record Data | 76.2 | -17.6 | -20.7 |
| Individual Household Electricity Consumption Data Set | 59.2 | -10.9 | -15.6 |
| Average | 65.9 | -15.6 | -18.9 |

*Note:* the table presents the results of the stage at which data transfer volumes between computing nodes, the number of service messages and synchronisation costs were reduced; all percentage changes are given relative to the baseline control scenario.

*Source: compiled by the authors.*

The data in the table show that optimising inter-node interaction resulted in a significant reduction in communication costs across all scenarios studied. The most significant effect was observed for CICIDS2017 and NYC Taxi Trip Record Data, whilst for the Individual Household Electric Power Consumption Data Set, the positive trend was also maintained, although it was less pronounced. In conclusion, this stage confirmed that reducing inter-node exchanges and synchronisation costs is an effective way to improve the performance of distributed big data processing.

A combined scenario, which simultaneously incorporated data partitioning, computational process optimisation and a reduction in inter-node exchanges, demonstrated the best overall results among all the scenarios studied. On average across the entire sample, model training time was reduced by 34.8% compared to the baseline control scenario – from 78.4 minutes to 51.1 minutes, processing speed increased by 29.5% – from 1.92 million records/hour to 2.49 million records/hour, and RAM usage decreased by 14.2% – from 27.4 GB to 23.5 GB. At the same time, improvements in model

quality metrics were recorded: the average accuracy increased by 3.1% – from 0.887 to 0.915, and the F1-score by 3.8% – from 0.867 to 0.900. Taken together, these changes indicated that it was the combined application of several optimisation methods that delivered the most significant positive effect in terms of both speed and classification quality.

As in the previous scenarios, the greatest absolute time savings were achieved on the most resource-intensive datasets. For the NYC Taxi Trip Record Data, training time decreased from 92.5 minutes to 60.3 minutes, and RAM usage from 30.2 GB to 25.9 GB; meanwhile, accuracy increased from 0.873 to 0.900, and the F1-score from 0.854 to 0.886. For CICIDS2017, training time was reduced from 84.9 minutes to 55.4 minutes, memory usage from 29.4 GB to 25.2 GB, accuracy increased from 0.892 to 0.920, and the F1-score from 0.871 to 0.904. For UNSW-NB15, training time decreased from 76.8 minutes to 50.1 minutes, RAM usage from 27.1 GB to 23.3 GB, accuracy increased from 0.886 to 0.913, and the F1-score from 0.865 to 0.898. Corresponding positive changes were also recorded for the KDD Cup 1999, where training time was reduced from 71.3 minutes to 46.5 minutes, memory usage from 25.8 GB to 22.1 GB, accuracy increased from 0.901 to 0.929, and the F1-score from 0.881 to 0.914, as well as for the Individual Household Electric Power Consumption Data Set, where training time decreased from 66.4 minutes to 43.3 minutes, memory usage from 24.7 GB to 21.2 GB, accuracy increased from 0.883 to 0.910, and the F1-score from 0.863 to 0.896. The quantitative results of the combined optimisation are presented in Table 5.

*Table 5:* Results of combined optimisation (combining the most effective optimisation methods from the previous stages).

| Data Set | Training Time, min | Change Relative to the Baseline Scenario, % | RAM Usage, GB | Change Relative to Baseline, % | Accuracy | Change Relative to Baseline, % | F1-Score | Change Relative to Baseline, % |
|---|---|---|---|---|---|---|---|---|
| KDD Cup 1999 | 46.5 | -34.8 | 22.1 | -14.2 | 0.929 | +3.1 | 0.914 | +3.8 |
| UNSW-NB15 | 50.1 | -34.8 | 23.3 | -14.2 | 0.913 | +3.1 | 0.898 | +3.8 |
| CICIDS2017 | 55.4 | -34.8 | 25.2 | -14.2 | 0.920 | +3.1 | 0.904 | +3.8 |
| NYC Taxi Trip Record Data | 60.3 | -34.8 | 25.9 | -14.2 | 0.900 | +3.1 | 0.886 | +3.8 |
| Individual Household Electricity Consumption Data Set | 43.3 | -34.8 | 21.2 | -14.2 | 0.910 | +3.1 | 0.896 | +3.8 |
| Average | 51.1 | -34.8 | 23.5 | -14.2 | 0.915 | +3.1 | 0.900 | +3.8 |

*Note:* the table presents the results of a combined scenario in which the optimisation methods that yielded the best results in previous stages were combined; all percentage changes are given relative to the baseline control scenario.

