The Future of Education in Bangladesh:Reshaping Education for a Global Job Market

Md.Aminul Islam Asif[1] and Dewan Md. Farid[2⋆]

Department of Computer Science & Engineering, Southeast University
251/A Tejgaon I/A, Dhaka 1208, Bangladesh
2023000000111@seu.edu.bd

Abstract. Bangladesh's education system remains at a pivotal juncture,demanding urgent reform to stay relevant in an increasingly competitive global talent market. From 6 June to 3, August 2024 .a wave of student-led activism - culminating in the resignation of Prime Minister Sheikh Hasina- revealed widespread frustration with obsolete institutions(universities and colleges) and entrenched disparities.This paper identifies a growing mismatch between the current education system and international job demands,Analyzing world best practices and offering policy reforms,it proposes a pathway to future success of students.

Keywords: Artificial Intelligence · Deep Learning · Intrusion Classification.

1 Introduction

Education in Bangladesh is at a turning point. The rise of digital technology, global economies, and demand for 21st-century skills is challenging old systems based on memorization and outdated content. As the global job market changes, Bangladesh must rethink its education system to prepare students for new opportunities.

While Bangladesh has made progress in access to education—especially in primary enrollment and gender equality—issues of quality, relevance, and practical skills remain. Many students still leave school without the abilities needed for today's digital and knowledge-based economy.

The Fourth Industrial Revolution has introduced Artificial Intelligence (AI), robotics, and big data, changing how people work. To stay competitive, Bangladesh must update its curriculum, support teacher training, improve vocational programs, and use education technology (EdTech).

The COVID-19 pandemic made these needs more urgent. It revealed gaps in digital access and infrastructure, especially in rural areas. At the same time, it showed the potential of blended and online learning for flexible and resilient education.

⋆ Supervisor

This paper explores how Bangladesh can reshape its education system to match global job market needs. It looks at the role of technology, policy change, and partnerships between schools and industry. The goal is to suggest a practical, inclusive, and future-ready education model.

Main Contributions

The main contributions of this paper are:

- Highlighting the gap between current education outcomes and global job market demands.
- Examining how technology and EdTech can modernize education.
- Reviewing policy examples and global models that Bangladesh can follow.
- Recommending clear steps to align education with future job skills.

Paper Organization

The structure of the paper is as follows: Section **??** presents a review of related work. Section 3 discusses the research methodology adopted. Section **??** provides a detailed analysis of Bangladesh's current education landscape and global market trends. Section **??** outlines strategic recommendations. Finally, Section 6 concludes the paper with a summary and future outlook.

2 Related Works

Anish Halimaa et al. [**?**] propose a hybrid model based on Support Vector Machine (SVM) and Naïve Bayes classifier. The main objective of this system is to increase the accuracy rate and lessen the false alarm rate. Yakub Kayode et al. [**?**] conducted a performance assessment of different ensemble-based ML classifiers, including AdaBoost (AB), Extreme Gradient Boosting (XGB), Gradient Boosted Machine (GBM), CatBoost, KNN, and QDA for Intrusion Detection System (IDS). Hasan Alkahtani et al. [**?**] proposed a hybrid CNN-LSTM model in order to design a system aimed at detecting intrusions in the Internet of Things (IoT) environment, demonstrating enhanced accuracy. Bo Cao et al. [**?**] used a CNN and GRU-based hybrid deep neural network for intrusion detection and achieved more satisfactory results. In response to the ongoing and changing malicious threats, Muhammad Ashfaq Khan et al. [**?**] proposed a deep learning-based hybrid intrusion detection system called HRCNNIDS. The proposed system utilizes convolutional neural networks to capture regional features and temporal features through recurrent neural networks. The experimental results show that it has a high detection rate for identifying malicious attacks. Most of these models are trained centrally which can be computationally burdensome and pose privacy concerns for data.

