# A Novel Distributed Machine Learning Model to Detect Attacks on Edge Computing Network

Trong-Minh Hoang[1], Trang-Linh Le Thi[2], and Nguyen Minh Quy[3,*]

[1] Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
[2] Electric Power University, Hanoi, Vietnam
[3] Hung Yen University of Technology and Education, Hungyen, Vietnam
*Correspondence: minhquy@utehy.edu.vn

***Abstract*—To meet the growing number and variety of IoT devices in 5G and 6G network environments, the development of edge computing technology is a powerful strategy for offloading processes in data servers by processing at the network and nearby the user. Besides its benefits, several challenges related to decentralized operations for improving performance or security tasks have been identified. A new research direction for distributed operating solutions has emerged from these issues, leading to applying Distributed Machine Learning (DML) techniques for edge computing. It takes advantage of the capacity of edge devices to handle increased data volumes, reduce connection bottlenecks, and enhance data privacy. The designs of DML architectures have to use optimized algorithms (e.g., high accuracy and rapid convergence) and effectively use hardware resources to overcome large-scale problems. However, the trade-off between accuracy and data set volume is always the biggest challenge for practical scenarios. Hence, this paper proposes a novel attack detection model based on the DML technique to detect attacks at network edge devices. A modified voting algorithm is applied to core logic operation between sever and workers in a partition learning fashion. The results of numerical simulations on the UNSW-NB15 dataset have proved that our proposed model is suitable for edge computing and gives better attack detection results than other state of the art solutions.**

***Keywords*—edge computing, intrusion detection system, distributed machine learning, voting algorithm, attacks**

## I. INTRODUCTION

The Internet of Things is anticipated to be a significant Internet innovation. The cooperation between IoT systems and intelligent computing has brought a series of exciting conveniences to our lives. On the other hand, IoT systems are vulnerable to several security threats, including malware, exploits, DoS (Denial-of-Service), and a backdoor. These attacks can cause problems on the Internet of Things, smart environment services, and devices. To protect a communication system, an Intrusion Detection System (IDS) is responsible for detecting imminent and potential threats or attacks. Therefore, developing intelligent IDS systems to cope with attacks on IoT is an essential task for both researchers and implementers [1]. Resource-constrained IoT devices, such as sensors, actuators, and IoT gateways, have gained popularity. IoT applications create vast amounts of data in real time, which is a desirable objective for AI systems [2]. However, it is nearly impossible to implement machine learning models on IoT end devices. A traditional approach includes directly processing data on a cloud server, which worsens latency, increases connection costs, and poses privacy issues. Hence, the edge computing solution has been introduced, where shared computing devices are placed close to the IoT devices where data is generated and at the network's edge. By allowing computations to be executed closer to the data sources, latency and security problems can be eliminated. Deploying machine learning systems on edge computing devices mitigates the mentioned problems [3]. However, this approach has also generated new requirements, including implementing favorite machine learning models [4, 5]. A Machine Learning (ML) system is one of the most effective solutions for extracting information and making decisions from data. IDSs based on machine learning have come to the forefront of intrusion detection research. ML enables systems to learn and improve by utilizing historical data. Unfortunately, the computational constraints of resource-constrained IoT devices restrict the deployment of ML algorithms on these devices when large-scale data sets in ML systems invoke computational challenges [6]. Conversely, it is frequently impossible for all edge devices to transmit their data to a parameter server for a centralized machine. Hence, it is desirable to introduce distributed learning algorithms that enable devices to build a unified learning model with local training cooperatively. This strategy reduces the amount of training data on the edge device, reduces the amount of data communication across connections, and enhances privacy. Besides the advantages, a lot of challenges have remained to face, including DML frameworks, parallel and distributed ML algorithms, privacy protection, and architecture [7]. Among these DML models, the partition learning model differs from the traditional reinforcement learning model in that it performs model partitioning and does not require edge devices to refresh the entire model. Hence, paralleling the CPU, disk, and network bandwidth can increase the system's efficiency. Otherwise, in the

Manuscript received October 17, 2022; revised November 17, 2022; accepted December 21, 2022; published February 28, 2023.

security area, concerning trade-offs between the limitations of the distributed model and the accuracy of attack detection are open issues [8]. To tackle the above-mentioned problems, an intelligent IDS architecture suitable for edge computing meets both the limited resources of edge devices and the attack detection rate as an urgent task. Therefore, this paper proposes a novel distributed machine learning model based on a modified voting algorithm to detect anomalies in Edge networks as IoT gateway devices. In our proposed model, a partition learning model is applied to IDS on edge devices, each edge device makes its own decision (worker), and the final decision in a server is achieved by a modified voting algorithm to enhance the precise attack detection decisions. The following are the main contributions of this study:

- Construct a DLM solution suitable for resource-constrained edge computing devices in the partition learning model approach.
- Deploy various ML algorithms on worker nodes to detect anomalies, then build a voting-base decision algorithm to make a final decision.
- Test with Reconnaissance attack on dataset UNSW-NB15 to demonstrate the efficiency of the proposed model.

