Journal on
Internet of Things

Tech Science Press

Doi:10.32604/jiot.2025.060786

ARTICLE

Optimization of Machine Learning Methods for Intrusion Detection in IoT

Alireza Bahmani*

Central Organization, Islamic Azad University, Tehran, 1955847781, Iran

*Corresponding Author: Alireza Bahmani. Email: bahmani@afranet.ir

Received: 09 November 2024; Accepted: 11 March 2025; Published: 24 June 2025

ABSTRACT: With the development of the Internet of Things (IoT) technology and its widespread integration in various aspects of life, the risks associated with cyberattacks on these systems have increased significantly. Vulnerabilities in IoT devices, stemming from insecure designs and software weaknesses, have made attacks on them more complex and dangerous compared to traditional networks. Conventional intrusion detection systems are not fully capable of identifying and managing these risks in the IoT environment, making research and evaluation of suitable intrusion detection systems for IoT crucial. In this study, deep learning, multi-layer perceptron (MLP), Random Forest (RF), Extreme Gradient Boosting (XGBoost), and their extensions were implemented to identify the most effective method. The results of the experiments conducted in this research demonstrate that while deep learning methods with regularization and dropout techniques attained an accuracy of 0.88, the optimized RF classifier using HalvingGridSearchCV achieved the highest accuracy of 0.91. The study also emphasized the importance of balancing accuracy and computational efficiency, especially in resource-constrained IoT environments, where the optimized RF classifier with acceptable computational time emerged as a practical solution. These methods were evaluated on the CICIoT2023 dataset. These findings contribute valuable insights for future research in the field of IoT security and underscore the potential of optimized machine learning techniques in enhancing intrusion detection capabilities.

KEYWORDS: Internet of Things (IoT); deep learning; random forest; extreme gradient boosting (XGBoost); multi-layer perceptron (MLP)

1 Introduction

Currently, the Internet of Things (IoT) is recognized as one of the most widely utilized technologies globally. The IoT is primarily defined as a connected network of diverse components that deliver intelligent systems and services [1]. Specifically, the IoT refers to a set of objects equipped with sensors, electronic devices, and internet connections that can communicate and exchange data among themselves. As wireless communication technologies have advanced, developers have been able to create low-cost IoT nodes that support wireless data collection, analysis, and transmission. Thanks to the ease, affordability, and accessibility of IoT devices in everyday life, this technology is developing and expanding, and the number of devices connected to it is constantly increasing. In this way, the IoT has created wide-ranging effects in social, commercial, and economic fields. Considering the growth and application of the IoT in life and the widespread attacks on its networks and devices, the importance of cybersecurity in this field is strongly felt. This issue has become a fundamental and critical challenge for managing the IoT [2].

Security is very important in the IoT because any device or connected object can lead to the portability of sensitive information, control of devices, or even physical risks for users in case of unauthorized access by

malicious actors. Therefore, intrusion detection or an Intrusion Detection System (IDS) is very important in the IoT. These systems are able to automatically identify unwanted activities and malicious intrusions and apply appropriate defensive measures [3].

The use of artificial intelligence, machine learning, and data mining technologies is very effective in detecting intrusions in the IoT. These technologies can identify common patterns and anomalies and help identify potential intrusions.

In fact, the use of data mining and machine learning techniques in detecting intrusions in the IoT helps to protect sensitive systems and information. Machine learning gives machines and systems the power to learn and improve from past experiences. By using data mining techniques, it is possible to help analyze normal and unusual behaviors and patterns in IoT data. In this way, by analyzing data from connected devices and sensors, attack patterns can be identified and predicted.

2 Related Works

In the research [4], various machine learning and deep learning algorithms, including Random Forest (RF), Convolutional Neural Networks (CNN), and Multi-Layer Perceptron (MLP), were explored for use in IoT networks. The RF and CNN methods yielded the best results on the BoT-IoT dataset, demonstrating superior accuracy and Area Under the Curve (AUC) metrics for multi-class classification. In the article [5], the application of evolutionary computing and a Grammatical Evolution (GE) algorithm for identifying cyber attacks in IoT environments was examined. Experiments were conducted on newly available datasets, confirming the effectiveness of the proposed method. In this paper [6], an intrusion detection system using a combination of multi-objective optimization was presented to detect DDoS attacks in IoT. The performance of the system was evaluated on the latest dataset on DDoS attacks, and an accuracy of 99.03% was recorded. The research in [7] proposed an intrusion detection method based on an ensemble trees approach, which incorporated a Decision Tree (DT) classifier alongside Random Forest (RF). This method demonstrated strong performance while maintaining low computational resource requirements compared to other existing techniques.

In the article [8], a feature selection and JRip classification mechanism was proposed for intrusion detection systems. The proposed system of this paper achieved better performance than the original feature set by selecting 16 and 19 features in the IoT-BoT and KDD Cup 1999 datasets, respectively. In the article [9], a method based on the combination of two convolutional neural networks (CNN-CNN) was proposed to detect IoT network attacks. The first CNN is used for feature selection, and the second CNN is used for IoT attack detection. In the study [10], the Chi-Square statistical method was used for feature selection, and ensemble classifier methods were used to detect attacks in the TON_IoT dataset. Among the investigated methods, including XGBoost, bagging, extra trees (ET), RF, and AdaBoost, the XGBoost method showed the highest performance on this dataset. In the article [11], a two-level intrusion detection model in IoT was proposed. In this method, at level 1, the abnormality of flow is detected, and at level 2, malicious activity is classified. This model was tested on the IoT botnet dataset, and the decision tree and RF recorded the best results for the two levels, respectively.

In the article [12], after applying the data preprocessing methods, the Refined Long Short-Term Memory (RLSTM) model was proposed to detect DoS attacks in IoT networks. The RLSTM model was applied to the NSL-KDD and CICIDS-2017 datasets and achieved good results. A hybrid deep learning network (HDL) combining CNN and LSTM methods was used for the intrusion detection system in the paper [13]. In addition, the SMOTE method and the Tomek-Links sampling method were used to address data imbalance, which improved the results compared to the basic methods.

