Journal of Computer Science

Research Article

A Distillation GAN Model for Network Intrusion Detection

Chandrakala C B[1], Sai Sreevall[1] and Raghudathesh G P[2]

[1]*Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, 576104, India*
[2]*Manipal School of Information Sciences, Manipal Academy of Higher Education, Manipal, 576104, India*

Article history
Received: 10-06-2025
Revised: 20-10-2025
Accepted: 27-12-2025

Corresponding Author:
Raghudathesh G P
Manipal School of Information Sciences, Manipal Academy of Higher Education, Manipal, 576104, India
Email: raghudathesh.gp@manipal.edu

Abstract: This study presents a distillation-based Generative Adversarial Network (GAN) model designed for intrusion detection in network systems. A Teacher-Student architecture is proposed, in which a single teacher generator is utilized to train both the teacher discriminator and the student discriminator. The model was trained to identify anomalies present in network traffic intrusions by tuning the min-max loss function in GANs. The imbalance caused by the underrepresented attack classes in the training dataset is corrected using the generator. Extensive experiments on the CIC-IDS-2017 and CSE-CIC-IDS2018 datasets within the framework of the proposed method showed the effectiveness of the approach. The student network on the CICIDS-2017 dataset showed an overall accuracy of 86.8% and an F1-score of 88.2%, comparable to that of the teacher, with an accuracy of 89.1% and an F1-score of 88.7%, with a 40% decrease in the number of parameters. For the CSE-CIC-IDS2018 dataset, the student model showed a competitive performance, with accuracy ranging from 73.10-88.20% and 70.24-86.68% for F1-score, respectively, and aligning with and even outperforming the metrics of a teacher (accuracies of 76.55-89.26% and F1-scores of 72.91-87.17%).

Keywords: Network Intrusion Detection, Traffic Attacks, Deep Learning, Generative Adversarial Networks, Distillation, SDG9

Introduction

Network Intrusion Detection Systems (IDS) are essential instruments providing secure digital infrastructures of various types, and they are used to protect enterprise networks, Internet of Things (IoT) ecosystems, smart cities, industrial control systems, and 5G-based communication networks. In such systems, the attacks that can be detected, analyzed, and prevented, which include intrusions, viruses, DDoS, etc., are exposed, thus our data and operations will be safe. With the increased use of IoT gadgets, people working remotely, and the use of cloud computing, the networks are demanding a higher level of protection than ever. The arrival of 5G network with low latency and high data rates means the security solutions need to be deployed at the edge of the network infrastructure. The analysis on the edge devices implies that the lower the delays, the lighter the network remains, and threats can be more readily identified and managed. This will save bandwidth for the important task, as information will be processed on the phone and therefore, prevent huge quantities from being sent to the server for processing. This makes IDS an important element of the current cybersecurity strategies.

One of the key issues that persists in today's sophisticated cyber threats is that conventional IDS has failed to address effectively. The traditional IDS uses mainly signature-based detection techniques, where the traffic that enters the network is matched with a predefined database of known attack signatures. Although the strategy is effective in detecting well-documented threats that can be detected precisely, it has weaknesses in the case of new and Advanced Persistent Threats (APTs) that do not have the appropriate signatures. Due to the dynamic characteristics of cyberattacks, as well as the decentralized structure of IoT and the edge devices, the attack space is growing, and the centralized security models are becoming inadequate. Nevertheless, it is always difficult to acquire the appropriate information to train an IDS. The bulk of publicly available data still contains many anomalies compared to typical traffic, meaning it does not reveal much variety in the forms of attacks, such as completely new viruses or novel attack methods. When models are not accurate, they fail to match real-world

scenarios, making it difficult to make practical decisions in various systems.

Researchers now often utilize Machine Learning (ML) and Deep Learning (DL) to address the issues listed above, as they make processes more flexible and automate tasks that were previously manual. Vinayakumar et al. (2019) explored a DNN model that helps detect unknown and unpredictable cyberattacks by using the NSL-KDD dataset. Although Mirsky et al. (2018) introduced Kitsune, this approach enhanced the ability to find known threats; it was not as effective against unseen ones. It required a lot of computing power, which prevents it from being used on small devices. Additionally, the research by Buczak and Guven (2016) examined the combination of different ML models for IDS, achieving strong results for balanced datasets but struggling to process massive data quickly. While they can spot anomalies in complicated data and are exempt from manual fixing of features, these ML/DL-based tools are commonly slowed down by the need for substantial computing resources in pre-trained models. These kinds of models, created for powerful cloud networks, do not meet the needs at the edge, where small amounts of power and fast responsiveness are needed.

The performance of ML/DL based IDS is strongly influenced by having large and accurate datasets. Nevertheless, researchers may encounter obstacles when collecting the data needed to examine their suggested solutions. Concerns with privacy can stop people from exchanging certain datasets. Also, since most publicly accessible datasets have less variety in attacks and types of traffic, it is hard to see the threats found in real networks. Because most of the traffic in these datasets is benign and only a small part is from attacks, there are serious challenges when building robust and accurate ML/DL models on this kind of data.

CICIDS2017 fixes some shortcomings found in other networks by offering a diverse and complete set of traffic data that includes today's common attack types:

- Brute-Force Attack: A Brute-Force Attack is when the attackers use a guessing method to obtain login credentials. It's very helpful for attackers where security is weak, since it allows them to gain illicit access
- Heartbleed Attack: A serious issue in the OpenSSL program used to handle the TLS security protocol. Cyber attackers exploit the bug by sending an incorrect heartbeat to a weakness in the server, using only a little information but making the length appear much larger, which allows the server to share additional data
- Botnet: A type of cyberattack where hackers take control of a network of connected computers. Once in control of these systems, they use these devices and their connections to steal information or send out spam
- DoS Attack: This attack aims to temporarily make the targeted network resource unavailable. The target machine is flooded with requests in an attempt to disrupt the network services and prevent the users from accessing the resource
- DDoS Attack: This attack is similar to the previous attack; the key difference is that a DDoS attack utilizes multiple devices on a network that usually form a botnet to flood the target resource with network traffic
- Web Attack: This attack targets vulnerabilities in a website to gain unauthorized access or alter the website's information. One of such attacks is SQL Injection, where the attacker transmits strings of SQL commands to the database to extract information. Another form of web attack is the Cross-site Scripting (XSS), where the attacker introduces some malicious script into a site of trust
- Infiltration Attack: With this attack, an attacker uses the vulnerability of the software and opens a backdoor through which the attacker can further execute attacks, which include IP sweeps, full port scans, and service enumeration, including NMAP

Although there are different types of network attacks that exist, the present study aims to identify the intrusions listed above, based on the CICIDS2017 dataset, to test the proposed IDS. Due to the enormous growth in the area of IoT and edge devices, the area of attack on networks has greatly increased. These devices are inherently susceptible and deployed in decentralized situations, posing a difficulty to the effectiveness of central security. What is more, with the introduction of 5G networks, which feature a lower latency and a higher data throughput, the necessity to have a security system at the edge becomes even more prominent. The proximity of data to the point of origin greatly reduces bottlenecks and increases threat detection, as well as decreases network congestion by reducing the volume of data that needs to be transmitted by the system to centralized points. This method maintains bandwidth for the most important applications. Since modern cyberattacks spread so rapidly, edge-based detection is essential to the mitigation of the threat of widespread breaches.

The introduction of 5G networks results in increased data volume and velocity, therefore, requires IDS to utilize AI and ML in real-time threat evaluation, reducing the dependency on cloud-based processing. Recent developments in AI/ML-based IDS have been aimed at building models to identify known patterns of attacks and signatures. These models are, however, normally unable

to keep up with the dynamic and ever-changing complex cyber threats. Also, the majority of the pre-trained deep learning models are resource-intensive and computationally heavy, and this will not be practical in resource-limited edge computing.