The rest of the paper is organized as follows. Section II presents related works. The proposed model is detailed in Section III. Section IV presents our validation of the proposed model on a practical dataset. Section V presents the decision-making process over voting the conclusions and our future works are presented in Section VI.

## II. Related Works

IDSs are an essential technology for network system security. Since machine learning techniques can capture the complexities of cyber attacks, ML-based IDS provides various practical benefits. Unfortunately, edge devices frequently have constrained resources such as limited energy sources, computational power, and memory. Hence, some proposed ML-based IDS models focus on data pre-processing to achieve efficient data reduction of a dataset to tailor it for resource-constrained devices.

The authors of [9] proposed a lightweight ML-based IDS model called IMPACT (IMPersonation Attack deteCTion) is proposed, that working on the AWID dataset [10] (Aegean WiFi Intrusion Dataset). The authors reduce the number of features through feature extraction and selection using a Stacked Autoencoder (SAE), Mutual Information (MI), and C4.8 wrapper. To reduce the impact of an unbalanced number of samples of different attack kinds in model training samples on model performance, the DL-IDS has proposed to improve robustness by employing a category weight optimization method [11] and then testing on the CICIDS2017 dataset [12].

Also this direction, accordingly, the authors of [13] proposed a novel IDS model based on DNN (Deep Neural Network) to select the feasible features before processing networking data. This study used the KDD99 dataset [14] for testing. The simulation results have shown that it enhances the detection accuracy rate by up to 99.4% compared to existing solutions.

The authors of [15] proposed the Random Forest (RF) method − the Multilayer Perceptron (MLP) neural network feature selection method and built on the Cerebellar Model Articulation Controller (CMAC) neuron network to detect DDOS on the UNSW-NB15 dataset. Although these results significantly improve, since edge computing operates in a distributed manner, lots of noisy data and data growth have created new challenges to be solved [16].

Distributed Machine Learning (DML) techniques, such as federated learning, partitioned learning, and distributed reinforcement learning, are mainly solutions for edge computing. Federated Learning (FL) is a famous architecture of DML for the decentralized generation of generic ML models, its related technologies and protocols, and several application scenarios, including cyberattack detection [8, 17]. The critical difference between partitioned learning and FL is that FL does not perform model partitioning and demands edge devices to renew the entire model jointly. While single-agent RL is typically described as a Markov decision process, Multi-agent Reinforcement Learning (MARL) and Multi-agent Deep Reinforcement Learning (MADRL) are typically framed as stochastic games with state transitions that account for the collaborative action of all agents [18]. Adapting the constrained resource of edge devices in the IoT environment, partition learning is the essential approach to provide an efficient DML algorithm while reducing the workload at distributed agent devices. A well-known framework of partitioned decentralized learning is the Parameter Server framework [19, 20]. The parameter server framework breaks a large-scale model optimization problem into distinct target functions, such as linear regression [21] and Support Vector Machine (SVM) [22]. However, the optimal efficiency of global and local updates in regards to time, number of active nodes, and sensitivity of fixed control parameters under different data distributions and node numbers.

Numerous works utilizing historical datasets, such as the KDD CUP 99, NSL-KDD, DARPA, and the ADFA dataset, have developed the best anomaly-based IDS. Recently, the UNSW-NB15 dataset [23] was made available to the research community. This data collection covers nine distinct types of current attacks and a large variety of everyday actions. The configuration of the simulation testbed relied on the generation of network traffic that evolved through time to replicate the actual network traffic of the present day. This data collection contains 49 features with class labels that pertain to network traffic characteristics depending on the flow between hosts and the packet header [24]. The authors of [25] proposed a novel distributed ensemble design based on IDS employing fog computing that includes k-nearest neighbors, XGBoost, and Gaussian naive Bayes as individual learners at the first level. Random Forest uses the prediction results from the first level to determine the final classification at the second level. The experimental results revealed that the proposed distributed IDS with UNSWNB15 can achieve a higher detection rate, up to 68.98% for analysis, 92.25% for reconnaissance, and 85.42% for DoS attacks. To adapt edge devices, the

authors of [26] proposed an ML-based IDS employing the limited feature space: Support Vector Machine (SVM), k-Nearest Neighbor (kNN), Logistic Regression (LR), Artificial Neural Network (ANN), and Decision Tree (DT). The results revealed on the UNSW-NB15 dataset that the XGBoost-based feature selection method permits methods such as the DT to increase their test accuracy for the binary classification scheme from around 88% to 90%.