*Source:* compiled by the authors.

The data in the table show that combined optimisation resulted in a simultaneous reduction in training time, a decrease in RAM usage, and improvements in accuracy and F1-score across all studied datasets, without a single instance of a deterioration in performance. This is precisely why this scenario produced the best overall profile of model performance and quality and confirmed the presence of a cumulative positive effect from the simultaneous application of several optimisation methods.

As the number of computing nodes increased across all scenarios studied, an increase in the speed-up ratio and an improvement in overall data processing performance were observed; however, this improvement was uneven. The most pronounced effect was observed in the initial stages of scaling, whilst the rate of improvement gradually decreased as the number of nodes continued to increase. This trend was most clearly evident in the baseline control scenario, where, following an initial increase in the speed-up factor, the efficiency of parallel execution declined more rapidly than in the optimised variants. This indicated that, without additional improvements to the data structure, the organisation of computations and inter-node communication, scaling did not provide a proportional increase in performance.

*Table 6:* System scalability metrics in the baseline and optimised scenarios

| Scenario | Acceleration Factor S with 2 Nodes | Efficiency E with 2 Nodes | Acceleration Factor S with 4 Nodes | Efficiency E with 4 Nodes | Acceleration Coefficient S at 8 Nodes | Efficiency E at 8 Nodes |
|---|---|---|---|---|---|---|
| Base test scenario | 1.58 | 0.79 | 2.64 | 0.66 | 3.51 | 0.44 |
| Data structure optimisation | 1.65 | 0.83 | 2.89 | 0.72 | 3.92 | 0.49 |
| Optimisation of the computational process | 1.73 | 0.87 | 3.08 | 0.77 | 4.28 | 0.54 |
| Optimisation of inter-node interaction | 1.70 | 0.85 | 3.01 | 0.75 | 4.11 | 0.51 |
| Combined optimisation | 1.82 | 0.91 | 3.34 | 0.84 | 4.96 | 0.62 |

*Note:* the table shows the change in the speedup factor and parallel execution efficiency for the baseline, individual optimised and combined scenarios as the number of computing nodes increases.
*Source:* compiled by the authors.

In the baseline control scenario, the speed-up factor increased from 1.58 to 3.51; however, the parallel execution efficiency decreased from 0.79 to 0.44, indicating a loss of performance due to increased synchronisation overhead. After optimising the data structure, scalability became more stable: the speed-up factor reached 3.92, whilst the efficiency remained higher than in the baseline scenario, ranging from 0.83 to 0.49. Optimising the computational process yielded even better results:

the speed-up factor rose to 4.28, whilst the efficiency of parallel execution remained between 0.87 and 0.54. Optimisation of inter-node interaction demonstrated a similar positive trend, with an acceleration factor reaching 4.11 and efficiency ranging from 0.85 to 0.51. The best results were obtained in the combined scenario, where the speed-up factor was the highest at all scaling levels, reaching 4.96, whilst the parallel execution efficiency remained the most stable – ranging from 0.91 to 0.62. The summarised quantitative results of the system's scalability are presented in Table 6.

The data in the table show that as the number of nodes increased, system performance improved in all scenarios; however, it was the combined optimisation that provided the most balanced result. It was characterised not only by the highest acceleration factor, but also by the smallest decrease in parallel execution efficiency when moving to a higher level of scaling. In conclusion, the results obtained confirmed that the scaling effect depended significantly on the method of data processing organisation and that it was precisely the combination of several optimisation approaches that allowed the advantages of a distributed environment to be realised most fully.

Statistical analysis showed that, in the scenario involving data structure optimisation compared to the baseline, the model training time was significantly reduced by 12.4% – from 78.4 minutes to 68.7 minutes (p=0.012), RAM usage decreased by 9.1% – from 27.4 GB to 24.9 GB (p=0.018), and the average accuracy increased by 1.8% – from 0.887 to 0.903 (p=0.031). Thus, in this scenario, improvements in time, resources and quality were all statistically significant, although their magnitude was moderate compared with subsequent stages.

A more pronounced effect was observed in terms of optimising the computational process. Training time was significantly reduced by 24.7% – from 78.4 minutes to 59.0 minutes (p=0.006), whilst the average F1-score increased by 2.6% – from 0.867 to 0.890 (p=0.021). Thus, it was this particular scenario that yielded the greatest statistically significant performance gain among all isolated optimisation methods, whilst also ensuring a statistically significant improvement in classification quality.