This paper proposes a light ML model to be used in edge deployment. In our approach, a new knowledge distillation application is introduced to the Generative Adversarial Networks (GANs), which is concerned with the creation of a multi-class discriminator neural network to detect intrusions. The proposed method uses a teacher-student framework to distill a compact student network from a large teacher GAN, which is capable of operating effectively on edge machines. The resulting student discriminator network not only ensures high detection accuracy but also demonstrates zero-shot detection ability, allowing it to identify previously unknown attacks effectively:

- Introducing a new knowledge distillation method to GANs through a teacher-student framework, which allows for the development of a lightweight model that can be deployed to edge networks
- Developed a GAN architecture that has a multi-class discriminator that can well identify various forms of intrusion attacks. The model uses a min-max loss function, which has the ability to identify simultaneous losses across multiple classes
- The generator of the teacher GAN is used to train the student discriminator, and the challenges with the training dataset imbalances are also tackled by balancing out the underrepresented classes
- The results of the experiment show that the student network is as good a detector as the teacher network and cuts the model size by a factor of 40%
- The student network demonstrates a robust performance for the zero-shot detection cases, having the ability to detect previously unknown attacks

Related Work

Network IDS plays a central role in the protection of digital infrastructures against the increasing wave of cyberspace attacks, particularly as networks are developed with the spread of IoT networks, 5G networks, and edge computing systems. The paper concentrates on the use of ML and DL algorithms in the form of GAN to create small-sized IDS models to be deployed on the edges. The rationale of such a literature review is to fill the gaps of the conventional IDS that include their inability to identify new attacks, and the computational cost of implementing resource-demanding ML/DL models on edge devices. The scope covers both the literature related to ML/DL as an IDS, and their suitability and performance, as well as compatibility with edges, but does not cover general cyberspace challenges such as malware, or general network security protocols unless it is directly connected to intrusion detection.

Conventional IDSs are mainly signature-based techniques that scan network traffic against an array of known attack signatures. In Bace and Mell (2001), the authors emphasized the precision of these systems for documented threats but specified their inability to detect novel attacks, a limitation worsened by the rapid evolution of cyber threats. Roesch (1999) introduced SNORT, an open-source IDS that identifies known threats but struggles with zero-day exploits due to its reliance on predefined signatures. In Paxson (1999), the author developed Bro (now Zeek), enhancing protocol analysis, but it faces scalability issues in high-throughput environments. Scarfone and Mell (2007) discussed the challenges of maintaining signature databases, indicating the need for frequent updates to counter new threats. The author in Axelsson (2000) analyzed the base-rate fallacy in IDS, showing how high benign traffic volumes lead to false positives in signature-based systems. The research of Ko et al. (1994) introduced an early IDS framework, which places foundational concepts of signature matching but reveals susceptibility to evasion techniques. In Debar et al. (1992), the authors proposed a rule-based IDS model, highlighting its effectiveness for known patterns but its rigidity against evolving threats. Ptacek and Newsham (1998) demonstrated how signature-based IDS could be bypassed using packet fragmentation, exposing inherent vulnerabilities. In Denning (1987), the concept of intrusion detection was pioneered with a model based on audit records, establishing a baseline for signature-based systems but lacking adaptability to new attack vectors. The combination of these works suggests the strengths and ongoing limitations of conventional IDS, which drove the transition to more dynamic approaches.

ML-based IDS addresses traditional shortcomings by leveraging data-driven adaptability. Sommer and Paxson (2010) applied decision trees to the KDD Cup 1999 dataset, achieving moderate success but highlighting challenges with imbalanced data and feature selection. In Buczak and Guven (2016), ensemble ML methods were reviewed, showing robust performance on balanced datasets like NSL-KDD, although real-time scalability remained elusive. Tsai et al. (2009) explored Support Vector Machine (SVM), indicating its effectiveness with high-dimensional data, but cited computational complexity as a barrier. In García-Teodoro et al. (2009), anomaly-based ML IDS emphasizes trade-offs between detection accuracy and false alarm rates. In research, Lee and Stolfo (2000)

developed a data mining framework for IDS, improving the detection of rare attacks but requiring extensive pre-processing. Mukkamala et al. (2005) compared neural networks and SVMs, finding neural networks more adaptable but computationally intensive. In Sabhnani and Serpen (2003), K-nearest neighbors (KNN) on KDD data demonstrate simplicity but poor performance on noisy datasets. These studies highlight ML's potential to enhance IDS while revealing challenges in computational efficiency, data quality, and real-time applicability.

DL advances IDS by automating feature extraction and detecting complex patterns. Vinayakumar et al. (2019) proposed a Deep Neural Network (DNN) for NSL-KDD, which improves known attack detection but falls short against unseen threats due to overfitting. In Mirsky et al. (2018), the authors introduced Kitsune, an autoencoder-based IDS that excels in anomaly detection but requires significant resources. Zhang et al. (2019) applied CNNs to CIC-IDS-2017, achieving 85% accuracy but facing high computational demands. The work in Potluri and Diedrich (2016) evaluated CNNs on the KDD dataset, noting the need for large training data to mitigate over-fitting. Tang et al. (2016) utilized Deep Belief Networks (DBNs) on NSL-KDD, which enhances detection for specific attacks but struggles with resource constraints. In Yin et al. (2017), a Recurrent Neural Network (RNN) model was proposed, leveraging temporal dependencies for sequential attack detection, but the scalability was limited. Javaid et al. (2016) developed a self-taught learning approach using sparse autoencoders, which improves anomaly detection but requires extensive tuning. Kim et al. (2017) applied Long Short-Term Memory (LSTM) networks to KDD, excelling in temporal analysis but facing high latency. Ferrag et al. (2020) surveyed DL-based IDS, noting superior performance on modern datasets like CSE-CIC-IDS2018, but highlighting the resource demands. These works demonstrate the power of DL in IDS while exposing deployment challenges critical to edge-focused solutions.

GANs help IDS by producing artificial data and designing complex types of attacks. Goodfellow et al. (2014) first proposed GANs and showed that they could be helpful for anomaly detection. Zenati et al. (2018) employed GANs to identify abnormalities, but encountered mode collapse due to the system instability. Hu and Tan (2022) addressed the data imbalance in CIC-IDS-2017 by applying GANs and successfully enhanced the detection mechanism, although training was quite time-consuming. Li et al. (2019) introduced a GAN-based IDS that can handle multiple classes, but it required a significant amount of resources to operate. Ferdowsi and Saad (2019) generated artificial traffic for IoT in GANs, which was useful for detecting attacks but lacked various attack activities. Ring et al. (2019) explains that GANs in cybersecurity are beneficial, providing examples of data enhancement and resolving issues related to training consistency. Wasserstein GANs were employed by Choi et al. (2019) on NSL-KDD, which successfully stabilized the methods but required careful adjustment of various parameters. Yang et al. (2019b) used GANs to locate network anomalies; however, dealing with a large amount of data became challenging. They demonstrate that GANs serve as useful tools for IDS, aiming to detect multiple types of attacks. Aldhaheri and Alhuzali (2023) proposes a framework, SGAN-IDS, using self-attention mechanisms and GAN to produce adversarial network traffic that bypasses machine learning based IDS. To generate realistic malicious flows, the system utilizes a generator, and to streamline the generated flows, it employs a discriminator that is informed by the feedback of five black-box IDS models. The results obtained in the experiments with the CICIDS2017 and NSL-KDD datasets indicate that SGAN-IDS decreased the detection rate by an average of 15.93, which means that the method has a high evasion capability. Self-attention enhanced the authenticity and heterogeneity of artificial attack flows. Overall, SGAN-IDS highlights the weaknesses of ML-based IDS and the potential for developing more effective defense mechanisms. Zhang et al. (2020) proposed a small-sized generative model for efficient anomaly detection. It utilizes two GANs in a teacher-student system, whereby knowledge is gradually condensed from a pre-trained teacher to a lightweight student. The method is built upon a two-step learning process, i.e., knowledge transfer and fine-tuning are combined to improve student performance and significantly lower computational cost. The results of CIFAR-10, MNIST, and FMNIST show that the oat model compression ratios of 700:1 are better. This solution includes a gradual notion of GAN-based distillation, which is adaptable to the intrusion detection system in resource-constrained settings.