Through the above-surveyed studies, the distributed machine learning model is a possible implementation direction on edge devices in the edge computing domain. However, the trade-off between the quantity of training data and accuracy has always been a major challenge for researchers. In this work, a DML-based IDS system is proposed with a partitioning dataset method to tailor the limited resources on edge devices, and the overall accuracy of the proposed model will be improved through the modified voting algorithm for final decisions. Indeed, we implement a complete machine learning algorithms model (workers) at each edge node to solve the challenges above in parameter communication between edge nodes and parameter servers. The decision to identify the attack is transmitted to the server by the edge nodes then the server uses a modified voting algorithm to make the final decision and update its workers. The detailed techniques are presented in the next sections.

## III. Proposed Model

### A. The Outline of the System Model

The IDS system is deployed on the edge device using the DML mechanism depicted in Fig. 1. Inwhere, resource-limited gateways play a role in edge computing devices. Then, the data from obtained sensors is passed through these gateways. A gateway device contains an ML-based IDS with a database part that is a portion of the system's common database. Here, we apply three distinct that have detailed analysis in Section II, including MultiLayer Perceptron (MLP) [15], Support Vector Machine (SVM) [22], and Random Forest (RF) [25] algorithms for training models among three and five different worker types. In more detail, each worker is defined by a tuple [*ML type, training model, feature*].

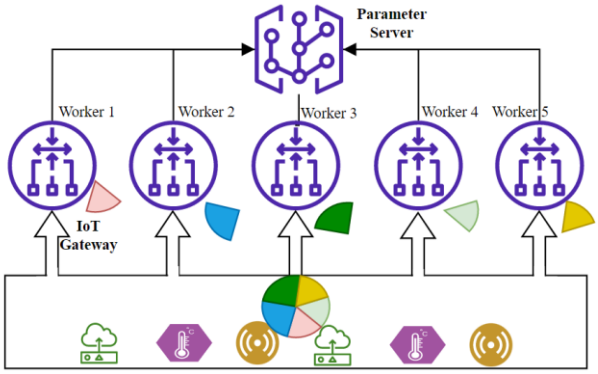


Figure 1. A typical IDS at edge computing structure.

The algorithms mentioned above define the attacks in the partitioned database. The binary decision (0, 1) corresponds to the state of either being attacked or unattacked. Hence, this partition learning system is also referred to as a binary decision system. The decision outputs of the system are sent to the server at edge computing for optimal decision problem computation. The algorithm for multi-decision optimization is presented as follows.

### B. Optimizing the Binary Multiple Decisions System

Assume all workers are independent. A binary decision system of a worker consists of two assumed input states (0, 1) and two output decisions (0, 1). So, the output accuracy of the decision is a conditional probability function as Pi(decision/assumed input). Denote *H0* and *H1* as the event corresponded to input values as 0 and 1. It is divided into 4 cases, as follows:

Case 1: decision=0, input = 0 → $p_i(0/0)$
Case 2: decision=0, input = 1 → $p_i(0/1)$
Case 3: decision=0, input = 1 → $p_i(0/1) = 1 - p_i(1/1)$
Case 4: decision=1, input = 0 → $p_i(1/0) = 1 - p_i(0/0)$

Cases 3 and 4 are called false alarm probabilities. The optimization criterion for the binary multiple decisions system is a linear combination of the probability of each worker, expressed by Eq. (1):

$$J(\alpha)=\alpha p_{sys}(0|0) + (1-\alpha)p_{sys}^{(N-n+1)}(1|1), 0 < \alpha < 1 \quad (1)$$

$\alpha$ is the weighting factor that characterizes the system's preference for accepting the correct assumptions. The binary multiple decisions system comprises $N$ workers, with $H0$ events voted on by more than $(n-1)$ workers and $H1$ events voted on by $(N-n)$ workers. The conditional probability of a binary multiple-decision system is calculated according to Eq. (2) [2].

$$p_{sys}^{(n)}(0|0) = \sum_{k=0}^{N-n} C_N^n \, p^{N-k}(0|0) q^k(1|0) \quad (2)$$

$$p_{sys}^{(N-n+1)}(1|1) = \sum_{k=0}^{N-n} C_N^n \, p^{N-k}(0|0) q^k(1|0), \quad n = \overline{1,N} \quad (3)$$

The binary multiple decisions system optimizations as presented in Eq. (4).

$$n_{opt} = argmax(\alpha p_{sys}^{(n)}(0|0) + (1-\alpha)p_{sys}^{(N-n+1)}(1|1), \quad 0 < \alpha < 1, n = \overline{1,N} \quad (4)$$

***Type 1.*** The worker's decision is independent, and the conditional probability is equal.

We can see that the optimization process of the binary multiple decision systems is simulated over the below example, presented in Tables I–III. Table I and Table III illustrate that the system has 5 and 7 workers, respectively, while Table II presents the conditional probabilities of a worker.

