Computer Science • 26(2) 2025 https://doi.org/10.7494/csci.2025.26.2.6450

Komal Jakotiya
Vishal Shirsath
SharanBasava Inamdar

INTRUSION DETECTION WITH MACHINE LEARNING: A TWO-STEP FEDERATED APPROACH USING THE CIC IoT 2023 DATASET

Abstract *The main objective of the planned effort is to provide analytical analyses of current intrusion detection systems grounded on ML algorithms. Furthermore, examined in this work are the useful data sets and several techniques already in use to develop an effective IDS using single, hybrid, and ensemble machine learning algorithms. The approaches in the literature have then been investigated under several criteria to provide a clear road and direction for the next projects that will be successful. Nowadays, companies of all kinds include an intrusion detection system (IDS), which inhibits cybercrime to protect the network, resources, and private data. Many strategies have been suggested and implemented up till now to prevent uncivil behaviour. Since machine learning (ML) approaches are successful, the proposed approach applied several ML models for the intrusion detection system. The CIC IoT 2023 Dataset is the one applied in this paper, and a two-step process for Intrusion detection was proposed. Tested with several techniques including random forest, XGBoost, logistic regression, MLP model, and RNN. Following fine-tuning, the federated learning model using neural networks had the best accuracy – 99.84%.*

Keywords LSTM, MLP, XG Boost, federated learning, logistic regression, RNN

Citation Computer Science 26(2) 2025: 79–97

1. Introduction

When it comes to the field of cybersecurity, one of the most important tasks that must be completed is the detection of breaches in computer networks. This is done to guarantee the availability, integrity, and confidentiality of information and resources. Intrusion Detection Systems, often known as IDS, are a type of security mechanism that monitors network traffic to identify any indications of unauthorized access, hostile actions, or unusual behavior [15]. Because of the ever-increasing attack surface that modern networks present, as well as the growth of complex cyber threats, there is an urgent need for intrusion detection solutions that are both robust and adaptable. When it comes to intrusion detection, traditional methods, which are based on predetermined rules and signatures, frequently struggle to keep up with the continually changing threat landscape. As a consequence of this, there is an increasing interest in utilizing cutting-edge methods such as ML (machine learning) to enhance the capabilities of intrusion detection. Machine learning algorithms are able to evaluate enormous amounts of network data, derive significant patterns, and recognize tiny deviations that are suggestive of hostile behavior. This research is centered on the utilization of machine learning techniques for the purpose of intrusion detection. The objective is to improve the accuracy, scalability, and responsiveness of intrusion detection systems (IDS). Our goal is to construct intelligent systems that are capable of independently identifying and categorizing a wide range of intrusions [18]. This will be accomplished by training machine learning models on labeled datasets that comprise both legitimate and malicious network traffic. The major purpose of this research is to investigate the effectiveness of various machine learning methods, including decision trees, random forests, support vector machines, and neural networks, in identifying intrusions that occur inside network traffic. The purpose of this study is to evaluate the performance of these algorithms in terms of detection accuracy, false positive rates, and computing efficiency [17]. This evaluation will be accomplished by empirical analysis and experimentation. By contributing to the advancement of the current state of the art in machine learning-based intrusion detection, the purpose of this research is to contribute to the creation of cybersecurity solutions that are more effective and adaptable. In conclusion, the insights that were acquired from this study can be used to inform the planning and execution of intrusion detection systems that are better able to protect against the shifting techniques that cyber adversaries employ [3]. There are a lot of different types of intrusions into computer systems and networks, each with its own method and goal. Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks flood a target with too much data. Man-in-the-middle (MitM) attacks happen when someone listens in on two people talking to each other and changes what they say [7]. Assailants can get into a system without being allowed and increase their access rights by taking advantage of security holes. These are both types of system-based intrusions. Malware attacks, which use viruses to damage, disrupt, or get into systems without permission, are also popular. Application-based attacks, such as SQL injection and Cross-Site Scripting (XSS), use

flaws in web applications to change databases or run malicious scripts. Insider threats are very dangerous, because bad insiders can do harm on purpose, and good insiders can compromise security by accident. Finally, to get around security measures, "physical intrusions" and "social engineering" attacks trick people or directly get into physical systems [28]. Knowing about these different kinds helps you come up with complete safety plans. Different types of IDS is shown in Figure 1.

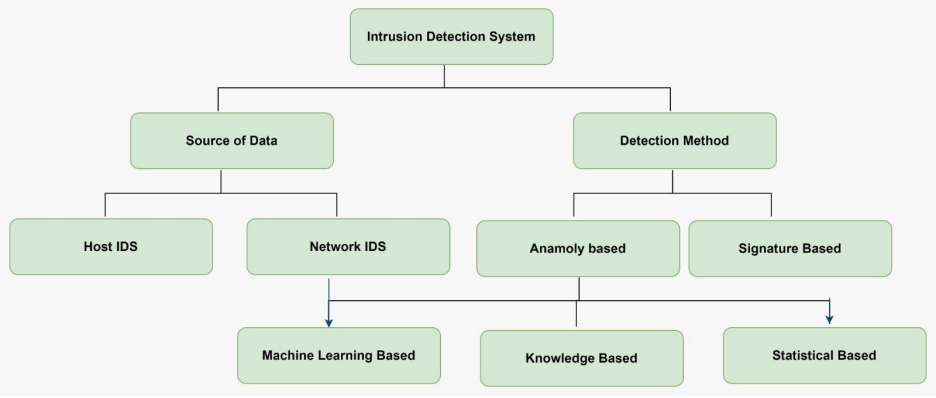

Figure 1. Different types of IDS

a. Anomaly-based Methodology: IDSs that use anomaly-based detection identify intrusions by looking at changes from usual patterns or behavior. They create a reference point for typical behavior and mark any changes as possible intrusions [9]. Using statistical models, machine learning techniques, or behavioural analysis approaches, anomaly-based intrusion detection systems (IDS) hunt abnormal activities.