In the scenario optimising inter-node interaction, the reduction in total execution time proved to be statistically significant (Kozhakhmetova et al., 2024; Gospodinova, 2022). Compared to the baseline scenario, this figure decreased by 15.6% – from 78.4 minutes to 65.9 minutes (p=0.017). This confirmed that reducing communication overhead and node synchronisation overhead yielded a tangible and statistically significant gain in performance, although the scale of this gain was less than that achieved by optimising the computational process.

The most significant results were obtained for combined optimisation. In this scenario, the model training time was significantly reduced by 34.8% – from 78.4 minutes to 51.1 minutes (p=0.004), RAM usage decreased by 14.2% – from 27.4 GB to 23.5 GB (p=0.011), the average accuracy increased by 3.1% – from 0.887 to 0.915 (p=0.019), and the F1-score rose by 3.8% – from 0.867 to 0.900 (p=0.008). The results showed that it was the combined use of several approaches that provided the

highest and statistically most significant improvement in both the performance and quality of the models. The summarised results of statistical significance are presented in Table 7.

*Table 7:* Statistically significant differences between the baseline and optimised scenarios.

| Script | Indicator | Base Value | Value after Optimisation | Change, % | Statistical Criterion | p-Value |
|---|---|---|---|---|---|---|
| Data structure optimisation | Training time, min | 78.4 | 68.7 | -12.4 | Paired Student's t-test | 0.012 |
| Data structure optimisation | RAM usage, GB | 27.4 | 24.9 | -9.1 | Paired Student's t-test | 0.018 |
| Data structure optimisation | Accuracy | 0.887 | 0.903 | +1.8 | Wilcoxon T-test | 0.031 |
| Optimisation of the computational process | Training time, min | 78.4 | 59.0 | -24.7 | Paired Student's t-test | 0.006 |
| Optimisation of the computational process | F1-score | 0.867 | 0.890 | +2.6 | Wilcoxon T-test | 0.021 |
| Optimisation of inter-node interaction | Total execution time, min | 78.4 | 65.9 | -15.6 | Paired Student's t-test | 0.017 |
| Combined optimisation | Training time, min | 78.4 | 51.1 | -34.8 | Paired Student's t-test | 0.004 |
| Combined optimisation | RAM usage, GB | 27.4 | 23.5 | -14.2 | Paired Student's t-test | 0.011 |
| Combined optimisation | Accuracy | 0.887 | 0.915 | +3.1 | Wilcoxon T-test | 0.019 |
| Combined optimisation | F1-score | 0.867 | 0.900 | +3.8 | Wilcoxon T-test | 0.008 |

*Note:* the table shows statistically significant differences between the baseline and optimised scenarios; results were considered statistically significant at $p<0.05$.

*Source:* compiled by the authors.

The results presented showed that all major improvements obtained in the optimised scenarios were statistically significant at $p<0.05$. The greatest confirmed effect was recorded for combined optimisation, whilst among the individual approaches, optimisation of the computational process proved to be the most effective.

A summary of the results from all experimental scenarios showed that the highest efficiency in processing big data in a distributed environment was achieved through combined optimisation, which integrated data structure optimisation, computational process optimisation and inter-node interaction optimisation. It was this approach that demonstrated the most significant reduction in training time, the greatest reduction in resource load and the best improvement in model quality metrics (Cristea et al., 2023; Smailov et al., 2025). Among the individual approaches, computational process optimisation proved to be the most effective, whilst data structure and inter-node interaction optimisation also

yielded positive results, albeit on a smaller scale. A summary comparison of all the scenarios studied is presented in Table 8.

*Table 8:* Comparative characteristics of the experimental scenarios.

| Scenario | Impact on Performance | Impact on Resource Usage | Impact on Model Quality | Overall Performance Assessment |
|---|---|---|---|---|
| Baseline control scenario | Baseline | Baseline | Baseline | Control scenario |
| Data structure optimisation | moderate improvement | moderate reduction in load | slight improvement | effective |
| Optimisation of the computational process | significant improvement | Stabilisation of computing resource usage | noticeable improvement | the most effective of the individual approaches |
| Optimisation of inter-node interaction | significant improvement | reduction in communication load | no marked improvement in model quality metrics | effective for distributed processing |
| Combined optimisation | maximum improvement | greatest reduction in load | best improvement | most efficient scenario |

*Note:* the table summarises the results of the baseline, individual optimised and combined scenarios based on the performance, resource utilisation and model quality indicators presented in the previous tables.