TABLE I. AN ILLUSTRATION OF THE SYSTEM HAS 7 WORKERS

| $n$ | $p_{sys}^{(n)}(0\|0)$ | $p_{sys}^{(N-n+1)}(1\|1)$ | $J(\alpha=0.5)$ | $J(\alpha=0.9)$ | $J(\alpha=0.1)$ |
|---|---|---|---|---|---|
| *n=1* | 1.00000 | 0.02799 | 0.51400 | 0.90280 | 0.00000 |
| *n=2* | 1.00000 | 0.15863 | 0.57932 | 0.91586 | 0.24277 |
| *n=3* | 1.00000 | 0.41990 | 0.70995 | 0.94199 | 0.47791 |
| *n=4* | 0.99999 | 0.71021 | 0.8551 | 0.97101 | 0.73919 |
| *n=5* | 0.99974 | 0.90374 | 0.95174 | 0.99104 | 0.91334 |
| *n=6* | 0.99214 | 0.98116 | 0.98665 | 0.99104 | 0.98226 |
| *n=7* | 0.86813 | 0.99836 | 0.93324 | 0.88115 | 0.98534 |

TABLE II. THE CONDITIONAL PROBABILITIES OF A WORKER

| $i$ | $i=1$ | $i=2$ | $i=3$ | $i=4$ | $i=5$ | $i=6$ | $i=7$ |
|---|---|---|---|---|---|---|---|
| $p_i(0\|0)$ | 0.99 | 0.98 | 0.97 | 0.96 | 0.95 | 0.94 | 0.93 |
| $p_i(1\|1)$ | 0.69 | 0.68 | 0.67 | 0.66 | 0.65 | 0.64 | 0.63 |

Workers whose probabilities correspond to the *H0*, *H1* events as follows: $p_i(0/0) = 0.98$, $p_i(1/1) = 0.6$, $i = 1,7$.

***Type 2.*** The worker's decision is independent, and the conditional probability is unequal. We have,

$$p_{sys}^{(n)}(0|0) = \prod_{i=1}^{N} p_i(0|0) + \sum^{C_N^1} p_{i_1}(0|0)p_{i_2}(0|0) \dots$$
$$\dots p_{i_{N-1}}(0|0)q_{i_N}(1|0) + \dots + \sum^{C_N^{N-n}} p_{i_1}(0|0)p_{i_2}(0|0) \dots$$
$$\dots p_{i_{N-n}}(0|0)q_{i_{N-n+1}}(1|0)q_{i_{N-n+2}}(1|0) \dots p_{i_N}(1|0) \quad (5)$$

$$p_{sys}^{(N-n+1)}(1|1) = \prod_{i=1}^{N} p_i(1|1) + \sum^{C_N^1} p_{i_1}(1|1)p_{i_2}(1|1) \dots$$
$$\dots p_{i_{N-1}}(1|1)q_{i_N}(0|1) + \dots + \sum^{C_N^{N-n}} p_{i_1}(1|1)p_{i_2}(1|1) \dots$$
$$\dots p_{i_{N-n}}(1|1)q_{i_{N-n+1}}(1|0)q_{i_{N-n+2}}(0|1) \dots p_{i_N}(0|1) \quad (6)$$

***Type 3.*** The worker's decision is dependent, and the conditional probability is unequal.

The approach to solving the above problem according to the statistical test method is made as follows: a random sequence is an input to the workers of the binary multiple decision $(syst, X(m))$, where $m$ is the number of random sequences, and the input value takes only one of the values 0 or 1. The output value of each expert will be the input of the probability calculation block of the binary multiple decisions system $(\hat{p}_i(0|0), \hat{p}_i(1|1), i = \overline{1,N})$. In this case, the optimal formula for the probability of the binary multiple decisions system is presented in Eq. (7).

$$n_{opt} = argmax(\alpha\hat{p}_{sys}^{(n)}(0|0) + (1-\alpha)\hat{p}_{sys}^{(N-n+1)}(1|1), 0 < \alpha < 1), n = \overline{1,N} \quad (7)$$

Symbol $M0(m)$, $M1(m)$ represent 0 and 1 values of input chain $m$, respectively. Eq. (8) determines the conditional probability of each worker.

$$\hat{p}_i(0|0) = (M0_i(m)|M0(m)),$$
$$\hat{p}_i(1|1) = (M1_i(m)|M1(m)), n = \overline{1,N} \quad (8)$$