Knowledge Distillation (KD) compresses large models for efficiency in resource-constrained environments. Hinton et al. (2015) pioneered distillation, transferring knowledge from teacher to student models. In Gou et al. (2021), distillation techniques for DL were reviewed, highlighting edge deployment potential. The work in Mishra and Marr (2017) distilled CNNs, reducing parameters by 30% with minimal loss in accuracy, making it applicable to IDS. Yang et al. (2019a) applied distillation to IDS, reducing model size by 35% on NSL-KDD, albeit at the expense of some accuracy. In Sanh et al. (2019), DistilBERT was introduced, reducing size and

inference time, relevant for IDS efficiency. The work of Polino et al. (2018) used distillation for adversarial training, enhancing robustness while simplifying models. The research by Chen et al. (2020) combined GANs with distillation for anomaly detection, achieving efficiency but lacking multi-class support in Bucilă et al. (2006) proposed a model compression via distillation, laying early groundwork for IDS applications. Phuong and Lampert (2019) distilled DNNs for faster inference, offering a blueprint for real-time IDS. In Wang et al. (2018), distillation to LSTM models reduces complexity while retaining temporal analysis capabilities. It demonstrates that distillation enables the creation of IDS models that can be easily deployed on edge networks. Wisanwanichthan and Thammawichai (2025) showed that knowledge distillation from a large teacher model to a compact student model reduces the parameters by 93% while maintaining comparable accuracy to that of the teacher across five IDS datasets. The refined student models achieved better inference times (7-11%) and improved detection (up to a 6.93% point F1 increase). The work highlights KD as a useful methodology for constructing lightweight, real-time IDS to be implemented in IoT and UAV scenarios with limited computational capabilities.

Edge-deployed IDS can detect threats instantly in widespread networks. Shone et al. (2018) developed a deep autoencoder for NSL-KDD, which was useful yet not tailored to fit the constraints of edge equipment. The team in Alrawashdeh and Purdy (2016) employed Restricted Boltzmann Machines and got high accuracy, but the method suffers from slow speed. ML and rules were used in Mao et al. (2009) to develop a streamlined IoT IDS that performs real-time monitoring with little extra effort. Work Ilse et al. (2018) presented a federated edge IDS that enables team training while ensuring data privacy. In Riku and Timo (2022), the authors used Tiny-ML to confirm that microcontroller-based intrusion detection was possible. Studies show why it is essential to have an IDS that can run efficiently at the network's edge.

Recent research findings in Table 1 demonstrate that diverse utilization of deep learning architectures, such as CNNs, LSTMs, GANs, and Transformers, is employed for anomaly detection across various domains. These studies together highlight the effectiveness of each one of these architectures, which depend on the nature of the dataset used and the characteristics of the anomalies being identified. These tables highlight how CNNs perform well for vision-based tasks, while LSTMs do well in forecasting, GANs in time-series data, and Transformers in handling datasets with differing class sizes.

Table 1: Literature summary of anomaly detection and attack generation studies using different modelling techniques

Author	Dataset	Proposed Model Type	Proposed Model Type	Time-Series	2-Class	Multi-Class
Bharathi and Makhija (2024)	NB15	Ensemble of Random Forest and Xgbm	Recursive feature elimination	✗	-	-
Mushtaq et al. (2022)	CIC-IDS2017 and CSE-CIC-IDS2018	AE-LSTM	Anomaly detection	✓	✓	✗
Yadav and Kalpana (2022)	NSL-KDD, CIC-IDS2017, and CSE-CIC-IDS2018	RNDAE and Yolov3	Vision-based	-	✓	✗
Althubiti et al. (2018)	CSIC 2010 HTTP	LSTM RNN	-	-	✓	✗
Toupas et al. (2019)	CIC-IDS2017	Fully Connected DNN	Limited experiments	✗	✗	✓
Altunay and Albayrak (2023)	UNSW-NB15 and X-IIoTID	CNN + LSTM	Not suitable at the edge	✗	✗	✓
M. et al. (2024)	CIC-IoT-2023	Transformer Model	-	✗	✗	✓
Potluri et al. (2018)	NSL-KDD and UNSW-NB15	CNN	Novel attacks	✗	✓	✗
Alom et al. (2015)	NSL-KDD dataset	DBN	Particle swarm optimization and deep belief networks	-	-	-
Zhao et al. (2024)	CIC-IDS2017	GANs to deal with Class Imbalance	Multiple GANs for anomaly data generation	-	-	✓[a]
Constantin et al. (2024)	UPB Flow dataset	GANs to deal with Class Imbalance	Extensive experiments. Multiple GANs for anomaly data generation	-	-	✓[a]
Ali et al. (2024)	CIC-IDS2017, IoT-23, NSL-KDD	GANs to deal with Class Imbalance	Extensive experiments. Teacher-Student model	-	-	✓[a]
Lin et al. (2022)	NSL-KDD	GANs for attack generation	Limited experiments	-	✗	✓

In this review, IDS methods are traced from signatures to the use of machine learning, deep learning, and GANs. Working against threats that have been seen before, traditional IDS is helped by ML and DL models, which are more flexible. Still, they cannot address overfitting, data disproportion, and require substantial computing resources, which restricts real-time applications. GANs enhance data handling, detection, and distillation, enabling edge deployment, but have the disadvantage of instability and resource intensity. KD offers model compression for efficiency, yet few integrate it with GANs or DL for edge environments, such as IoT and UAVs. The present condition demonstrates significant advancements, but there are still difficulties implementing an effective, multi-class IDS on the edge devices. There are issues that still exist in optimizing GANs to cater to multi-class detection and addressing imbalanced datasets in real-time scenarios. Future studies would be able to optimize the GAN models, parallel training, and experiment on different datasets. These gaps are addressed by the proposed distillation GAN model, which has 86.8% accuracy on CIC-IDS-2017 with a 40% reduction in the number of parameters, connecting with the body of existing knowledge as it provides an effective, edge-ready IDS solution.

The Distillation GAN Model

Recently, Generative Adversarial Networks (GANs) have gained significant traction in the deep learning community due to their high-quality outputs generated through the intriguing framework of two-player minimax games, where a generator and a discriminator compete against each other. The generator learns the real data distribution by reaching an equilibrium point of the minimax game. It is interesting to note that at the same equilibrium point, the trained discriminator model also will have the highest fidelity and is capable of producing very accurate classification results in a binary classification setting. Building upon this intuition, we develop a multi-class Teacher Discriminator by pooling multiple binary-class discriminator models, where each binary-class model will be trained to specialize in identifying a designated anomaly or attack class. For each input, we compute the loss of the overall model as the minimal loss across all or K out of N binary class models and train the teacher discriminator to minimize the total loss over the entire training set. We apply L1-regularization to avoid any possible mode collapse. Subsequently, we built a light-weight student network that emulates the teacher architecture but with a smaller number of parameters in each layer. The student network is trained against the real data and the teacher generator.

Let $G(., \theta)$ represent the generator network with parameter vector θ. Consider M discriminators, $M \leq K$, denoted by $\{D_m(;\emptyset_m)\}_{m=1}^{M}$, where K is the number of classes of intrusion attacks. The discriminators are trained with the following loss functions:

$$\mathcal{L}_T^{D_r}(x) = -\sum_{i=1}^{N_d}\sum_{m=1}^{M}\chi_{i,m}\log\big(D_m(x_i;\phi_m)\big)$$

$$\sum_{m=1}^{M}\chi_{i,m} \leq K, \quad (1)$$

Where N_d are the samples from the real data, since the fake example generated by the generator has to be detected by all the discriminators correctly, we add the following additional loss term:

$$\mathcal{L}_T^{D_g}(x) = -\sum_{j=1}^{N_g}\sum_{m=1}^{M}\Big(1-\log\Big(D_m\big(G(Z_j);\phi_m\big)\Big)\Big) \quad (2)$$

Where Z_j is the fake generated against the sample x.

The generator is trained to counter the loss in Eq. 2 over the chosen number of discriminators. Or equivalently, the generator loss is given by:

$$\mathcal{L}_T^{G}(x) = -\sum_{i=1}^{N_g}\sum_{m=1}^{M}\psi_{i,m}\log\big(1-D_m(G(z_i);\phi_m)\big)$$

$$\sum_{m=1}^{M}\phi_{i,m} \leq K, \quad (3)$$

In the above expressions, $\phi(i,m)$ and $\psi(i,m)$ are indicator variables which equal *1* if i=m, and equal 0, otherwise.

