


## Article

# Reinventing Web Security: An Enhanced Cycle-Consistent Generative Adversarial Network Approach to Intrusion Detection

Menghao Fang <sup>1,†</sup>, Yixiang Wang <sup>2,†</sup>, Liangbin Yang <sup>1</sup>, Haorui Wu <sup>1</sup>, Zilin Yin <sup>2</sup>, Xiang Liu <sup>2</sup>, Zexian Xie <sup>1</sup> and Zixiao Kong <sup>1,\*</sup> 

<sup>1</sup> School of Cyber Science and Engineering, University of International Relations, Beijing 100091, China; menghao@uir.edu.cn (M.F.); ylb@uir.edu.cn (L.Y.); wuhaorui@uir.edu.cn (H.W.); zxxie@uir.edu.cn (Z.X.)

<sup>2</sup> Marine Engineering of College, Dalian Maritime University, Dalian 116026, China; bmw8369m@dlmu.edu.cn (Y.W.); a1307403667@dlmu.edu.cn (Z.Y.); liuxiang0611@dlmu.edu.cn (X.L.)

\* Correspondence: kongzixiao@uir.edu.cn

<sup>†</sup> These authors contributed equally to this work.

**Abstract:** Web3.0, as the link between the physical and digital domains, faces increasing security threats due to its inherent complexity and openness. Traditional intrusion detection systems (IDSs) encounter formidable challenges in grappling with the multidimensional and nonlinear traffic data characteristic of the Web3.0 environment. Such challenges include insufficient samples of attack data, inadequate feature extraction, and resultant inaccuracies in model classification. Moreover, the scarcity of certain traffic data available for analysis by IDSs impedes the system's capacity to document instances of malicious behavior. In response to these exigencies, this paper presents a novel approach to Web3.0 intrusion detection, predicated on the utilization of cycle-consistent generative adversarial networks (CycleGANs). Leveraging the data transformation capabilities of its generator, this method facilitates bidirectional conversion between normal Web3.0 behavioral data and potentially intrusive behavioral data. This transformative process not only augments the diversity and volume of recorded intrusive behaviors but also clandestinely simulates various attack scenarios. Furthermore, through fostering mutual competition and learning between the discriminator and generator, the approach enhances the ability to discern the defining characteristics of potential intrusive behaviors, thereby bolstering the accuracy of intrusion detection. To substantiate the efficacy of the CycleGAN-based intrusion detection method, simulation experiments were conducted utilizing public datasets, including KDD CUP 1999 (KDD), CIC-DDOS2019, CIC-IDS2018, and SR-BH 2020. The experimental findings evince the method's remarkable accuracies across the four datasets, attaining rates of 99.81%, 97.79%, 89.25%, and 95.15%, respectively, while concurrently maintaining low false-positive rates. This research contributes novel insights and methodologies toward the advancement of Web3.0 intrusion detection through the application of CycleGAN technology, which is poised to play a pivotal role in fortifying the security landscape of Web3.0.

**Keywords:** Web3.0; CycleGAN; intrusion detection; deep learning; data augmentation



**Citation:** Fang, M.; Wang, Y.; Yang, L.; Wu, H.; Yin, Z.; Liu, X.; Xie, Z.; Kong, Z. Reinventing Web Security: An Enhanced Cycle-Consistent Generative Adversarial Network Approach to Intrusion Detection. *Electronics* **2024**, *13*, 1711. <https://doi.org/10.3390/electronics13091711>

Academic Editor: Aryya Gangopadhyay

Received: 26 March 2024

Revised: 23 April 2024

Accepted: 25 April 2024

Published: 29 April 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

As one of the most significant technological trends of the 21st century, Web3.0 has profoundly impacted human life and work. With the interconnection of various physical devices, sensors, and embedded systems, we have entered a new digital era where the interaction of smart devices and data sharing have become crucial. The essence of Web3.0 lies in its decentralization and the concept of user-controlled data, making data security and privacy protection focal points. However, the rapid development of Web3.0 has also brought about a series of serious security challenges, necessitating continuous exploration of innovative solutions to ensure its sustainable development and security [1].

The SolarWinds supply chain attack was one of the most notable events between 2020 and 2021, affecting thousands of companies and government organizations. Hackers successfully infiltrated many customers' network systems by tampering with SolarWinds' software updates. This event serves as a significant warning for the security of Web3.0. While Web3.0's decentralized nature and smart contracts offer innovative potential, they also bring new security challenges.

Web3.0 encompasses decentralized networks, smart contracts, digital assets, and other areas, facing various security challenges including, but not limited to, the security of decentralized networks, smart contract vulnerabilities, the secure management of digital assets, and supply chain attacks [2]. Therefore, the demand for Web3.0 security technologies continues to grow. By integrating intrusion detection technologies, potential intrusions and attacks in Web3.0 networks can be effectively monitored and defended against. This integration enables Web3.0 networks to better protect user data, smart contracts, and digital assets, thereby enhancing network security and trustworthiness [3]. Currently, cutting-edge security technologies include blockchain [4], edge computing [5], threat intelligence [6], and intrusion detection techniques. Intrusion detection systems (IDSs) can monitor sensitive data [7], prevent the leakage of private data, detect DoS attacks [8], and reduce the risks of data misuse [9] and privacy violations [10].

Intrusion detection systems are crucial in today's Web security field for monitoring and identifying potential network intrusions and security threats. Although intrusion detection technologies have made significant progress in recent decades, they still face multiple challenges and limitations that affect performance, accuracy, and availability. False positives and false negatives [11] frequently occur, reducing the credibility and effectiveness of the system. Traditional intrusion detection systems typically rely on known attack patterns and signatures to detect threats, but zero-day vulnerabilities and advanced persistent threats (APTs) use new attack methods [12], making it difficult for traditional approaches to identify these threats. To address these issues, machine learning and deep learning techniques have been introduced to improve the accuracy and adaptability of intrusion detection systems to better handle diverse threats.

This research proposes a Web intrusion detection system based on cycle-consistent adversarial network models [13], which have outstanding capabilities in anomalous traffic detection and data augmentation. The model maps the process of data anomalies caused by network traffic attacks to the generator's process of converting normal data into anomalous data, achieving the goal of data augmentation. Meanwhile, the adversarial learning between the discriminator and generator improves the discriminator's ability to identify anomalous data.

Experiments have verified that the proposed method can accurately detect various attacks on the Web. We evaluated the model on the KDD99 [14], CIC-IDS2018 [15], CIC-DDOS2019 [16], and SB-RH 2020 [17] datasets. On the KDD99 dataset, the model achieved a high accuracy of 99.81%; on the CIC-DDOS2019 dataset, the accuracy reached 97.79%; on the CIC-IDS2018 dataset, it was 89.25%; and on the SR-BH 2020 dataset, it was 95.15%. The results demonstrate that compared to LSTM, RNN, MLP, and other deep learning models, this model significantly improves performance.

The main contributions of this paper are as follows:

- (1) Network traffic analysis is often limited by insufficient attack samples. To address this data scarcity issue, this model introduces cycle-consistent adversarial networks (CycleGAN) to convert data across domains. In attack detection tasks, normal network traffic and malicious attack traffic are viewed as two domains, and CycleGAN enables mutual conversion between them to generate more training samples. In this way, the training dataset is effectively expanded, improving the model's generalization capability.

- (2) CycleGAN is utilized not only for data augmentation, but also for feature learning and transfer. This model trains a CycleGAN model to convert normal traffic into feature representations of malicious traffic, then uses these feature representations to train the

intrusion detection model. This allows the model to learn richer features from malicious traffic and improves detection performance.

