International Journal of Safety and Security Engineering
Vol. 15, No. 3, March, 2025, pp. 563-572
Journal homepage: http://iieta.org/journals/ijsse

Parallel and GPU-Based Optimization of XGBoost and Neural Networks for Effective Landmine Classification

Lesia Mochurad[1], Nataliya Shakhovska[1,2], Jamil Abedalrahim Jamil Alsayaydeh[3*], Mohd Faizal Yusof[4]

[1] Department of Artificial Intelligence, Lviv Polytechnic National University, Lviv 79013, Ukraine
[2] Brunel University, London UB8 3PH, UK
[3] Department of Engineering Technology, Fakulti Teknologi Dan Kejuruteraan Elektronik Dan Komputer (FTKEK), Universiti Teknikal Malaysia Melaka (UTeM), Melaka 76100, Malaysia
[4] Faculty of Resilience, Rabdan Academy, Abu Dhabi 22401, United Arab Emirates

Corresponding Author Email: jamil@utem.edu.my

https://doi.org/10.18280/ijsse.150315

Received: 10 February 2025
Revised: 20 March 2025
Accepted: 25 March 2025
Available online: 31 March 2025

Keywords:
mine clearance, classification, neural networks, humanitarian demining, GPU acceleration, parallel computing

ABSTRACT

The problem of mine clearance in both open and closed areas remains highly relevant in the modern world, especially in the context of military conflicts, humanitarian crises, and post-war reconstruction processes. Traditional mine detection methods require significant human and technical resources, making the demining process costly, time-consuming, and potentially dangerous for operators. Therefore, there is a need to develop automated systems capable of ensuring high accuracy, efficiency, and speed in identifying explosive objects, thereby enhancing the safety of those conducting the operations. Existing landmine classification methods face limitations in speed, scalability, and deployment feasibility due to computational constraints and lack of optimization. This paper presents a mine classification method based on a combination of neural networks and gradient boosting, aimed at improving the accuracy and speed of the recognition process. Two main optimization strategies are proposed: (1) data-driven and algorithmic parallelization, which improve training speed and computational efficiency; and (2) GPU-accelerated model training to leverage parallel processing capabilities. A series of experiments were conducted, and the results confirmed the effectiveness of the proposed methods. For open environments, the classification accuracy reached 94.32% for gradient boosting and 93.89% for neural networks, while for closed environments, the accuracy was 93.25% and 92.75%, respectively. The optimization allowed for a fivefold increase in model training speed due to parallel computations and GPU data processing, making the proposed method suitable for real-world applications. An analysis of the results indicates the potential of this approach not only for further improvement of automated mine clearance systems but also for solving other classification and object identification tasks in complex environments.

1. INTRODUCTION

The presence of mines in any area threatens human life and health and impedes movement and access to resources and infrastructure. Mine clearance is a key step in restoring peaceful life in affected areas and ensuring the safety of the population. Mines are indiscriminate weapons with unfortunate and far-reaching consequences even after conflicts end. They have significant social, economic, and environmental impacts as munitions and tactical barriers [1]. While relatively cheap and easy to deploy, they are time-consuming, dangerous, and expensive to remove. According to the 2022 report [2], sixty countries and territories remain contaminated by anti-personnel landmines, with fifty casualties reported in 2021. Of the 5,544 people killed in 2021, 4,200 were civilians. In Myanmar alone, since 2023, 1052 cases of landmines among civilians have been officially recorded [3]. The publication [4] shows what types of ammunition and explosives have been used in Ukraine since the beginning of 24 February 2022. The article [5] describes that since 2022 to date, about 1,000 injuries and fatalities have been recorded with landmines and that about 1-2 million mines are currently present on the territory of Ukraine.

Traditional demining methods [6], which involve the direct use of people and animals - are slow, insufficiently effective, and dangerous for rescue workers. It takes a long time to train such a demining specialist. Also, do not forget the possibility that a protective suit may not withstand an explosion and, as a result, the deminer will be injured. The deminer also needs to know the exact location of the mine, and in today's warfare, these weapons are increasingly being camouflaged so that the human eye will not be able to recognize the mine among the grass, mounds of earth, etc. To solve this problem, artificial intelligence methods can be applied [7]. Their use can significantly improve demining processes and reduce risks to human life and health. With the growing threat of mines

around the world [8], as well as the rapid development of artificial intelligence technologies, this research is extremely relevant and important for improving the efficiency and safety of the demining process.

This paper examines the application of artificial intelligence methods to mine action, in particular, the classification of different types of mines based on magnetometer data. The analysis covers both theoretical aspects and practical implementations, which makes it possible to assess the advantages and limitations of using artificial intelligence in this area, as well as outline the prospects for further research and technology development. Particular attention is paid to the optimization of the model training process, including the implementation and comparison of different approaches [9-11]. Special emphasis is placed on the importance of parallel computing techniques, which significantly accelerate the training and inference phases, enabling real-time or near-real-time classification [12-14]. A description of the implemented methods, their results, and an analysis of the feasibility of their application are presented. Additionally, quantitative metrics are provided, demonstrating that optimization of the training process has a minimal impact on the quality of the results. At the same time, parallelization ensures computational efficiency without compromising accuracy, keeping the differences within the limits of permissible errors.

The purpose of this paper is to investigate the effectiveness of machine learning models, in particular neural networks and XGBoost, for classifying mines or their absence in open and closed environments, as well as to analyze the impact of optimization methods (computational parallelization, data distribution, GPU usage) on the learning speed and accuracy of the models. The proposed approaches demonstrate the advantages of using modern computing architectures in machine learning tasks.

The main contribution of the work:

•An optimized neural network architecture is proposed that, due to thread parallelization, provides a significant reduction in training time by a factor of five compared to standard methods. This allows for efficient use of available computing resources without reducing the accuracy of the model, making the proposed approach competitive in tasks requiring rapid processing of large amounts of data.

•A combined XGBoost optimization approach is developed and implemented, integrating thread parallelization and GPU computing. This results in a significant acceleration of the training process, up to 5-6 times compared to the standard implementation, while maintaining high accuracy of the results.

•A comparative analysis of the performance of the neural network and XGBoost for classification tasks in both open and closed environments is carried out. The results show that although XGBoost provides faster training, the neural network has a greater potential for adaptation and improvement, especially in tasks requiring work with heterogeneous data.

•For the first time, a comprehensive study of the impact of optimization approaches on models for classifying open and closed environments is performed. The proposed solutions have demonstrated efficiency and competitiveness compared to existing methods, opening up new opportunities for their application in real-world environment analysis.