b. Misuse detection systems, also called as signature-based intrusion detection systems, match seen events to a database of known attack patterns or signatures to identify intrusions. If the signature of an incoming event matches one in the database, it is deemed most likely invasive. Although signature-based intrusion detection systems are excellent in seeing known attacks, they could find difficulty spotting fresh or undiscovered threats [23].

c. Network-based Intrusion Detection System (NIDS): NIDS keep an eye on network traffic to look for unusual or suspicious behavior. They can identify many kinds of network-based attacks, including malware transfer, denial-of-service (DoS) attacks, and port scans, by analyzing packets as they move over the network. To track traffic at critical junctures in the network, NIDS is positioned strategically [4].

d. Host-based Intrusion Detection System (HIDS): These systems are placed on specific hosts or network endpoints. Rather than keeping an eye on network traffic, they concentrate on the actions and characteristics of the host systems [22]. HIDS scans for intrusions or questionable activity directed at specific hosts by examining system logs, file integrity, registry modifications, and other host-specific data.

e. Application-based Intrusion Detection System (AppIDS): AppIDS are designed to monitor and safeguard particular services or applications. To find abnormalities or attacks aimed at particular applications, they examine data and traffic at the application level [8]. databases, Websites, and business apps are just a few examples of the vital applications for which AppIDS offers granular visibility and security.

Contributions of the work

- The study makes sure that the data is balanced by sampling to match all groups in the CIC IoT 2023 dataset.
- Several pre-processing methods were employed to balance the dataset and generalize it. This makes data more reliable and fixes the common problem of class imbalance in intruder detection datasets.
- A comparative study with different machine learning models was made.
- A novel and a two-step model is proposed for detecting the IDS attacks more accurately.
- The study looks at the model in great detail and compares it to a dataset to see how well it works. It also looks at other methods and gives specific performance metrics.

1.1. Organization of paper

The paper will be organized in the following sections. Relevant current work related to the demonstrated approach is covered in Section 2. Section 3 goes into specifics of the dataset, including features like an overview and the pre-processing of the dataset. Explaining the model architectures is Section 4. An introduction to pseudo codes is provided in Section 5. The findings comparison and analysis are provided in Section 6 The conclusion of the work is given in Section 7. The future work on the topic is explained in Section 8.

2. Literature survey

Several works were studied and some prominent done in this field related to IDS attacks and detection are discussed in Table 1. The strengths and weaknesses of the models are discussed as below.

Table 1
Literature survey

Citation	Year	Dataset used	Method used	Weakness	Strength of the proposed model
[10]	2024	CIC IOT 2023	MLP, Auto-encoders	The methodology received lesser accuracy. It used GSK in pre-processing which does not address the issue of data imbalance properly	It received higher accuracy than the existing model. Due to the use of under-sampling and SMOTE, the issue of dataset imbalance is addressed

Table 1 cont.

Citation	Year	Dataset used	Method used	Weakness	Strength of the proposed model
[26]	2024	KDD CUP 99, CIC-IDS-2017, and IOT-23	Auto encoder-based hybrid detection model	Will be good while working with less data. The dataset used is not the latest and updated as they do not cover modern attacks	The proposed model can be used for large data. The dataset used is an up-to-date model which consists of modern attacks and features
[19]	2024	SDN-IOT	ML Models (KNN, AdaBoost, Logistic Regression)	Received comparatively lower accuracy than the proposed model	A new and complex dataset has been used. Comparatively received higher accuracy
[21]	2023	UNSW-NB15	A hybrid of LSTM and CNN	The hybrid model might be more complex and require more resources and computation. The dataset worked on is not recent, so it might not be able to predict new attacks	Less complex as it works based on local models and aggregation. The dataset worked on here is the latest dataset
[5]	2023	NSL-KDD, UNSW-NB15, and CI-CIDS2017	DT, RF, and Boosting Algorithms	It does not use modern datasets, which include features of IoT network traffic	Includes modern datasets
[1]	2023	TON-IoT, Edge-IIoTset, and UNSW-NB15	Deepak-IoT	Lesser accuracy compared to the proposed model	Received relatively higher accuracy
[11]	2022	Gas pipeline and UNSW-NB15	LSTM-Autoencoder	It requires centralized data which may cause privacy concerns	The proposed model keeps data only at the local level, so no privacy concerns
[13]	2020	Not mentioned	Random Forest	Performed only binary classification and received an accuracy of 86.7%	Performed multi-class classification of types of attacks
[12]	2020	UNSW-NB15	Tree-based classifiers	Performed only binary classification	Performed multi-class classification of attacks

Table 1 cont.