(3) For novel attacks with unknown attack patterns and signatures, such as zero-day vulnerabilities and advanced persistent threats (APTs), the discriminator of this model can distinguish between normal network traffic and unknown attack traffic by learning the characteristics of normal traffic.

(4) The methodology proposed in this paper exhibits excellent performance. Simulation experiments were conducted using public datasets, namely KDD CUP 1999 (KDD), CIC-DDoS2019, CIC-IDS2018, and SR-BH 2020. The experimental results demonstrate high accuracy rates on these datasets, reaching 99.81%, 97.79%, 89.25%, and 95.15%, respectively, with concurrently low false-positive rates.

The remainder of this paper is organized as follows: Section 2 will elaborate on the background and motivations of this research in detail to highlight the rationale behind the CycleGAN-based intrusion detection method. Section 3 will introduce the proposed methods in detail, including the technical details of CycleGAN-based data augmentation and feature learning, as well as attack detection methods. Section 4 will present the experimental results, comprehensively evaluating the performance of the CycleGAN-based intrusion detection system in depth and comparing it with traditional methods. Finally, Section 5 concludes the paper, objectively summarizing the limitations, contributions, and future research directions.

## 2. Background and Related Work

### 2.1. DNN and CycleGAN

Deep neural networks (DNNs) are a biologically inspired machine learning model that mimic biological neural networks. They consist of multiple stacked layers of neural units, with each layer containing multiple neurones interconnected through adjustable weight connections. The early origins of neural networks can be traced back to the 1950s–1960s, including Frank Rosenblatt’s perceptron [18] and Marvin Minsky and Seymour Papert’s research on the limitations of perceptrons [19]. However, progress on early DNNs was limited until Yann LeCun proposed the convolutional neural network (CNN) model LeNet-5 in 1998 [20], which was trained using backpropagation. However, DNNs underperformed compared to traditional machine learning algorithms and training deep networks was challenging. In 2012, Alex Krizhevsky’s AlexNet [21] marked a major breakthrough for DNNs, successfully introducing deep neural networks to the field of image recognition and completely transforming the field. AlexNet overturned traditional image classification, and DNNs started to emerge, followed by many network architectures like VGGNet [22], GoogleNet [23], and ResNet [24]. These models have superior classification capabilities, and are used to analyze network traffic to identify malicious attacks, such as Swarna Priya R.M. et al. [25] using deep neural networks to classify and predict unknown network attacks.

Since they were first proposed in 2014, generative adversarial networks (GANs) [26] have been widely applied in anomaly detection. GANs are a deep learning model composed of a generator and discriminator that compete with each other, continuously adjusting parameters so the generator can produce more realistic data to improve the discriminator’s accuracy. In 2016, researchers proposed the pix2pix model [27], which utilizes adversarial training similar to GANs and can convert input images to associated output images, such as converting line drawings to colored images. However, pix2pix requires paired training data, making it unsuitable for some cases. The key innovation of cycle-consistent GANs (CycleGAN) is the ability to achieve unpaired cross-domain image translation, converting images from one domain to another without paired datasets. Compared to DiscoGAN [28], proposed in the same year, which can also perform cross-domain translation, DiscoGAN requires paired training data. In addition to anomaly detection capabilities, CycleGAN has the unique ability of data-type conversion. CycleGAN represents an important milestone in the development of GANs, enabling more practical and efficient data-type conversion

through continuous improvements and optimizations. Compared to other deep learning models, such as LSTM, AE [29], CNN, and GCN [30], which can also be used for anomaly detection tasks, CycleGAN also enables exceptional cross-domain conversion.

## 2.2. Intrusion Detection Based on Deep Learning

The concept of intrusion detection was first proposed by James Anderson [31] in 1980, who described a method to monitor and detect anomalous activities in computer systems, which can be seen as an early intrusion detection system prototype. Another early work was the host-based intrusion detection model proposed by Dorothy Denning [32] in 1987, which focused on detecting abnormal or anomalous behaviors in computer systems.

In the 1990s, researchers began to use traditional neural networks such as MLPs [33] for anomaly detection in networks. With the rise in deep learning, the performance of deep neural network-based intrusion detection systems has greatly improved and has become a major approach. In recent years, many deep neural network intrusion detection systems have emerged, such as the system by Ghulam Muhammad et al. [34], which combines autoencoders and deep neural networks, learns features unsupervised, and then, supervised, trains the DNN to extract deep features for classification.

Yanqing Yang et al. [35] proposed the SAVAER-DNN intrusion detection model, using the SAVAER decoder, to generate low-frequency and unknown attack samples, increasing data diversity and balancing the dataset. The model can detect both known and unknown attacks, improving the detection rate for low-frequency attacks. Neelu Khare et al. [36] combined deep learning and machine learning, improving detection performance by optimizing the dataset. Chaofei Tang et al. [37] proposed the SAAE-DNN intrusion detection model, using the SAAE encoder to automatically extract features and initialize DNN weights, improving detection accuracy.

Mohammad Al-Fawareh et al. [38] proposed the PCA-DNN model to detect anomalous network behaviors, addressing issues like high false alarm rates, long detection times, and zero-day attacks. Ankit Thakkar et al. [39] analyzed the impact of  $L1$ ,  $L2$ , elastic net regularization and dropout techniques on DNN intrusion detection performance. K. Narayana Rao et al. [40] proposed a two-stage hybrid approach, where in the first stage  $L1$  regularization sparsifies the autoencoder, and in the second stage the DNN predicts and classifies attacks, achieving high detection rates.

E. Balamurugan et al. [41] proposed the IDSGT-DNN framework, which incorporates attacker and defender mechanisms to process attack and normal data. Ankit Thakkar et al. [42] proposed a new feature selection technique by integrating differences, fusing the differences between standard deviation, mean, and median to improve DNN-IDS performance. The following year, Ankit Thakkar et al. [43] used a machine learning-driven deep neural network to classify unbalanced intrusion data, addressing the class imbalance issue in intrusion detection datasets.

Since its advent in 2014, generative adversarial networks (GANs) have gained much attention; although initially used for image tasks, they have expanded into multi-disciplinary research. In network security, they are especially used for intrusion detection tasks dealing with imbalanced datasets [44]. As data samples are mostly imbalanced in most cases, causing intrusion detection models to be biased towards majority classes, to address this, Vikash Kumar et al. [45] proposed a Wasserstein conditional GAN (WCGAN) combined with an XGBoost classifier. They used gradient penalty with WCGAN to stabilize model training, enabling the model to generate highly similar minority class samples.

Recent related research shows that, on the one hand, some researchers adopt deep neural networks (DNNs) to analyze Web3.0 network traffic to improve the ability of intrusion detection systems to distinguish between normal and malicious traffic. On the other hand, researchers combine machine learning and deep learning algorithms to improve detection performance. In addition, some researchers focus on techniques like regularization and dropout to improve DNN model performance. Finally, to address imbalanced dataset issues, some researchers use generative adversarial networks (GANs)

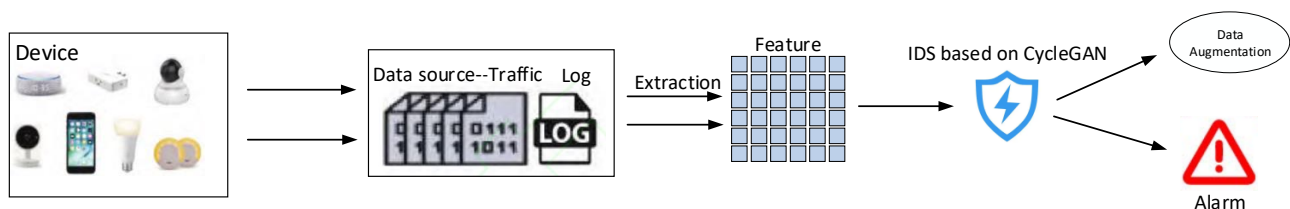
to simultaneously handle sample generation and attack behavior detection. The explanation table for the relevant work is shown in Table 1.