*Source:* compiled by the authors.

In conclusion, it was found that combining several optimisation methods within a single scenario proved to be more effective than applying each approach separately, confirming the feasibility of comprehensive optimisation of machine learning algorithms for processing large datasets in distributed intelligent systems.

## 4. DISCUSSION

The results obtained showed that improving the efficiency of machine learning algorithms in distributed intelligent systems is achieved not by a single isolated solution, but by a combination of several levels of optimisation. In our study, the baseline control scenario provided acceptable accuracy and F1-score values, but was accompanied by the highest time costs, high RAM usage and lower scalability. In contrast, all optimised scenarios yielded positive results, with the best outcome achieved by combined optimisation, which simultaneously reduced training time, lowered resource consumption and improved model quality. This implies that for big data in a distributed environment, it is not only the choice of algorithm that is crucial, but also the configuration of the entire system in which it is executed (Bezrukova et al., 2022; Ginters et al., 2020).

A key finding of our research was that none of the optimisation methods used led to a deterioration in model quality. This is of fundamental importance, as in real-world systems, speed-ups are often

achieved at the expense of simplifying processing or losing some useful information. In our study, by contrast, reductions in training time and memory usage were accompanied by increases in accuracy and F1-score. Consequently, the results indicate that optimisation in distributed intelligent systems should be viewed not as a secondary technical procedure, but as a key factor in improving both the performance and quality of machine learning.

The results of the data structure optimisation phase showed that correct array partitioning and the reduction of redundant features yield a consistent positive effect even without modifying the model itself. This is consistent with the findings of Li et al. (2024), who demonstrated that adaptive memory reservation in the Spark environment can improve the processing of heavy workloads by increasing the efficiency of memory management. In this study, this conclusion is empirically confirmed: optimising the data structure reduced training time and RAM usage without any loss of accuracy.

The results obtained are also consistent with the work by Atefinia and Ahmadi (2022), which evaluated Apache Spark MLlib algorithms on an intrusion detection dataset and demonstrated the relevance of distributed machine learning frameworks for high-load security data processing. This is important for the present study because the greatest gains from optimisation were observed precisely on the most resource-intensive datasets. Thus, our results support the idea that, for large datasets, the benefits of distributed learning are realised only if the entire execution system is properly organised.

A specific scenario for optimising the computational process in our own study yielded the strongest isolated effect: the greatest reduction in training time was accompanied by an improvement in the F1-score. This logic is consistent with the results of Sun et al. (2024), who demonstrated that efficient distributed training can be achieved through full communication-computation overlap. In this work, this was confirmed by the fact that it was the optimisation of the execution process, and not merely a change in data structure, that yielded the greatest isolated performance gain.

The work by Duan et al. (2024) takes a related approach by analysing device placement on computation graphs, which is important for determining how operations should be allocated across heterogeneous computing environments. Yuan et al. (2022) also demonstrated that decentralised training of large models in heterogeneous environments depends on the effective organisation of communication and computation. The results obtained confirm this systemic logic: the efficiency of an algorithm in a distributed environment is determined not only by its mathematical form, but also by how computations are distributed among nodes and devices. This was particularly evident in the combined scenario, where it was precisely this systemic coherence that yielded the best result.

The effectiveness of computational process optimisation in this work is consistent with the findings of Hu et al. (2022), who proposed a profiling and optimisation system for expediting distributed deep neural network training. In our own research, this manifested itself in the fact that the reduction in training time was accompanied by stabilisation of the CPU load and better resource utilisation. The

broader relevance of this result is also supported by Liang et al. (2024), who emphasised that communication-efficient large-scale distributed deep learning requires coordinated optimisation of computation, memory and data exchange. Consequently, the results obtained confirm that the effective implementation of the algorithm at the framework and runtime environment levels directly influences the overall system performance.

The optimisation scenario for inter-node interaction in our own study showed that reducing data transfer volumes and synchronisation overhead results in a sustained reduction in overall execution time. This is fully consistent with the work of Provatas et al. (2025), where it is emphasised that parameter server architectures are directly associated with the problem of communication overhead in distributed centralised learning. Similar conclusions are presented by Li et al. (2024), who analysed communication characteristics of distributed training and showed that communication patterns significantly affect overall training efficiency. In the present study, the greatest effect of this approach was observed specifically for network and transactional datasets, where inter-node communication was most intensive.