TABLE III. THE CONDITIONAL PROBABILITY VALUE OF THE SYSTEM

| $n$ | $p_{sys}^{(n)}(0\|0)$ | $p_{sys}^{(N-n+1)}(1\|1)$ | $J(\alpha=0.5)$ | $J(\alpha=0.9)$ | $J(\alpha=0.1)$ |
|---|---|---|---|---|---|
| *n=1* | 1.00000 | 0.05438 | 0.52719 | 0.90544 | 0.14894 |
| *n=2* | 1.00000 | 0.25099 | 0.62550 | 0.92510 | 0.32589 |
| *n=3* | 1.00000 | 0.55527 | 0.77764 | 0.95553 | 0.59974 |
| *n=4* | 0.99994 | 0.81653 | 0.98160 | 0.98160 | 0.83487 |
| *n=5* | 0.99824 | 0.95095 | 0.99351 | 0.99351 | 0.95568 |
| *n=6* | 0.97152 | 0.99239 | 0.97361 | 0.97361 | 0.99030 |
| *n=7* | 0.75031 | 0.99948 | 0.77523 | 0.77523 | 0.97456 |

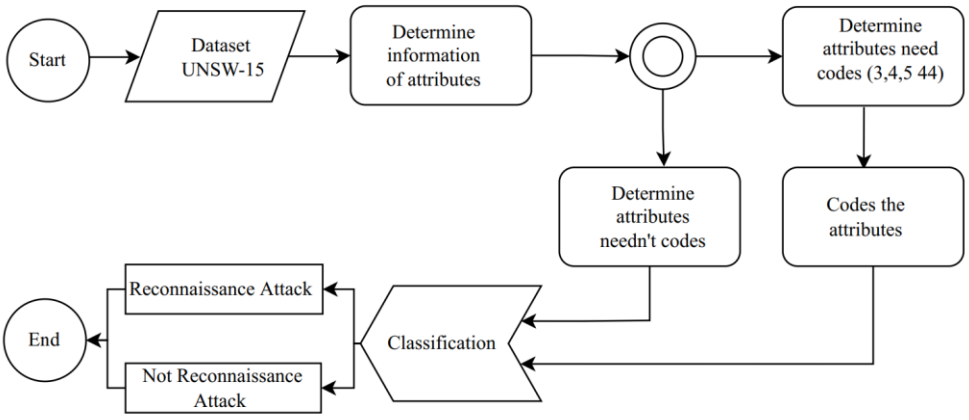


Figure 2. Data prepossessing on the UNSW-NB15 dataset.

The conditional probability of a binary multiple decisions system is calculated by.

**IF** $X(m+1)=0$ THEN the number of workers has correct decision $=L0$, and $M0_{sys}^{(n)}(m)=M0_{sys}^{(n)}(m)+1, n=\overline{1,L0}$

**IF** $X(m+1)=1$ THEN the number of workers has correct decision $=L1$, and $M1_{sys}^{(n)}(m)=M1_{sys}^{(n)}(m)+1, n=\overline{1,L1}$

We have,

$$\hat{p}_{sys}^{(n)}(0|0)=\left(M0_{sys}^{(n)}(m)\middle|M0(m)\right),$$

$$\hat{p}_{sys}^{(n)}(1|1)=\left(M1_{sys}^{(n)}(m)\middle|M1(m)\right), n=\overline{1,N} \quad (9)$$

Eq. (9) is the basis for optimizing the binary multiple decisions system next section describes the probability simulation experiment of this case's binary multiple decisions system

## IV. Experimental Results

### A. Data Prepossessing

To evaluate the efficacy of the proposed method, we utilize the UNSW-NB15 database, one of many IoT attack datasets created in 2015. This dataset is comprised of 2.540.044 records saved in four CSV files. After deleting duplicate records, the number of remaining records is 2.059.419, and all records are divided into four files containing only data regarding common information and attack kinds. The UNSW-NB 15 [20−21, 23] dataset's attacks are classified into nine categories: *normal*, *fuzzers*, *analysis*, *backdoors*, *denial of service*, *exploits*, *generic*, *reconnaissance*, *shellcode*, and *worm*. Each record consists of 44 properties concerning network traffic of five value types: *identifier*, *integer*, *real number*, *time*, and *binary*, with the latter two containing information about the attack type for each property. Fig. 2 depicts the data pre-processing for the UNSWNB15 dataset. In these experimental simulations, we use attack classification and detection for the reconnaissance attack type as an example.

### B. The Binary Multiple Decisions System Architecture

The binary multiple-decision system is made up of workers. We tested the cases of 3 workers and 5 workers. Each specialist has a different training and characterization system. This binary multiple-decision system's learning and testing process occurs as follows: from the UNSW-NB 15 dataset, after pre-processing, it is divided at 80% to serve the training process and 20% to practice and test. The dataset used for retraining is divided into 3 or 5 equal parts to serve the learning process of each expert. The testing process of all workers uses 20% of the test data extracted from the original data set. The composition of the files for training and testing is shown in Table IX.

The training and testing of the workers of the binary multiple decisions system are performed according to the data presented in Table IV and Table V. We use the MATLAB environment containing the application program packages as Neural Network Toolbox.