In the study [14], extensive experiments were conducted on the Canadian Institute for Cybersecurity Internet of Things Dataset 2023 (CICIoT2023) dataset. The results of the widely used BERT [15], DistilBERT [16], and XLNet [17] models were investigated on this dataset under the same conditions. These models achieved accuracies of 0.85, 0.89, and 0.88, respectively. In addition, the results of recent models proposed in the field of research were investigated on this dataset. The paper [18] presents the Deep Learning Bidirectional Long Short-Term Memory (DL-BiLSTM) model, a lightweight intrusion detection system for IoT networks that addresses the challenges of limited resources. By combining deep neural networks (DNNs) with bidirectional long short-term memory (BiLSTM), the model extracts complex network features and captures cyber-attack patterns. The study [19] examines the impact of the Zeroth Order Optimization (ZOO) adversarial attack on machine learning models using the IoTID20 and CIC-IoT-2023 datasets. The XGBoost and MLP-Mimic models demonstrated the best results based on the RF and Catboost models, respectively. The results highlight the need to strengthen machine learning techniques for improved cybersecurity. In the research [20], deep learning models such as DNN, CNN, and RNN were applied to different network traffic flows, and the RNN model achieved the highest accuracy on the CIDIoT2023 dataset. The results demonstrate the effectiveness of the proposed method in detecting cyberattacks in real-world IoT environments. A summary of the reviewed research, including the method, dataset, and evaluation results, is given in Table 1.

Table 1: Summary of reviewed research

Research	Methods	Dataset	Evaluation
[4]	RF	BoT-IoT	AUC Normal/1
	CNN		AUC Normal/0.99
[5]	GE algorithm	IoT-MQTT (Bi)	Accuracy/97.94
		IoT-MQTT (Uni)	Accuracy/96.70
		BoT-IoT	Accuracy/99.98
[6]	Deep learning + NSGA-II	CISIDS2017	Accuracy/99.03
[7]	Ensemble trees	NF-ToN-IoT-v2	AUC//93
[8]	Feature selection + JRip	IoT-BoT	Accuracy/99.9993
		KDD Cup 1999	Accuracy/99.9920
[9]	CNN-CNN	BoT IoT 2020	Accuracy/98.04
[10]	XGBoost	TON_IoT	Accuracy/100
[11]	DT + RF	IoT botnet	Accuracy/99.90
[12]	RLSTM	NSL-KDD	Accuracy/99.22
		CICIDS-2017	Accuracy/98.60
[13]	STL-HDL	CIDDS-001	Accuracy/99.83
		UNS-NB15	Accuracy/99.17
[15]	CICIoT2023	BERT	Accuracy/0.85
[16]	CICIoT2023	DistilBERT	Accuracy/0.89
[17]	CICIoT2023	XLNet	Accuracy/0.88
[18]	CICIoT2023	DL-BiLSTM	Accuracy/0.87
[19]	CICIoT2023	MLP-Mimic + Catboost	Accuracy/0.89
[20]	CICIoT2023		Accuracy/0.90

3 Intrusion Detection System for IoT

According to the studies, the development of intrusion detection systems based on different machine learning and deep learning approaches has been the main focus of research studies to address security challenges in IoT networks. Each of the techniques used shows different performance depending on the dataset and the prediction process. In this section, the flowchart, the description of the dataset, and the learning techniques used in this research are presented. In this research, the effectiveness of deep learning, MLP, RF, and XGBoost methods and their extensions based on the network intrusion detection dataset were compared. The experimentation process and the implementation of different deep learning and machine learning methods are shown in Fig. 1.

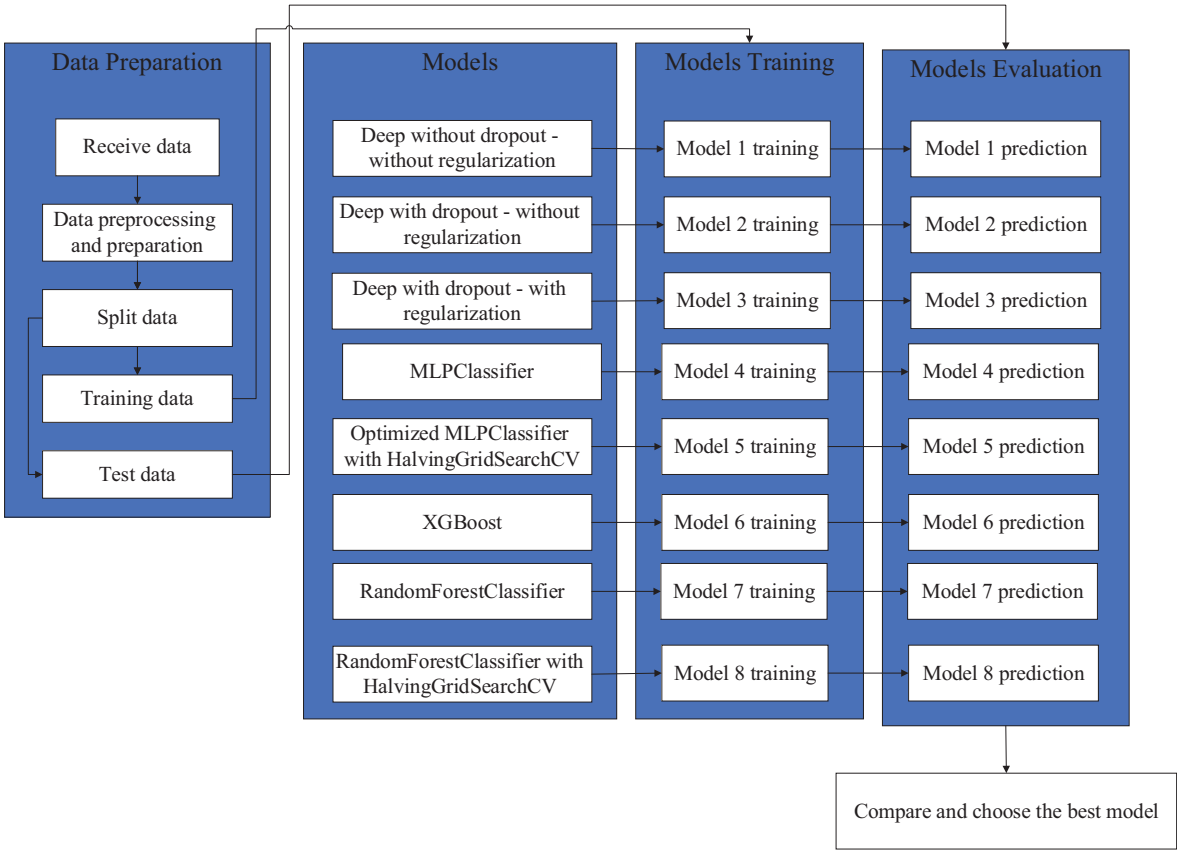

Figure 1: Experimentation process

4 Dataset

The CICIoT2023 dataset is used in this research, which is a new and realistic dataset in the field of IoT attacks. This dataset can be used to create prediction models for different types of intrusion attacks and design intrusion detection systems in IoT networks. This dataset, which divides attacks into different categories, was developed in a 2023 study [21] and contains 1,191,264 network samples for intrusions and 47 features of each intrusion. To collect this dataset, thirty-three distinct attacks were performed on an IoT topology consisting of 105 different devices. This dataset is accessible from the Kaggle website.