The article consists of several main sections: the “Related Works” section analyses existing research in the field of mine classification and model optimization, while the “Problem Statement” section identifies the main challenges faced by current methods. In the “Proposed Methodology” section, we propose an approach that includes a detailed description of the datasets used, the neural network architecture, parallelization, and optimization tasks for XGBoost. In the “Analysis of Numerical Experiments” section, we present the results of experiments with both approaches, comparing the accuracy and training time. The final section “Discussion” presents the interpretation of the results and recommendations for further research. Handle more complex data, and integration with XAI methods.

2. RELATED WORKS

Barnawi et al. [15] used data collected by a UAV equipped with a magnetic magnetometer. This data contains information about the earth's magnetic field, which can be used to detect mines. To analyze the data, the authors use deep learning methods, in particular, neural networks. These models are trained to recognize the characteristics of mine objects based on magnetic data. Data processing is performed at the edges of the network, which reduces latency and preserves bandwidth. This is especially important for real-time mine detection tasks. The downside is the cost of this application and implementation. However, the proposed approach provided an accuracy of 97.8%, which is a significant result for mine signature recognition.

Article [16] is devoted to the research and development of methods for detecting and classifying mines based on ground penetrating radar (GPR) data [17, 18] using multimodal feature fusion. The authors describe the support vector method used to solve the problem for a dataset with multimodal feature fusion. The authors achieved an accuracy of 91.1% on the validation data.

Šipoš and Gleich [19] described the process of radar development and construction, including the materials used, characteristics, and principles of operation. An important part is the description of methods for optimizing energy consumption and reducing equipment weight. This study achieved a detection accuracy of 92.5%.

Jiao et al. [20] proposed a cellular decomposition extraction method for planning a coverage path in a polygonal area. The proposed method divides a complex polygon into different subregions, and then a path is determined for each subregion. The value of this work is to optimize the drone's route, thereby covering the maximum scanning area to collect data that will be used for training. The proposed approach was able to cover 98.5% of the test area.

Recent advances in drone-based remote sensing using lightweight multispectral and thermal infrared sensors allow for the rapid detection of landmine contamination at long distances. The methodology was proposed to detect dispersed plastic mines that use liquid explosives packed in a plastic or plastic case [21]. This makes it impossible to detect such explosives with a metal detector. Therefore, image processing techniques were used to show search results in a specific area. The authors used Faster R-CNN, which showed an accuracy of 99.3% for the test set and 71.5% for the validation set.

Pryshchenko et al. [22] described an approach to object detection and classification using a special set of ultra-wideband (UWB) pulsed GPR systems. The authors used a GPR system with one transmitting and four receiving antennas, which allowed them to collect signals received at different angles. The sums and differences of the signals received by two of the four antennas were merged into one

long signal, which increased the number of reflections and improved the accuracy of object detection. After that, an artificial neural network was used to classify the detected objects as mines or other objects [23]. The results showed that the proposed approach improved the accuracy of mine detection and classification compared to traditional methods. During the work, the authors achieved an accuracy of 90% for the validation data. The accuracy of mine detection was 98.5%, and the accuracy of classification of mines and other objects was 96.5%. In addition, the proposed approach reduced the number of false positives by 30%.

Vivoli et al. [24] presented a deep learning approach to detecting surface mines using real-time optical imagery. The authors used a dataset of optical images of mines and non-mines and trained a deep convolutional neural network (CNN) to classify the images as mines or non-mines. The results showed that the proposed approach achieved a high mine detection accuracy of 98.5% and a low false positive rate of 1.5%. The approach also enabled real-time mine detection, with a processing time of less than 100 milliseconds per image. The authors conclude that the proposed approach has the potential to significantly improve the efficiency and accuracy of mine detection and can be used for real-time mine detection in a variety of environments.

The study [25] analyses the effectiveness of pre-trained CNN models for classifying B-scan GPR images, particularly for detecting objects below the surface. Experimental results on the DECKGPRHv1.0 dataset showed that ResNet achieves the highest accuracy, demonstrating high transfer learning efficiency without fine-tuning. This research can be useful for demining applications where GPR images are used to identify underground objects, including mines, through automated signal classification using deep learning. At the same time, the work has certain limitations, including the use of only one dataset, the lack of comparison with newer architectures, and the lack of analysis of model performance and robustness to real-world conditions.

The authors of the study [26] focused on improving the analysis of sonar images for detecting objects in the underwater environment using CNN (VGG-16) with a weighted feature fusion technique, which allows achieving an accuracy of 86-91%. However, this approach has several limitations: it is focused exclusively on underwater conditions, does not include computational optimization to speed up the model, and is inferior in accuracy to other methods such as gradient boosting and neural networks used in the second study. In addition, the lack of GPU acceleration and parallel computing may make it difficult to use the method in real-world settings where the speed and adaptability of the algorithm are important [27].

In summary, this research is relevant due to the need for effective, cost-efficient, and scalable approaches to mine classification in open and closed environments, which is an important task for ensuring the security and reconstruction of territories in the post-conflict period [28]. Existing methods often have limitations in speed, accuracy, or adaptability to heterogeneous conditions, and their implementation may be impractical due to the high costs or complexity of reproduction [29]. In this context, our optimized approaches based on modern artificial intelligence methods and computing architectures can significantly improve the efficiency of model training, ensuring their accuracy and reducing costs, which opens up prospects for the widespread use of such technologies in humanitarian demining.

3. PROBLEM STATEMENT

The task of mine detection and classification based on numerical data from GPR is to find the optimal function f that minimizes the penalty for incorrect mine predictions. This can be achieved by training the model on existing data and validating its performance on new data. Formally, the problem can be formulated as follows.

Let $X = \{x_1, x_2, ..., x_n\}$ be a vector of numerical features obtained from GPR, and $Y = \{y_1, y_2, ..., y_n\}$ be a set of corresponding labels, where $y_i > 0$ if there is a mine at location x_i and $y_i = 0$ if there is not. The task is to find a function $f: XR$ that minimizes the loss function $L(f)$, which measures the average number of classification errors:

$$L(f) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 + \lambda * R(f)$$

where, $\frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2$ is the classification error loss function, where $f(x_i)$ is the model prediction for the i-th feature vector, y_i is the corresponding label. n is the total number of samples in the training set, λ is the regularisation coefficient, which determines the balance between minimizing classification errors and limiting model complexity, and $R(f)$ is a regularisation function that usually measures model complexity. The purpose of regularisation is to prevent overfitting of the model by reducing its complexity.