While these central models may provide superior accuracy, they can also lead to classifier overload due to the high volume of traffic generated by all IoT devices

on the network. Therefore, there is a shift towards distributed, decentralized and similar approaches for IDS. Recently, Federated Learning (FL) [**?**] has been adopted for IDS to address centralization and isolated data issues. Nguyen et al. [**?**] proposed an anomaly detection system for detecting compromised IoT devices using federated learning by aggregating anomaly-detection profiles for intrusion detection. Agrawal et al. [**?**] provides a comprehensive survey of FL-based solutions in IDS emphasizing on security, privacy, and reliability aspects. Liu et al. [**?**] proposed an Attention Mechanism-based Convolutional Neural NetworkLong Short-Term Memory (AMCNN-LSTM) model to accurately detect anomalies and also reduce the communication overhead by 50% compared to the centralized system. Thu Huong et al. [**?**] proposed a hybrid model of Variational Autoencoder (VAE) and Long-Short Term Memory (LSTM) in FL mode to cope with anomaly detection for time-series data in Industrial Control Systems. However, intrusion detection models trained using Federated Learning exhibit poor performance and higher false alarm rates when there is limited training data. Therefore, we explore the use of Generative Adversarial Networks (GANs) to build intrusion detection models.

Ferdowsi et al. [**?**] have proposed an adversarial network to identify intrusions in IoT devices. The proposed distributed GAN-based IDS has up to 20% higher accuracy, 25% higher precision, and 60% lower false positive rate compared to a standalone GAN-based IDS. In a study on Intrusion Detection Systems (IDS) focusing on Generative Adversarial Networks (GAN), Lee et al.[**?**] proposed a GAN model designed to resample minority classes with a high degree of accuracy. Xu et al. [**?**] proposed a new training strategy based on a Bidirectional GAN (Bi-GAN) more suited to detect intrusions. This model consists of a one-class binary classifier for the trained encoder and discriminator to use for detecting anomalous traffic from normal traffic. Currently, there appears to be a lack of Generative Adversarial Network (GAN) based intrusion detection models that are specifically designed for a federated scenario. For instance, Markovic et al. [**?**] introduced a Random Forest-based intrusion detection system using the Federated approach. The proposed framework was evaluated on KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 datasets. Similarly, Idrissi et al. [**?**] developed a FL-based framework, called Fed-ANIDS. Within their proposed framework three variants of AEs (autoencoder) were employed to improve detection accuracy while preserving privacy through the use of FL. The proposed framework was evaluated on USTC-TFC2016, CIC-IDS2017, and CSE-CIC-IDS2018 datasets.

3 Methodology

3.1 Federated Learning

Federated Learning (FL) is a cutting-edge privacy-preserving paradigm that facilitates the training of machine learning (ML) models on individual edge devices within a distributed network. The main advantage of FL lies in its ability to enable model training without the exchange of training data between clients or with the central server. By doing so, this approach offers a level of privacy and

security that surpasses traditional centralized machine learning methods. In traditional centralized machine learning, all the clients are connected to the central server and send their local data to it. The central server then combines all the data from clients and performs machine learning tasks using this collective data. Centralized training is computationally efficient as participating clients only send their local data to the server and the rest is done by the server. However, despite its efficiency, several issues must be addressed. Centralized learning requires significant computing resources and may encounter scalability issues when dealing with large volumes of data and complex models. Additionally, privacy concerns arise when handling personal and sensitive data, as the client's data is stored centrally, posing a high risk of potential breaches. On the other hand, in federated learning (FL), data remains decentralized, and model training takes place on edge devices. The clients' data remains on their respective devices, and the model training is carried out by the edge devices themselves. Initially, a general or pre-trained model is distributed to all clients. Each client then trains this model locally using their own data. After training, the local model updates are transmitted to the central server. Subsequently, the central server aggregates all the model updates and utilizes them to update the global model, which is then shared with client devices. This continuous process ensures that all clients are consistently updated with new and evolving knowledge. It continues for multiple rounds until the desired performance is achieved. The general federated learning architecture is shown in Fig.1. The federated learning formulation [**?**] is given below:

$$f(w) = \sum_{k=1}^{k} \frac{n_k}{n} F_k(w) \quad \textit{Where} \quad F_k(w) = \frac{1}{n_k} \sum_{i \,\epsilon\, p_k} f_i(w) \tag{1}$$

In equation 1, f_i represents a loss function of prediction for input x_i to an expected output y_i with weight vectors w. K is the number of participants in the current learning round and F_k is the local objective function of kth participant.For total number of samples n, n_k is the number of samples present locally in k_{th} participant. Similarly, P_k with $n_k = |P_k|$, is the partitioned assigned to k_{th} participant from whole dataset P.