The total loss associated with the teacher discriminator network is given by:

$$\mathcal{L}_T = \alpha\mathcal{L}_T^{D_r}(x) + \beta\mathcal{L}_T^{D_g}(x) \quad (4)$$

Where α and β are hyperparameters that assign relative weights to loss over real and generated data, respectively. The teacher discriminator network acts as an adversary against the generator network in the underlying minimax game.

Since the student network emulates the teacher network but with significantly fewer parameters, we replace the parameter vectors in the teacher discriminator with limited parameters and denoted by $\emptyset_m^s$. Hence, the loss terms for the student network are:

$$\mathcal{L}_T^{D_r}(x) = -\sum_{i=1}^{N_d}\sum_{m=1}^{M}\chi_{i,m}^{s}\log\big(D_m(x_i;\phi_m^s)\big)$$

$$\sum_{m=1}^{M}\chi_{i,m} \leq K, \quad (5)$$

$$\mathcal{L}_S^{D_g}(x) = -\sum_{j=1}^{N_g}\sum_{m=1}^{M}\Big(1-\log\Big(D_m\big(G(Z_j);\phi_m^s\big)\Big)\Big) \quad (6)$$

Where $\chi_{i,m}^{s}$ is the indicator variable of the student network that corresponds to $\chi_{i,m}$. Now the total loss corresponding to the student network:

$$\mathcal{L}_S = \alpha_s \mathcal{L}_S^{D_r}(x) + \beta_s \mathcal{L}_S^{D_g}(x) \quad (7)$$

Where α_s and β_s are hyperparameters that assign relative weights to losses over real and generated data in the student network:

$$\mathcal{L}_{GAN} = w_S \mathcal{L}_S + w_T \mathcal{L}_T \quad (8)$$

Where the weights w_S and w_T are gradually adjusted during iterations such that $w_S >> w_T$ when the Teacher network is fully trained.

Materials and Methods

For the proposed GAN-based teacher-student model, we utilized the CIS-IDS-2017 dataset to build the GAN model and the CIS-IDS-2018 dataset to evaluate its performance across various computing environments. We conducted 300 rounds of training using a computer equipped with 52GB of RAM and a GPU with 16GB of memory, which was ideal for processing the large dataset of 1,653,421 samples.

On the other hand, the model was taught on a CPU computer with 4GB of RAM, as is typical of a device at the edge. We arranged 300 rounds of training to coincide with the time recommended by the teacher. We were able to compare the students' performance with the teacher, which helped us address the challenges of using the model on edge devices.

Teacher Student GAN Network Architecture

The proposed approach introduces an enhanced adaptation of GANs, conceptualized initially by Goodfellow et al. (2014), to address challenges of the multi-class intrusion detection in network security. The architecture is designed to incorporate several discriminators, each specialized in identifying a distinct intrusion class, including the Brute-Force, Botnet, DoS, or Web Attacks. The input to each of these discriminators consists of samples that are generated by the teacher generator, tailored to the corresponding intrusion class.

The teacher model is the primary network that deals with learning the specific feature representation of various types of intrusion classes, including Brute-force, Botnet, DoS, and Web Attacks. It features a generator that produces synthetic samples from the input sample noise and a group of discriminators, each specialized in recognizing a specific intrusion class, thereby enabling the ability to classify attacks of diverse types. Building on the teacher model, a student model is introduced to obtain model compression without noticeable performance loss. The student network follows the same topology as the teacher model but has fewer nodes and links in each layer of the discriminators, resulting in a 40% reduction in node parameters. The student generator is conditioned on the knowledge distilled into the teacher network and would effectively learn compact yet discriminative representations of features for use in real-time or edge-based intrusion detection systems.

The architecture is visually detailed in Figure 1, illustrating the hierarchical relationship between the Teacher and Student models. Figure 2 shows that the training process starts with gathering of multi-class anomaly training data, the benign data are labeled as "no anomaly", and random sampling of relabeled batches is performed. These batches are then fed to the Teacher generator to produce transformed samples, and subsequently, these are fed to the discriminators to train with a multi-class GAN loss function.

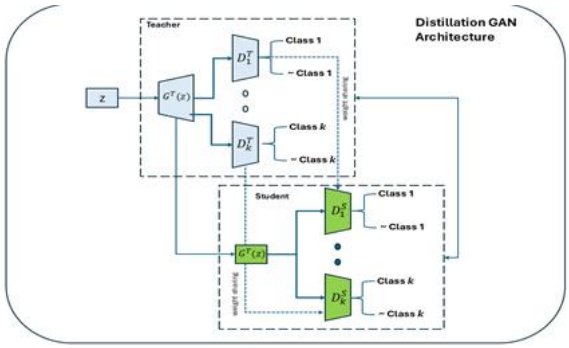

Fig. 1: Proposed Distillation GAN architecture, in which the student model is trained on the teacher GAN through knowledge transfer between the two corresponding class discriminators

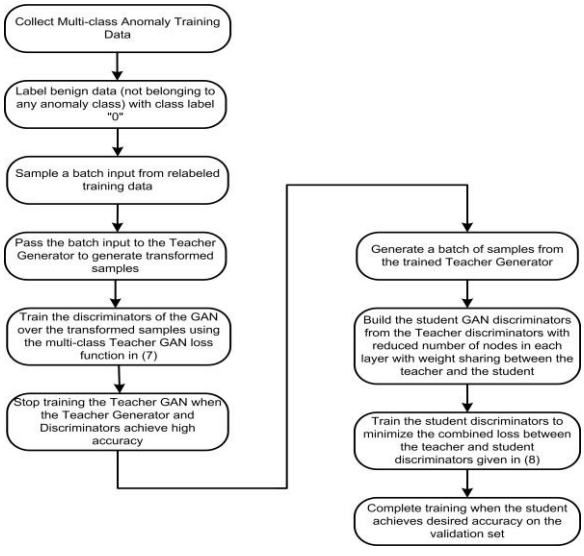

Fig. 2: Distillation GAN training process, which shows how the teacher training, student knowledge distillation, and convergence are performed in a linear sequence according to the accuracy of the validation

The Training of the Teacher GAN will be continued until the accuracy of both the generator and the discriminators is high. The Student GAN discriminators are then built by reducing nodes in each of the layers and sharing weights with the Teacher discriminators. The difference between the teacher and student output is then minimized by optimizing the student network using a loss function (as defined in Eq. 8). The training stops once the student model meets the desired accuracy on the validation set, as shown in the flowchart. This process is followed so that the student model can mimic the teacher's performance at a 40% parameter reduction, which will be deployed at the edges without significant intrusion detection compromises. This distillation approach allows the student model to provide estimations comparable with the performance of the teacher model, maintaining computational efficiency, so that it can be easily applied in practice in low-power or edge implementations of network IDS.

Figure 2 represents the general course of the training process. The first stage consists of training a highly capable teacher using multi-class anomaly data, where label "0" is indicated by benign data. The teacher generator generates conditioned samples of classes, and its discriminators are trained to distinguish between real and generated data of various anomaly classes. After the teacher has reached high reliability and consistent performance, the knowledge is then transferred to a smaller Student GAN using model distillation. The designed student architecture incorporates a lower complexity and partial weight sharing, which is trained on an adversarial and distillation loss. The teacher can use synthetic data to balance underrepresented classes to ensure equal representation of classes in each mini-batch. This data-level balancing is further supported by soft-label distillation, and this allows the student discriminators to align their outputs to the teacher. As a result, the student model gets to learn discriminative boundaries to the rare classes and is not over-fitting to the majority classes. The joint training goal, which consists of adversarial, classification, and distillation terms, is to make sure that the student performs at a similar level as the teacher and is also efficient. The process is useful in easing the imbalance in data and boosting the ability to classify anomalies in a variety of classes.

Dataset

The present work is based on the CIS-IDS-2017 dataset to create GAN models. Table 2 demonstrates that the CIS-IDS-2017 dataset developed by the Canadian Institute of Cybersecurity is the best to evaluate intrusion detection systems. It contains 1,653,421 labeled network traffic samples, and the sample sizes of these classes (C1 to C5) are 440,012 (C5), 1,910,433 (C1), which is extremely disproportional. The dataset represents real-world network environments, such as normal traffic and different types of attacks, such as DoS, DDoS, brute force, web attacks, and botnets, gathered in a testbed simulating a current enterprise network.