Citation	Year	Dataset used	Method used	Weakness	Strength of the proposed model
[24]	2019	UNSW-NB15	Deep Neural Network	Less effective results than binary classification in multi-class classification	Gives a higher accuracy result in multi-class classification and improves generalization
[25]	2019	UNSW-NB15	ANN	Lower accuracy of 84%	Proposed model gains more accuracy compared to this
[20]	2019	UNSW-NB15	LSTM	Lower accuracy around 72% for multi-class classification of attacks	Higher accuracy compared to the model and the dataset used is more advanced and recent, helping create a good model
[2]	2017	KDD99 and UNSW-NB15	Logistic Regression	Complex patterns and relationships found in complex IDS datasets escape this linear relationship, which logistic regression considers, which could result in less accurate detection of complicated assaults	The CIT dataset used here contains more complex features, and Federated Learning uses data from different sources without centralizing it, enhancing the power of the models. This makes the model more effective at managing diverse kinds of attacks

2.1. Summary of the existing literature

Above literature survey The 2017–2024 field of intrusion detection effort utilizing machine learning and deep learning approaches spans various datasets including NSL – KDD, UNSWNB15, CICIDS2017, TON – IoT, and Edge – IIoTset. Over these years, several algorithms, including random forest, regression, LSTM, CNN, deep neural networks, etc., were applied. The table below briefly addresses their shortcomings and how the suggested model addresses them.

3. Dataset

3.1. Dataset description

This work uses a CIC IoT 2023-based dataset. The IoT Dataset 2023 developed by the Canadian Institute for Cybersecurity aims to support the evolution of security analytics in Internet of Things environments. Included are data on 33 various types of attacks – DDoS, DoS, Recon, Web-based, Brute Force, Spoofing, and Mirai – that were executed on a network of 105 Internet of Things devices. The dataset enables evaluation of machine learning and deep learning algorithms for IoT network traffic classification and detection.

A chunk of one million rows of the dataset is taken to work on. In-depth network traffic characteristics that are crucial for IoT security research are included in the IoT Dataset. Flag numbers (e.g., "fin_flag_number," "syn_flag_number," "rst_flag_number") and counts (e.g., "ack_count," "syn_count," "fin_count") are among the dataset's key fields. Columns including "flow_duration," "header_length," "protocol_type," "duration," "rate," "srate," and "drate" are also included. With functions like "Tot sum," "Min," "Max," "AVG," "Std," "Tot size," "IAT," "Number," "Magnitude," "Radius," "Covariance," "Variance," and "Weight," statistical metrics are well-represented. Columns exclusive to a protocol include "HTTP," "HTTPS," "DNS," and "Telnet."

A "label" column within the dataset further classifies seven different kinds of DDoS attacks: "DDoS-RSTFINFlood" (0), "DDoS-ICMP_Flood" (1), "DDoS-SynonymousIP_Flood" (2), "DDoS-SYN_Flood" (3), "DDoS-PSHACK_Flood" (4), "DDoS-TCP_Flood" (5), and "DDoS-UDP_Flood" (6).

3.2. Data preprocessing for DDoS attack identification

A significant imbalance is found in the initial label distribution. The initial count is as follows: "DDoS-ICMP_Flood" (158,671), "DDoS-UDP_Flood" (119,350), "DDoS-TCP_Flood" (99,442), "DDoS-PSHACK_Flood" (90,465), "DDoS-SYN_Flood" (89,842), "DDoS-RSTFINFlood" (89,078), and "DDoS-SynonymousIP_Flood" (79,263). We used an undersampling strategy [13] to correct this imbalance and guarantee a balanced dataset from models. By lowering the number of instances in the majority classes to the size of the smallest class, an equal representation of all classes was achieved. Following undersampling, each class had 79,263 instances and the label distribution was evenly balanced. The graphs before and after under-sampling are given in Figure 2 and Figure 3 respectively.

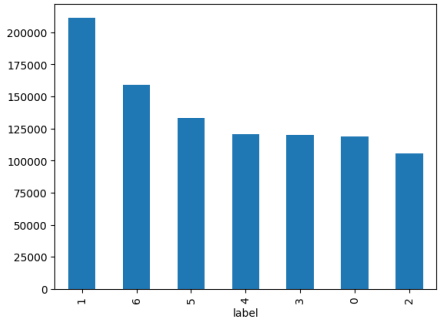

Figure 2. Before under sampling

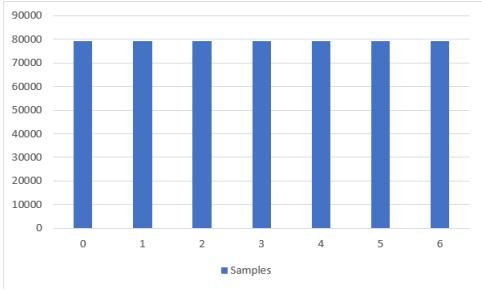

Figure 3. After under sampling

This preprocessing step was essential to minimize the errors brought about by class imbalance, which improves the accuracy and dependability of our prediction models in identifying different kinds of DDoS attacks. After pre-processing, the count of labels is discussed in Table 2.

Table 2
Sample table with sample data

Labels	Label numbers	Initial count	Count after under-sampling
RSTFINFlood	0	158,671	79,263
DDoS-ICMP_Flood	1	119,350	79,263
DDoS-SynonymousIP_Flood	2	99,442	79,263
DDoS-SYN_Flood	3	90,465	79,263
DDoS-PSHACK_Flood	4	89,842	79,263
DDoS-TCP_Flood	5	89,078	79,263
DDoS-UDP_Flood	6	79,263	79,263

To address the issue of class imbalance effectively and ensure that the model generalizes well to real-world scenarios, a multi-step approach was adopted in the methodology. Initially, undersampling was employed to reduce the size of the majority class. This step aimed to bring all classes to a similar level, ensuring that the model does not become biased towards the majority class. However, undersampling alone could result in a loss of valuable information from the majority class, potentially leading to reduced model performance and generalizability.