Just like other ML & DL projects, frameworks, and libraries are necessary for training algorithms in federated learning. For example, Google has introduced TensorFlow Federated (TFF), a Python-based, open-source framework designed for training machine learning models on decentralized data. IBM has also developed its own FL framework called IBM Federated Learning. Another popular open-source library is PySyft, created by OpenMined. In this study, we employed Flower, a powerful and open-source Python library specifically designed for Federated Learning. Flower supports various deep learning frameworks, including TensorFlow, PyTorch, and Keras. It provides a flexible and extensible architecture for building federated learning systems.

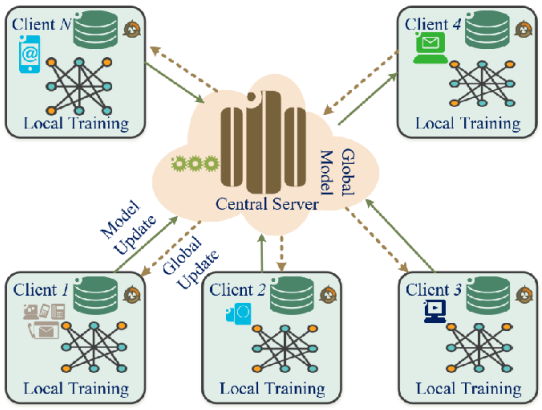


Fig. 1. A Generalised Federated Learning Architecture.

3.2 Averaging Algorithms

The FedAvg algorithm, short for Federated Averaging, is a widely used method for training models in a federated environment. It is based on a federated version of stochastic gradient descent (SGD), known as FedSGD. It operates by allowing each device to perform a local training iteration on its own data, followed by averaging the updated model parameters across all devices and distributing the average back to each device for the next iteration. This process is repeated until either convergence or a stopping criterion is reached. The goal is to balance the local model updates with the global average, resulting in a model that can generalize well across the entire federated population. The pseudocode of FedAvg is shown in Algorithm 1.

Algorithm 1 Federated Averaging Algorithm (FedAvg).The K clients are indexed by k ; B is the local minibatch size, E is the number of local epochs, η is the learning rate and ℓ is the local loss function.

Server executes:
initialize w_0 each round t = 1,2...
    $m \leftarrow$ max (C . K, 1)
    $S_t \leftarrow$ (random set of m clients)
each client $k \in S_t$ **in parallel**
$w_{t+1}^k \leftarrow$ ClientUpdate (k, w_t)
$w_{t+1} \leftarrow \sum_{k=1}^{k} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate (k, w): //Run on client k
$\mathcal{B} \leftarrow$ (split $\mathcal{P}$k into batches of size B) each local epoch i from 1 to E batch $b \in B$
$w \leftarrow w - \eta \nabla l(w; b)$
return w to Server

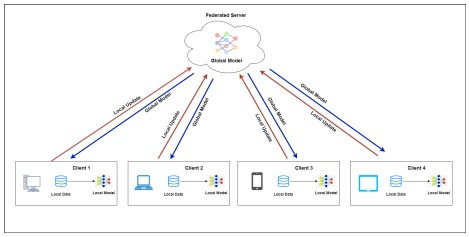
3.3 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) was first introduced by Ian J. Goodfellow [**?**] in 2014. GANs are machine learning (ML) models that utilize deep learning methods to accurately predict by contesting two neural networks against each other. GANs, are a type of neural network used for generative modeling and consist of two networks: (1) the Generator, G and (2) the Discriminator, D. The role of the generator is to create real-like fake samples, while the discriminator aims to distinguish between real and fake samples. These two networks are trained together in an adversarial process where the generator, G attempts to deceive the discriminator, while the discriminator strives to improve its ability to identify fake samples. As the training progresses, the generator gets better at creating realistic fake samples and the discriminator improves at identifying them. Ultimately, the generator model becomes capable of generating new samples that are indistinguishable from real ones, and the discriminator's accuracy decreases as it begins to misclassify fake data as real.

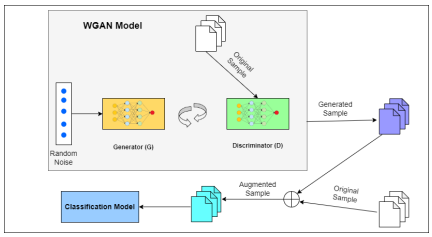
A WGAN [**?**], which stands for Wasserstein Generative Adversarial Network, represents a variant of traditional GANs. It introduces a novel loss function namely Wasserstein-1 distance (Earth Mover's distance) to address the challenges encountered when training standard GANs, such as mode collapse and vanishing gradients. Wasserstein distance provides a more informative gradient signal, leading to more stable training and fewer issues with mode collapse. Due to the improved training stability, WGANs can generate higher-quality synthetic data that better approximates the real data distribution.