4. PROPOSED METHODOLOGY

4.1 Dataset overview

In the course of analyzing the available datasets, we found datasets containing the results of GPR scanning [30]. The first one describes the results of scanning an open area from above, and the second one describes the results of a robot or drone (not precisely described) in an enclosed area. The data is a feature vector with the target characteristic M - the type of mine or its absence. In total, there are 4 types of mines (anti-personnel, anti-tank, etc.) and 5 types of mines. The data distribution is shown in Figure 1 and Table 1.

The features described are distance, voltage, and surface type, which were collected from the GPR scan. A more detailed description is not possible because the data sample was already provided normalized, without a description of the data before normalization [31]. No description of other features is provided, it can be assumed that these may be the time of the electromagnetic wave acquisition and the frequency at which the radar operates or other characteristics obtained during the scan [32]. There are about 300,000 records in each dataset - for open and closed spaces.

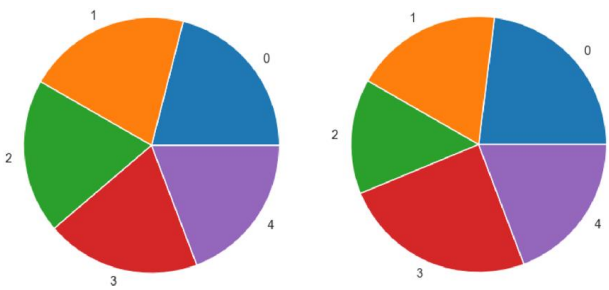

Figure 1. Class distribution: Open space and enclosed space

Table 1. Class distribution in percentages

Class Number	Distribution (Open Space)	Distribution (Enclosed Space)
0	21	23
1	20.71	18.71
2	19.526	14.52
3	19.526	24.52
4	19.23	19.23

Among the disadvantages of both datasets is the lack of primary data and the absence of coordinates. The latter is especially important for integrating this model into real systems, so recording exactly where any non-standard signal was found is a mandatory requirement for these systems.

All input features in both datasets were already normalized using min-max scaling to the [0,1] range. To improve training efficiency and reduce noise, redundant features were excluded based on low variance analysis. Feature vectors were additionally standardized to ensure uniform contribution to model training. Class imbalance was handled by applying stratified sampling during training set generation, ensuring proportional representation of each class in parallelized subsets.

4.2 Proposed neural network

To solve this problem - classification of mines (or their absence, which is also a separate class) - a neural network of the following architecture was used (see Figure 2).

Input layer - 5 neurons - the number of features in the dataset;

FCL = Dense(64, activation='LeakyReLU');

Dropout layer = 0.2 to avoid overfitting by discarding random weights;

FCL = Dense(64, activation='LeakyReLU' , kernel_regulariser = regularizers.L1L2(l1=1e-5, l2=1e-4));

Dropout layer = 0.3;

FCL = Dense(16, activation = 'LeakyReLU', kernel_regulariser=regularizers.L1L2(l1=1e-5, l2=1e-4));

Dropout layer = 0.3;

FCL = Dense(8, activation = 'LeakyReLU', kernel_regulariser=regularizers.L1L2 (l1=1e-5, l2=1e-4));

Dropout layer = 0.3;

Output layer - 5 neurons, according to the number of classes, activation - softmax.

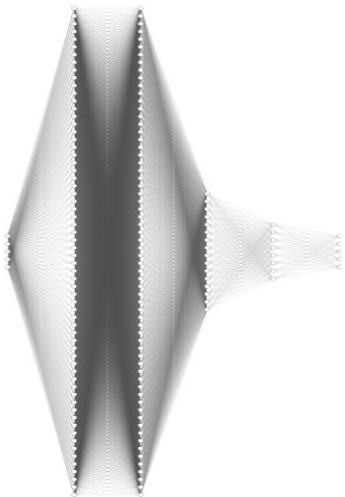

Figure 2. Network architecture

Batch normalization is applied after each layer to improve both training speed and result quality. LeakyReLU activation functions were used in the hidden layers to prevent gradient vanishing during training. ReLU was not used because the input contains values less than zero, which would lead to a large number of inactive neurons. LeakyReLU addresses this issue by allowing a small gradient for negative values. If ReLU were used, the sum of all neurons would be equal to 1, which is not a good practice as it constrains the layer's total output. The final layer utilizes softmax, which produces class probabilities, ensuring their sum equals 1. Tangent and sigmoid activation functions were not considered since they are primarily suited for binary classification, which does not align with the given task.

Regularization is implemented to enhance the model's generalization and prevent overfitting. This feedforward neural network architecture was proposed because, with additional loss functions such as sparse_categorical_crossentropy and the dataset's limited number of features, the classification task can be effectively solved. In contrast, convolutional networks would not be able to extract patterns from such a small number of input features. A residual network could be considered with an increased number of layers; however, this would significantly complicate the architecture and increase training time. Moreover, both residual and convolutional networks are primarily used for image processing, which is not the focus of this study.

4.3 Parallelization tasks

This paper implements two approaches to parallelization: data-driven and algorithmic parallelization. The general concept is shown in the diagram below (see Figure 3). Data-driven parallelization involves dividing the training set into n equal parts and gradually training each of them. Upon completion of the training, the results are summarised and the output is a decision on whether the sample belongs to the mine class or not.

Algorithmic parallelization involves the independent training of several models on different subsets of data corresponding to separate classes. Each model specializes in detecting the characteristics of objects of a particular class. After the parallel training is completed, the results of the models are aggregated, and the class of the object is determined by the maximum value of the corresponding function. Since the number of classes in this task is 5, the corresponding number of separate processes is used. Depending on the distribution of data in the dataset, the speed of model training may vary due to the uneven number of samples in different classes. Therefore, the total execution time is determined by the upper bound, i.e., until all processes are completed.

To implement parallel training of neural networks, separate processes are used, each of which is responsible for training its own model and generating results. Processes operate in isolation, have their own memory area, and execute independently of each other. Unlike threads, each process can contain several threads, which can significantly improve performance. Threads are high-level primitives that provide asynchronous computing and data handling.

Figure 3 shows a visualisation of the approach using multiple neural networks. It is worth noting that it does not represent a single continuous network, but rather individual

models, of which five are used in this paper, with the architecture described above.