To mitigate these potential drawbacks, SMOTE (Synthetic Minority Oversampling Technique) was integrated into the process. After performing undersampling to manage the majority class, SMOTE was applied to the minority class. This technique generates synthetic samples by interpolating between existing data points, which helps increase the representation of the minority class without simply duplicating existing data. By creating more diverse training data, the model is better equipped to learn the underlying patterns and relationships, thereby enhancing its performance and robustness.

Additionally, to prevent the risk of overfitting associated with synthetic data generation, regularization techniques, such as L2 regularization, were implemented to penalize overly complex models. This regularization strategy reduces the likelihood that the model will fit too closely to the training data, including the synthetic samples created by SMOTE, thereby improving its ability to generalize to unseen, real-world data. This balanced approach combining undersampling, SMOTE, and regularization provides a comprehensive strategy to address class imbalance while maintaining both fairness and generalizability in the model's performance.

4. Methodology

The proposed intrusion detection methodology employs a two-step approach with specialized algorithms to enhance the accuracy of detecting and classifying attacks:

1. **Step 1**: Determine whether the attack is known or unknown.
2. **Step 2**: Identify the specific type of attack if it is known.

The first detection is done using autoencoders. The Autoencoder consists of an encoder that compresses the input and a decoder that tries to reconstruct the original input from this compressed representation. The architecture is kept relatively simple to focus on learning the normal patterns. The Autoencoder is trained using only the data for known attack patterns. During training, the model learns to reconstruct these patterns with minimal error

4.1. Detection of unknown attacks

- **Reconstruction Error**: When new data is fed into the trained Autoencoder, the model attempts to reconstruct it. If the data is significantly different from what the model has seen during training (i.e., a new or unknown attack), the reconstruction error will be high.
- **Threshold Setting**: A threshold is set on the reconstruction error. Inputs with an error above this threshold are classified as "unknown attacks." Else classified as "known".

If the attack is unidentified, a manual analysis is conducted, halting progress to the next step. But if the attack is known, then the second step is performed, to detect what type of attack was done. For this, The first step in the proposed methodology of anomaly detection is shown in Figure 4.

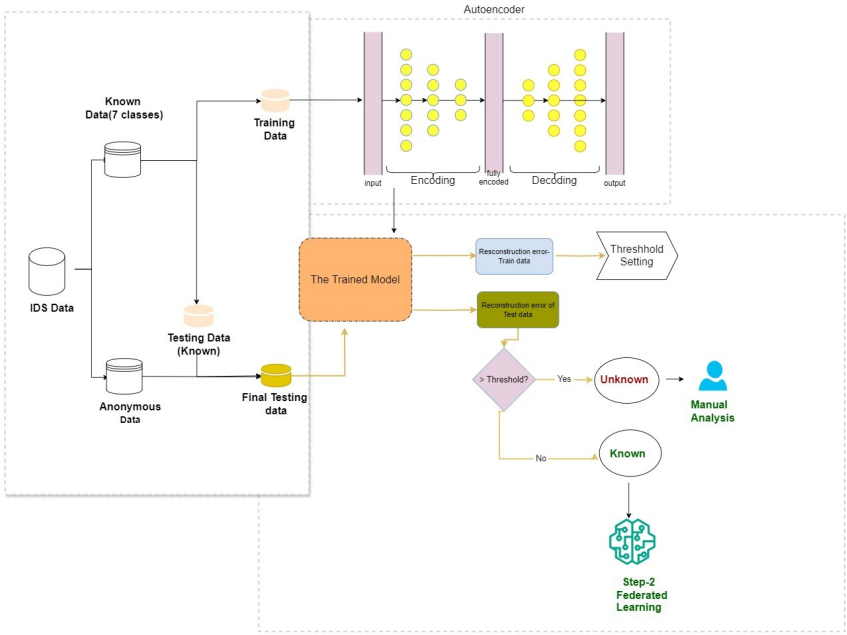

Figure 4. Step 1 of proposed methedology

After the first step, if the attack is identified as known, the process advances to the second step, where the data is fed into the federated learning framework. The

model is tested on the testing set and thereafter evaluated after several iterations. This part clarifies the specific facts and framework of federated learning with neural networks. Federated Learning is a decentralized approach to machine learning, where the training process is distributed across multiple devices or computers. Unlike conventional machine learning methods [27–29], which involve sending data collected from devices to a central server for model training [12], federated learning keeps the data on each device, thereby preserving data privacy. In federated learning, a global model is iteratively optimized by training it locally on each device using the data stored on that device. Instead of sending raw data to a central server, only the changes to the model parameters (such as gradients) are shared with the central server [6]. This allows the central model to be improved across multiple rounds by aggregating the locally computed updates from many devices, without exposing the underlying data. The core process of federated learning involves training local models on the data residing on each client device and then securely combining these updates to refine a global model [16]. This process, which maintains data privacy while optimizing the global model, Federated learning detail architecture is illustrated in Figure 5 below.

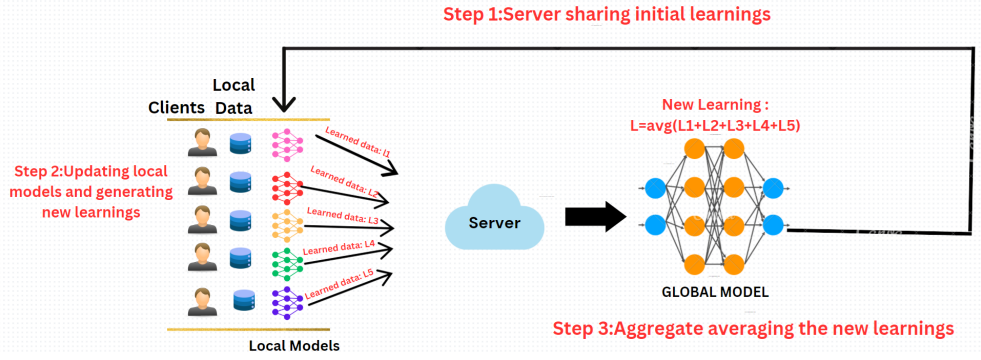

Figure 5. Federated Learning that is employed in Step 2

Client: Any device or local entity holding data and capable of making computations (e.g., smartphones, local servers). Server: The central entity that handles the federated learning process, aggregates model updates from clients, and shares with the global model. Local Model: A machine learning model based on the client's local data. Global Model: The aggregated model after combining information from various clients.