5 Performance Metrics

In this research, after its collection and preprocessing, the data was divided into two groups of training and experimental samples with a ratio of 70% to 30%. In the following, the four criteria of Accuracy, Precision, Recall, and F1-score were used to evaluate the performance. In evaluating machine learning models, these four important criteria help to analyze and compare the performance of the models. These criteria consist of true positive (TP), false positive (FP), false negative (FN), and true negative (TN) components and are defined as follows:

Accuracy is the ratio of the number of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{FN} + \text{TN})} \tag{1}$$

Precision is the ratio of the number of correct positive predictions to the total number of positive predictions.

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{2}$$

Recall is the ratio of the number of correct positive predictions to the total number of true positives.

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{3}$$

The F1-score is the harmonic mean of positive precision and recall and is used as a balanced measure to evaluate imbalanced models.

$$\text{F1} - \text{score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{4}$$

6 Results and Discussion

In this section, we present the results obtained from various models, including deep learning, MLP, RF, and XGBoost, along with their extensions. Each model examined in this research will be detailed in its own respective section.

6.1 Deep without Dropout-without Regularization

In this section, the deep network, which consists of dense layers (known as a fully connected neural network), is used to solve the problem of intrusion detection on the investigated dataset. In this type of network, each neuron in one layer is connected to all neurons in the next layer. Deep networks are usually able to identify complex patterns in data and can achieve good accuracy in predictions and classifications. These methods can exploit large amounts of data, identify complex and nonlinear features, and provide better performance by learning from them. The detailed specifications of the implemented deep network are given in Model 1.

As shown in the code, this model is a fully connected deep neural network (Deep Neural Network) that uses several Dense layers with ReLU activation. The network is designed without using special techniques such as Dropout or regularization. Dense layers mean fully connected layers where all neurons are connected to the neurons of the next layer. Due to the lack of Dropout and regularization, this model is prone to overfitting because the network may overfit to the details of the training data.

Below are the results obtained from this deep model. Fig. 2 shows the confusion matrix obtained from this model. As can be seen in this figure, the deep model did not detect any cases in the Benign Traffic, MITM, and Recon classes. Therefore, there are no rows related to these classes in this figure. In Table 2, the results obtained from this method are given in terms of Accuracy, Precision, Recall, and F1-score. As can be seen in this table, in the Benign Traffic, MITM, and Recon classes, the values of Precision, Recall, and F1-score criteria are equal to zero. The total accuracy obtained from this method is equal to 0.81. Fig. 3 shows the loss and accuracy obtained from the deep model during different epochs.

Model 1: Deep model without dropout-without regularization		
Layer (Type)	**Output shape**	**Number of parameters**
Dense (dense_12)	(None, 32)	704
Dense (dense_13)	(None, 64)	2112
Dropout (dropout_4)	(None, 64)	0
Dense (dense_14)	(None, 128)	8320
Dense (dense_15)	(None, 64)	8256
Dropout (dropout_5)	(None, 64)	0
Dense (dense_16)	(None, 128)	8320
Dense (dense_17)	(None, 64)	8256
Dense (dense_18)	(None, 64)	4160
Dropout (dropout_6)	(None, 64)	0
Dense (dense_19)	(None, 32)	2080
Dense (dense_20)	(None, 32)	1056
Dense (dense_21)	(None, 16)	528
Dropout (dropout_7)	(None, 16)	0
Dense (dense_22)	(None, 32)	544
Dense (dense_23)	(None, 64)	2112
Dense (dense_24) (Output Layer)	(None, 6)	390
Total params: 46,838 (182.96 KB)		
Trainable params: 46,838 (182.96 KB)		
Non-trainable params: 0 (0.00 B)		

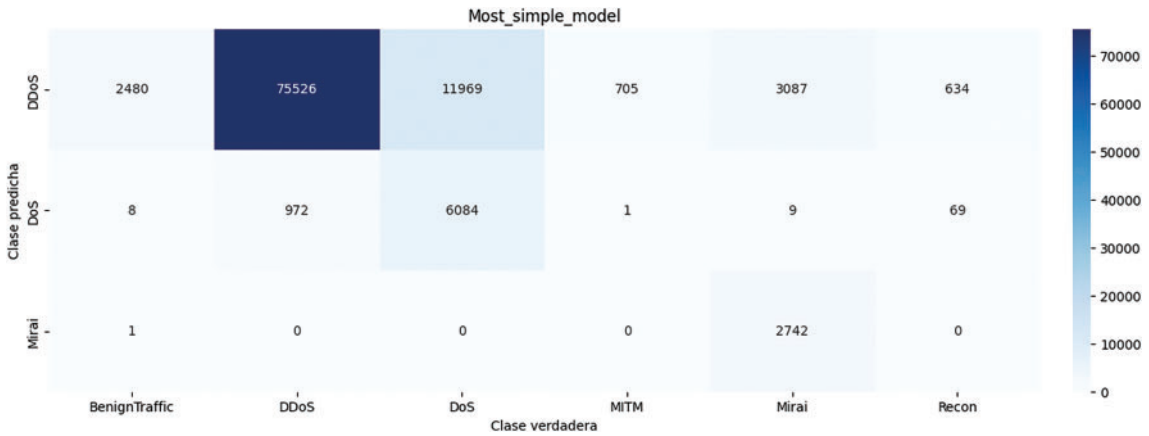

Figure 2: Confusion matrix obtained from deep model without dropout-without regularization

Table 2: The results obtained from the deep model without dropout-without regularization

	Precision	Recall	F1-Score
BenignTraffic	0.00	0.00	0.00
DDoS	0.80	0.99	0.88
DoS	0.85	0.34	0.48
MITM	0.00	0.00	0.00
Mirai	1.00	0.47	0.64
Recon	0.00	0.00	0.00
Accuracy		0.81	