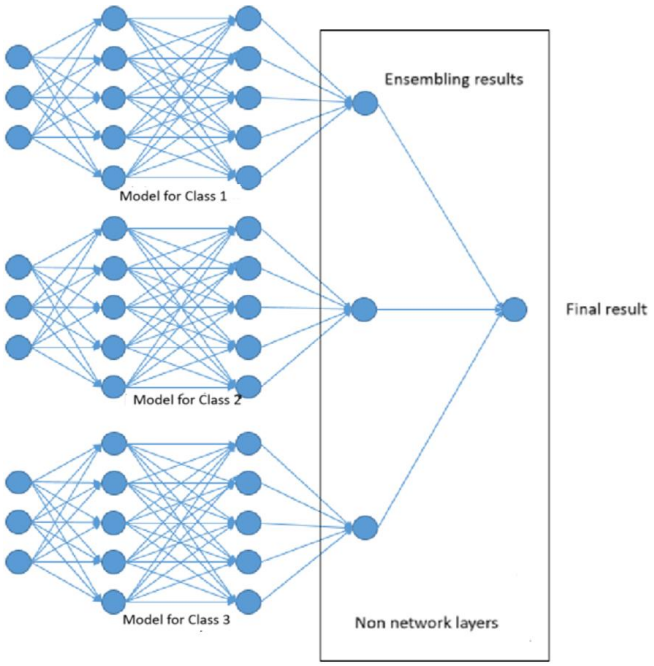

Figure 3. Algorithmic parallelization representation

To clearly differentiate the data-driven and algorithmic parallelization strategies, the following pseudocode (see Algorithm 1) outlines the respective workflows. It reflects the two types of parallelization implemented in this study.

Algorithm 1: *Pseudocode for data-driven vs. algorithmic parallelization workflow*

```
BEGIN
  SELECT parallelization_strategy
  IF strategy == "Data-Driven Parallelization" THEN
    SPLIT dataset into n equal chunks
    FOR each chunk IN parallel:
      Train identical model on chunk
    END FOR
    Aggregate all model outputs (e.g., majority vote or averaging)
  ELSE IF strategy == "Algorithmic Parallelization" THEN
    FOR each class c IN parallel:
      Extract data for class c
      Train model specific to class c
    END FOR
    Aggregate outputs from all class-specific models using max-confidence selection
  END IF
END
```

The data-driven strategy uniformly splits the dataset and trains identical models independently, aggregating their predictions at the end. The algorithmic strategy trains distinct models, each specialized in one class, and then combines their outputs using the class with the highest prediction score.

4.4 The case of XGBoost

To evaluate the effectiveness of the proposed neural network, its performance was compared with XGBoost (eXtreme Gradient Boosting), a gradient boosting algorithm that uses decision tree ensembles [33, 34]. The main characteristics of XGBoost are boosting with sequential model training, where each subsequent model corrects the errors of the previous one, gradient descent to minimize the loss function by updating the model in the direction of decreasing the gradient, regularisation using the built-in L1 (Lasso) and L2 (Ridge) methods, that control the coefficient values and eliminate redundant features, processing of missing values by taking into account missing data when generating splits in decision trees, and optimization of speed and performance through special optimizations for fast learning on large amounts of data. The XGBoost algorithm is based on an ensemble of decision trees added in stages to improve forecasting. Each new tree generates a partitioning based on the loss gradient of previous models, which helps to improve forecast accuracy and reduce errors. Thus, comparing XGBoost with a neural network allows us to evaluate the effectiveness of the proposed approach in solving the mine classification problem.

Algorithmically, you can split the training into several threads. Each thread will receive a corresponding unoccupied branch for computation, and when a branch ends with a leaf and has no children, the thread is released and ready for the next branch, if there are any unbuilt branches in the queue by that time [35]. In this paper, we use the number of threads for XGBoost = 8.

It is also proposed to use this algorithmic approach for distributed data. Instead of training 1 XGBoost on the entire sample, the data will be divided into n uniform sets and the results of their training will be bagged. In this case, the use of threads for branching is also used.

In comparison, XGBoost will be used with a histogram algorithm. Instead of using an exact algorithm with a greedy search for the best partition that uses CPU, which can be slow for large datasets, gpu_hist uses a histogram-based algorithm that is faster and requires less memory. For a detailed description of the features of histogram computing, see study [36]. Note that these manipulations are possible only for GPUs from NVIDIA and CUDA 3.5+ installed, as their solutions have tensor kernels, which are currently not available in AMD and Intel GPUs.

When training the XGBoost model using gpu_hist, the data is divided into groups using histograms. Each group contains data with similar values. The algorithm then calculates the average of each group and uses it to split the data into two subgroups. This process is repeated until a decision tree is created.

Since the histogram algorithm uses a grouping of data, it requires less memory than the exact algorithm. In addition, since the calculations are performed in parallel for each group, the histogram algorithm can be much faster than the exact algorithm, especially for large data sets.

4.5 Data parallelization

Both methods will use data-driven parallelization - the dataset will be divided into equal *n* (2, 3, 4, ...) parts and used in the models for training. In contrast to algorithmic parallelization, we do not allocate specific classes to the models, i.e. each model will receive the same amount of training data as input, with the ratio of classes being arbitrary - in some parts, records of one class will prevail due to the distribution, in others, there will be a uniform distribution.

Thus, the output will be a number of models, depending on how many parts the training data was divided into (see Figure 4).

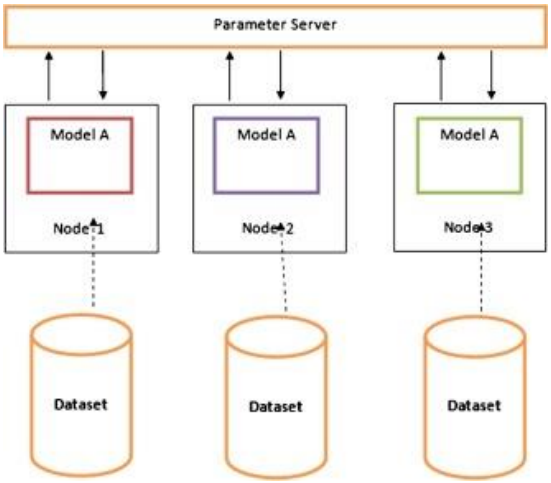

Figure 4. Visualization of data distribution [37]

5. ANALYSIS OF NUMERICAL EXPERIMENTS

This section will gradually describe the results for each of the datasets. First, for the outdoor area, then for the indoor area.