Federated Averaging (FedAvg): A popular algorithm used to combine model updates. The proposed federating method includes training local models on multiple clients, combining the model weights using weighted averaging, and fine-tuning the aggregated global model. In this model, there are 5 clients. The distribution of data among clients is as follows.

Each clientś training data is made using stratified sampling to ensure that the class distribution is consistent across all clients. Stratified sampling divides the dataset into groups based on class names and then takes an equal-sized sample from each group. This makes sure that all subsets (each clientś dataset) match the original datasetś class distribution. This method keeps the original class numbers in each subset, which makes sure that everything is unbiased and equal. The client data is stored in a list, with each dataset having 443,872 rows and 46 columns.

4.2. Federated model architecture

This work uses dense neural networks for federated learning, which matches the distributed nature of the training process. We choose a dense neural network model, in which every neuron in one layer is coupled to every neuron in the next layer. The DNN consists of one input layer,2 hidden layers, and an output layer. This architecture is well-suited for handling complex connections within data, making it widely applicable across various machine-learning tasks. L2 regularization and dropout were introduced in dense layers to prevent overfitting. They penalize the weights and can drop or deactivate neurons so that the model can get generalized. The parameters used for training the client that is a dense neural network are given in the Table 3.

Table 3
Training parameters

Parameters	Values
Optimizer	Adam
Loss function	Sparse categorical entropy
Activation function	Softmax
Epochs	10

$$Loss = \frac{1}{N}\sum_{i=1}^{N}\log(p_i \cdot y_i) \tag{1}$$

$$softmax(a_i) = \frac{e^{a_i}}{\sum_{j=1}^{n} e^{a_j}} \tag{2}$$

Each client locally trains the neural network model on its own data for 10 iterations. After training, the model updates are saved locally, and instead of sharing raw data, clients only share the updated model weights. These weights are then aggregated centrally to update the global model, ensuring data privacy while improving the overall model collaboratively. Five clients are involved in this federated learning setting. This makes sure that the training is spread out and interconnected, using data from different sources while protecting privacy [14].

4.3. Aggregating weights

Through federated learning, every client trains its model locally, and weighted averaging is used to combine the weights from all client models to update the global

model. This approach guarantees that clients having more data have a bigger impact on the global model. The global model that has been compiled is then improved on a combined dataset by using information from every client to improve generalization Steps involved in aggregating weights are:

1. **Launch Aggregated Weights:**
2. The collected weights for every model layer are stored in an empty list or array.
3. **Sum Weights**

Compute the weighted total of the weights from every client model for every layer in the models. The quantity of samples each client has scaled its weights. This guarantees a bigger impact on clients with more data on the global model.

$$W^{(l)} = \sum n \cdot w \tag{3}$$

Dividing the weighted total of the weights by the overall number of samples across all clients yields the average weights for every layer.

$$w = \frac{W}{\sum n} \tag{4}$$

4.4. Global model and fine-tuning

A new global model having the same architecture as the client models will be built after the averaged weights for each layer have been determined. Then these averaged weights are used to set the weights of the world model. This procedure guarantee that each client's local training gives the information and patterns that are included into the global model. Effectively reflecting the total contributions of all clients, weighted by the amount of data each client gave, the global model's weights are set to the averaged weights. This aggregation technique that guarantees clients with more data have a relatively bigger effect on the global model.

After aggregating the weights from all clients, the global model is further fine-tuned using a separate dataset that is not client specific. This fine-tuning process leverages the combined knowledge embedded in the client model updates, improving the global model's generalization and performance on unseen data while maintaining client data privacy. This means the global model can become more accurate because it learns to generalize better to new, unseen data. Training on a larger and more diverse dataset allows the model to adapt to different situations and data patterns. Fine-tuning is a crucial step in federated learning because it ensures the model works well across various contexts and data distributions.

5. Pseudo code

Algorithm 1: Federated Learning with Neural Network

1. Input: Number of clients $K = 5$, Number of global iterations T, Learning rate η, Initial model parameters w_0
2. Output: Final model parameters w_T

3. Initialize global model parameters w_0
4. **for** each global iteration $t = 0, 1, \ldots, T-1$ **do**
   (a) Sample a subset of clients $S_t \subseteq \{1, 2, \ldots, K\}$
   (b) **for** each client $k \in S_t$ **in parallel do**
      i. Download current global model parameters w_t to client k
      ii. Client k trains the neural network locally using its local data to compute w_k
   (c) **end for**
   (d) Aggregate local model updates at the server:

$$w_{t+1} = \frac{1}{|S_t|} \sum_{k \in S_t} w_k$$

5. **end for**
6. **return** w_T

6. Results

In this work, the CIC IoT dataset 2023 which consists of 7 different classes of attacks with different have been tested with models like random forest, XG Boost, logistic regression, MLP, RNN, and federated learning. Out of all these, the proposed federated learning model with a dense neural network received the highest accuracy. It outperformed the remaining models with training accuracy, testing accuracy, precision, and F1-Score.