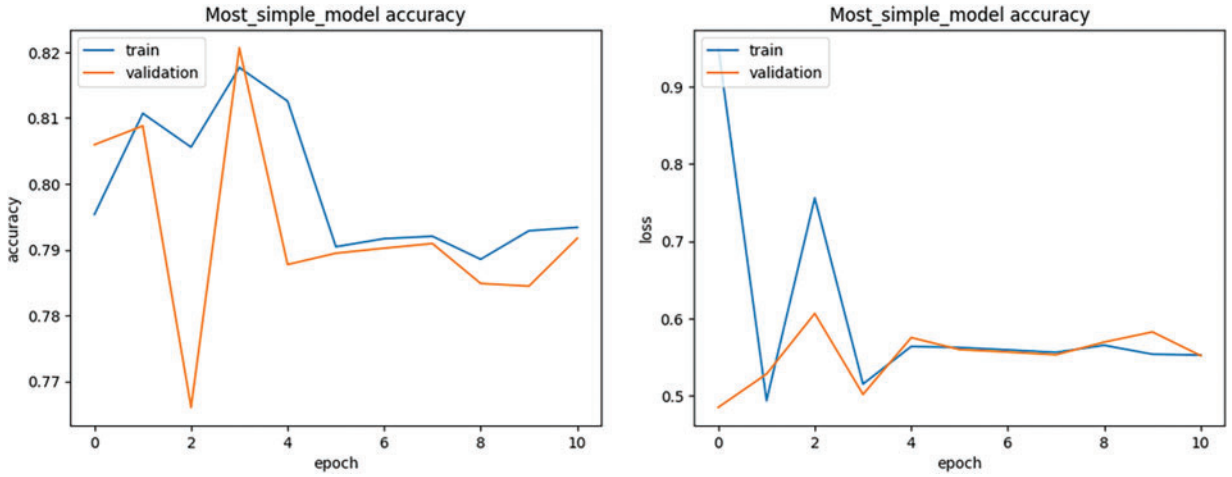

Figure 3: Loss and accuracy obtained during different epochs from the deep model without dropout-without regularization

6.2 Deep with Dropout-without Regularization

In this section, the deep network used in the previous section was developed by adding Dropout. The main advantage of adding Dropout to a deep network consisting only of dense layers is to reduce the probability of overfitting. Overfitting occurs when the model overfits the training data and fails to perform well on new data. With Dropout, at each training step, a number of neurons are randomly deactivated. This makes the model unable to depend on specific neurons and, as a result, learn better from the general features of the data. In general, Dropout can help improve the performance of the model on test data and increase its generalizability. The detailed specifications of the deep network implemented in this section are given in Model 2.

As shown in the code, this model is similar to the first model, but Dropout has been added to reduce the risk of overfitting. Dropout makes the model not dependent on certain neurons by randomly deactivating neurons. The Dropout value is different in layers, which helps the model learn features more effectively during training and avoid overfitting.

Here are the results obtained from this deep learning model. Fig. 4 displays the confusion matrix generated by this method. As seen in this figure, the deep model of this section reliably identified the data in the Recon class, which was not detected at all in the previous model. However, the total accuracy obtained

from this method is equal to 0.81, like the previous model. In Table 3, the results obtained from this method are given in terms of Accuracy, Precision, Recall, and F1-score. Fig. 5 shows the loss and accuracy obtained from the deep model during different epochs.

Model 2: Deep with dropout-without regularization		
Layer (Type)	**Output shape**	**Number of parameters**
Dense (dense_25)	(None, 32)	704
Dense (dense_26)	(None, 64)	2112
Dense (dense_27)	(None, 128)	8320
Dense (dense_28)	(None, 64)	8256
Dense (dense_29)	(None, 128)	8320
Dense (dense_30)	(None, 64)	8256
Dense (dense_31)	(None, 64)	4160
Dense (dense_32)	(None, 32)	2080
Dense (dense_33)	(None, 32)	1056
Dense (dense_34)	(None, 16)	528
Dense (dense_35)	(None, 32)	544
Dense (dense_36)	(None, 64)	2112
Dense (dense_37) (Output Layer)	(None, 6)	390
Total params: 46,838 (182.96 KB)		
Trainable params: 46,838 (182.96 KB)		
Non-trainable params: 0 (0.00 B)		

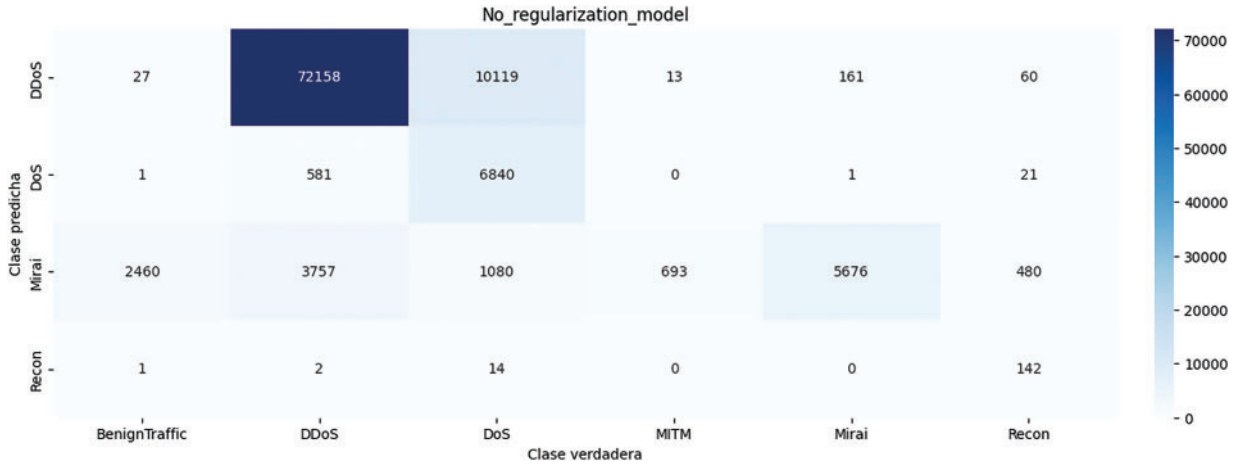

Figure 4: Confusion matrix obtained from deep model with dropout-without regularization

Table 3: The results obtained from the deep model with dropout-without regularization

	Precision	Recall	F1-Score
BenignTraffic	0.00	0.00	0.00
DDoS	0.87	0.94	0.91
DoS	0.92	0.38	0.54
MITM	0.00	0.00	0.00
Mirai	0.40	0.97	0.57
Recon	0.89	0.20	0.33
Accuracy		0.81	