To ensure the reproducibility of the experiments, all model training and evaluation were conducted on a local machine equipped with an Intel Core i7-12700K processor (12 cores, 20 threads), an NVIDIA GeForce RTX 3080 GPU with CUDA Compute Capability 8.6, and 32 GB of DDR4 RAM. The software environment included Python 3.10, TensorFlow 2.11 with the Keras API, CUDA Toolkit 11.7, cuDNN 8.4, and XGBoost version 1.7.4. The operating system used was Ubuntu 22.04 LTS. GPU memory management was handled by enabling memory growth in TensorFlow, while model training performance and GPU utilization were monitored using NVIDIA System Management Interface (nvidia-smi). The specified versions and hardware configurations were selected to ensure compatibility and stability for GPU-accelerated training, particularly when using the gpu_hist algorithm in XGBoost, and for efficient multi-process execution in neural network experiments.

5.1 Neural network

Initially, the network was trained in its standard form, after which the aforementioned methods were applied. Training in the standard form refers to using the standard dataset and a single neural network with the architecture described above.

As shown in Figure 5, the parallelized algorithm demonstrated the best performance, with a training time of 67.61 seconds. Compared to the standard approach, this resulted in an approximately fivefold speedup.

Data-driven parallelization (splitting into 2 and 4 equal parts) also provided a performance improvement, although modest. However, when splitting into 2 parts, the training time exceeded that of the standard approach. This may be attributed to uneven resource distribution on the computer, where certain processes may have received higher priority than the network training. Nevertheless, the 2-second difference can be considered as an experimental error.

The situation for indoor environments is similar, but the total training time for each case has slightly increased (see Figure 6). The distributed algorithm is still in the lead in terms of performance, followed by the data-driven algorithm. For this dataset, the speedup was 4 times, which is a significant optimization. As for the obtained metrics, all results for both cases are presented in Tables 2 and 3.

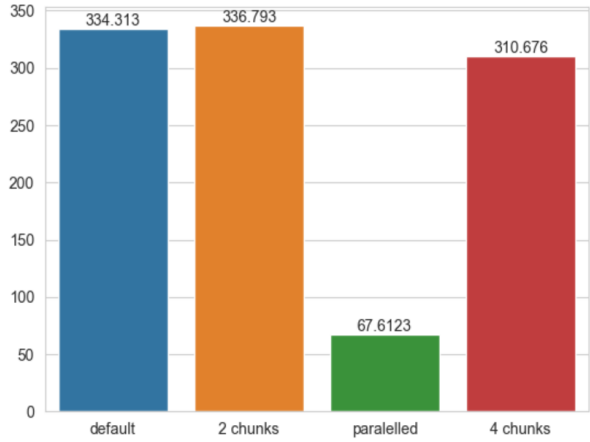

Figure 5. Training time of the network for the open space dataset

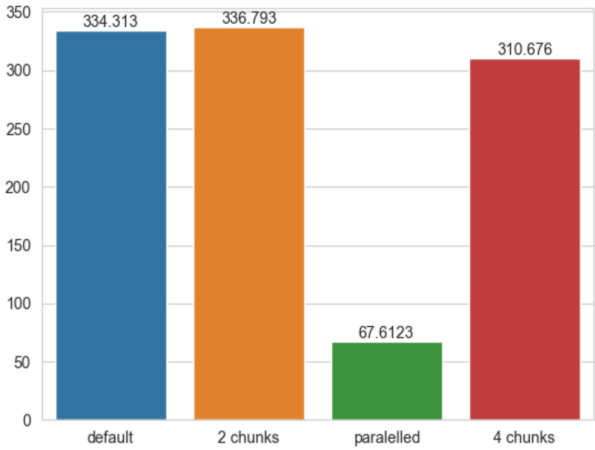

Figure 6. Training time of the network for the closed space dataset

Table 2. Results for the open space dataset

Model Type	R^2	MSE	MAE	Training Time (s)
Standard	93.561	0.2106	0.0746	334.31
2 data chunks	93.456	0.2102	0.0734	336.79
Parallelized	93.892	0.2000	0.0698	67.61
4 data chunks	93.252	0.2120	0.0758	310.67

Table 3. Results for the closed space dataset

Model Type	R^2	MSE	MAE	Training Time (s)
Standard	92.326	0.2232	0.0812	353.42
2 data chunks	92.421	0.2212	0.0804	335.86
Parallelized	92.758	0.2195	0.0783	82.48
4 data chunks	92.227	0.2251	0.0828	317.67

From Tables 2 and 3, it can be concluded that the accuracy across the different approaches varies by approximately 0.5%, which can be considered as experimental error. However, for both datasets, the best results were achieved with parallel

training, with accuracy rates of 93.89% and 92.76%, respectively. The same pattern is observed with the other metrics as well.

5.2 XGBoost

Figures 7-10 present diagrams illustrating the results of training speed evaluation. The metrics for both datasets are provided in Tables 4-5. For the open terrain case, the best results were achieved using an integrated optimization strategy that combined thread-level parallelization with data-level splitting. The first optimization stage involved parallelization using threads, which resulted in an approximate 10% performance improvement. The second stage involved splitting the data into two parts and applying parallelization again. The combination of both techniques led to a fourfold increase in training speed compared to the standard approach. Further splitting the data into four parts did not yield significant performance benefits, so these results are not discussed in detail.

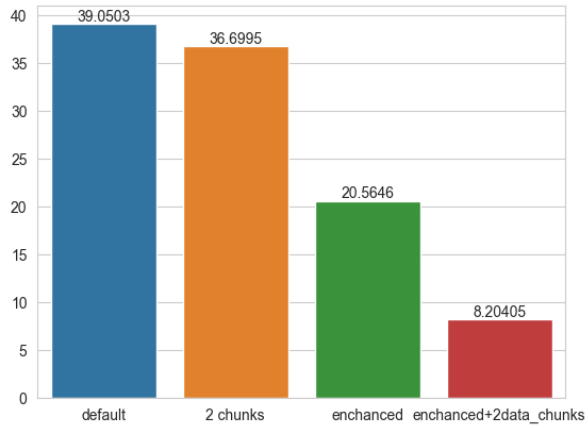

Figure 7. Training time of the XGBoost for the open space dataset

Figure 8 shows the results for the same dataset using similar techniques but with GPU computing (except for the first approach). The increase in efficiency is much greater: for the parallelised approach, the training speed increased by a factor of 6, and the combined method demonstrated particularly high efficiency.