We also used the CSE-CIC-IDS2018 dataset to test our GAN-based teacher-student model. Developed by the Communications Security Establishment (CSE) in collaboration with the Canadian Institute of Cybersecurity (CIC), the data set is an excellent resource to test the intrusion detection systems. It has 16,233,002 samples of network traffic, which are all classified in eight different categories. These groups differ significantly in size, as attacks represent only about 17% of the samples, making the dataset unbalanced. The data resembles actual situations, including normal operations, as well as Brute-Force, Heartbleed, Botnet, DoS, DDoS, Web Attacks, and Infiltration. They originate from a setup designed to align with the structure of a modern business network. All samples consist of 80 features, including packet and byte counts, as well as calculated measures such as the duration of a flow and the time interval between packets.

The dataset was generated using the CICFlowMeter-V3 tool and is particularly helpful in addressing today's cybersecurity challenges. There was a significant difference in the ratio of attack and normal samples, which presented challenges for machine learning. Fortunately, techniques such as oversampling and GAN-based enhancements were very helpful. Because the CSE-CIC-IDS2018 dataset is widely used among researchers and facilitates easy comparison of intrusion detection tools, it is suitable for testing the teacher-student model developed in this work.

Table 2: Distribution of attack and benign traffic in the CIC-IDS-2017 dataset

Record ID	Class Name	Count	Relative Frequency (%)
1	Heartbleed	11	0.0004
2	Infiltration	36	0.0014
3	Bot	1948	0.08
4	Web Attacks	2143	0.085
5	DoS Slowhttptest	3219	0.12
6	DoS slowloris	5238	0.20
7	SSH-Patator	5385	0.21
8	FTP-Patator	5931	0.23
9	DoS GoldenEye	10283	0.41
10	DDoS	90694	3.6
11	PortScan	128008	5.08
12	DoS Hulk	172838	6.87
13	BENIGN	2089692	83.07

Results and Discussion

The performance of the GAN-based model on the CIS IDS-2017 data set is summarized in Table 3, and Figure 3 shows the teacher-student precision (P), recall (R), F1 score (F1), false positive rate (FPA), and per-class accuracy (Acc).

The data set comprises1, 653, 421 samples, distributed across five classes (C1 to C5), with C1 having the largest sample size (1,910,433) and C5 the smallest (440,012). The teacher's accuracy, reflecting the model's performance with labeled data, ranges from 86.2% (C5) to 91.6% (C2), with an overall average of 89.1%. The student's accuracy, based on pseudo-labels generated by the teacher model, variesfrom81.1% (C5) to 90.9% (C2), with an overall average of 86.8%. The F1-scores further validate these findings: the teacher's F1-score (F1-Score (T)) ranges from 82.0% (C5) to 91.0% (C2), averaging 88.7% the student's F1-score (F1-Score (S)) ranges from 78.2% (C5) to 90.1% (C2), averaging 88.2%.

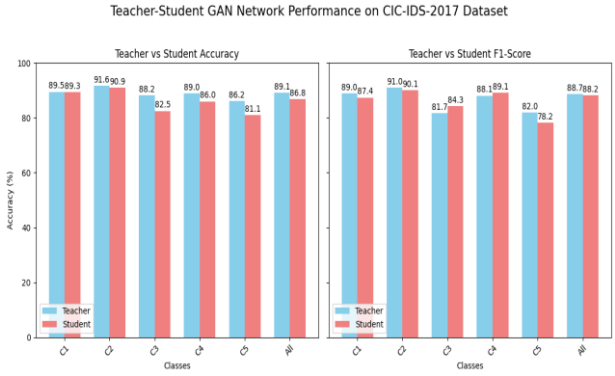

Fig. 3: Teacher Students GAN Network accuracy and F1-score for CICS-IDS-2017 dataset

Table 3: CIC-IDS-2017: Combined Teacher & Student subclass metrics with Macro/Micro average and False Positive

			Teacher					Student				
Class	Subclass	Cnt.	P	R	F1	FPR	Acc.	P	R	F1	FPR	Acc.
C1	C1-1 (Benign)	1908467	88.42	89.55	88.98	10.92		85.56	89.36	87.42	14.04	
C1	C1-2 (Bot)	1966	97.38	84.79	90.65	0.03		88.25	83.42	85.76	0.09	
C1	TOTAL (class-level)	1910433		89.50	89.00				89.30	87.40		
C1	MACRO-AVG (per class)	-	92.90	87.17	89.82	5.48	89.50	86.90	86.39	86.59	7.06	89.30
C1	MICRO-AVG (per class)	-	92.84	89.55	91.18	10.79		85.57	89.36	87.42	13.95	
C2	C2-1 (Benign)	71733	92.76	90.35	91.54	8.84		91.78	90.15	90.96	9.61	
C2	C2-2(Portscan)	87674	88.58	92.62	90.56	7.65		87.38	91.51	89.40	8.64	
C2	TOTAL (class-level)	159407		91.60	91.00				90.90	90.10		
C2	MACRO-AVG (per class)	-	90.67	91.49	91.05	8.24	91.60	89.58	90.83	90.18	9.12	90.90
C2	MICRO-AVG (per class)	-	90.49	91.57	91.06	8.21		89.47	90.88	90.23	9.14	
C3	C3-1 (Benign)	168186	88.68	88.57	88.62	11.46		88.29	82.30	85.19	15.14	
C3	C3-2 (WebAttack–BruteForce)	1507	74.38	90.56	81.71	0.95		92.80	88.14	90.41	0.40	
C3	C3-3 (WebAttack–XSS)	673	75.94	91.66	83.13	0.42	88.20	94.89	89.12	91.90	0.19	82.50
C3	TOTAL (class-level)	170366		88.20	81.70				82.50	84.30		
C3	MACRO-AVG (per class)	-	79.67	90.26	84.49	4.28		92.00	86.52	89.17	5.24	
C3	MICRO-AVG (per class)	-	88.51	89.50	85.76	11.31		88.33	82.31	85.19	14.83	
C4	C4-1 (Benign)	440031	86.30	89.38	87.81	12.06		92.56	85.98	89.14	10.52	
C4	C4-2 (DoS Hulk)	231073	90.68	88.50	89.58	6.13	89.00	85.20	86.07	85.63	7.95	86.00
C4	C4-3 (DoS GoldenEye)	10293	95.69	88.47	91.93	1.85		82.82	85.93	84.34	3.63	
C4	C4-4 (DoS Slowloris)	5796	92.49	90.26	91.36	2.65		87.38	86.34	86.86	4.24	
C4	C4-5 (DoS slowhttptest)	5510	95.33	86.06	90.45	2.78		86.67	85.68	86.17	4.48	
C4	TOTAL (class-level)	692703		89.00	88.10				86.00	89.10		
C4	MACRO-AVG (per class)	-	92.50	88.13	90.23	5.49		86.93	85.60	86.43	6.36	
C4	MICRO-AVG (per class)	-	88.64	89.13	88.56	10.31		90.64	85.98	88.86	10.33	
C5	C5-1 (Benign)	432074	88.56	86.48	87.51	13.06		87.11	81.28	84.09	16.60	
C5	C5-2 (FTP-Patator)	7938	78.52	86.48	82.35	1.69	86.20	74.41	81.30	77.70	2.43	81.10
C5	C5-3 (SSH-Patator)	5897	78.63	86.46	82.35	2.03		74.51	81.27	77.75	2.80	
C5	TOTAL (class-level)	440012		86.20	82.00				81.10	78.20		
C5	MACRO-AVG (per class)	-	81.90	86.47	84.07	5.59		78.68	81.28	79.85	7.28	
C5	MICRO-AVG (per class)	-	88.14	86.48	87.49	13.02		86.99	81.28	84.07	16.44	

The results show the GAN-based model performs robustly across the CIS-IDS-2017 dataset, with an overall teacher's accuracy of 89.1% and an F1-score of 88.7%, demonstrating a strong balance between precision and recall. The student performance is marginally lower with an accuracy of 86.8 and 88.2% F1-score, but it is still competitive, indicating that the pseudo-labeling method is very useful in utilizing the predictions of a teacher. It is interesting to mention that the application of a teacher-student model with GAN contributed to obtaining high F1-scores despite the imbalanced dataset. When the fused GAN classifier was used, the accuracy of the models was sufficiently high and did not significantly decrease. The highest performance of class C2 has been found in all metrics, probably because of its vast size in data, and this offers sufficient training data to the machine learning

models. On the other hand, C5 has the worst results and its accuracies, and F1-scores are much lower than those of the other classes. This is because it has a small dataset (440,012 samples), making it underfit or failing to represent its features.