**Table 1.** Related work.

IDS	Dataset	Supervised/Unsupervised	Application Scenarios	Advantages	Disadvantages	Algorithm Complexity	Accuracy
Stacked Autoencoder-Based IDS [36]	KDDCup99, NSL-KDD, Aegean Wi-Fi intrusion Dataset	Semi-supervised	Financial Transactions	Has made innovative contributions in the field of financial transactions and achieved high results in this area.	It has a high model complexity and therefore requires a lot of time and data to train.	High	94.2%, 99.7%, 99.9%
SAVAER-DNN [37]	NSL-KDD, UNSW-NB15	supervised	Network Monitoring	More effective in detecting low-frequency and unknown attacks.	The data augmentation scheme is unstable. And it requires a large amount of training set data.	High	89.4%, 93.0%
SMO-DNN [38]	KDD Cup 99, NSL-KDD	supervised	Network Monitoring	Can use less training data and achieve better results.	The generalization of the model is low.	Middle	99.4%, 92.0%
SAAE-DNN [39]	NSL-KDD	supervised	Simulation and Simulation	Low model complexity.	Low accuracy.	Low	87.7%
PCA-DNN [40]	CSE-CI-UNB 2018	supervised	Network Monitoring	Less computational resources required, strong model generalization ability.	When subjected to a large number of attacks, its performance will be weakened.	Middle	97.0%
SAE-DNNL1 [42]	UNSW-NB15	Semi-supervised	Network Monitoring	Applying sparsity regularization to weights enables compressed feature extraction for more comprehensive feature capture.	Complex transformations of data features can lead to poor generalization ability of the model.	High	99.9%
IDSGT-DNN [43]	CICIDS-2017	supervised	Cloud Computing	Collecting models and policies can effectively reduce training resources.	The applicability of the strategy needs to be considered.	Middle	97.9%
DNN- feature selection technique [44]	NSL-KDD, UNSW _NB-15, CIC-IDS-2017	supervised	Network Monitoring	A simplified feature subset composed of features with high distinguishability and bias can be derived.	None.	Middle	99.84%, 89.03%, 99.80%
Ours	KDDCup99, CIC-DDoS2019, CIC-IDS2018, SR-BH 2020	Semi-supervised	Network Monitoring	The required amount of training data is small, the training time is short, and the training resources are limited.	Weak generalization ability and lack of interpretability.	Low	99.81%, 97.79%, 89.25%, 95.15%

### 3. Intrusion Detection Framework

In this section, we will introduce the intrusion detection framework proposed in this paper and its functionality in the context of Web3.0. Web3.0 applications involve interactions with decentralized networks, smart contracts, and digital assets, which generate network traffic that may contain malicious payloads. The model proposed in this paper analyzes this Web3.0 traffic by extracting features, performs data augmentation, and detects malicious activities. The reference architecture of the intrusion detection model for Web3.0 proposed in this paper is as shown in Figure 1.



**Figure 1.** The architecture of Web intrusion detection methods.

#### 3.1. Dataset Definition

In this paper, normal data are defined as the source domain, represented by dataset  $X : \{x_i\}_{i=1}^N$ ; anomalous data are defined as the target domain, represented by dataset  $Y : \{y_j\}_{j=1}^M$ . Taking the KDD99 dataset as an example, data with the normal label are considered the source domain, data with the Back attack-type label are considered the



target domain, and data of other attack types are considered other domains, represented by the Other dataset:  $\{\text{other}_i\}_{i=1}^N$ .

### 3.2. Cycle-Consistent Generative Adversarial Network

The intrusion detection model proposed in this research is based on an unsupervised learning method, using cycle-consistent adversarial networks (CycleGANs). CycleGAN is an image-to-image translation method that does not require paired training data. For given datasets from two domains, CycleGAN can translate between the two domains without needing to match data pairs one-to-one. It works by learning to map data from one domain to the other, and then back to the original domain, while preserving consistency between the original data and reconstructed data. This adversarial generative network-based technique enables CycleGAN to achieve high-quality cross-domain data translation. Due to its superior generalization even on small datasets, it can outperform traditional methods. In this paper, we apply this method to attack-type conversion, translating normal data to anomalous data, in order to effectively augment the dataset.

$L_{\text{GAN}}(G, D_Y, X, Y)$  is the generator loss,  $L_{\text{GAN}}(F, D_X, Y, X)$  is the discriminator loss, and  $L_{\text{cyc}}(G, F)$  is the cycle consistency loss in the overall objective function of the cycle-consistent adversarial network:

$$L(G, F, D_X, D_Y) = L_{\text{GAN}}(G, D_Y, X, Y) + L_{\text{GAN}}(F, D_X, Y, X) + \lambda L_{\text{cyc}}(G, F) \quad (1)$$

### 3.3. Intrusion Detection Network Model

The cycle-consistent adversarial network in this paper consists of two discriminators and two generators, all implemented using the same multilayer perceptron (MLP) network structure for training. The generator includes three hidden layers with 128, 256, and 512 neurons, respectively, and the input and output layers have equal numbers of neurons. The discriminator has two hidden layers with 512 and 256 neurons, respectively, and the output layer has 1 neuron. The two generators achieve data translation from the source domain to the target domain and vice versa. The two discriminators judge whether the data belong to the source or target domain.

In this paper, the intrusion detection dataset is divided into normal traffic dataset  $X : \{x_i\}_{i=1}^N$  and anomalous traffic dataset  $Y : \{y_j\}_{j=1}^M$ , which have a non-paired relationship. The goal of the network model is to learn a mapping  $G_{X \rightarrow Y} : X \rightarrow Y$  so that the generator can continuously optimize to eventually translate samples  $X$  to  $Y$ ; meanwhile, it learns an inverse mapping  $D_{G_{Y \rightarrow X}} : Y \rightarrow X$  to reconstruct  $X$  from  $Y$ ,  $G_{Y \rightarrow X}(G_{X \rightarrow Y}(X)) \approx X$ . Discriminators  $D_X$  and  $D_Y$  are introduced, where  $D_X$  distinguishes between data  $\{x\}$  and  $\{G_{Y \rightarrow X}(y)\}$ , and  $D_Y$  distinguishes between  $\{y\}$  and  $\{G_{X \rightarrow Y}(x)\}$ . To ensure that the core content is transferred during translation instead of just the type, a cycle consistency loss function  $L_{\text{cyc}}(G_{X \rightarrow Y}, G_{Y \rightarrow X})$  is added to preserve the key information of  $X$ . Data from other domains do not participate in the translation, and discriminators  $E$  and  $F$  distinguish target domain data from other domain data by learning the features of Other:  $\{\text{other}_i\}_{i=1}^N$ . The model training process is illustrated in Figure 2.