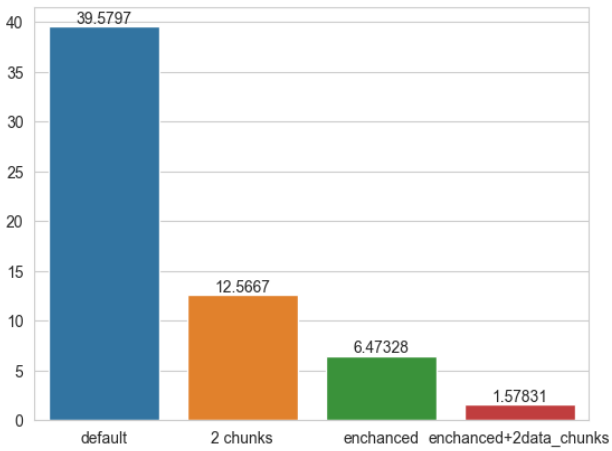

Figure 8. Training time of the XGBoost (gpu_hist) for the open space dataset

For indoor environments, the trend remains the same, with a combined approach for both cases - standard computing and GPU - proving to be the best.

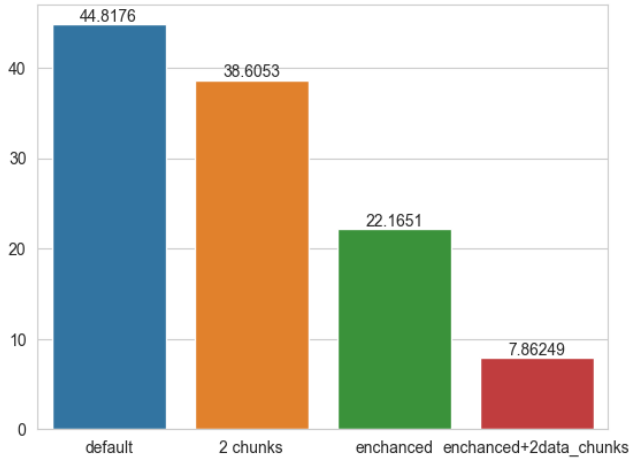

Figure 9. Training time of the XGBoost for the closed space dataset

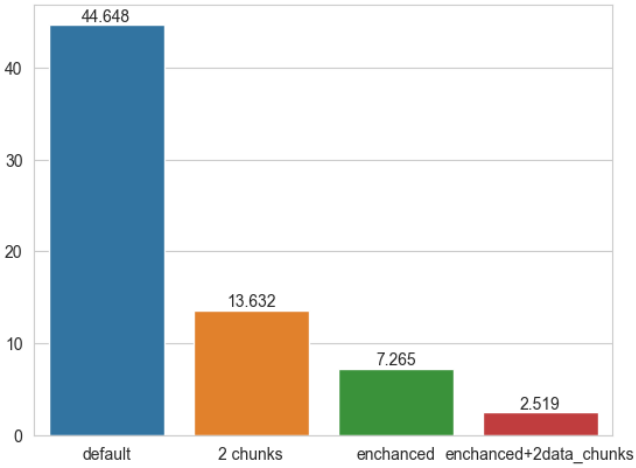

Figure 10. Training time of the XGBoost (gpu_hist) for the closed space dataset

As for the metrics, the results are presented in Tables 4 and 5.

Table 4. Results for the open space dataset (XGBoost)

Model Type	R^2	MSE	MAE	Training Time (s)
XGBOOST	94.2274	0.1975	0.0653	39.05
2 data chunks	94.2524	0.1963	0.0641	36.69
Parallelized	94.3266	0.1959	0.0628	20.56
Parallel + 2 data chunks	94.2485	0.1967	0.0638	8.2

Table 5. Results for the closed space dataset (XGBoost)

Model Type	R^2	MSE	MAE	Training Time (s)
XGBOOST	93.154	0.20021	0.0751	44.81
2 data chunks	93.142	0.20034	0.0758	38.6
Parallelized	93.258	0.20009	0.0743	22.16
Parallel + 2 data chunks	93.145	0.20031	0.0749	7.86

Analyzing the obtained results, it can be noted that the accuracy and other metrics did not experience significant changes. The highest accuracy values were observed when using the parallelized approach, reaching 94.32% and 93.25% for the two datasets, respectively. Additionally, compared to the neural network, the XGBoost model demonstrated slightly higher values for all metrics, indicating the potential for further improvement of the neural network architecture. The highest training speed was achieved using a combined optimization approach that integrated thread-level parallelization and GPU-accelerated histogram-based training (gpu_hist), which accelerated the process by 5-6 times without sacrificing accuracy or other metrics. The results of training using gpu_hist are presented below in Tables 6 and 7.

Table 6. Results for the open space dataset (GPU)

Model Type	R^2	MSE	MAE	Training Time (s)
XGBOOST	94.2270	0.1969	0.0652	39.57
2 data chunks	94.2522	0.1960	0.0641	12.56
Parallelized	94.3268	0.1965	0.0627	6.47
Parallel + 2 data chunks	94.2483	0.1962	0.0638	1.57

Table 7. Results for the closed space dataset (GPU)

Model Type	R^2	MSE	MAE	Training Time (s)
XGBOOST	93.154	0.20021	0.0752	44.64
2 data chunks	93.141	0.20032	0.0754	13.63
Parallelized	93.259	0.20008	0.0745	7.26
Parallel + 2 data chunks	93.144	0.20033	0.0749	2.51

As can be seen, in the worst case, the results changed by only 0.001, which is not a critical deviation.

6. DISCUSSION

In this paper, a solution based on a neural network and the XGBoost algorithm were proposed and implemented, which was applied to the datasets of scanning open and closed environments. The obtained results demonstrate the feasibility of using the proposed approaches since the following accuracy rates were achieved on the test data:

•Open environment: XGBoost - 94.32%, neural network - 93.89%.

•Closed environment: XGBoost - 93.25%, neural network - 92.75%.

Compared to other studies, the proposed methods provide a significant improvement in classification accuracy - from 4% to 20%, depending on the sources compared. The observed superior performance of XGBoost over neural networks in closed environments can be attributed to several factors. First, XGBoost, being a tree-based ensemble method, handles sparse or less informative features more efficiently. In closed environments, the feature variance may be lower or the signal-to-noise ratio less favorable due to reflections and structural interferences. Neural networks, which rely heavily on pattern abstraction and parameter tuning, may underperform when faced with subtle class separations or overlapping distributions. In contrast, XGBoost effectively captures such conditional patterns through gradient-based tree boosting, making it better suited for scenarios where feature importance is highly localized or unevenly distributed. At the same time, it is worth noting that the analyzed studies did not consider classification in closed environments, while this paper takes such scenarios into account, although the results for closed spaces were slightly lower than for open spaces.

The obtained results can be improved by increasing computational resources, since in this study the training was performed on a local machine using 5 processes for the neural network and 8 threads for XGBoost. Further experiments with scalable configurations may help to improve the performance of the models.