Table 4
Comparison of different models

Model	Accuracy	F1-Score	Recall	Loss	Confusion matrix
Federated Learning (NN)	99.84	0.92931	0.9301552	0.0055	Figures 6, 7
Random Forest	95%	0.92931	0.9301552	–	Figure 8
XGBoost	95.96%	0.9296	0.9303	–	Figure 9
Logistic Regression	92.73%	0.9274	0.9276	–	Figure 10
MLP	93.09%	–	–	0.1420	Figure 11
RNN	91.21%	–	–	0.1977	Figure 12

This proposed model which used neural networks as global and local models received a Global model fine-tuning accuracy of 0.9984 loss of 0.0055 and recall of 0.99830. In the below Figure 6 ROC curve of the model is given and in Figure 7 confusion matrix for the predictions made is given. The comparison for all the model's performance is given in Table 4, hyper parameters in Table 5, and computational resources in Figure 13.

Table 5
Hyper parameters

Model	Hyperparameter	Value	Description
Federated Learning	Learning Rate	0.001	Controls the step size in the optimization process.
	Batch Size	32	Determines the number of samples used in each update.
Random Forest	Number of Trees	100	The number of decision trees in the ensemble.
	Max Depth	10	The maximum depth of each decision tree.
XGBoost	Learning Rate	0.1	Controls the step size in the gradient boosting process.
	Max Depth	6	The maximum depth of each tree in the ensemble.
Logistic Regression	Regularization	L2	Applies L2 regularization to prevent overfitting.
	C	1	Inverse regularization strength. Higher values mean stronger regularization.
MLP	Learning Rate	0.01	Controls the step size in the optimization process.
	Hidden Layers	2	The number of hidden layers in the neural network.
RNN	Learning Rate	0.01	Controls the step size in the optimization process.
	Hidden Units	100	The number of units in each hidden layer.