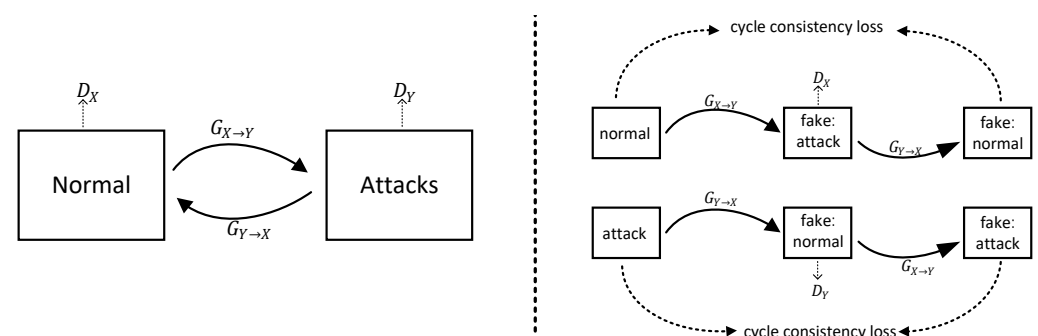


Figure 2. Model training process diagram.

The mean squared error (MSE) and L1 loss functions are used in this paper. MSE is a commonly used loss function in regression tasks that measures the average squared difference between the predicted and actual values. A lower MSE value indicates smaller differences between predicted and true values, and better model performance.

$$\text{MSE} = \frac{1}{n} \sum (x_i - y_i)^2 \quad (2)$$

The L1 loss function is known as minimizing absolute error. It has good robustness and is not overly affected by data with large errors. In this paper, the cycle consistency loss function  $L_{\text{cyc}}$  is represented using the L1 loss function.

$$\text{MAE} = \frac{1}{n} \sum |x_i - y_i| \quad (3)$$

While the generators translate between the target and source domains, the key information should not be lost. Therefore, this paper chooses to train the two generators together with the cycle consistency loss function.

$$L_G(G_{X \rightarrow Y}, G_{Y \rightarrow X}, D_X, D_Y) = \text{MSE}(D_X(G_{Y \rightarrow X}(y)), 1) + \text{MSE}(D_Y(G_{X \rightarrow Y}(x)), 1) \\ + \text{MAE}(G_{Y \rightarrow X}(G_{X \rightarrow Y}(x)), x) + \text{MAE}(G_{X \rightarrow Y}(G_{Y \rightarrow X}(y)), y) \quad (4)$$

The loss function of discriminator  $D_X$  is used to train discriminator  $D_X$ 's ability to distinguish between normal data and data of other types.

$$L_{D_X} = \text{MSE}(D_X(x), 1) + \text{MSE}(D_X(y), 0) \\ + \text{MSE}(D_X(G_{Y \rightarrow X}(y)), 0) + \text{MSE}(D_X(\text{other}), 0) \quad (5)$$

The loss function of discriminator  $D_Y$  is used to train discriminator  $D_Y$ 's ability to distinguish between anomalous data and data of other types.

$$L_{D_Y} = \text{MSE}(D_Y(y), 1) + \text{MSE}(D_Y(x), 0) \\ + \text{MSE}(D_Y(G_{X \rightarrow Y}(x)), 0) + \text{MSE}(D_Y(\text{other}), 0) \quad (6)$$

Through this method, the ability of generator  $G_{X \rightarrow Y}(x)$  to convert normal data into anomalous data can be enhanced, thereby expanding the anomalous data training set. At the same time, it also enhances generator  $G_{Y \rightarrow X}(y)$ 's ability to convert anomalous data into normal data, expanding the normal dataset.

After training, discriminators  $D_X$  and  $D_Y$  can distinguish between normal and anomalous data on the test set. For test data, discriminators  $D_X$  and  $D_Y$  are used to judge the category of traffic data, respectively. If  $D_X(\text{data}) > D_Y(\text{data})$ , the data are judged as normal; if  $D_X(\text{data}) < D_Y(\text{data})$ , they are judged as anomalous.

As indicated in the Algorithm 1 provided, before inputting data into the model, the training set is first divided into normal data  $X : \{x_i\}_{i=1}^N$ , a specific type of anomaly data  $Y : \{y_i\}_{i=1}^N$ , and other types of anomaly data  $\{\text{other}_i\}_{i=1}^N$ . Then, the generator  $G_{X \rightarrow Y}$ , inverse mapping  $G_{Y \rightarrow X}(y)$ , and discriminators  $D_X$  and  $D_Y$  are defined. The dataset is then fed into the model, where the parameters of generators  $G_{X \rightarrow Y}$  and  $G_{Y \rightarrow X}(y)$  are optimized using optimizer  $L_G$ , and the parameters of discriminators  $D_X$  and  $D_Y$  are optimized using loss functions  $L_{G_X}$  and  $L_{G_Y}$ . In step 2, two already trained generators are used for data augmentation to expand the dataset. In step 3, the trained dataset is used for data classification, where if  $D_X(\text{Data}) > D_Y(\text{Data})$ , the data are classified as normal; if  $D_X(\text{Data}) < D_Y(\text{Data})$ , they are classified as anomalous data.

**Algorithm 1** CycleGANIDS Data training

- 
- 1: Input:  $X$  and  $Y$ , Other ( $X : \{x_i\}_{i=1}^N, Y : \{y_j\}_{j=1}^N, \{\text{other}_i\}_{i=1}^N$ )
  - 2: Train: Generator ( $G_{X \rightarrow Y}$ ), Inverse mapping ( $G_{Y \rightarrow X}(y)$ ), Determiner ( $D_X$  and  $D_Y$ )
  - 3: Step1 Training network
  - 4: **while**  $i < \text{iterations}$  **do**
  - 5: Optimize the parameters of generators  $G_{X \rightarrow Y}$  and  $G_{Y \rightarrow X}(y)$ :  $L_G(G_{X \rightarrow Y}, G_{Y \rightarrow X}, D_X, D_Y)$
  - 6: Optimize the parameters of discriminator  $D_X$ :  $L_{D_X}$
  - 7: Optimize the parameters of discriminator  $D_Y$ :  $L_{D_Y}$
  - 8: **end while**
  - 9: Step2 Using generative networks for data augmentation and expansion of datasets
- $$g\_back = G_{X \rightarrow Y}(x), g\_normal = G_{Y \rightarrow X}(y),$$
- 10: Step3 Use judgment network
  - 11: When data are fed into the discriminator, if  $D_X > D_Y$  the data are normal, if  $D_X < D_Y$  the data are anomalous.
- 

**3.3.1. Data Augmentation of the Model**

Data augmentation refers to techniques that transform or make small modifications to existing data to synthesize new data, thereby expanding the dataset capacity. Data augmentation is commonly used to alleviate insufficient data issues in deep learning, and has been widely applied in image and natural language processing, expanding to intrusion detection [46]. Domestic and foreign researchers have adopted various data augmentation techniques, such as adding noise, rotating, flipping, cropping images, etc. [21]. However, existing methods are limited to processing image and speech data, and cannot effectively expand network traffic data.

To address the above issues, the model proposed in this research adopts CycleGAN technology to achieve conversion from normal traffic to attack traffic, thereby generating more diverse attack data. This method not only expands the dataset scale, but also enhances the model's ability to detect new attacks. The inspiration comes from the infection mechanism of computer viruses [47]: after being infected, the computer loses normal functionality due to some reason and is controlled by the virus to attack other computers, but can resume normal operation after cleanup. Similarly, this paper maps the "infection" and "cleanup" processes to generators  $D_X$  and  $D_Y$ . Generator  $D_X$  can infect normal data to expand the anomalous dataset, while inverse generator  $D_Y$  can purify anomalous data to generate new normal data, expanding the normal dataset. This traffic conversion based on adversarial networks can effectively augment the data needed for intrusion detection systems. The data augmentation process is illustrated in Figure 3.