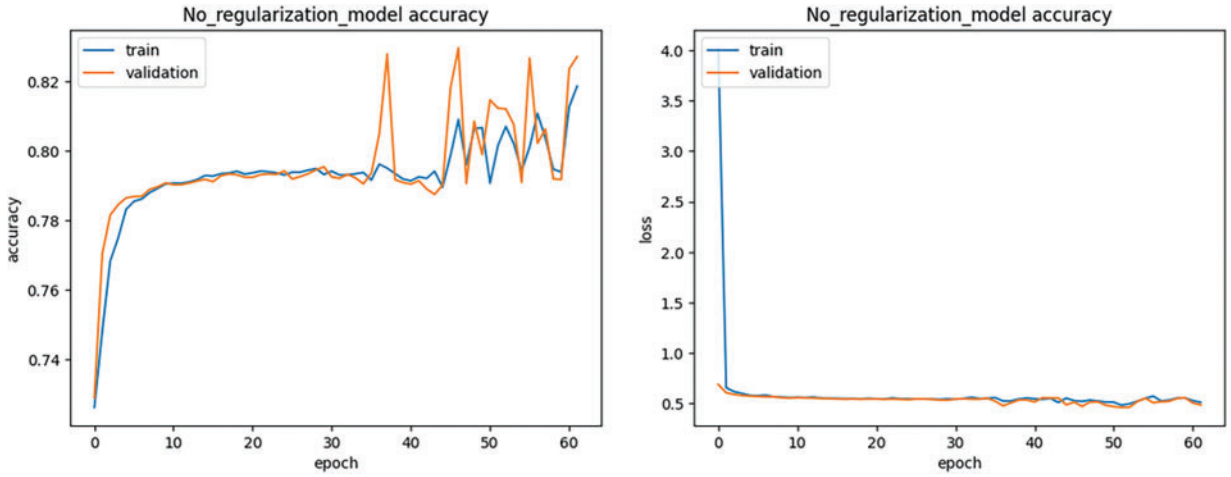

Figure 5: Loss and accuracy obtained during different epochs from the deep model with dropout-without regularization

6.3 Deep with Dropout-with Regularization

In this section, the regularization parameter is also added to the deep network developed with Dropout. In general, adding regularization to Dense networks can help to improve the performance of the network from different aspects and increase their robustness and stability. In this section, a penalty is added for the sum of squares of the weights, thus helping to avoid overfitting the training data. By applying regularization, the complexity of the network is reduced, and this can improve the efficiency of the network and its ability to learn and generalize to new data. The detailed specifications of the deep network implemented in this section are given in Model 3.

Below are the results obtained from this deep model. Fig. 6 shows the confusion matrix obtained from this method. As can be seen in this figure, the deep model of this part identified the data in the Benign Traffic class well, which was not detected in the previous model. The total accuracy obtained from this method also increased significantly and reached the value of 0.88. In Table 4, the results obtained from this method are given in terms of Accuracy, Precision, Recall, and F1-score. Fig. 7 shows the loss and accuracy obtained from the deep model during different epochs.

Model 3: Deep model with dropout-with regularization		
Layer (Type)	**Output shape**	**Number of parameters**
Dense (dense_38)	(None, 64)	1408
Dropout (dropout_8)	(None, 64)	0
Dense (dense_39)	(None, 64)	4160
Dense (dense_40)	(None, 128)	8320
Dropout (dropout_9)	(None, 128)	0
Dense (dense_41)	(None, 64)	8256
Dense (dense_42)	(None, 128)	8320
Dense (dense_43)	(None, 64)	8256
Dense (dense_44)	(None, 64)	4160
Dropout (dropout_10)	(None, 64)	0
Dense (dense_45)	(None, 32)	2080
Dense (dense_46)	(None, 32)	1056
Dense (dense_47)	(None, 16)	528
Dropout (dropout_11)	(None, 16)	0
Dense (dense_48)	(None, 32)	544
Dense (dense_49)	(None, 64)	2112
Dense (dense_50) (Output Layer)	(None, 6)	390
Total params: 49,590 (193.71 KB)		
Trainable params: 49,590 (193.71 KB)		
Non-trainable params: 0 (0.00 B)		

Figure 6: Confusion matrix obtained from deep model with dropout-with regularization

Table 4: The results obtained from the deep model with dropout-with regularization

	Precision	Recall	F1-Score
BenignTraffic	0.65	0.94	0.77
DDoS	0.89	0.97	0.93
DoS	0.77	0.53	0.63
MITM	0.00	0.00	0.00
Mirai	0.99	0.94	0.97
Recon	0.00	0.00	0.00
Accuracy		0.88	

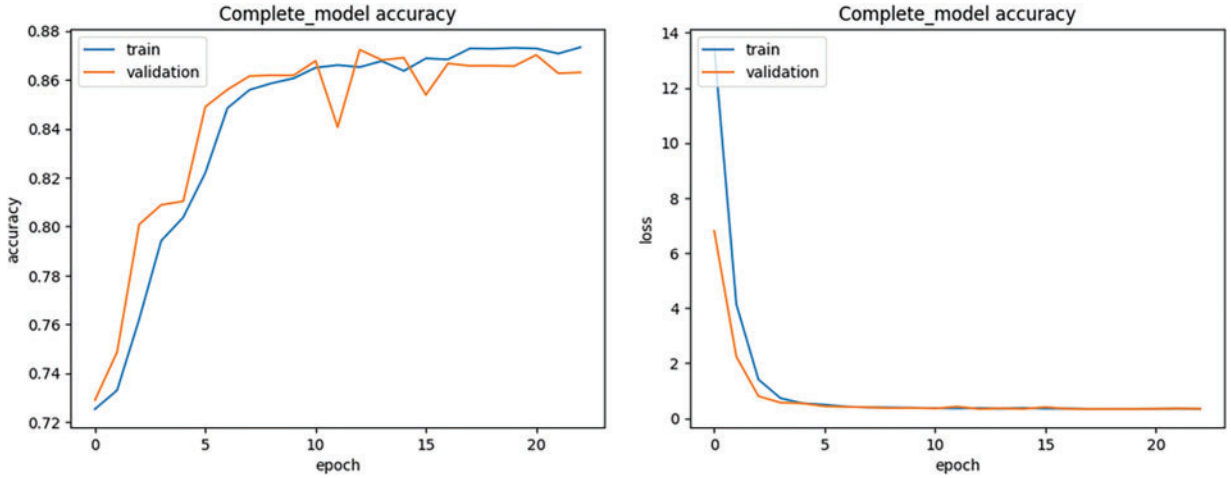

Figure 7: Loss and accuracy obtained during different epochs from the deep model with dropout-with regularization

6.4 MLPClassifier

MLP is a type of neural network that consists of several layers of neurons. This network includes an input layer, several hidden layers, and an output layer. Having several hidden layers, this method can learn complex and non-linear patterns and can adapt and learn from different types of data. In this research, the values of random_state and max_iter in MLPClassifier were considered equal to 1 and 0, respectively. Table 5 shows the results obtained from the MLP model. The accuracy obtained from this method is equal to 0.83.