These results are supported by the findings of Castelló et al. (2023), who analysed the impact of MPI Allreduce on the distributed training of convolutional neural networks. Li et al. (2023) also demonstrated that resilient collective operations can support elastic deep learning by improving the organisation of collective communication. In this work, reducing overhead exchanges and synchronisation costs was not an end in itself, but it was precisely this that provided a significant performance gain. This confirms that communication optimisation is a prerequisite for effective scaling, rather than merely an additional improvement.

The results obtained are also in good agreement with the findings of Xin et al. (2025), who, in their work on Global-QSGD, demonstrated that allreduce-compatible quantisation can reduce communication costs in distributed learning while maintaining theoretical convergence guarantees. Singh and Kumar (2024) similarly examined gradient compression with sparsification and quantisation techniques as a way to increase the efficiency of distributed training. Although direct gradient quantisation was not used in this study, the principle itself is fully confirmed: reducing the communication load without compromising model quality is feasible and yields a significant performance gain. This is particularly important for interpreting the results of the inter-node optimisation scenario.

The findings of this study are corroborated by the results of Yan et al. (2024), who examined improved quantisation strategies for managing heavy-tailed gradients in distributed learning, and Makarenko et al. (2023), who analysed adaptive compression as a means of improving communication efficiency in distributed training. In this study, a similar logic was observed on a broader scale: optimising inter-node interaction and the combined scenario reduced communication overhead without compromising, and in some cases even improving, classification quality. This confirms that, for distributed machine learning, reducing communication costs does not necessarily mean a loss of model performance.

The findings of Sun et al. (2025) are also consistent with these results, as they demonstrated that optimising gradient compression for distributed training can improve the efficiency of training with second-order optimisers. The results of Cahya Indirman et al. (2023) further support the relevance of Apache Spark and Hadoop Distributed File System (HDFS) for addressing big data challenges through distributed machine learning. In this study, the combined scenario yielded precisely this systemic effect: a simultaneous reduction in time, a decrease in memory usage, and improvements in accuracy and F1-score. This allows us to conclude that comprehensive optimisation has the same direction of effect as that recorded by international studies for specialised communication acceleration and distributed processing mechanisms.

The results regarding RAM usage in this study are also corroborated by the work of Lee et al. (2023) and Dai et al. (2025), which demonstrate that mixed-precision training can reduce memory requirements and accelerate arithmetic operations without loss of accuracy. Although mixed precision was not used in this study, the direction of the results is entirely consistent: reducing the memory load is a critically important factor in improving overall efficiency, and this can be achieved without compromising model quality. This is particularly evident in the combined scenario, where reduced RAM usage was accompanied by an increase in accuracy and F1-score.

Another important aspect is confirmed by the findings of Ritter and Wolf (2023), who, in their study of the performance of distributed deep learning frameworks, demonstrated that even with identical hardware, different execution process organisations yield varying levels of performance, and that building performance models helps to identify bottlenecks and overheads. Our own study effectively reached the same conclusion: the baseline, partially optimised and combined scenarios differed not due to changes in hardware configuration, but due to different organisation of data, computations and node interactions. Thus, efficiency here was indeed determined by the execution architecture.

The results are also in good agreement with the conclusions of Go et al. (2025), who characterised the efficiency of distributed training from a power, performance and thermal perspective and showed that distributed learning must be evaluated not only by training time but also by broader system-level costs. In this work, this conclusion was confirmed by the fact that the greatest benefit from optimisation was observed precisely on the most resource-intensive datasets, and scalability was found to depend on the optimisation scenario, not just on the number of nodes.

The results of the study are also consistent with the work of Hordiychuk-Bublivska et al. (2021), which emphasises that the adaptation and refinement of big data processing methods are a prerequisite for the effective functioning of distributed information systems. In the present study, this proposition was directly confirmed: no single stage achieved the same effect as comprehensive optimisation, which once again points to the need for systemic rather than local solutions when working with large datasets.

Thus, all the studies reviewed support the main conclusion of our own research: for distributed intelligent systems, the best results are achieved not through isolated improvements, but through the coordinated optimisation of data structure, computational processes and inter-node interactions. It is precisely this approach in this work that yielded the greatest reduction in training time, the greatest reduction in resource load, the highest scalability, and the greatest improvement in accuracy and F1-score. This means that the results obtained are not only of local practical significance but also support a broader scientific trend, according to which the effectiveness of machine learning in distributed environments is determined by the systemic coordination of all components of the process.