**3.3.2. Model Discrimination**

Intrusion detection is the core functionality of the model in this paper, and it is used to monitor and detect potential intrusive behaviors. As an active security protection technology, intrusion detection can monitor internal attacks, external attacks, and misoperations in real time and take interception and response measures before the network system is threatened.

The discrimination process of this model has two approaches. The first approach is used to distinguish between a single attack type and normal data, for example, training a single discriminator to differentiate between normal data and Back attack, or between normal data and Pod attack. The second approach involves training multiple discriminators, with each discriminator corresponding to one attack type. Then, unknown data are fed sequentially into the multiple discriminators, and the discriminator associated with the maximum value is output, thereby determining the attack type of the unknown traffic.

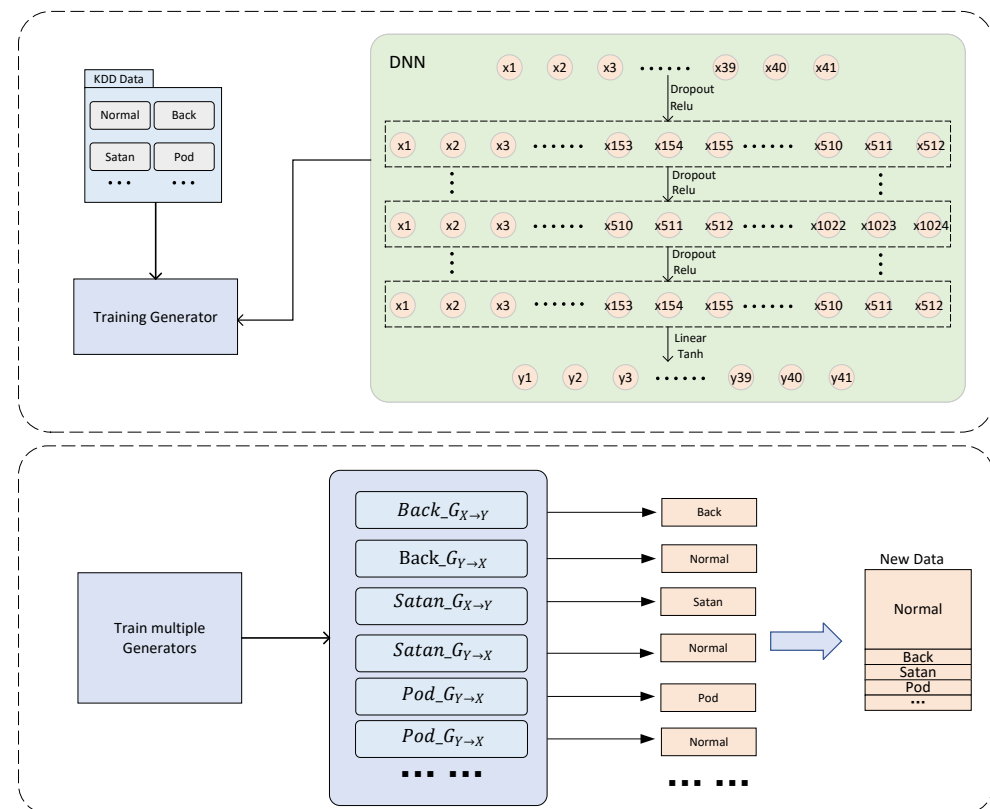


### (1) Single-category attack detection

For detecting a single type of attack traffic, such as backdoor attack, the model trains an adversarial discriminator  $D_y$ . The discrimination objective of  $D_y$  is to distinguish between normal traffic  $X$  and anomalous traffic  $Y$ , where  $Y$  refers specifically to backdoor attack traffic. Through adversarial learning,  $D_y$  obtains feature expressions of normal traffic to identify differences between normal traffic  $X$  and traffic  $Y$  containing backdoor attack features. After training,  $D_y$  can discriminate new unknown network traffic, judging whether anomalous backdoor attack features exist based on its determination.

### (2) Multi-classification attack detection

For detecting multiple types of attacks, the model trains multiple adversarial discriminators  $\{D_1, D_2, \dots, D_n\}$ , with each discriminator  $D_i$  corresponding to a known attack type  $Y_i$ . During testing, new unknown network traffic is fed sequentially into each adversarial discriminator  $D_i$  for judgement. The discrimination probabilities of different  $D_i$ s are compared, and the attack type corresponding to the discriminator with maximum probability  $P_{\max}$  is selected as the most likely attack type for that traffic flow. The detection process is illustrated in Figure 4.

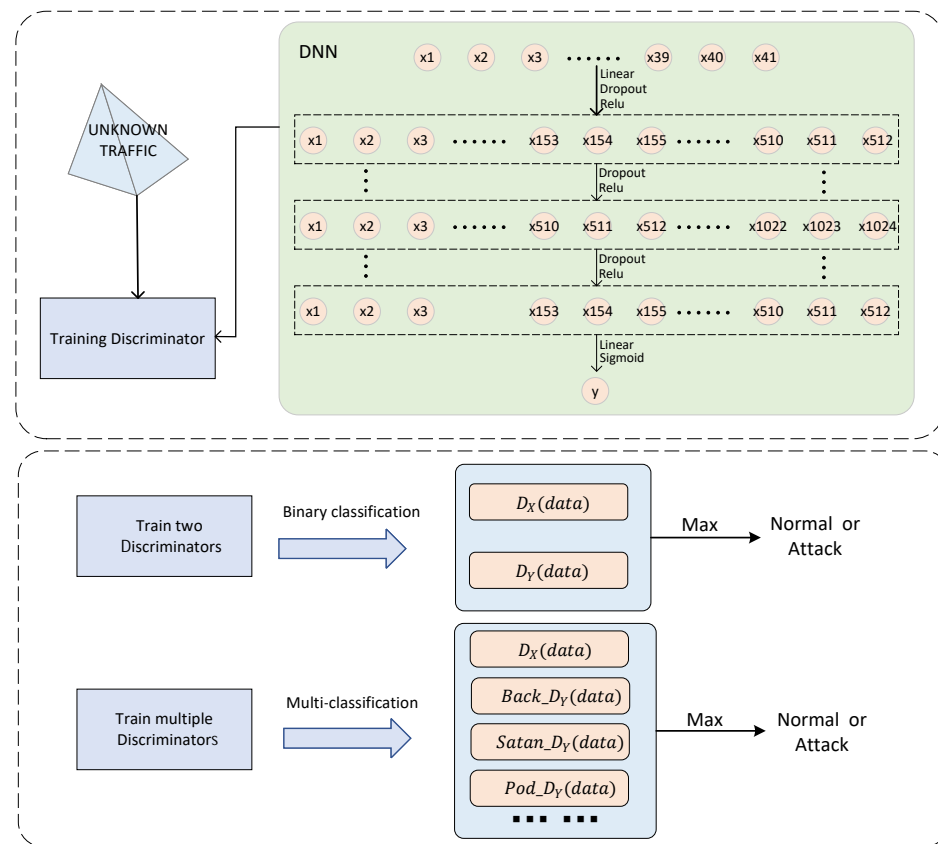


**Figure 3.** Data augmentation.

By training discriminators that distinguish between normal traffic and various attack traffic types, the model can detect known and zero-day attacks that may exist in unknown traffic. Compared to simply matching attack signatures, this adversarial deep learning discriminator-based approach can better detect complex network intrusion behaviors.