These findings support the application of GAN model-based discriminators in cases where the detection of the type of intrusion at a finer level is essential to achieve successful implementation in the field of network security. It is natural that the student model performs with slightly reduced accuracy than the teacher model due to the small scale of degradation caused by pseudo-labels in the model, as they cause certain noise and uncertainty. A small difference of about 2.3\% in the accuracy and 0.5% the F1-score is an indication of model stability and the ability of the GAN system to produce credible synthetic labels.

The main objective was to create a student network that can mimic the performance of the teacher network, to implement it safely and practically on an edge device. The outcomes of the above results show that the student network can be applied to inference at the edge. Additionally, to confirm the effectiveness of the proposed teacher-student model, the performance assessments were performed with the help of the CSE-CIC-IDS2018 dataset, which is summarized in Table 4. The accuracy and F1-score of the teacher-students is presented in Fig. 4. The fact that the student model outperforms the teacher in certain subsets (e.g., the student model has an accuracy of 88.20% in comparison to that of the teacher of 86.47%) gives prominence to the fact that the distillation approach is effective in maintaining the detection capabilities without making the computationally heavy. Nevertheless, lower scores in certain subsets (e.g., 73.10% of the students in the fourth subset) suggest difficulties with certain types of attacks or uneven distributions and thus require further research. The next steps in work can be an expansion of the data against underrepresented classes, including C5, or the use of data balancing methods to enhance performance in general. Additionally, exploring advanced GAN architectures or hybrid models may further enhance the student's accuracy, bringing it closer to the teacher's baseline. Thus, the GAN-based teacher-student model demonstrates robust performance across both datasets, with the student model achieving near-teacher performance despite a 40% parameter reduction, as noted in the study. The CIC-IDS-2017 results highlight consistent performance across classes, although class imbalance affects lower-sample classes, such as C5. The CSE-CIC-IDS2018 results show variability, likely due to differing class compositions, but the student model's ability to match or exceed the teacher in some of the subsets underscores the effectiveness of the distillation approach for the edge deployment in resource-constrained environments.

Table 4: CSE-CIC-IDS2018: Teacher/Student subclass metrics with Macro/Micro averaging and FPR

			Teacher					Student				
Class	Subclass	Cnt.	P	R	F1	FPR	Acc.	P	R	F1	FPR	Acc.
C1	C1-1 (Benign)	753769	88.6	87.1	87.8	7.9		86.8	89.4	88.1	8.6	
C1	C1-2 (Bot)	294815	91.9	85.8	88.8	4.3		89.5	85.0	87.2	5.7	
C1	TOTAL (class-level)	1,048,575		86.47	84.03		86.47		88.20	84.11		88.20
C1	MACRO-AVG (per class)	-	90.25	86.45	88.30	6.10		88.15	87.20	87.65	7.15	
C1	MICRO-AVG (per class)	-	89.33	86.47	87.80	6.94		87.53	88.20	87.91	8.10	
C2	C2-1 (Benign)	739411	87.9	86.0	86.9	8.8		89.1	88.5	88.8	7.6	
C2	C2-2(Portscan)	309164	90.6	84.6	87.5	6.1		88.8	87.0	87.9	8.9	
C2	TOTAL (class-level)	1,048,575		85.46	86.30		85.46		88.20	84.72		88.20
C2	MACRO-AVG (per class)	-	89.25	85.30	87.20	7.45		88.95	87.75	88.35	8.25	
C2	MICRO-AVG (per class)	-	88.56	85.46	86.82	8.07		89.02	88.20	88.57	8.06	
C3	C3-1 (Benign)	5567239	89.1	87.3	88.2	6.2		87.6	86.6	87.1	7.2	
C3	C3-2 (WebAttack–BruteForce)	1730205	86.7	88.5	87.6	4.8		86.2	84.4	85.3	5.9	
C3	C3-3 (WebAttack–XSS)	651304	87.9	86.1	87.0	8.2	87.25	88.4	82.7	85.4	9.9	86.30
C3	TOTAL (class-level)	7,948,748		87.25	87.17				86.30	84.86		
C3	MACRO-AVG (per class)	-	87.90	87.30	87.60	6.40		87.40	84.57	85.93	7.67	
C3	MICRO-AVG (per class)	-	88.35	87.25	87.98	6.01		87.13	86.30	86.18	7.21	
C4	C4-1 (Benign)	737590	83.4	77.9	80.6	10.8		85.2	72.6	78.4	11.1	
C4	C4-2 (DoS Hulk)	219544	81.6	75.8	78.6	7.3		80.1	71.8	75.7	9.8	
C4	C4-3 (DoS GoldenEye)	19729	84.5	74.2	79.0	5.9		82.8	70.4	76.1	6.7	
C4	C4-4 (DoS Slowloris)	5223	86.2	73.9	79.6	4.1	76.55	83.1	69.8	75.8	5.5	73.10
C4	C4-5 (DoS slowhttptest)	66389	82.9	76.0	79.3	8.6		79.4	72.6	75.8	9.1	
C4	TOTAL (class-level)	1,048,575		76.55	72.91				73.10	70.24		
C4	MACRO-AVG (per class)	-	83.72	75.56	79.42	7.34		82.12	71.44	76.36	8.44	
C4	MICRO-AVG (per class)	-	83.31	76.55	79.90	9.93		83.84	73.10	78.24	10.41	
C5	C5-1 (Benign)	738592	86.6	84.1	85.3	4.8		87.4	86.4	86.9	5.6	
C5	C5-2 (FTP-Patator)	157340	89.8	82.6	86.1	3.1		90.5	83.1	86.7	4.1	
C5	C5-3 (SSH-Patator)	152643	89.1	83.4	86.2	5.6		90.1	82.6	86.2	7.3	86.17
C5	TOTAL (class-level)	1,048,575		83.42	86.30		83.42		86.17	86.68		
C5	MACRO-AVG (per class)	-	88.50	83.37	85.87	4.50		89.33	84.03	86.60	5.67	
C5	MICRO-AVG (per class)	-	87.17	83.42	85.12	4.57		88.16	86.17	87.26	5.78	

Table 4: Continued

C6	C6-1(Benign)	740112	90.3	89.8	90.0	6.7		86.1	82.8	84.4	7.4	
C6	C6-2(HOIC)	308463	88.2	87.7	87.9	4.2		85.8	82.5	84.1	3.8	
C6	TOTAL (class-level)	1,048,575		89.26	81.62		89.26		82.83	81.11		82.83
C6	MACRO-AVG (perclass)	-	89.25	88.75	88.95	5.45		85.95	82.65	84.25	5.60	
C6	MICRO-AVG (perclass)	-	89.65	89.26	89.50	6.01		86.03	82.83	84.60	6.43	
C7	C7-1(Benign)	813307	81.2	79.4	80.3	10.6		85.6	83.7	84.6	9.7	
C7	C7-2(LOIC-HTTP)	235268	83.9	78.6	81.1	7.1		86.9	82.6	84.7	5.3	
C7	TOTAL (class-level)	1,048,575		79.205	77.31		79.2		83.17	80.36		83.17
C7	MACRO-AVG (perclass)	-	82.55	79.00	80.70	8.85		86.25	83.15	84.65	7.50	
C7	MICRO-AVG (perclass)	-	81.66	79.20	80.22	9.85		86.09	83.17	84.29	8.93	