Table 5: The obtained results from MLPClassifier

	Precision	Recall	F1-Score
BenignTraffic	0.58	0.80	0.67
DDoS	0.87	0.94	0.90
DoS	0.65	0.37	0.47
MITM	0.38	0.40	0.39
Mirai	0.97	0.95	0.96
Recon	0.26	0.34	0.29
Accuracy		0.83	

6.5 Optimized MLPClassifier with HalvingGridSearchCV

In this section, we aim to improve the performance of the MLP using the HalvingGridSearchCV method. This method seeks to find the best values for the parameters of a model by searching over the specified parameter values through successive halving. The search strategy in this method starts by evaluating all candidates with a small number of resources and iteratively selects the best candidates using more and more resources.

The results obtained from this method are given in Table 6. According to the obtained results, the optimized MLP method did not achieve improvement compared to the MLP model, obtaining an accuracy of 0.83.

Table 6: The obtained results from optimized MLPClassifier with HalvingGridSearchCV

	Precision	**Recall**	**F1-Score**
BenignTraffic	0.48	0.16	0.23
DDoS	0.82	0.99	0.90
DoS	0.89	0.39	0.54
MITM	0.08	0.00	0.00
Mirai	1.00	0.62	0.77
Recon	0.69	0.28	0.40
Accuracy		0.83	

6.6 XGBoost

XGBoost is a decision tree-based ensemble learning algorithm that uses boosting techniques. This method is specifically designed to optimize speed and performance and can effectively work with large and complex data. In this research, the values of n_estimators, learning_rate, max_depth, and random_state in XGBoost were considered equal to 100, 1.0, 1, and 0, respectively. Table 7 shows the results obtained from the XGBoost model. The accuracy obtained from this method is equal to 0.85.

Table 7: The obtained results from XGBoost

	Precision	**Recall**	**F1-Score**
BenignTraffic	0.00	0.00	0.00
DDoS	0.90	0.95	0.92
DoS	0.73	0.56	0.63
MITM	0.00	0.00	0.00
Mirai	0.58	0.99	0.73
Recon	0.23	0.01	0.02
Accuracy		0.85	

6.7 RandomForestClassifier

RF is an ensemble learning algorithm that consists of a set of decision trees. By creating several decision trees and combining their predictions, this method increases prediction accuracy and helps to reduce the probability of overfitting. RF is very popular due to its simplicity and high performance in various problems, especially in high-dimensional data and complex features.

In this research, the values of max_depth and random_state in RF were considered equal to 2 and 0, respectively. Table 8 shows the results obtained from the RandomForestClassifier model. The accuracy obtained from this method is equal to 0.75. This was the worst accuracy achieved by the models investigated in this research. The implemented RandomForestClassifier model did not recognize samples of the MITM, Mirai, and Recon classes.

Table 8: The obtained results from the RandomForestClassifier model

	Precision	Recall	F1-Score
BenignTraffic	0.69	0.90	0.78
DDoS	0.76	1.00	0.86
DoS	0.17	0.00	0.00
MITM	0.00	0.00	0.00
Mirai	0.00	0.00	0.00
Recon	0.00	0.00	0.00
Accuracy		0.75	

6.8 Optimized RandomForestClassifier with HalvingGridSearchCV

In RF, determining parameter values is crucial for the model's quality and accuracy. This section discusses the optimization of parameter values for this method using HalvingGridSearchCV. These parameters can include the number of trees, the depth of each tree, the number of features used in each tree, etc. By properly setting the parameters, it is possible to avoid overfitting (if the number of trees is large) or underfitting (if the number of trees is small). This increases the accuracy of the model in predicting new data. The results obtained from this model are given in Table 9. The accuracy obtained from this method is equal to 0.91%. The results obtained from the optimized RandomForestClassifier method have improved significantly compared to the accuracy obtained from the initial RandomForestClassifier.

Table 9: The obtained results from optimized RandomForestClassifier with HalvingGridSearchCV

	Precision	Recall	F1-Score
BenignTraffic	0.83	0.96	0.89
DDoS	0.91	0.98	0.94
DoS	0.86	0.61	0.71
MITM	0.90	0.70	0.79
Mirai	1.00	0.99	0.99
Recon	0.79	0.64	0.71
Accuracy		0.91	

7 Comparison of the Obtained Results

Table 10 presents the results of the implementations conducted in this research. First, the results obtained from three implemented deep models are presented. In this study, the performance of the deep method improved by implementing regularization and Dropout techniques; however, the best result obtained is equal to 0.88 accuracy. Next, the MLP, RF, and XGBoost methods were examined, and the results were improved by adjusting their parameters using the HalvingGridSearchCV method.

Table 10: Comparing the performance of implemented machine learning methods

Method	Accuracy	Elapsed Time (s)
Deep without dropout-without regularization	0.81%	118.57
Deep with dropout-without regularization	0.81%	162.84
Deep with dropout-with regularization	0.88%	368.91
MLPClassifier	0.83%	29.11
Optimized MLPClassifier with HalvingGridSearchCV	0.83%	289.77
XGBoost	0.85%	93.17
RandomForestClassifier	0.75%	60.15
Optimized RandomForestClassifier with HalvingGridSearchCV	0.91%	47.75

Fig. 8 presents a comparison of the examined methods. Based on the evaluations conducted, the highest accuracy of 0.91% was achieved by the optimized RandomForestClassifier. This method benefits from hyperparameter tuning, which contributes to improving the model's performance. The improved deep learning method with regularization and dropout achieved the best performance after the optimized RandomForestClassifier (with an accuracy of 0.88%), followed by XGBoost (with an accuracy of 0.85%).