3.4 Proposed Architecture

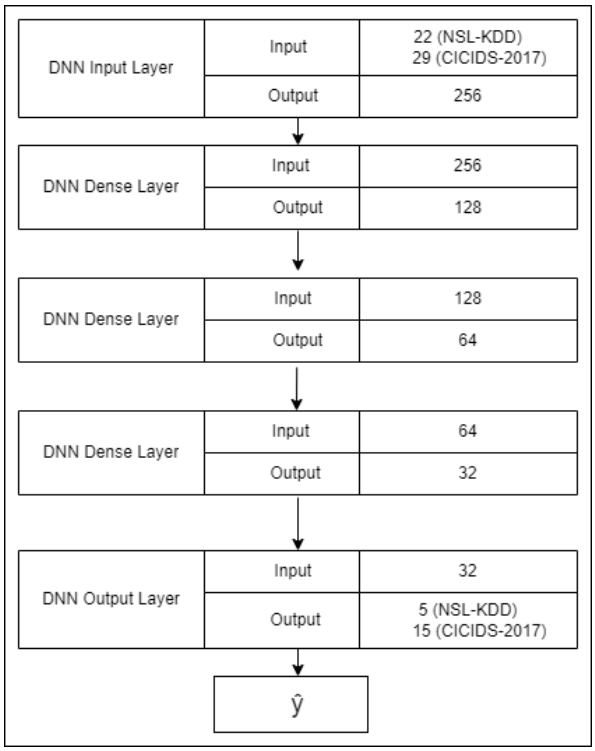
In this research, the objective is to develop a real-time, and effective IDS that is not only effective but also operates in real-time. To achieve this, we utilize DL techniques to develop a model that can learn and recognize normal network activity based on regular network usage. Building an effective IDS for IoT systems faces several challenges, such as resource constraints, imbalanced data, and privacy concerns. To address these challenges, we propose integrating two advanced technologies with complementary benefits: Federated Learning (FL) and Generative Adversarial Networks (GANs). It is well-known that the accuracy of intrusion detection models is directly proportional to the amount of training data, particularly for ML or DL models. However, each IoT device has limited data, leading to weak individual models in scenarios. Additionally, it is not advisable to centralize the collection of IoT device data for model training, especially when dealing with sensitive information. Furthermore, imbalanced data on IoT devices presents a significant bottleneck in training a successful FL model. Considering these problems, we propose a FL-based IDS that leverages the use of GANs and focuses on the minor class samples. In our approach, each IoT device is equipped with a generator network and a discriminator network, which together form a GAN network.



Four Clients Participating in FL System.



Client Model.



DNN Classification Model.

Fig. 2. Architecture of Proposed Framework.

The generator on each IoT device analyzes local data and generates similar traffic patterns, providing sufficient data samples for the local discriminator of that device. Once trained, the generator produces real-like synthetic samples, along with original samples, which are subsequently fed into the DNN model for classification. After that, the gradients are transmitted to the central server, which aggregates the parameters and updates the global model accordingly. The updated global model is then sent back to each IoT device. The complete ar-

chitecture of our proposed framework is shown in Fig.2. The distributed nature of the proposed IDS ensures reduced time and computational requirements, enabling the rapid classification of attacks.

4 IDS Datasets

NSL-KDD is a well-known dataset. This dataset was released by the University of New Brunswick as a cleaned-up and revised version of the KDD'99 dataset. The dataset has 43 features per record out of which 41 features describe the traffic input, while the remaining two features are labels indicating whether the input is normal or an attack, and the severity of the traffic input. The training set has 125,973 instances and the testing set has 22,544 instances. There are four classes of attacks in the data set: Denial of Service (DoS), Probe, User to Root (U2R), and Remote to Local (R2L). A list of all the attack types is presented in Table 1.

Table 1. Distribution of NSL-KDD dataset before and after sample generation.

Attack Class	No. of Records	After Sample Generation
Normal	77,054	77,054
DoS	53,092	53,092
R2L	10,931	10,931
Probes	7249	12,369
U2R	191	10,303

CIC-IDS2017 datset was created by the Canadian Institute for Cybersecurity in 2017. The CIC-IDS2017 dataset contains five days of network traffic containing normal and attack information. The dataset has 2830743 instances and 79 features containing 15 class labels (1 Benign + 14 attack labels). The diverse nature of this dataset makes it the ideal choice for network traffic analysis. A list of all the attack types is presented in Table 2.