TABLE IV. Composition of the Dataset for 3 Workers

| | Attacks | No attacks |
|---|---|---|
| Worker 1 (MLP1) | 2725 | 40897 |
| Worker 2 (MLP2) | 2725 | 40900 |
| Worker 3 (MLP3) | 2723 | 40874 |
| Testing data | 2043 | 30727 |

TABLE V. Composition of the Dataset for 5 Workers

| | Attacks | No attacks |
|---|---|---|
| Worker 1 (MLP1) | 1634 | 24544 |
| Worker 2 (MLP2) | 1634 | 24545 |
| Worker 3 (MLP3) | 1634 | 24539 |
| Worker 4 (SVM) | 1634 | 24540 |
| Worker 5 (RF) | 1637 | 24555 |
| Testing data | 2043 | 30727 |

In our experiment, Binary Multiple Decisions System (BMDS) made up of workers based on MLP multiplayer neural network, the training process uses three layers (*15-10-1*, *30-20-1*, *50-30-1*, *100-50-1*, *100-100-1*, *150-100-1*, *200-100-1*, *200-150-1*) and four layers (*30-20-10-1*). Where the first number in the symbols is presented by the number of neurons in the first layer, the second number in the symbols is presented by the number of neurons in the second layer, and so on. The number of input attributes for all networks is 42. The threshold for classification will run from 0.1 to 0.9 in 0.01-step increments. The detection rates of the workers are presented in Table VI and Table VII.

TABLE VI. Parameters for the System Have 3 Workers

| No | Parameters | | | | |
|---|---|---|---|---|---|
| | Type | Method | Threshold | A | No.A |
| Worker 1 | MLP1 | Trainlm | 0.74 | 73.86 | 96.41 |
| Worker 2 | MLP2 | Trainlm | 0.74 | 85.12 | 94.32 |
| Worker 3 | MLP3 | Trainlm | 0.74 | 90.70 | 93.91 |

TABLE VII. Parameters for the System Have 5 Workers

| No | Parameters | | | | |
|---|---|---|---|---|---|
| | Type | Method | Threshold | A | No.A |
| Worker 1 | MLP1 | Trainlm | 0.6 | 84.68 | 92.39 |
| Worker 2 | MLP2 | Trainlm | 0.6 | 80.52 | 94.08 |
| Worker 3 | MLP3 | Trainlm | 0.74 | 90.70 | 93.91 |
| Worker 4 | SVM | RBF | - | 88.06 | 90.86 |
| Worker 5 | | All | - | 87.66 | 93.37 |

## V. Decision-Making Process over Voting

According to the traditional method, the majority voting algorithm is applied to the results of the workers. If 2 out of 3 workers decide whether to attack or not, the system will decide by a majority. According to this voting rule, the detection rate of Reconnaissance attacks is 85.71 %, and non-Reconnaissance attacks are 95.06 %. In our proposed algorithm, any decision to identify an attack from a worker will lead to the overall decision of the whole system that an attack occurs. According to this voting rule, the detection rate of reconnaissance attacks is 96.23%, and non-reconnaissance attacks is 91.07%. In addition to this

voting rule, Table VIII gives the results of other voting rules.

TABLE IX. THE DECISION RULE FOR 3 WORKERS SYSTEM

| Vote | No.A | No.NA | FP | No.D | RA | RNA |
|---|---|---|---|---|---|---|
| Case 1 | 2043 | 30727 | 2744 | 1966 | 96.23 | 91.07 |
| Case 2 | 2043 | 30727 | 1519 | 1751 | 85.71 | 95.06 |
| Case 3 | 2043 | 30727 | 459 | 1384 | 67.74 | 98.51 |

where,

*No.NA* is the number of non-reconnaissance (R) attacks in the data set test.

*FP* is false positives.

*No.D* is a detected attack number.

*RA* is rate detected attack.

*RNA* is rate detected no attack.

Case 1: 3 workers voted no attack;

Case 2: 2 or 3 workers voted no attack;

Case 3: 1 or 2 or 3 workers voted no attack;

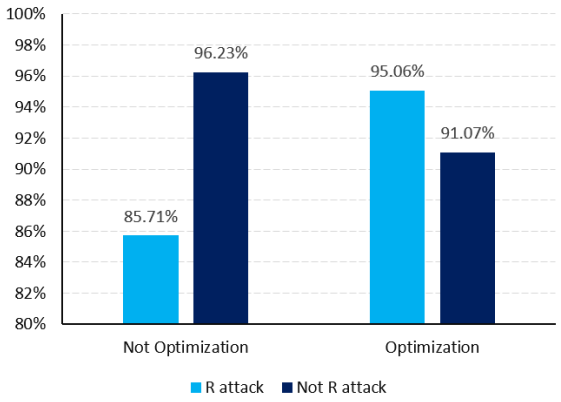


Figure 3. Optimizing according to different rules of 3 workers.