Suppose a cloud service provider is under a DDoS attack. Attackers use a large number of zombie computers to send a massive amount of malicious traffic to the servers of the cloud service provider, exhausting the bandwidth and resources of the servers and preventing normal users from accessing the cloud service. A CycleGAN-based intrusion detection system can be employed to detect such DDoS attacks. The system initially monitors and analyzes the traffic entering the network of the cloud service provider in real

time. It utilizes a pre-trained discriminator to classify the traffic, distinguishing between normal and anomalous traffic.



**Figure 4.** Classification detection.

In this scenario, the traffic sent by the attackers might exhibit certain characteristics, such as a high volume of requests from geographically diverse IP addresses, abnormally high request frequencies, and targets concentrated on specific services or ports. The system uses the discriminator model to recognize and analyze these features, identifying traffic that may likely be part of a DDoS attack.

Furthermore, the system can collect anomalous traffic, capturing its characteristics, and through CycleGAN's data augmentation transformation technology, learn the features of similar anomalous traffic for better future detection and analysis.

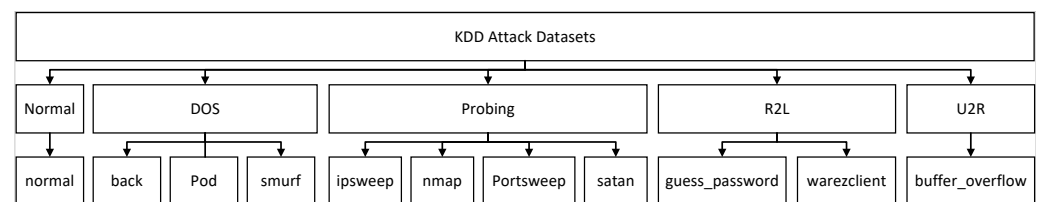
Once the system detects an abnormal traffic pattern, it immediately takes measures to address it. For example, the system can automatically tag the attacking traffic, diverting it to a dedicated firewall or traffic scrubbing equipment for further analysis and mitigation. Additionally, the system can alert network administrators and record detailed information about the attack for subsequent investigation and analysis.

#### 4. Performance Analysis

In this section, we mainly discuss the experiments we conducted to verify the binary anomalous detection capabilities of the intrusion detection model. In the experiments, three public datasets, including KDD, CIC-DDOS2019, CIC-IDS2018 and SR-BH 2020, were used, and the data preprocessing process is shown in detail. Next, the metrics used to evaluate model performance are introduced, and the performance results of the model are presented. Finally, through comparison with the experimental results of LSTM, CNN, MLP, and other models, the superiority of this model in performance is validated.

#### 4.1. Dataset

The first evaluation dataset used in this experiment is from the Third International Knowledge Discovery and Data Mining Tools Competition in 1999, which aimed to build robust intrusion detection systems. The dataset simulates 9 weeks of network connection and system audit data to mimic various user types and different network traffic and attack methods, making it close to real network environments. The dataset contains four anomaly types: DOS, Probing, R2L, and others. Each traffic sample has 41 features, where 1–9 represent basic TCP connection features, 10–22 are content features of TCP connections, 23–31 are time-based network traffic statistical features calculated within a 2 s time window, and 32–41 are host-based network traffic statistical features used to evaluate attacks lasting more than two seconds, as shown in Figure 5.



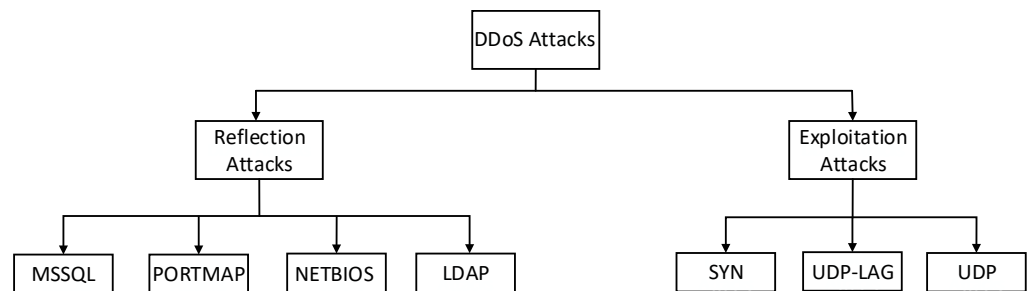
**Figure 5.** KDD99 dataset attack distribution.

The second evaluation dataset used in this experiment is the CIC-IDS2018 dataset (2018 Intrusion Detection Evaluation Dataset) developed by the Canadian Institute for Cybersecurity (CIC). The dataset provides raw data (PCAPs) as well as network traffic analysis results based on timestamps, source IP, destination IP, source port, destination port, protocol, and attack labels. The dataset includes abstracted behavior of twenty-five users, based on HTTP, HTTPS, FTP, SSH, and email protocols. Brute force attack types include FTP, SSH, DoS, Heartbleed, Web, infiltration, botnet, and DDoS. The table summarizes the traffic information recorded each day. In this study, only samples from the Friday—2 March 2018, Wednesday—14 February 2018, and Friday—16 February 2018 datasets were used for analysis, as shown in Table 2 [48].

**Table 2.** CIC-IDS2018 Dataset Attack Distribution.

File Name (Record Date)	Attack Type
Thursday—1 March 2018	Benign, Infiltration
Friday—2 March 2018	Benign, Bot
Wednesday—14 February 2018	Benign, SSH-Bruteforce, FTP-BruteForce
Thursday—15 February 2018	Benign, DoS-GoldenEye, DoS-Slowloris
Friday—16 February 2018	Benign, DoS attack-hulk, DoS attacks-SlowHTTPTest
Tuesday—20 February 2018	Benign, DDoS attacks-LOIC-HTTP, DDoS-LOIC-UDP
Wednesday—21 February 2018	Benign, DDoS-LOIC-UDP, DDoS-HOIC
Thursday—22 February 2018	Benign, Brute Force-Web, Brute Force-XSS, SQL Injection
Friday—23 February 2018	Benign, Brute Force-Web, burte Force-XSS, SQL Injection
Wednesday—28 February 2018	Benign, Infiltration

The third evaluation dataset used in this experiment is the recently released CIC-DDoS2019 DDoS evaluation dataset (2019) from the Canadian Institute for Cybersecurity (CIC). The dataset underwent network traffic analysis using the CICFLOWMeter-V3 tool, and the results contain traffic tokens based on timestamps, source IP, destination IP, etc. The dataset covers various types of DOS attacks found in real network environments, including LDAP, MSSQL, NetBIOS, Portmap, SYN, UDP, and UDALag, with a total of 88 features. The CIC-DDoS2019 dataset is publicly available on the Canadian Institute for Cybersecurity website in PCAP and CSV flow format, and can be used to evaluate the ability to detect the latest DDoS attacks, as shown in Figure 6.



**Figure 6.** CIC-DDOS2019 dataset attack distribution.

The final dataset utilized in this experiment is the SR-BH 2020 dataset, which is designed to test and evaluate different algorithms and models. This dataset consists of Web requests collected from a Wordpress Web server installed on a virtual machine and exposed to the Internet during the period of 12 July 2020. It is a specialized multi-label dataset for Web attack detection, comprising 907,814 requests, of which 525,195 are normal requests and 382,619 are anomalous requests. Each record includes 24 distinct features and a set of 13 labels. Table 3 below provides detailed information about the classification of Web requests under specific CAPEC categories.