Oversampling using WGAN Recently, there has been an increasing interest in GAN-based oversampling techniques in deep learning. The Wasserstein GAN is a type of GAN that is known for its superior performance. It avoids the issues of vanishing or exploding gradients during the training process. This architecture generates data samples with low noise and requires minimal adjustments to the hyperparameters. For these reasons, we have chosen to use the WGAN architecture in this study.

Table 2. Distribution of CIC-IDS2017 dataset before and after sample generation.

Attack Class	No. of Records	After Sample Generation
Benign	2,273,097	2,273,097
DoS Hulk	230124	230124
PortScan	158,930	158,930
DDoS	128,027	128,027
DoS GoldenEye	10,293	10,293
DoS Slowloris	5796	15,796
DoS Slowhttptest	5499	15,499
FTP Patator	7938	17,938
SSH Patator	5897	15,897
Bot	1966	11,966
Web Attack-Brute Force	1507	11,507
Web Attack-XSS	652	10,652
Web Attack-Sql Injection	21	10,021
Infiltration	36	10,036
Heartbleed	11	10,011

Feature selection ANOVA (Analysis of Variance) is a commonly used filter-based feature selection method. ANOVA F-test calculates the F-statistic between each feature and the target variable. By calculating the F-statistic for each feature, we can determine which ones are most informative and should be included in our analysis. Features with high F-statistics are likely to have a strong relationship with the target variable, and including them in our analysis can improve our model's accuracy and predictive power. Xiaodong et al. [**?**] have identified the most significant features from two popular datasets, NSL-KDD and CICIDS-2017. Specifically, they selected 22 features from the NSL-KDD dataset and 39 features from the CICIDS-2017 dataset.

5 Experimental Results

In this section, we present the results of experiments conducted on the NSL-KDD and CIC-IDS2017 datasets using the Flower Federated Learning environment. The purpose of these experiments was to evaluate the performance of the proposed model in both binary and multi-class classification scenarios. To model the Intrusion Detection System (IDS), we employed FL and GAN techniques. Specifically, each client device was equipped with two Deep Neural Networks

(DNNs): a WGAN and a Classifier. Each client underwent individual training using both its own data and synthetic data generated by its local GAN. Furthermore, all clients were trained collectively using the federated learning approach, employing the same DNN model. Accuracy, Precision, Recall, and F1-Score were utilized as the metrics for comparing the performance of the FL model against the individual models.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

$$f1_{score} = 2 * \frac{Recall * Precison}{Recall + Precison} \tag{5}$$

Where, True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

5.1 Binary Classification

In Binary classification, we modeled the DNN as the binary classifier to detect normal and anomaly samples. Table. 3 and Table. 4 present the results from binary classification on NSL-KDD and CIC-IDS2017 datasets.The tables display the results of Four participating clients and their aggregated results using the Federated approach. To balance the distribution of data in these datasets and prevent bias, the NSL-KDD and CICIDS-2017 datasets were enhanced by including synthetic data for minority samples. The datasets were then divided into 3 sets: Train, Validation, and Test with 70%, 15%, and 15% respectively. From the training set all the normal samples and attack samples were distributed randomly to all client devices. When using the Federated Learning approach, it can be challenging to track scores from a large number of clients involved in the process. To make it more manageable, we decided to experiment with only four clients. We assessed their individual performance on their own data and then compared the results with the Federated Learning aggregated model. To achieve this, Federated Learning models utilize a technique known as "Federated Averaging" which combines model updates from different clients to produce a new shared model. In order to validate the binary classification on the NSL-KDD and CICIDS-2017 datasets, we conducted a similar experiment. As shown in table 3, the accuracy and precision of each client are above 98% and the FL model using the NSL-KDD dataset achieves 98.51%. In table 4, the accuracy and precision are recorded as 99% for each client and 98.9% for the FL model with the CICIDS-2017 dataset. The results showed that there is no significant difference and the Federated Learning (FL) system exhibits remarkable binary classification performance.

Table 3. FL Performance in Binary Classification with NSL-KDD dataset.

Metrics	FL-Client1	FL-Client2	FL-Client3	FL-Client4	FL-Model
Accuracy	99.01	98.78	98.36	98.43	98.51
Precision	98.27	98.27	98.37	98.43	98.58
Recall	98.25	98.28	98.35	98.42	98.58
F1-Score	98.26	98.27	98.36	98.43	98.58

Table 4. FL Performance in Binary Classification with CICIDS-2017 dataset.