The decision of the binary multiple decisions system can choose one of the three options in Table VIII according to the criteria appropriate to the system. If the binary multiple decisions system decides according to the first row of Table VIII, that means that the binary multiple decisions system prioritizes detecting Reconnaissance attacks. If the binary multiple decisions system makes the $3^{rd}$ row decision in Table VIII, the priority is to detect attacks that are not Reconnaissance attacks. The result of line 2 is the application of the majority voting decision-making algorithm. Fig. 3 shows the detection results of the binary multiple decisions system when the decision is based on the majority vote and the modified voting algorithm based on a rule selection in Table VIII.

TABLE IX. THE DECISION RULE FOR 5 WORKERS SYSTEM

| Vote | No.A | No.NA | FP | No.D | RA | RNA |
|---|---|---|---|---|---|---|
| Case 1 | 2043 | 30727 | 3868 | 2018 | 98.8 | 87.4 |
| Case 2 | 2043 | 30727 | 2699 | 1983 | 97.1 | 91.2 |
| Case 3 | 2043 | 30727 | 2195 | 1871 | 91.6 | 92.9 |
| Case 4 | 2043 | 30727 | 1497 | 1629 | 79.7 | 95.1 |
| Case 5 | 2043 | 30727 | 490 | 1301 | 63.7 | 98.4 |

where,

Case 1: 5 workers voted no attack;

Case 2: 4 or 5 workers voted no attack;

Case 3: 3 or 4 or 5 workers voted no attack;

Case 4: 2 or 3 or 4 or 5 workers voted no attack;

Case 5: 1 or 2 or 3 or 4 or 5 workers voted no attack;

The results obtained after optimizing binary multiple-decision systems in this study are higher than the results in the references [26, 27]. The authors of [26] achieved a rate of around 69.9%, and in [27], around 75.6%. Our proposed model brings up the accuracy of 97.06%, as presented in Fig. 4.

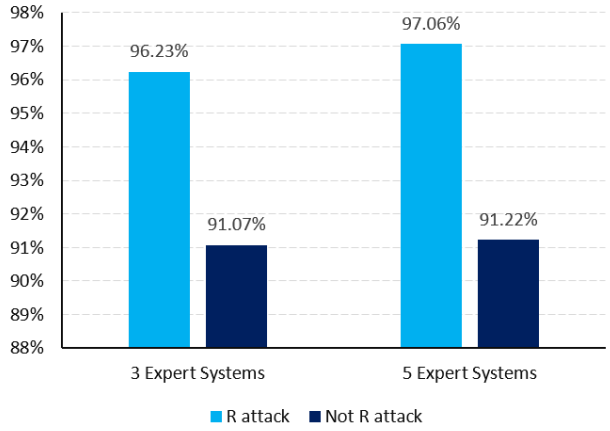


Figure 4. Comparison between systems of 3 and 5 workers.

## VI. CONCLUSION

In recent years, edge computing has emerged as a useful strategy for many internet of things applications that require low latency and privacy. In contrast, IoT application attacks' rising sophistication and intensity have forced new requirements on attack detection systems. IDS solutions based on machine learning have been developed to overcome these challenges. However, distributed machine learning techniques must be implemented to overcome the IoT edge devices' resource limitations. This paper proposes a new approach to DML architecture based on a partition learning approach to increase attack detection accuracy with partition datasets and various ML methods. Using the UNSW-NB15 dataset and the worker system's decision rules, our experimental results are better than those of other approaches. In future studies, we will combine our proposed method into edge-based smart healthcare systems to enhance privacy and security for patient's data.

## CONFLICT OF INTEREST

The authors declare no conflict of interest.

## AUTHOR CONTRIBUTIONS

The authors conducted the research together; Trong-Minh Hoang and Trang-Linh Le Thi proposed models and performed simulations, Trong-Minh Hoang and Nguyen Minh Quy analyzed the data; The authors wrote the paper

together; Nguyen Minh Quy proofread this paper. All authors had approved the final version.

## REFERENCES


[1] E. M. Faisal, A. I. Awad, and H. F. Hamed, "Intrusion detection systems for IoT-based smart environments: A survey," *Journal of Cloud Computing*, no. 1, pp. 1–20, 2018.

[2] Z. Chang, S. Liu, X. Xiong, *et al*., "A survey of recent advances in edge-computing-powered artificial intelligence of things," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13849–13875, 2021.

[3] X. Wang, Y. Han, C. Wang, *et al.*, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[4] Z. Zhou, X. Chen, E. Li, *et al*., "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

[5] G. Carvalho, B. Cabral, V. Pereira, *et al*., "Edge computing: Current trends, research challenges and future directions," *Computing*, vol. 103, pp. 993–1023, 2021.