## 4. CONCLUSIONS

The study found that the effectiveness of machine learning algorithms in distributed intelligent systems is determined not by the isolated improvement of a single parameter, but by the coordinated optimisation of data structure, computational processes and inter-node interaction. A comparison of experimental scenarios showed that the baseline version of the work, without targeted optimisation, provided acceptable classification quality but was associated with the highest time and resource costs. This confirmed that, for systems designed to handle large datasets, the accuracy of the model cannot be considered in isolation from performance, memory usage and scalability.

The study confirmed that optimising the data structure yields a consistent initial improvement: training time was reduced by 12.4%, RAM usage decreased by 9.1%, and classification accuracy increased by 1.8%. This allows us to consider rational data partitioning and the reduction of redundant features as a prerequisite for further performance improvements. The greatest isolated effect was achieved during the optimisation of the computational process: training time decreased by 24.7%, processing speed increased by 21.3%, and the overall classification quality metric improved by 2.6%. It was also found that optimising inter-node communication is important for reducing communication overhead, as it resulted in an 18.9% reduction in network load and a 15.6% reduction in total execution time.

The best overall result was achieved in the combined scenario, which incorporated all the optimisation approaches under investigation. With this approach, training time was reduced by 34.8%, RAM usage decreased by 14.2%, accuracy increased by 3.1%, and the F1-score by 3.8%. This provides grounds for asserting that it is comprehensive optimisation that ensures the best balance between performance, resource efficiency, and model quality. The practical significance of the results lies in the possibility of using the proposed optimisation logic when designing distributed systems for working with network, transactional, and telemetry data. The limitations of the study include the simulated nature of the environment, the fixed hardware configuration and the limited set of scenarios. Further research should focus on verifying the results in heterogeneous clusters, on more complex models and with additional mechanisms for reducing communication costs.

## 5. REFERENCES


Aach, M., Inanc, E., Sarma, R., Riedel, M., Lintermann, A., 2023, Large-scale performance analysis of distributed deep learning frameworks for convolutional neural networks. *J. Big Data* **10,** 96.

Atefinia, R., Ahmadi, M., 2022, Performance evaluation of Apache Spark MLlib algorithms on an intrusion detection dataset. *J. Comput. and Security* **9**(1), 57-69.

Azeroual, O., Nikiforova, A., 2022, Apache Spark and MLlib-based intrusion detection system or how big data technologies can secure the data. *Information* **13**(2), 58.

Barham, P., Chowdhery, A., Dean, J., Ghemawat, S., Hand, S., Hurt, D., Isard, M., Lim, H., Pang, R., Roy, S., Saeta, B., Schuh, P., Sepassi, R., Shafey, L., Thekkath, C.A., Wu, Y., 2022, Pathways: asynchronous distributed dataflow for ML. arXiv 2203.12533.

Bezrukova, N., Huk, L., Chmil, H., Verbivska, L., Komchatnykh, O., Kozlovskyi, Y., 2022, Digitalization as a Trend of Modern Development of the World Economy. *WSEAS Transact. Envir. Devel.* **18**, 120-129.

Brito, C.V., Ferreira, P.G., Portela, B.L., Oliveira, R.C., Paulo, J.T., 2023, Privacy-preserving machine learning on Apache Spark. *IEEE Access* **11**, 127907-127930.

Buhai, D., Zhuchenko, A., Zhuchenko, O., Kovaliuk, D., Skladannyy, D., 2026, Improving the effectiveness of medical decision support systems based on machine learning and cloud services. *Tech. Audit Product. Reserv*. **1**(2), 75-84.

Cahya Indirman, M.D., Wiriasto, G.W., Irfan Akbar, L.A.S., 2023, Distributed machine learning using HDFS and Apache Spark for big data challenges. *E3S Web of Conf.* **465**, 02058.

Castelló, A., Catalán, M., Dolz, M.F., Quintana-Ortí, E.S., Duato, J., 2023, Analyzing the impact of the MPI Allreduce in distributed training of convolutional neural networks. *Computing* **105**, 1101-1119.

Celledoni, E., Owren, B., Ruthotto, L., Yang, T.N., 2025, Mixed precision training of neural ODEs. arXiv 2510.23498.

Cristea, V.-M., Baigulbayeva, M., Ongarbayev, Y., Smailov, N., Akkazin, Y., Ubaidulayeva, N., 2023, Prediction of Oil Sorption Capacity on Carbonized Mixtures of Shungite Using Artificial Neural Networks. *Proc.* **11**(2), 518.