**Table 3.** Number of Web requests by CAPEC classification.

CAPEC Classification	Number of Web Requests	% of Total Requests
000-Noraml	525,195	57.85%
272-Protocol Manipulation	9153	1.00%
242-Code Injection	15,827	1.74%
88-OS Command Injection	7482	0.82%
126-Path Traversal	20,992	2.31%
66-SQL Injection	250,311	27.57%
16-Dictionary-based Password Attack	1847	0.20%
310-Scanning for Vulnerable Software	2718	0.30%
153-Input Data Manipulation	2272	0.25%
274-HTTP Verb Tampering	5437	0.60%
194-Fake the source of data	56,145	6.18%
34-HTTP Response Splitting	19,738	2.17%
33-HTTP Request Smuggling	1059	0.12%
TOTAL	918,176	

#### 4.2. Data Preprocessing

Before model training, training sets for three datasets need to be properly processed to improve model performance. Specifically, character features first need to be processed, since character data cannot be directly input into neural networks, such as the “LDAP” in labels and “xxx-xxx-xxx” in time features. Next, numerical data are normalized.

1. The KDD99 and CICDDOS2019 datasets have 41 and 88 features, respectively, with the final label features being character type. For ease of neural network training, these label features need to be removed or converted to the numeric type. The CICIDS2018 dataset has a timestamp as the third feature, containing the year, month, and day, so this character feature was removed in this experiment.

2. Before data modeling and analysis, data are usually standardized to eliminate the influence of different feature dimensions, and the standardized data are used for analysis. The purpose of standardization is to make each feature have similar magnitudes and be dimensionless. Data normalization is mainly used to solve the problem of features with different properties, because directly summing indicators of different properties cannot correctly reflect their combined effect. Through standardization, the data properties of different features can be adjusted to make their impacts on the evaluation results more consistent, in order to obtain the correct model.

$$x' = \frac{x - \bar{x}}{\sigma} \quad (7)$$

3. Before data analysis, normalization is also required to map feature values into the range of 0–1, in order to compare and weight features of different magnitudes. Normalization is a method to simplify computation by transforming the original dimensional expression into a dimensionless expression, making it a pure quantity. This helps process features with different units or magnitudes for unified calculation and analysis.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (8)$$

#### 4.3. Evaluation Indicators

This paper adopts six performance metrics—precision, recall, F1-score, accuracy, false-negative rate (FNR), and false-positive rate (FPR)—to evaluate model performance. These metrics are calculated based on four measurements: true positive (TP), true negative (TN), false positive (FP), and false negative (FN):

True Positive (TP): correctly predict positive samples as positive classes.

True Negative (TN): correctly predict negative samples as negative classes.

False Positive (FP): mispredict negative samples as positive classes.

False Negative (FN): mispredict positive samples as negative classes.

The calculation formula for accuracy is the proportion of correctly predicted samples to the total number of samples:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \quad (9)$$

The calculation formula for accuracy is the proportion of correctly predicted samples to the total number of samples:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (10)$$

The recall rate refers to the probability of correctly predicting positive samples among all positive samples, which is the ratio of correctly predicting the number of positive samples to all positive samples. It reflects the proportion of actual positive classes predicted as positive classes, and its formula is as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (11)$$

The F1-score is a metric used in statistics to measure the accuracy of binary classification (or multi-task binary classification) models. It takes into account both the precision and recall of the classification model.

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (12)$$

The false-negative rate (FNR) and false-positive rate (FPR) are important metrics to measure the performance of intrusion detection systems. The FNR refers to the probability that the system mistakenly identifies normal behavior as intrusive under normal conditions. The FPR refers to the probability that the system fails to correctly identify intrusive behavior when intrusion is present.

$$\begin{aligned} \text{FNR} &= \frac{FN}{TP + FN} = 1 - \text{Precision} \\ \text{FPR} &= \frac{FP}{TP + FP} = 1 - \text{Recall} \end{aligned} \quad (13)$$

#### 4.4. Experimental Analysis

We divided the dataset's normal and abnormal samples into training, validation, and testing sets, respectively, with the number of abnormal samples being the same as



the number of normal samples. The test set and training set in this study were both obtained through random sampling from the KDD99, CIC-IDS2018, CIC-DDOS2019, and SR-BH 2020 datasets. For each training, a specific number of anomalous samples and the same number of normal samples were randomly selected from the training set for model training, while the validation set was used to fine-tune model parameters, and finally the test set was used to evaluate model performance. For example, 3000 “Neptune” attack samples and 3000 “normal” samples were used to train the model, improving the model’s ability to detect neptune attacks and perform normal–anomalous data conversion. Then, 500 “normal” samples and 500 “neptune” samples were extracted as the test set to evaluate model performance, as shown in Tables 4–7.

**Table 4.** KDD99 dataset division.

Dataset	Label	Training Set	Validation Set	Test Set
KDD99	back	1900	302	302
	neptune	3000	500	500
	guess_passwd	3000	500	500
	pod	230	30	30
	teardrop	900	80	80
	Portssweep	900	140	140
	ipsweep	1000	246	246
	satan	1300	285	285
	nmap	205	25	25
	warezclient	800	210	210
	buffer_overflow	24	5	5
	smurf	3000	500	500
	Normal	16,259	2823	2823

**Table 5.** CIC-DDOS2019 dataset division.

Dataset	Label	Training Set	Validation Set	Test Set
CIC-DDOS2019	LDAP	3500	500	500
	UDP	3500	500	500
	MSSQL	4000	500	500
	NetBIOS	4000	500	500
	Portmap	4000	500	500
	UDPLag	1100	250	250
	SYN	4000	500	500
	Normal	24,100	3250	3250

**Table 6.** CIC-IDS2018 dataset division.

Dataset	Label	Training Set	Validation Set	Test Set
CIC-IDS2018	Bot	3500	800	800
	SSH-Bruteforce	3500	800	800
	FTP-Bruteforce	3500	800	800
	Dos attacks-Hulk	3500	800	800
	DoS attacks-SlowHTTPTest	3500	800	800
	Benign	17,500	4000	4000

**Table 7.** SR-BH 2020 dataset division.

Dataset	Label	Training Set	Test Set
SR-BH 2020	272-Protocol Manipulation	4000	500
	242-Code Injection	4000	500
	88-OS Command Injection	4000	500
	126-Path Traversal	4000	500
	66-SQL Injection	4000	500
	16-Dictionary-based Password Attack	1300	500
	310-Scanning for Vulnerable Software	2200	500
	153-Input Data Manipulation	1700	500
	274-HTTP Verb Tampering	4000	500
	194-Fake the Source of Data	4000	500
	34-HTTP Response Splitting	4000	500
	33-HTTP Request Smuggling	550	500

Through multiple experiments adjusting different parameters, the optimal model parameter settings were obtained. During the experiments, it should be noted that due to

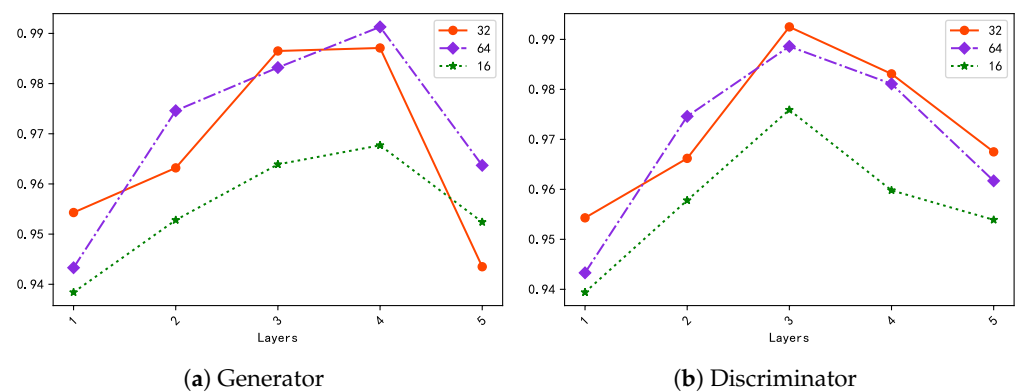
the different feature dimensions of the three datasets, the number of neurons in the model's input layer needed to be adjusted accordingly before model training. For the KDD99, CIC-IDS2018, CIC-DDOS2018, and SR-BH 2020 datasets, the input layer dimensions of the model were set to 41, 76, 79, and 20, respectively. Next, the generators and discriminators of the model were trained and tested for performance. The results found that when the sample size was greater than 1500, a batch size of 64 gave the optimal model performance; otherwise, a batch size of 32 was better, as shown in Table 8.