[6] H. Kim, H. Nam, W. Jung, *et al*., "Performance analysis of CNN frameworks for GPUs," in *Proc. IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2017, pp. 55–64.

[7] S. Hu, X. Chen, W. Ni, *et al*., "Distributed machine learning for wireless communication networks: Techniques, architectures, and applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1458–1493, 2021.

[8] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Comm. Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[9] S. J. Lee, P. D. Yoo, A. T. Asyhari, *et al*., "IMPACT: Impersonation attack detection via edge computing using deep autoencoder and feature abstraction," *IEEE Access*, vol. 8, pp. 65520–65529, 2020.

[10] C. Kolias, G. Kambourakis, A. Stavrou, *et al*., "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 184–208, 2016.

[11] P. Sun, P. Liu, Q. Li, *et al*., "DL-IDS: Extracting features using CNN-LSTM hybrid network for the intrusion detection system," *Security and Communication Networks*, 2020.

[12] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th International Conference on Information Systems Security and Privacy (ICISSP)*, Purtogal, 2018.

[13] L. H. Li, R. Ahmad, W. C. Tsai, *et al*., "A feature selection based DNN for intrusion detection system," in *Proc. 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2021, pp. 1–8.

[14] S. P. RM, P. K. R. Maddikunta, M. Parimala, *et al*., "An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture," *Computer Communications*, vol. 160, pp. 139–149, July 2020.

[15] L. L. T. Trang, V. T. Nguyen, Q. H. Dinh, *et al*., "Comparison of data dimension reduction methods in the problem of detecting attacks," in *Proc. International Conference on Advanced Technologies for Communications (ATC)*, 2021, pp. 324–327.

[16] A. S. Almogren, "Intrusion detection in edge-of-things computing," *Journal of Parallel and Distributed Computing*, vol. 137, pp. 259–265, 2020.

[17] H. B. McMahan, E. Moore, D. Ramage, *et al*. (2016). Federated learning of deep networks using model averaging. [Online]. Available: http://arxiv.org/pdf/1602.05629v1

[18] A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1226–1252, 2021.

[19] M. Li, L. Zhou, Z. Yang, *et al*., "Parameter server for distributed machine learning," in *Proc. NIPS Workshop on Big Learning, Lake Tahoe, USA*, Dec. 2013.

[20] M. Li, D. G. Andersen, J. W. Park, *et al*., "Scaling distributed machine learning with the parameter server," in *Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Broomfeld, USA, 2014.

[21] J. Geng, B. Zhang, L. M. Huie, *et al.*, "Online change-point detection of linear regression models," *IEEE Transactions on Signal Processing*, vol. 67, no. 12, pp. 3316–3329, 2019.

[22] Y. Song, J. Liang, and F. Wang, "An accelerator for support vector machines based on the local geometrical information and data partition," *Int. J. Mach. Learn. Cyber*., vol. 10, no. 9, pp. 2389–2400, 2019.

[23] The UNSW-NB15 Dataset. (May 7, 2022). [Online]. Available: https://research.unsw.edu.au/projects/unsw-nb15-dataset

[24] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.

[25] K. Prabhat, G. P. Gupta, and R. Tripathi, "A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 10, pp. 9555–9572, 2021.

[26] N. Moustafa and J. Slay, "The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems," in *Proc. 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, 2015, pp. 25–31.

[27] A. Mahmoud, A. Shahraki, and A. Taherkordi, "Deep learning for Network Traffic Monitoring and Analysis (NTMA): A survey," *Computer Communications*, vol. 170, pp. 19–41, 2021.




**Trong-Minh Hoang** received a bachelor's degree in physic engineering (1994) and electronic and telecoms engineering (1999) from HUST, a master's degree in electronic and telecommunication engineering (2003), and a Ph.D degree in telecommunication engineering (2014) from PTIT. His research interests include routing, security, and network performance in mobile edge computing, wireless mobile networks, and 5G and beyond.

**Trang-Linh Le Thi** is currently a lecturer in the Faculty of Information Technology at Electric Power University, Viet Nam. She received an Engineer's degree in information system (2009) from MUCTR (Mendeleev University of Chemical Technology of Russia) and Ph.D. in System analysis, control and information processing in Moscow in 2019 from MIPT (Moscow Institute of Physics and Technology). Her research focuses on AI, Neural Networks, and Information Security.

**Nguyen Minh Quy** is the Council President of the Hung Yen University of Technology and Education. He received his B.S. in Information Technology from the Hanoi University of Science and Technology and his Master's degree in Software engineering from VNU University of Engineering and Technology, Vietnam. He obtained a Ph.D. degree in Software Engineering from the Hanoi University of Science and Technology, in 2015. His general research interests are High-Performance Computing, Mobile Communication Networks, and Software Engineering (Email: minhquy@utehy.edu.vn).