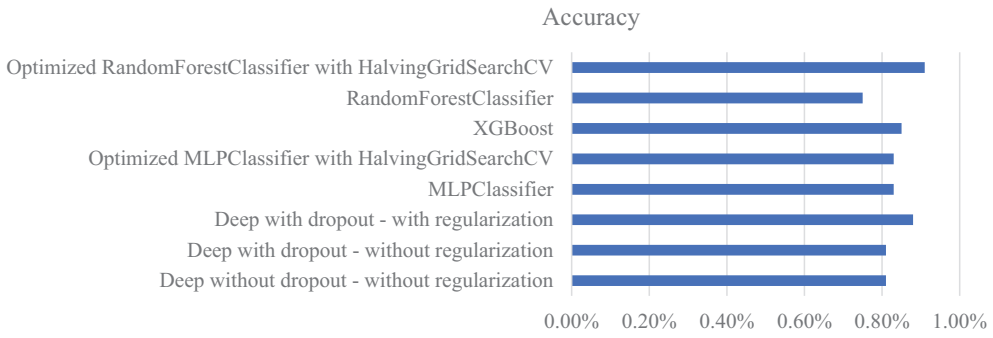

Figure 8: Comparison of the accuracy obtained from the implemented machine learning methods

In contrast, the MLP method likely showed lower performance due to the complexity of hyperparameter tuning in this scenario.

While the Optimized RandomForestClassifier achieved the highest accuracy, its computational cost (47.75 s) was more efficient compared to other top methods, such as XGBoost (93.17 s) and the improved deep learning method with regularization and dropout (368.91 s). The computational cost of methods like XGBoost could limit their practical application in such resource-constrained environments.

Striking a balance between accuracy and execution time is particularly important in real-time IoT applications. While the Optimized RandomForestClassifier achieved the highest accuracy, its execution time, compared to other models with either lower accuracy or much higher computational costs, makes it a more practical choice for IoT systems with limited resources.

In the continuation of this section, the performance of the implemented methods, in terms of precision, recall, and F1-score criteria, is presented in the form of diagrams in Figs. 9–11, respectively. The results show that the proposed method of this research generally performed well in all criteria and was able to identify many attacks with high accuracy. Using Hyperparameter Tuning, this model has been able to simulate rare

attacks well and was successful in identifying more complex classes such as MITM and Recon. After that, the "Deep with dropout-with regularization" model has been able to achieve favorable results, especially in identifying Benign Traffic, Mirai, and DDoS, by combining Dropout and Regularization in different layers. Compared to other models, XGBoost also performed well and was successful in identifying certain classes, such as DDoS and Mirai.

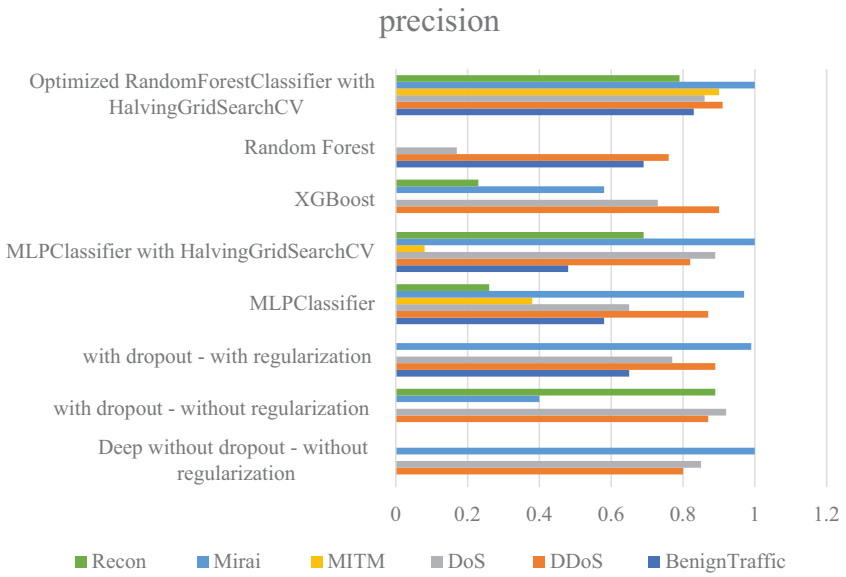

Figure 9: Comparison of the precision obtained from the implemented machine learning methods

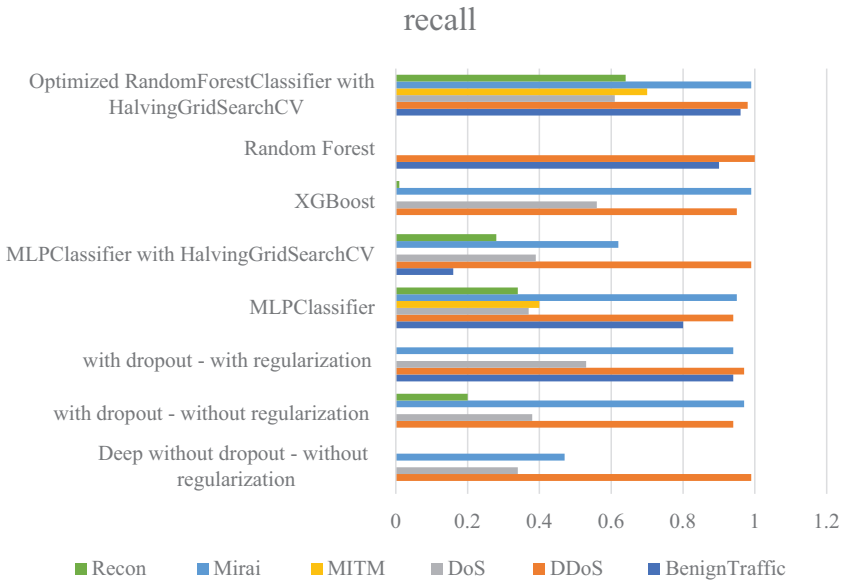

Figure 10: Comparison of the recall obtained from the implemented machine learning methods

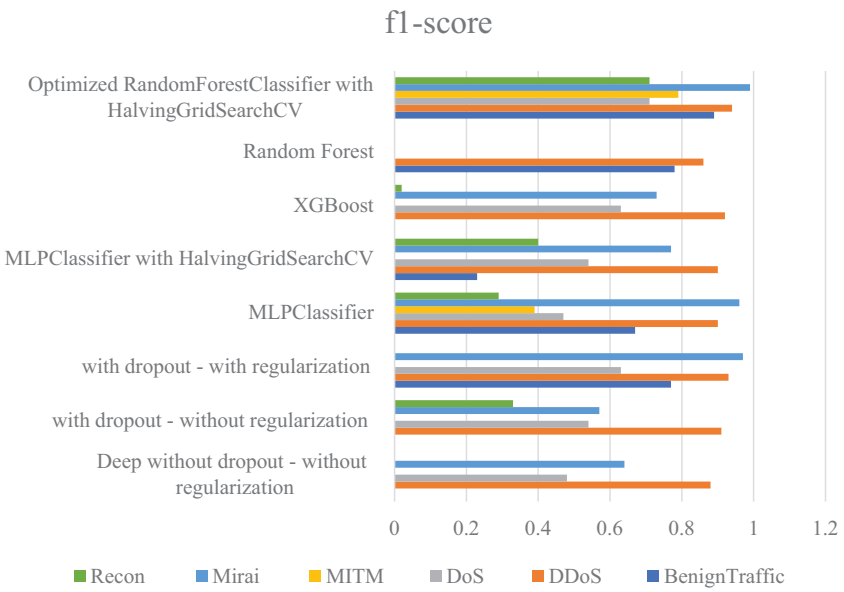

Figure 11: Comparison of the F1-score obtained from the implemented machine learning methods