Optimizing the neural network by parallelizing the processes significantly increased the training speed. As shown in the graphs, the overall performance increase was 4-5 times depending on the dataset. However, data partitioning as a separate approach did not provide a significant improvement, providing a speed increase of only ~10% compared to the standard model. At the same time, the accuracy remained stable at 92-93% for all the datasets under consideration.

For XGBoost, 8 threads were used to build decision trees and distribute data. The proposed combination of these two methods significantly accelerated the learning process. The use of GPU acceleration (gpu_hist) provided an addition 5-6 times speedup compared to the already parallelized approaches, which confirms the importance of GPU accelerators in the optimization process. Due to the GPU architecture and the ability to efficiently perform vector computations, they significantly outperform the CPU, which processes operations on individual numbers. At the same time, the classification accuracy remained unchanged (93-94%), which confirms the stability of the proposed methods. The same trend was observed for the MSE and MAE metrics.

The literature analysis has shown that most studies focus exclusively on the task of classifying or identifying mines, without focusing on optimizing the learning process. Thus, the results of this work can be used to further develop methods for optimizing and accelerating training processes in similar tasks. The obtained accuracy rates (92-94%) are competitive with existing solutions and can be applied to real-world testing scenarios.

7. CONCLUSIONS

Although XGBoost performed better in terms of training speed and metrics compared to the proposed neural network, a network based on neural approaches has several potential advantages. Firstly, it has better scalability to large and unstructured data such as audio signals, images, or videos, as it can automatically detect complex patterns that are difficult to analyze with XGBoost methods. Secondly, the proposed neural network is more adaptive to non-standard relationships and nonlinearities in the data, which can produce noise or complex features that are poorly handled by tree-based models. Thirdly, the proposed neural network can be integrated with explanatory artificial intelligence (XAI) methods, which ensures transparency and explainability of classification results. Thus, even though XGBoost is faster and more accurate in many cases, the proposed neural network offers significant advantages in scalability, adaptability, ability to handle more complex data, and integration with XAI methods.

Despite the promising results, the study has several

limitations. The datasets used are normalized and lack original spatial coordinates, which poses a challenge for direct real-world deployment where geospatial accuracy is critical. Moreover, the absence of raw signal information limits the generality of the models when applied to new sensor types or different environmental conditions. The models were tested in controlled settings with well-preprocessed data; hence, their robustness in dynamic or noisy real-world scenarios requires further validation. Future work should focus on evaluating these models on raw, heterogeneous datasets and under real-time constraints to ensure operational viability.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the Department of Artificial Intelligence Systems at Lviv Polytechnic National University for their valuable contributions and collaboration. The authors also appreciate the reviewers for their constructive and concise recommendations, which helped improve the presentation of the materials. Furthermore, the authors extend their appreciation to Universiti Teknikal Malaysia Melaka (UTeM) for its financial and institutional support and to the Ministry of Higher Education of Malaysia (MOHE) for their support in this research.

REFERENCES

[1] Deepak Raj, S., Ramesh Babu, H.S. (2022). Identification of intelligence requirements of military surveillance for a WSN framework and design of a situation aware selective resource use algorithm. Revue d'Intelligence Artificielle, 36(2): 251-261. https://doi.org/10.18280/ria.360209

[2] Wen, Z., Shi, J., He, B., Chen, J., Ramamohanarao, K., Li, Q. (2019). Exploiting GPUs for efficient gradient boosting decision tree training. IEEE Transactions on Parallel and Distributed Systems, 30(12): 2706-2717. https://doi.org/10.1109/TPDS.2019.2920131

[3] Al Jazeera. (2024). Myanmar deaths from mines, ordnance tripled in 2023: UN. https://www.aljazeera.com/news/2024/4/4/myanmar-deaths-from-mines-ordnance-tripled-in-2023-un.

[4] Wen, Z., He, B., Kotagiri, R., Lu, S., Shi, J. (2018). Efficient gradient boosted decision tree training on GPUs. In 2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Vancouver, Canada, pp. 234-243. https://doi.org/10.1109/IPDPS.2018.00033

[5] Farmer, B.M. (2024). Ukraine's landmine crisis—CBS News. https://www.cbsnews.com/news/ukraines-landmine-crisis-60-minutes/.

[6] Camacho-Sanchez, C., Yie-Pinedo, R., Galindo, G. (2023). Humanitarian demining for the clearance of landmine-affected areas. Socio-Economic Planning Sciences, 88: 101611. https://doi.org/10.1016/j.seps.2023.101611

[7] Agrawal, R., Sharma, K. (2024). An extensive review on significance of Explainable Artificial Intelligence models in discrete domains for informed decisions making. Revue d'Intelligence Artificielle, 38(3): 957-968. https://doi.org/10.18280/ria.380321

[8] Anghel, A., Papandreou, N., Parnell, T., De Palma, A., Pozidis, H. (2018). Benchmarking and optimization of gradient boosting decision tree algorithms. arXiv preprint arXiv:1809.04559. https://doi.org/10.48550/arXiv.1809.04559

[9] Mochurad, L., Shchur, G. (2021). Parallelization of cryptographic algorithm based on different parallel computing technologies. In IT&AS'2021: Symposium on Information Technologies & Applied Sciences, Bratislava, Slovakia, pp. 20-29.

[10] Trostianchyn, A.M., Izonin, I.V., Duriagina, Z.A., Tkachenko, R.O., Kulyk, V.V., Havrysh, B.M. (2022). Boosting-based model for solving Sm-Co alloy's maximum energy product prediction task. Archives of Materials Science and Engineering, 116(2): 71-80. https://doi.org/10.5604/01.3001.0016.1191

[11] Izonin, I., Muzyka, R., Tkachenko, R., Dronyuk, I., Yemets, K., Mitoulis, S.A. (2024). A method for reducing training time of ML-based cascade scheme for large-volume data analysis. Sensors, 24(15): 4762. https://doi.org/10.3390/s24154762

[12] Kovtun, V., Altameem, T., Al-Maitah, M., Kempa, W. (2024). Simple statistical tests selection based parallel computating method ensures the guaranteed global extremum identification. Journal of King Saud University - Science, 36(5): 103165. https://doi.org/10.1016/j.jksus.2024.103165

[13] Mochurad, L. (2024). Implementation and analysis of a parallel kalman filter algorithm for lidar localization based on CUDA technology. Frontiers in Robotics and AI, 11: 1341689. https://doi.org/10.3389/frobt.2024.1341689

[14] Mochurad, L., Solomiia, A. (2020). Optimizing the computational modeling of modern electronic optical systems. In Lecture Notes in Computational Intelligence and Decision Making: Proceedings of the XV International Scientific Conference "Intellectual Systems of Decision Making and Problems of Computational Intelligence" (ISDMCI'2019), Ukraine, pp. 597-608. https://doi.org/10.1007/978-3-030-26474-1_41

[15] Barnawi, A., Kumar, K., Kumar, N., Alzahrani, B., Almansour, A. (2024). A deep learning approach for landmines detection based on airborne magnetometry imaging and edge computing. Computer Modeling in Engineering & Sciences, 139(2): 2117-2137. https://doi.org/10.32604/cmes.2023.044184

[16] Genç, A. (2024). Multi-feature fusion for GPR-based landmine detection and classification. https://open.metu.edu.tr/handle/11511/28093.