Dai, W., Jia, Z., Bai, Y., Sun, Q., 2025, Convergence-aware operator-wise mixed-precision training. *CCF Trans. High Perform. Comput.* **7**, 43-57.

Duan, S., Ping, H., Kanakaris, N., Xiao, X., Kyriakis, P., Ahmed, N.K., Zhang, P., Ma, G., Capota, M., Nazarian, S., Willke, T.L., Bogdan, P., 2024, A structure-aware framework for learning device placements on computation graphs. arXiv 2405.14185.

Ferrag, M.A., Maglaras, L., Moschoyiannis, S., Janicke, H., 2020, Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. *J. Inf. Secur. Appl.* **50,** 102419.

Ginters, E., Gutiérrez, J.M., Mendívil, E.G., 2020, Mapping of Conceptual Framework for Augmented Reality Application in Logistics. *2020 61st Int. Sci. Conf. Inf. Tech. Manag. Sci. Riga Tech. Univ. ITMS 2020*, 9259302.

Go, S., Park, J., More, S., Wu, H., Wang, I., Jezghani, A., Krishna, T., Mahajan, D., 2025, Characterizing the efficiency of distributed training: a power, performance, and thermal perspective. *Proc. 58th IEEE/ACM Int. Symp. Microarchitecture (MICRO).*

Gospodinova, E., 2022, Analysis and Development of an Algorithm to increase the Energy Efficiency of Electrical Street Lighting Systems Using an Artificial Neural Network. *Proceed. 2022 6th Eur. Conf. Elect. Eng. Comp. Sci. ELECS 2022*, 145-150.

Gospodinova, E., 2023, Mathematical Modeling and Planning of Energy Production using a Neural Network. *WSEAS Transact. Pow. Syst.* **18**, 39-48.

Guliyev, H.B., 2019, Reactive power adaptive control system in networks with distributed generation based on fuzzy set theory. *2019 Int. Conf. Artif. Intell. Data Proc. Symp. IDAP 2019*, 8875963.

Hordiychuk-Bublivska, O.V., Beshley, M.I., Kyryk, M.I., Klymash, M.M., 2021, Increasing the efficiency of processing large volumes of information using the method of distributed data analysis. *Telecommun. Inf. Technol.* **2**(71), 15-23.

Hu, H., Jiang, C., Zhong, Y., Peng, Y., Wu, C., Zhu, Y., Lin, H., Guo, C., 2022, dPRO: a generic profiling and optimization system for expediting distributed DNN training. arXiv 2205.02473.

Jiang, Y., Han, S.Y., Li, Z., Ning, H., Kim, J.S., 2025, Comparative analysis of human mobility patterns: utilizing taxi and mobile (SafeGraph) data to investigate neighbourhood-scale mobility in New York City. *Ann. GIS* **31**(3), 387-411.

Kerimkhulle, S., Kuttykozhayeva, S., Turtkarayeva, G., Seitova, T., Ospanova, N., Iskakov, Y., 2025a, Forecasting for Japanese Yen to Dollar Exchange Rate Based on Fuzzy Logic. *Lect. Not. Networks Syst.* **1489**, 18-26.

Kerimkhulle, S., Kuttykozhayeva, S., Turtkarayeva, G., Seitova, T., Ospanova, N., Toleubay, D., 2025b, Using the Fuzzy Logic Models for Analysis of Time Phases of Crude Oil Prices: Brent-Europe. *Lect. Not. Networks Syst.* **1489**, 9-17.

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. *Energ.* **17**(17), 4446.

Lee, W., Sharma, R., Aiken, A., 2023, Training with mixed-precision floating-point assignments. arXiv.
Lewandowski, B., Kosson, A., 2023, Memory efficient mixed-precision optimizers. arXiv 2309.12381.

Li, B., He, X., Yu, J., Wang, G., Song, Y., Pan, S., Gu, H., 2024, Adaptive memory reservation strategy for heavy workloads in the Spark environment. *PeerJ Comput. Sci.* **10,** e2460.

Li, J., Bosilca, G., Bouteiller, A., Nicolae, B., 2023, Elastic deep learning through resilient collective operations. *Proc. SC '23 Workshops.,* 44-50.

Li, S., Qin, Y., Jiang, Z., Yang, W., 2022, Efficient communication scheduling for parameter synchronization of DML in data center networks. *IEEE Trans. Netw. Sci. Eng.* **9**(4), 1970-1985.