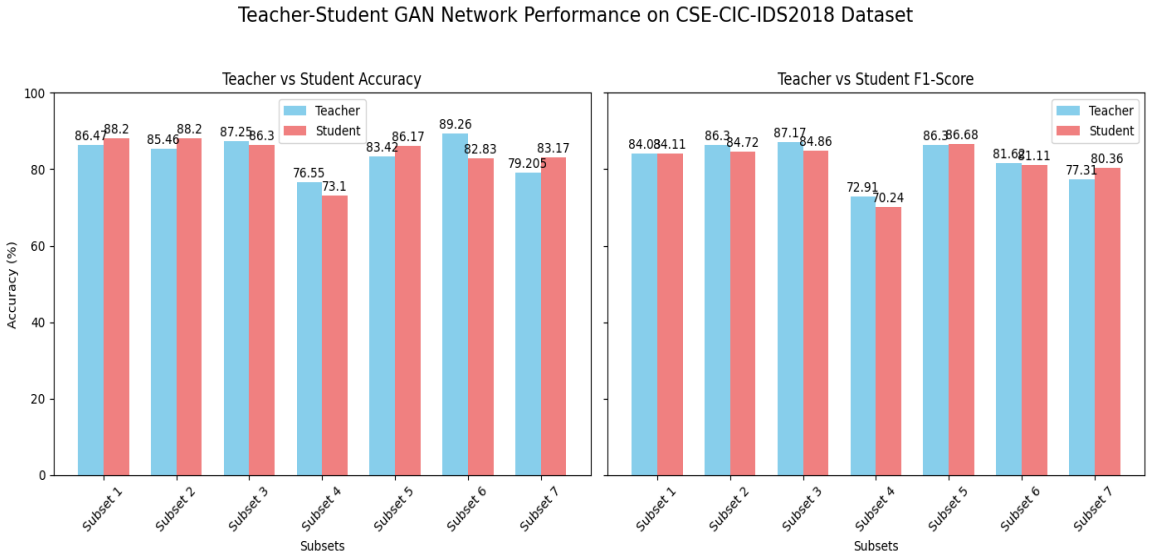

Fig. 4: Teacher Students GAN Network accuracy and F1-score for CICS-IDS-2018 dataset

Conclusion

In this work, a distillation-based GAN model was proposed, adapted to network intrusion detection. It involves initially training a teacher GAN and then using its generator to train its discriminators, which are the student GANs. The GAN model was optimized by introducing a weighted loss function for improved performance, suitable for deployment on edge devices on a network. The effectiveness of the model was proven using experimental analysis of CIC-IDS-2017 and CSE-CIC-IDS-2018 data sets. The student network on the CICIDS-2017 dataset showed an overall accuracy of 86.8% and an F1-score of 88.2%, comparable to that of the teacher, with an accuracy of 89.1% and an F1-score of 88.7%, with a 40% decrease in the number of parameters. For the CSE-CIC-IDS2018 dataset, the student model showed a competitive performance, with accuracy ranging from 73.10% - 88.20% and 70.24% - 86.68% for F1-score, respectively, and aligning with and even outperforming the metrics of a teacher's accuracy of 76.55% - 89.26% and F1-scores of 72.91% - 87.17%. Results highlighted the usability of the proposed model in a real-time resource-constrained environment to identify the intrusions in the network. Future work will focus on implementing a parallel training strategy to enhance the performance of the teacher network. Additionally, the performance of the GAN-based teacher-student architecture will be evaluated on other benchmark intrusion detection data sets, such as UNSW-NB15. Incorporating unsupervised or continual learning mechanisms could further improve adaptability to novel threats. Furthermore, the hardware implementation of the inference engine at the edge will be investigated to assess its real-world applicability. The proposed GAN framework was limited to supervised and semi-supervised learning settings.

Acknowledgment

We thank MAHE (Deemed to be University) for their support in the development of this work. We thank the publisher for giving us the opportunity to share our work and contribute to the field through this publication, and also extend special thanks to the editorial team for reviewing and editing the article.

Funding Information

This research received no external funding.

Author's Contributions

Chandrakala C B: Writer original draft, validation, software, resources, methodology, formal analysis, data curation, conceptualization, supervision.

Sai Sreevalli: Writer original draft, visualization, resources, methodology, conceptualization.

Raghudathesh G P: Writer original draft, methodology, data curation.

Ethics

This article is original and has not been published elsewhere. All authors have read and approved the final version of the manuscript, and there are no ethical issues with this research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

Aldhaheri, S., & Alhuzali, A. (2023). *Self-attention-based generative adversarial network against intrusion detection systems*. Sensors. https://doi.org/10.3390/s23187796

Ali, T., Eleyan, A., Bejaoui, T., & Al-Khalidi, M. (2024). *Lightweight Intrusion Detection System with GAN-Based Knowledge Distillation*. 1–7. https://doi.org/10.1109/smartnets61466.2024.10577682

Alom, Z., Bontupalli, V. R., & Taha, T. M. (2015). Intrusion Detection Using Deep Belief Network and Extreme Learning Machine. *International Journal of Monitoring and Surveillance Technologies Research*, *3*(2), 35–56. https://doi.org/10.4018/ijmstr.2015040103

Alrawashdeh, K., & Purdy, C. (2016). *Toward an Online Anomaly Intrusion Detection System Based on Deep Learning*. 195–200. https://doi.org/10.1109/icmla.2016.0040

Althubiti, S., Nick, W., Mason, J., Yuan, X., & Esterline, A. (2018). *Applying Long Short-Term Memory Recurrent Neural Network for Intrusion Detection*. 1–5. https://doi.org/10.1109/secon.2018.8478898

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

Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security*, *3*(3), 186–205. https://doi.org/10.1145/357830.357849

Bace, R., & Mell, P. (2001). *Intrusion detection systems.*

Bharathi, I., & Makhija, R. (2024). *Network Intrusion Detection System using Random Forest and Gradient Boosting Machines*. 1–5. https://doi.org/10.1109/conit61985.2024.10627542

Bucilă, C., Caruana, R., & Niculescu-Mizil, A. (2006). *Model compression*. 535–541. https://doi.org/10.1145/1150402.1150464

Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys & Tutorials*, *18*(2), 1153–1176. https://doi.org/10.1109/comst.2015.2494502

Chen, D., Li, Y., Qiu, M., Wang, Z., Li, B., Ding, B., Deng, H., Huang, J., Lin, W., & Zhou, J. (2020). *AdaBERT: Task-Adaptive BERT Compression with Differentiable Neural Architecture Search*. 2463–2469. https://doi.org/10.24963/ijcai.2020/341

Choi, H., Kim, M., Lee, G., & Kim, W. (2019). Unsupervised Learning Approach for Network Intrusion Detection System Using Autoencoders. *The Journal of Supercomputing*, *75*(9), 5597–5621. https://doi.org/10.1007/s11227-019-02805-w

Constantin, M. G., Stanciu, D.-C., Ștefan, L.-D., Dogariu, M., Mihăilescu, D., Ciobanu, G., Bergeron, M., Liu, W., Belov, K., Radu, O., & Ionescu, B. (2025). Exploring Generative Adversarial Networks for Augmenting Network Intrusion Detection Tasks. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *21*(1), 1–19. https://doi.org/10.1145/3689636

Debar, H., Becker, M., & Siboni, D. (2003). *A neural network component for an intrusion detection system*. 240–250. https://doi.org/10.1109/risp.1992.213257

Denning, D. (1987). An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*, *SE-13*(2), 222–232. https://doi.org/10.1109/tse.1987.232894

Ferdowsi, A., & Saad, W. (2019). *Generative Adversarial Networks for Distributed Intrusion Detection in the Internet of Things*. 1–5. https://doi.org/10.1109/globecom38437.2019.9014102

Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, *50*, 102419. https://doi.org/10.1016/j.jisa.2019.102419

García-Teodoro, P., Díaz-Verdejo, J. E., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, *28*(1–2), 18–28. https://doi.org/10.1016/j.cose.2008.08.003

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge Distillation: A Survey. *International Journal of Computer Vision*, *129*(6), 1789–1819. https://doi.org/10.1007/s11263-021-01453-z

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *ArXiv Preprint ArXiv:1503.02531*. https://doi.org/10.48550/arXiv.1503.02531

Hu, W., & Tan, Y. (2022). Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN. *Data Mining and Big Data. DMBD 2022*, *1745*, 409–423. https://doi.org/10.1007/978-981-19-8991-9_29

Ilse, M., Tomczak, J. M., & Welling, M. (2018). Attention-based deep multiple instance learning. *35th International Conference on Machine Learning, ICML*, 2127–2136.