[17] Yu, S., Yang, S., Chen, W., Mao, W. (2019). An electromagnetic detection method for grain silos based on finite difference time domain and ground penetration radar. Instrumentation Mesure Métrologie, 18(2): 159-164. https://doi.org/10.18280/i2m.180210

[18] Qiao, X., Yang, F., Zheng, J. (2019). Ground penetrating radar weak signals denoising via semi-soft threshold empirical wavelet transform. Ingénierie Des Systèmes d Information, 24(2): 207-213. https://doi.org/10.18280/isi.240213

[19] Šipoš, D., Gleich, D. (2020). A lightweight and low-power UAV-borne ground penetrating radar design for landmine detection. Sensors, 20(8): 2234. https://doi.org/10.3390/s20082234

[20] Jiao, Y.S., Wang, X.M., Chen, H., Li, Y. (2010).

Research on the coverage path planning of UAVs for polygon areas. In 2010 5th IEEE Conference on Industrial Electronics and Applications, Taichung, pp. 1467-1472. https://doi.org/10.1109/ICIEA.2010.5514816

[21] Baur, J., Steinberg, G., Nikulin, A., Chiu, K., De Smet, T.S. (2020). Applying deep learning to automate UAV-based detection of scatterable landmines. Remote Sensing, 12(5): 859. https://doi.org/10.3390/rs12050859

[22] Pryshchenko, O.A., Plakhtii, V., Dumin, O.M., Pochanin, G.P., Ruban, V.P., Capineri, L., Crawford, F. (2022). Implementation of an artificial intelligence approach to GPR systems for landmine detection. Remote Sensing, 14(17): 4421. https://doi.org/10.3390/rs14174421

[23] Alsayaydeh, J.A.J., Indra, W.A., Khang, W.A.Y., Shkarupylo, V., Jkatisan, D.A.P.P. (2019). Development of vehicle ignition using fingerprint. ARPN Journal of Engineering and Applied Sciences, 14(23): 4045-4053.

[24] Vivoli, E., Bertini, M., Capineri, L. (2024). Deep learning-based real-time detection of surface landmines using optical imaging. Remote Sensing, 16(4): 677. https://doi.org/10.3390/rs16040677

[25] Dikmen, M. (2022). Investigating transfer learning performances of deep learning models for classification of GPR b-scan images. Traitement du Signal, 39(5): 1761-1766. https://doi.org/10.18280/ts.390534

[26] Wang, Z., Huang, F. (2024). Evaluating the adaptability of deep learning-based multi-feature sonar image detection algorithms. Traitement du Signal, 41(3): 1223-1230. https://doi.org/10.18280/ts.410312

[27] Alsayaydeh, J.A.J., Khang, W.A.Y., Hossain, A.K.M.Z., Shkarupylo, V., Pusppanathan, J. (2020). The experimental studies of the automatic control methods of magnetic separators performance by magnetic product. ARPN Journal of Engineering and Applied Sciences, 15(7): 922-927.

[28] Shkarupylo, V., Blinov, I., Chemeris, A., Dusheba, V., Alsayaydeh, J.A., Oliinyk, A. (2021). Iterative approach to TLC model checker application. In 2021 IEEE 2nd KhPI Week on Advanced Technology (KhPIWeek), Kharkiv, Ukraine, pp. 283-287. https://doi.org/10.1109/KhPIWeek53812.2021.9570055

[29] Khan, A.A., Laghari, A.A., Baqasah, A.M., Bacarra, R., Alroobaea, R., Alsafyani, M., Alsayaydeh, J.A.J. (2025). BDLT-IoMT—A novel architecture: SVM machine learning for robust and secure data processing in Internet of Medical Things with blockchain cybersecurity.The Journal of Supercomputing, 81(1): 271. https://doi.org/10.1007/s11227-024-06782-7

[30] Parsa, P. (2025). Land Mines (Normalized Data) [Dataset]. https://www.kaggle.com/datasets/parsapzadeh/land-mines.

[31] Afifie, N.A., Khang, A.W.Y., Amin, A.F.B.M., Alsayaydeh, J.A.J., Indra, W.A., Herawan, S.G., Ramli, A.B. (2021). Evaluation method of mesh protocol over ESP32 and ESP8266. Baghdad Science Journal, 18(4S): 1397-1397. https://doi.org/10.21123/bsj.2021.18.4(Suppl.). 1397

[32] Alsayaydeh, J.A.J., bin Yusof, M.F., Halim, M.Z.B.A., Zainudin, M.N.S., Herawan, S.G. (2023). Patient health monitoring system development using ESP8266 and Arduino with IoT platform. International Journal of Advanced Computer Science and Applications, 14(4): 617-624. http://doi.org/10.14569/IJACSA.2023.0140467

[33] Tarwidi, D., Pudjaprasetya, S.R., Adytia, D., Apri, M. (2023). An optimized XGBoost-based machine learning method for predicting wave run-up on a sloping beach. MethodsX, 10: 102119. https://doi.org/10.1016/j.mex.2023.102119

[34] Chen, T., Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, California, USA, pp. 785-794. https://doi.org/10.1145/2939672.2939785

[35] Wen, H.T., Wu, H.Y., Liao, K.C. (2022). Using XGBoost regression to analyze the importance of input features applied to an artificial intelligence model for the biomass gasification system. Inventions, 7(4): 126. https://doi.org/10.3390/inventions7040126

[36] Lindskog, W., Prehofer, C., Singh, S. (2023). Histogram-based federated XGBoost using minimal variance sampling for federated tabular data. In 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC), Tartu, Estonia, pp. 182-189. https://doi.org/10.1109/FMEC59375.2023.10306242

[37] Li, X., Zhang, G., Li, K., Zheng, W. (2016). Deep learning and its parallelization. In Big Data, pp. 95-118. https://doi.org/10.1016/B978-0-12-805394-2.00004-0