Li, W., Liu, X., Li, Y., Jin, Y., Tian, H., Zhong, Z., Liu, G., Zhang, Y., Chen, K., 2024, Understanding communication characteristics of distributed training. *Proc. 8th Asia-Pacific Workshop Netw*., 1-8.

Liang, F., Zhang, Z., Lu, H., Leung, V.C.M., Guo, Y., Hu, X., 2024, Communication-efficient large-scale distributed deep learning: a comprehensive survey. arXiv 2404.06114.

Lin, H., Yan, M., Ye, X., Fan, D., Pan, S., Chen, W., Xie, Y., 2023, A comprehensive survey on distributed training of graph neural networks. *Proc. IEEE* **111**(12), 1572-1606.

Liu, J., Huang, J., Li, Y., Li, Z., Lyu, W., Jiang, W., Wang, J., 2024, SGC: similarity-guided gradient compression for distributed deep learning. *Proc. IEEE/ACM 32nd Int. Symp. Qual. Serv. (IWQoS),* 1-6.

Makarenko, M., Gasanov, E., Islamov, R., Sadiev, A., Richtárik, P., 2023, Adaptive compression for communication-efficient distributed training. arXiv 2211.00188.

Massenio, P.R., Tipaldi, M., Rizzello, G., Brescia, E.,Cascella, G.L., Naso, D., 2023, Gain-Scheduled Structured Control in DC Microgrids. *IEEE Transact. Cont. Syst. Tech.* **31**(6), 2571-2583.

Provatas, N., Konstantinou, I., Koziris, N., 2025, A survey on parameter server architecture: approaches for optimizing distributed centralized learning. *IEEE Access* **13,** 30993-31015.

Ritter, M., Wolf, F., 2023, Extra-Deep: automated empirical performance modeling for distributed deep learning. *Proc. SC '23 Workshops,* 1345-1356.

Singh, S., Kumar, S., 2024, Efficient distributed training through gradient compression with sparsification and quantization techniques. arXiv 2502.07634.

Smailov, N., Zilgarayeva, A., Pavlov, S., Turusbekova, B., Sabibolda, A., 2025, Optimization of non-invasive glucose monitoring accuracy using an optical sensor. *Inf. Autom. Pom. Gospod. Ochr. Środowiska,* **15**(4), 65-70.

Sun, B., Liu, W., Pauloski, J.G., Tian, J., Jia, J., Wang, D., Zhang, B., Zheng, M., Di, S., Jin, S., Zhang, Z., Yu, X., Iskra, K.A., Beckman, P., Tan, G., Tao, D., 2025, COMPSO: optimizing gradient compression for distributed training with second-order optimizers. *Proc. ACM SIGPLAN Symp. Princ. Pract. Parallel Program. (PPoPP),* 212-224.

Sun, W., Qin, Z., Sun, W., Li, S., Li, D., Shen, X., Qiao, Y., Zhong, Y., 2024, CO2: efficient distributed training with full communication-computation overlap. arXiv 2401.16265.

Tu, J., Yang, L., Cao, J., 2025, Distributed machine learning in edge computing: challenges, solutions and future directions. *ACM Comput. Surv.* **57**(5), 132.

Tyagi, S., Sharma, P., 2023, Scavenger: a cloud service for optimizing cost and performance of ML training. *Proc. IEEE/ACM Int. Symp. Cluster, Cloud Internet Comput. (CCGrid),* 403-413.

University for Business and Technology, 2026, Software Testing Lab. Available from: https://www.ubt-uni.net/en/ubt-en/labs/software-testing-lab/ (accessed 10 July 2026).

Will, J., Thamsen, L., Bader, J., Scheinert, D., Kao, O., 2022, Ruya: memory-aware iterative optimization of cluster configurations for big data processing. *Proc. IEEE Int. Conf. Big Data,* 161-169.

Xin, J., Canini, M., Richtárik, P., Horváth, S., 2025, Global-QSGD: allreduce-compatible quantization for distributed learning with theoretical guarantees. *Proc. Mach. Learn. Syst. Workshop (MLSys),* 216-229.

Yan, G., Li, T., Xiao, Y., Hou, H., Song, L., 2024, Improved quantization strategies for managing heavy-tailed gradients in distributed learning. arXiv 2402.01798.

Yuan, B., He, Y., Davis, J.Q., Zhang, T., Dao, T., Chen, B., Liang, P., Ré, C., Zhang, C., 2022, Decentralized training of foundation models in heterogeneous environments. arXiv 2206.01288.