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). *A Deep Learning Approach for Network Intrusion Detection System*. 1–6. https://doi.org/10.4108/eai.3-12-2015.2262516

Kim, J., Shin, N., Yeon Jo, S., & Hyun Kim, S. (2017). *Method of intrusion detection using deep neural network*. 313–316. https://doi.org/10.1109/bigcomp.2017.7881684

Ko, C., Fink, G. A., & Levitt, K. (1994). Automated detection of vulnerabilities in privileged programs by execution monitoring. *Tenth Annual Computer Security Applications Conference*, 134–144. https://doi.org/10.1109/csac.1994.367313

Lee, W., & Stolfo, S. J. (2000). A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, *3*(4), 227–261. https://doi.org/10.1145/382912.382914

Li, D., Chen, D., Jin, B., Shi, L., Goh, J., & Ng, S.-K. (2019). MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. *Artificial Neural Networks and Machine Learning – ICANN 2019: Text and Time Series*, *11730*, 703–716. https://doi.org/10.1007/978-3-030-30490-4_56

Lin, Z., Shi, Y., & Xue, Z. (2022). IDSGAN: Generative Adversarial Networks for Attack Generation Against Intrusion Detection. *Advances in Knowledge Discovery and Data Mining*, *13282*, 79–91. https://doi.org/10.1007/978-3-031-05981-0_7

Mao, C.-H., Lee, H.-M., Parikh, D., Chen, T., & Huang, S.-Y. (2009). Semi-supervised co-training and active learning based approach for multi-view intrusion detection. *SAC09: The 2009 ACM Symposium on Applied Computing*, 2042–2048. https://doi.org/10.1145/1529282.1529735

Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. *Network and Distributed System Security Symposium*, 1–15. https://doi.org/10.14722/ndss.2018.23204

Mishra, A., & Marr, Debbie. (2017). Apprentice: Using knowledge distillation techniques to improve lowprecision network accuracy. *ArXiv Preprint ArXiv:1711.05852*.

Mukkamala, S., Sung, A. H., & Abraham, A. (2005). Intrusion detection using an ensemble of intelligent paradigms. *Journal of Network and Computer Applications*, *28*(2), 167–182. https://doi.org/10.1016/j.jnca.2004.01.003

Mushtaq, E., Zameer, A., Umer, M., & Abbasi, A. A. (2022). A two-stage intrusion detection system with auto-encoder and LSTMs. *Applied Soft Computing*, *121*, 108768. https://doi.org/10.1016/j.asoc.2022.108768

Paxson, V. (1999). Bro: a system for detecting network intruders in real-time. *Computer Networks*, *31*(23–24), 2435–2463. https://doi.org/10.1016/s1389-1286(99)00112-7

Phuong, M., & Lampert, C. (2019). Towards understanding knowledge distillation. *36th International Conference on Machine Learning (ICML), ICML*, 5142–5151.

Polino, A., Pascanu, R., & Alistarh, D. (2018). Model compression via distillation and quantization. *6th International Conference on Learning Representations ICLR 2018 - Conference Track Proc*, 1–10.

Potluri, S., & Diedrich, C. (2016). Accelerated deep neural networks for enhanced Intrusion Detection System. *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–8. https://doi.org/10.1109/etfa.2016.7733515

Potluri, S., Ahmed, S., & Diedrich, C. (2018). Convolutional Neural Networks for Multi-class Intrusion Detection System. *Mining Intelligence and Knowledge Exploration*, *11308*, 225–238. https://doi.org/10.1007/978-3-030-05918-7_20

Ptacek, T., & Newsham, T. N. (1998). Insertion, evasion, and denial of service: Eluding network intrusion detection. *CEUR Workshop Proceedings*, 1–24.

Riku, I. and Timo, H. (2022). Tiny machine learning for resource-constrained microcontrollers. *Journal of Sensors*, 7437023. https://doi.org/https://doi.org/10.1155/2022/7437023

Ring, M., Wunderlich, S., Scheuring, D., Landes, D., & Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers & Security*, *86*, 147–167. https://doi.org/10.1016/j.cose.2019.06.005

Roesch, M. (1999). Snort - lightweight intrusion detection for networks. *Proceedings of the 13th USENIX Conference on System Administration*, 229–238.

Sabhnani, M., & Serpen, G. (2003). *Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context.* 209–215.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*. *1*, 5. https://doi.org/10.48550/arXiv.1910.01108

Scarfone, K., & Mell, P. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS)*. https://doi.org/10.6028/nist.sp.800-94

Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *2*(1), 41–50. https://doi.org/10.1109/tetci.2017.2772792

Sommer, R., & Paxson, V. (2010). *Outside the Closed World: On Using Machine Learning for Network Intrusion Detection*. 305–315. https://doi.org/10.1109/sp.2010.25

Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., & Ghogho, M. (2016). *Deep learning approach for Network Intrusion Detection in Software Defined Networking*. 258–263. https://doi.org/10.1109/wincom.2016.7777224

Toupas, P., Chamou, D., Giannoutakis, K. M., Drosou, A., & Tzovaras, D. (2019). *An Intrusion Detection System for Multi-class Classification Based on Deep Neural Networks*. 1253–1258. https://doi.org/10.1109/icmla.2019.00206

Tsai, C.-F., Hsu, Y.-F., Lin, C.-Y., & Lin, W.-Y. (2009). Intrusion detection by machine learning: A review. fFIG*Expert Systems with Applications*, *36*(10), 11994–12000. https://doi.org/10.1016/j.eswa.2009.05.029

Tseng, S.-M., Wang, Y.-Q., & Wang, Y.-C. (2024). Multi-Class Intrusion Detection Based on Transformer for IoT Networks Using CIC-IoT-2023 Dataset. *Future Internet*, *16*(8), 284. https://doi.org/10.3390/fi16080284

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, *7*, 41525–41550. https://doi.org/10.1109/access.2019.2895334

Wang, L., Zhang, W., He, X., & Zha, H. (2018). *Supervised Reinforcement Learning with Recurrent Neural Network for Dynamic Treatment Recommendation*. 2447–2456. https://doi.org/10.1145/3219819.3219961

Wisanwanichthan, T., & Thammawichai, M. (2025). A Lightweight Intrusion Detection System for IoT and UAV Using Deep Neural Networks with Knowledge Distillation. *Computers*, *14*(7), 291. https://doi.org/10.3390/computers14070291

Yadav, M., S., & Kalpana, R. (2022). Recurrent nonsymmetric deep auto encoder approach for network intrusion detection system. *Measurement: Sensors*, *24*, 100527. https://doi.org/10.1016/j.measen.2022.100527

Yang, C., Xie, L., Su, C., & Yuille, A. L. (2019). *Snapshot Distillation: Teacher-Student Optimization in One Generation*. 2854–2863. https://doi.org/10.1109/cvpr.2019.00297

Yang, Y., Zheng, K., Wu, C., & Yang, Y. (2019). Improving the Classification Effectiveness of Intrusion Detection by Using Improved Conditional Variational AutoEncoder and Deep Neural Network. *Sensors*, *19*(11), 2528. https://doi.org/10.3390/s19112528

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*, *5*, 21954–21961. https://doi.org/10.1109/access.2017.2762418

Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Chandrasekhar, V. R. (2018). Efficient gan-based anomaly detection. *ICLR Workshop*. https://doi.org/10.48550/arXiv.1802.06222

Zhang, Y., Li, P., & Wang, X. (2019). Intrusion Detection for IoT Based on Improved Genetic Algorithm and Deep Belief Network. *IEEE Access*, *7*, 31711–31722. https://doi.org/10.1109/access.2019.2903723

Zhang, Z., Chen, S., & Sun, L. (2020). *P-KDGAN: Progressive Knowledge Distillation with GANs for One-class Novelty Detection*. 3237–3243. https://doi.org/10.24963/ijcai.2020/448

Zhao, X., Fok, K. W., & Thing, V. L. L. (2024). Enhancing network intrusion detection performance using generative adversarial networks. *Computers & Security*, *145*, 104005. https://doi.org/10.1016/j.cose.2024.104005