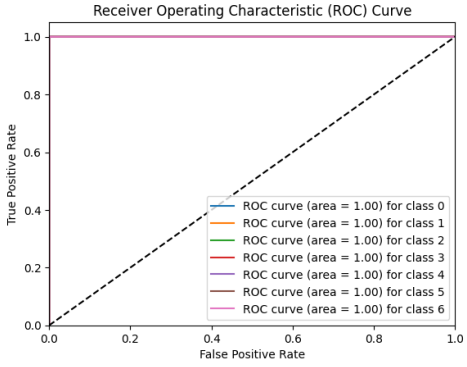

Figure 6. ROC curve

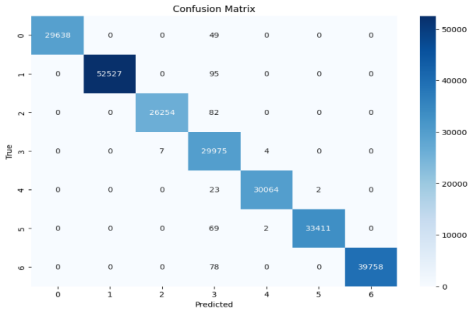

Figure 7. Confusion matrix for proposed model

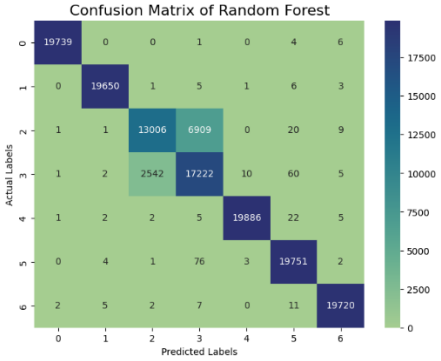

Figure 8. Confusion matrix for Random Forest

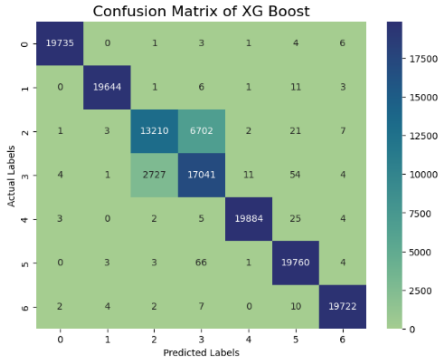

Figure 9. Confusion matrix for XG Boost

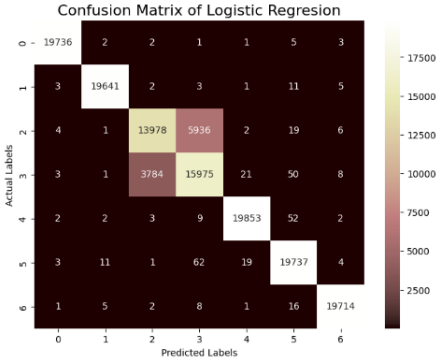

Figure 10. Confusion matrix for logistic regression

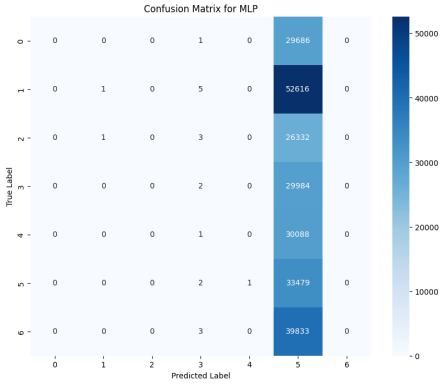

Figure 11. Confusion matrix for MLP

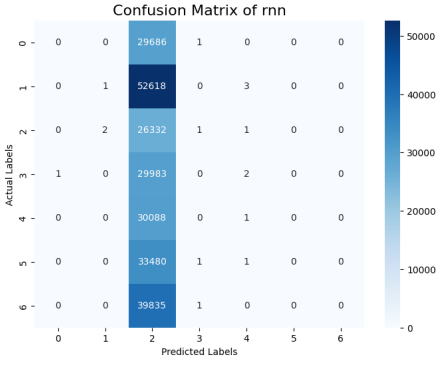

Figure 12. Confusion matrix for RNN

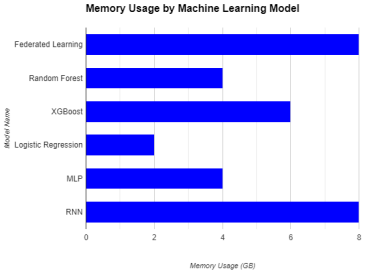

Figure 13. Graphical comparison of various methods computational resources

6.1. System configuration

The research was done on a Google Colab laptop, specifically the base model. It had an Intel i3 7th generation 2-core processor clocked at 2.30 GHz and 8 GB of RAM. This set-up for computing made it easy to do a full evaluation of how well the model worked at detecting intrusions and types.

7. Conclusion

In conclusion, the work done so far shows how correctly the federated learning model using dense neural networks as local and global models can predict the type of DDoS attack using the CIC 2023 IOT dataset by maintaining privacy. It will be able to predict 7 different types of DDoS attacks. Compared to the traditional models like random forest, XG boost, RNN, MLP and logistic regression federated learning model achieved higher accuracy. The data pre-processing step played a very vital role in increasing the performance of the model.

8. Future scope

Future studies should look into how scalable the federated learning architecture is to support larger datasets and a wider variety of Internet of Things devices, hence enhancing the generalization of the model. Developing real-time processing abilities will enable fast identification and prevention of DDoS attacks. Furthermore confirming the model's practical usefulness are sophisticated privacy-preserving methods that may be used to guarantee data security across decentralized networks.

Furthermore, the detection accuracy for unknown attack types may be much increased by combining hybrid models that integrate federated learning with methods like transfer learning. Entire assessments in various network settings will support the validity of the model in practical applications. For models used on Internet of Things devices with restricted power supplies, optimization is especially crucial. Using the developed models in practical settings will, at last, provide crucial information regarding their applicability in real life and areas that require improvement.

Acknowledgements

The research presented in this paper was partially supported by Ajeenkya DY Patil University (ADYPU) for their valuable support and resources, which significantly contributed to the successful completion of this research.

References

[1] Abdulla A.R., Jameel N.G.M.: A review on IoT intrusion detection systems using supervised machine learning: Techniques, datasets, and algorithms, *UHD Journal of Science and Technology*, vol. 7(1), pp. 53–65, 2023. doi: 10.21928/uhdjst.v7n1y2023.pp53-65.

[2] Adeyemo V.E., Abdullah A., JhanJhi N.Z., Supramaniam M., Balogun A.O.: Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: an empirical study, *International Journal of Advanced Computer Science and Applications*, vol. 10(9), 2019. doi: 10.14569/ijacsa.2019.0100969.

[3] Agoramoorthy M., Ali A., Sujatha D., Michael Raj. TF, Ramesh G.: An Analysis of Signature-Based Components in Hybrid Intrusion Detection Systems. In: *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, pp. 1–5, IEEE, 2023. doi: 10.1109/iccebs58601.2023.10449209.

[4] Ainurrochman, Nugroho A., Wahyuwidayat R., Sianturi S.T., Fauzi M., Ramadhan M.F., Pratomo B.A., Shiddiqi A.M.: Ensemble methods classifier comparison for anomaly based intrusion detection system on CIDDS-002 dataset. In: *2021 13th International Conference on Information & Communication Technology and System (ICTS)*, pp. 62–67, IEEE, 2021. doi: 10.1109/icts52701.2021.9608714.

[5] Altunay H.C., Albayrak Z.: A hybrid CNN+ LSTM-based intrusion detection system for industrial IoT networks, *Engineering Science and Technology, an International Journal*, vol. 38, 101322, 2023. doi: 10.1016/j.jestch.2022.101322.

[6] Assiri A.: Anomaly classification using genetic algorithm-based random forest model for network attack detection, *Computers, Materials & Continua*, vol. 66(1), 2021. doi: 10.32604/cmc.2020.013813.

[7] Borkar A., Donode A., Kumari A.: A survey on Intrusion Detection System (IDS) and Internal Intrusion Detection and protection system (IIDPS). In: *Proceedings of the International Conference on Inventive Computing and Informatics (ICICI 2017)*, pp. 949–953, IEEE, 2017. doi: 10.1109/icici.2017.8365277.

[8] Ding W., Abdel-Basset M., Mohamed R.: DeepAK-IoT: An effective deep learning model for cyberattack detection in IoT networks, *Information Sciences*, vol. 634, pp. 157–171, 2023. doi: 10.1016/j.ins.2023.03.052.

[9] Dong Y., Wang R., He J.: Real-time network intrusion detection system based on deep learning. In: *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 1–4, IEEE, 2019. doi: 10.1109/icsess47205.2019.9040718.

[10] Gheni H.Q., Al-Yaseen W.L.: Two-step data clustering for improved intrusion detection system using CICIoT2023 dataset, *e-Prime-Advances in Electrical Engineering, Electronics and Energy*, vol. 9, 100673, 2024. doi: 10.1016/j.prime.2024.100673.