8 Conclusion

This article explores various machine learning and deep learning methods for intrusion detection in IoT networks. Deep learning techniques, MLP, RF, XGBoost, and their extensions were implemented. These methods were applied to the CICIoT2023 dataset, which is a new dataset in the field of intrusion detection for IoT. The proposed deep learning method was developed using dropout and regularization techniques, while MLP and RF methods were optimized by finding the best values for the parameters using HalvingGridSearchCV. The results obtained from the methods applied to different classes of attacks in the IoT network were examined. The results showed that the Optimized RandomForestClassifier method achieved the best result with an accuracy of 0.91. This method successfully detected all types of attacks, while most of the other methods failed to detect attacks in at least one class. In future work, further studies will focus on the most effective machine learning techniques for intrusion detection systems in IoT. Our goal is to develop an intrusion detection system that can accurately detect all types of attacks in the IoT network and ensure better performance.

Acknowledgement: Special thanks to my friend Hossein Tafazolian for his insightful feedback.

Funding Statement: The author received no specific funding for this study.

Availability of Data and Materials: The data used in this study is available from the Kaggle site.

Ethics Approval: Not applicable.

Conflicts of Interest: The author declares no conflicts of interest to report regarding the present study.

References

1. Sarker IH, Khan AI, Abushark YB, Alsolami F. Internet of Things (IoT) security intelligence: a comprehensive overview, machine learning solutions and research directions. Mob Netw Appl. 2023;28(1):296–312. doi:10.1007/s11036-022-01937-3.

2. Ullah I, Mahmoud QH. Design and development of RNN anomaly detection model for IoT networks. IEEE Access. 2022;10:62722–50. doi:10.1109/ACCESS.2022.3176317.
3. HaddadPajouh H, Dehghantanha A, Parizi RM, Aledhari M, Karimipour H. A survey on Internet of Things security: requirements, challenges, and solutions. Internet Things. 2021;14:100129. doi:10.1016/j.iot.2019.100129.
4. Susilo B, Sari RF. Intrusion detection in IoT networks using deep learning algorithm. Information. 2020;11(5):279. doi:10.3390/info11050279.
5. Alyasiri H, Clark JA, Malik A, de Frein R. Grammatical evolution for detecting cyberattacks in Internet of Things environments. In: 2021 International Conference on Computer Communications and Networks (ICCCN); 2021 Jul 19–22; Athens, Greece. p. 1–6. doi:10.1109/icccn52240.2021.9522283.
6. Roopak M, Tian GY, Chambers J. An intrusion detection system against DDoS attacks in IoT networks. In: 2020 10th Annual Computing and Communication Workshop and Conference (CCWC); 2020 Jan 6–8; Las Vegas, NV, USA; p. 562–7. doi:10.1109/ccwc47524.2020.9031206.
7. Le TTH, Kim H, Kang H, Kim H. Classification and explanation for intrusion detection system based on ensemble trees and SHAP method. Sensors. 2022;22(3):1154. doi:10.3390/s22031154.
8. Nimbalkar P, Kshirsagar D. Feature selection for intrusion detection system in Internet-of-Things (IoT). ICT Express. 2021;7(2):177–81. doi:10.1016/j.icte.2021.04.012.
9. Alabsi BA, Anbar M, Rihan SDA. CNN-CNN: dual convolutional neural network approach for feature selection and attack detection on Internet of Things networks. Sensors. 2023;23(14):6507. doi:10.3390/s23146507.
10. Awotunde JB, Folorunso SO, Imoize AL, Odunuga JO, Lee CC, Li CT, et al. An ensemble tree-based model for intrusion detection in industrial Internet of Things networks. Appl Sci. 2023;13(4):2479. doi:10.3390/app13042479.
11. Ullah I, Mahmoud QH. A two-level flow-based anomalous activity detection system for IoT networks. Electronics. 2020;9(3):530. doi:10.3390/electronics9030530.
12. Adefemi Alimi KO, Ouahada K, Abu-Mahfouz AM, Rimer S, Alimi OA. Refined LSTM based intrusion detection for denial-of-service attack in Internet of Things. J Sens Actuator Netw. 2022;11(3):32. doi:10.3390/jsan11030032.
13. Al S, Dener M. STL-HDL: a new hybrid network intrusion detection system for imbalanced dataset on big data environment. Comput Secur. 2021;110:102435. doi:10.1016/j.cose.2021.102435.
14. Chen J, Xiao J, Xu J. VGGIncepNet: enhancing network intrusion detection and network security through non-image-to-image conversion and deep learning. Electronics. 2024;13(18):3639. doi:10.3390/electronics13183639.
15. Devlin J, Chang MW, Lee K, Toutanova LK. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers); 2019 Jun 2–7; Minneapolis, MN, USA. doi:10.18653/v1/N19-1423.
16. Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108. 2019. doi:10.48550/arXiv.1910.01108.
17. Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov R, Le QV. XLNet: generalized autoregressive pretraining for language understanding. arXiv:1906.08237. 2019. doi:10.5555/3454287.3454804.
18. Wang Z, Chen H, Yang S, Luo X, Li D, Wang J. A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. PeerJ Comput Sci. 2023;9:e1569. doi:10.7717/peerj-cs.1569.
19. Parfenov D, Grishina L, Zhigalov A, Parfenov A. Investigation of the impact effectiveness of adversarial data leakage attacks on the machine learning models. ITM Web Conf. 2024;59:04011. doi:10.1051/itmconf/20245904011.
20. Abbas S, Bouazzi I, Ojo S, Al Hejaili A, Sampedro GA, Almadhor A, et al. Evaluating deep learning variants for cyber-attacks detection and multi-class classification in IoT networks. PeerJ Comput Sci. 2024;10:e1793. doi:10.7717/peerj-cs.1793.
21. Neto ECP, Dadkhah S, Ferreira R, Zohourian A, Lu R, Ghorbani AA. CICIoT2023: a real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors. 2023;23(13):5941. doi:10.3390/s23135941.