Metrics	FL-Client1	FL-Client2	FL-Client3	FL-Client4	FL-Model
Accuracy	99.05	99.12	99.13	99.17	98.99
Precision	99.09	99.11	99.14	99.17	99.12
Recall	98.93	99.05	99.01	99.10	99.03
F1-Score	99.01	99.08	99.07	99.14	99.07

5.2 Multi-class Classification

As shown in Table. 5, each client achieved an accuracy exceeding 97%, while the FL model attained an accuracy of 96.22% using the NSL-KDD dataset. Table. 6 presents the results of multi-class classification for the CICIDS-2017 dataset, with each client achieving an accuracy and precision surpassing 98.50%, and the FL model achieving an accuracy of 98%. The FL system's performance remains impressive even in the context of multi-class classification, as evidenced by the results obtained from the NSL-KDD and CICIDS-2017 datasets. Furthermore, it's worth noting that the FL model exhibits commendable performance, even when dealing with complex datasets such as CICIDS-2017. This serves as a testament to the FL system's effectiveness in intrusion detection. Finally, we conducted an extensive performance analysis of our proposed model (FLGAN-IDS) in comparison to existing FL-based IDS. The detailed results of this comparison can be found in Table.7. To ensure a fair comparison, we exclusively selected IDS models that were evaluated using the same NSL-KDD and CIC-IDS2017 datasets utilized in our study. From the results, it is evident that our proposed model outperforms all existing methods using these datasets.

Table 5. FL Performance in Multi-class Classification with NSL-KDD dataset.

Metrics	FL-Client1	FL-Client2	FL-Client3	FL-Client4	FL-Model
Accuracy	97.33	97.03	97.17	97.07	96.22
Precision	95.09	94.36	94.97	94.46	94.29
Recall	93.60	93.25	94.16	93.60	92.24
F1-Score	94.19	93.73	94.40	93.96	93.13

Table 6. FL Performance in Multi-class Classification with CICIDS-2017 dataset.

Metrics	FL-Client1	FL-Client2	FL-Client3	FL-Client4	FL-Model
Accuracy	98.81	98.84	98.83	98.78	98.08
Precision	98.77	98.48	98.72	98.44	98.13
Recall	96.11	96.11	95.70	96.17	95.60
F1-Score	97.19	97.16	97.07	97.20	96.70

6 Conclusion and Future works

In recent years, the protection of user data privacy has emerged as a prominent concern for organizations. With the proliferation of IoT devices, the storage of large volumes of data for ML analysis poses a risk of privacy breaches. Fortunately, Federated Learning (FL) provides a solution by enabling privacy-preserving ML/DL tasks. In this research, we conducted several experiments to assess the effectiveness of FL in detecting attacks in IoT environments while preserving data privacy. Our findings indicate that FL is an effective method for detecting attacks in IoT environments, even when the devices are distributed and diverse. The data analysis was carried out at the clients' end, and no data sharing occurred between clients. The server served as an aggregator, using FedAvg algorithms for parameter aggregation. In each round, clients trained the DNN model using their local data (including their own data and synthetic data generated by their local GAN) and participated in the FL system. The results demonstrated that an FL model can deliver excellent performance in terms of accuracy, precision, recall, and F1-score for both binary and multi-class classification. It is important to note that this experiment involved only four clients, which is not an ideal scenario for an IoT environment where a large number of devices are typically present. In future, we intend to incorporate more clients and introduce new data augmentation techniques to further improve the outcomes.

Table 7. Performance comparison of the proposed models with other existing models.

Model	Dataset	Architecture	Accuracy (%)
FL-IDS [?]	NSL-KDD	RF based ML	93.51
FTML-IDS [?]	NSL-KDD	ML	98.11
Fed-IDS [?]	NSL-KDD	ML	83.09
FLGAN-IDS (Proposed)	NSL-KDD	DNN	98.51
Fed-ANIDS [?]	CIC-IDS2017	Autoencoders	93.36
FL-IDS [?]	CIC-IDS2017	RF based ML	93.51
Blockchain-based FLS [?]	CIC-IDS2017	Vanilla-Autoencoder	97.00
FLGAN-IDS (Proposed)	CIC-IDS2017	DNN	98.99

References