[11] Hanif S., Ilyas T., Zeeshan M.: Intrusion detection in IoT using artificial neural networks on UNSW-15 dataset. In: *2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life using ICT & IoT and AI (HONET-ICT)*, pp. 152–156, IEEE, 2019. doi: 10.1109/honet.2019.8908122.

[12] Khammassi C., Krichen S.: A GA-LR wrapper approach for feature selection in network intrusion detection, *Computers & Security*, vol. 70, pp. 255–277, 2017. doi: 10.1016/j.cose.2017.06.005.

[13] Khan I.A., Keshk M., Pi D., Khan N., Hussain Y., Soliman H.: Enhancing IIoT networks protection: A robust security model for attack detection in Internet Industrial Control Systems, *Ad Hoc Networks*, vol. 134, 102930, 2022. doi: 10.1016/j.adhoc.2022.102930.

[14] Kim T., Pak W.: Early detection of network intrusions using a GAN-based one-class classifier, *IEEE Access*, vol. 10, pp. 119357–119367, 2022. doi: 10.1109/access.2022.3221400.

[15] Kiran A., Prakash S.W., Kumar B.A., Likhitha, Sameeratmaja T., Charan U.S.S.R.: Intrusion Detection System Using Machine Learning. In: *2023 International Conference on Computer Communication and Informatics (ICCCI)*, pp. 1–4, IEEE, 2023. doi: 10.1109/iccci56745.2023.10128363.

[16] Kumar V., Sinha D., Das A.K., Pandey S.C., Goswami R.T.: An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset, *Cluster Computing*, vol. 23, pp. 1397–1418, 2020. doi: 10.1007/s10586-019-03008-x.

[17] Li J.: Network Intrusion Detection Algorithm and Simulation of Complex System in Internet Environment. In: *Proceedings of the 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA 2022)*, pp. 520–523, IEEE, 2022. doi: 10.1109/icirca54612.2022.9985720.

[18] Malek Z.S., Trivedi B., Shah A.: User behavior pattern-signature based intrusion detection. In: *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pp. 549–552, IEEE, 2020. doi: 10.1109/worlds450073.2020.9210368.

[19] Pande S., Khamparia A.: Explainable deep neural network based analysis on intrusion detection systems, *Computer Science*, vol. 24(1), pp. 97–111, 2023. doi: 10.7494/csci.2023.24.1.4551.

[20] Pande S.D., Lanke G.R., Soni M., Kulkarni M.A., Maaliw R.R., Singh P.P.: Deep Learning-Based Intrusion Detection Model for Network Security. In: *International Conference on Intelligent Computing and Networking*, pp. 377–386, Springer, 2023. doi: 10.1007/978-981-99-3177-4_27.

[21] Ramaiah M., Padma A., Vishnukumar R., Rahamathulla M.Y., Chithanuru V.: A hybrid wrapper technique enabled Network Intrusion Detection System for Software defined networking based IoT networks. In: *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*, pp. 1–6, IEEE, 2024. doi: 10.1109/aiiot58432.2024.10574755.

[22] Samrin R., Vasumathi D.: Review on anomaly based network intrusion detection system. In: *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pp. 141–147, IEEE, 2017. doi: 10.1109/iceeccot.2017.8284655.

[23] Shah A., Clachar S., Minimair M., Cook D.: Building multiclass classification baselines for anomaly-based network intrusion detection systems. In: *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 759–760, IEEE, 2020. doi: 10.1109/dsaa49011.2020.00102.

[24] Siddiqi M.A., Pak W.: Tier-based optimization for synthesized network intrusion detection system, *IEEE Access*, vol. 10, pp. 108530–108544, 2022. doi: 10.1109/access.2022.3213937.

[25] Vinayakumar R., Alazab M., Soman K.P., Poornachandran P., Al-Nemrat A., Venkatraman S.: Deep learning approach for intelligent intrusion detection system, *IEEE Access*, vol. 7, pp. 41525–41550, 2019. doi: 10.1109/access.2019.2895334.

[26] Wei N., Yin L., Tan J., Ruan C., Yin C., Sun Z., Luo X.: An Autoencoder-Based Hybrid Detection Model for Intrusion Detection With Small-Sample Problem, *IEEE Transactions on Network and Service Management*, pp. 2402–2412, 2023. doi: 10.1109/tnsm.2023.3334028.

[27] Yang L., Li J., Yin L., Sun Z., Zhao Y., Li Z.: Real-time intrusion detection in wireless network: A deep learning-based intelligent mechanism, *IEEE Access*, vol. 8, pp. 170128–170139, 2020. doi: 10.1109/access.2020.3019973.

[28] Zhan X., Yuan H., Wang X.: Research on block chain network intrusion detection system. In: *2019 International Conference on Computer Network, Electronic and Automation (ICCNEA)*, pp. 191–196, IEEE, 2019. doi: 10.1109/iccnea.2019.00045.

[29] Zhou C., Huang S., Xiong N., Yang S.H., Li H., Qin Y., Li X.: Design and analysis of multimodel-based anomaly intrusion detection systems in industrial process automation, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45(10), pp. 1345–1360, 2015. doi: 10.1109/tsmc.2015.2415763.

Affiliations

Komal Jakotiya
School of Engineering, ADYPU, Pune India, komal.jakotiya@adypu.edu.in

Vishal Shirsath
School of Engineering, ADYPU, Pune India, vss.csit@gmail.com

SharanBasava Inamdar
School of Engineering, ADYPU, Pune India

Received: 17.07.2024
Revised: 19.10.2024
Accepted: 6.01.2025