**Table 8.** Model parameters.

Parameter	Generator	Discriminator
Batch Size	64 or 32	64 or 32
Layers	3	4
Dropout	0.5	0.5
Learn Rate	0.00001	0.00001
Epoch	50	50

The experiments discovered that high-quality discriminators and generators can provide high-quality feature expressions to improve model computation speed and avoid gradient vanishing. Relu activation functions were used between the hidden layers of the discriminators and generators. Through multiple experiments adjusting the number of hidden layers, the results showed that four layers for the generator and three layers for the discriminator achieved the best balance between accuracy and computational resources.

Taking the KDD99 dataset as an example, this study experimented with the relationship between discriminator and generator performance and the number of network layers under different batch sizes. The results show that when the batch size is 16, 32, and 64, the generator with four layers and the discriminator with three layers strike the most suitable balance between accuracy and efficiency in terms of network structure selection, as shown in Figure 7.



**Figure 7.** Performance of generator and discriminator batch sizes.

#### 4.5. Experimental Result

First, this model was used to detect single anomaly types in the KDD99 and CIC-DDOS2019 datasets to evaluate model performance in anomalous sample identification and data augmentation. The results show that the model demonstrated good performance in detecting various anomaly types. On the KDD99 test set, the model accuracy (ACC) reached 100% at best, indicating perfect detection capability, while the lowest was 98.23% for nmap attack detection. On the CIC-DDOS2018 test set, the lowest detection accuracy of the model was 94.5% for SYN attack detection, while the highest was 99.81% for LDAP attack detection. However, on the ICI-IDS2018 test set, the model performed relatively poorly. Although precision reached 100%, the recall and overall accuracy (ACC) were only about 93% for FTP-Bruteforce attack detection, as shown in Tables 9–11.

**Table 9.** Single anomaly detection results for KDD99 dataset.

Dataset	Label	Accuracy	Precision	Recall	F1-Score
KDD99	back	1.0	1.0	1.0	1.0
	neptune	1.0	1.0	1.0	1.0
	guess_passwd	1.0	1.0	1.0	1.0
	pod	1.0	1.0	1.0	1.0
	teardrop	1.0	1.0	1.0	1.0
	PortswEEP	0.9856	1.0	0.9712	0.9854
	ipsweep	0.9979	1.0	0.9959	0.9969
	satan	0.9965	1.0	0.9929	0.9964
	nmap	0.9823	1.0	0.9611	0.9796
	warezclient	1.0	1.0	1.0	1.0
	buffer_overflow	1.0	1.0	1.0	1.0
	smurf	1.0	1.0	1.0	1.0

**Table 10.** CIC-DDOS single classification detection results.

Dataset	Label	Accuracy	Precision	Recall	F1-Score
CIC-DDOS2019	LDAP	0.9981	1.0	0.9962	0.9979
	UDP	0.9960	0.9979	0.9940	0.9959
	MSSQL	0.9890	0.9822	0.9960	0.9890
	NetBIOS	0.9889	0.9803	0.9980	0.9891
	Portmap	0.9980	1.0	0.9960	0.9979
	UDPLag	0.9670	0.9499	0.9860	0.9676
	SYN	0.9450	1.0	0.8900	0.9418

**Table 11.** CIC-IDS2018 single classification detection results.

Dataset	Label	Accuracy	Precision	Recall	F1-Score
CIC-IDS2018	Bot	0.9956	0.9913	0.9962	0.9955
	SSH-Bruteforce	0.9713	1.0	0.9425	0.9704
	FTP-Bruteforce	0.9334	1.0	0.9334	0.9657
	Dos attacks-Hulk	1.0	1.0	1.0	1.0
	DoS attacks-SlowHTTPTest	1.0	1.0	1.0	1.0

As shown in Table 12, our model conducted experiments on the SR-BH 2020 dataset targeting classical Web attacks. Due to the limited number of instances for labels such as Dictionary-based Password Attack, Scanning for Vulnerable Software, Input Data Manipulation, and HTTP Request Smuggling, each with fewer than 3000 instances, the test set for each label comprised 500 samples. The model performed exceptionally well for the single-label CAPEC classification within the SR-BH 2020 dataset. For most categories, it achieved extremely high levels of accuracy, precision, recall, F1-score, and MCC value, with some categories even reaching perfect scores. This indicates that the model possesses robust capability to identify and classify various attack patterns, achieving a good balance between precision and comprehensiveness. Overall, the model demonstrated satisfactory results in the single-label CAPEC classification task, providing reliable tools and support for attack detection and defense in the security domain.

**Table 12.** SR-BH 2020 dataset single-label CAPEC classification.

Dataset	Label	Accuracy	Precision	Recall	F1-Score
SR-BH 2020	272-Protocol Manipulation	0.907	0.925	0.913	0.916
	242-Code Injection	0.898	0.901	0.896	0.889
	88-OS Command Injection	0.962	1.0	0.924	0.960
	126-Path Traversal	0.998	1.0	0.996	0.998
	66-SQL Injection	0.993	0.990	0.996	0.993
	16-Dictionary-based Password Attack	0.973	0.949	1.0	0.974
	310-Scanning for Vulnerable Software	0.999	0.998	1.0	0.999
	153-Input Data Manipulation	0.999	1.0	0.998	0.999
	274-HTTP Verb Tampering	0.993	0.995	0.990	0.993
	194-Fake the Source of Data	0.980	0.962	1.0	0.981
	34-HTTP Response Splitting	0.987	1.0	0.974	0.987
	33-HTTP Request Smuggling	1.0	1.0	1.0	1.0

In addition, binary classification experiments were conducted on the KDD99, CIC-IDS2018, and CIC-DDOS2019 datasets using this model, and performance was compared

with mainstream deep learning models like LSTM AE, ResNet-101, and DNN on the test sets. The results show that in binary classification tasks, the performance of this model decreased to some extent compared to single anomaly detection, but was still superior. Specifically, on the KDD99 dataset, this model significantly outperformed the other three comparison methods in terms of precision, recall, F1-score, and accuracy. On the CIC-DDOS2019 dataset, the accuracy (ACC), recall (Rec), and F1-score of this model were better than other methods, but the precision was low, indicating the model may have misjudged anomalous samples as normal, leading to higher false positives (FPs). On the CIC-IDS2018 dataset, the model's performance was far from its performance in single anomaly detection experiments. Although the metrics were higher than those in CNN and MLP models, the accuracy and recall were lower than those in the LSTM model. The classification results are shown in Tables 13–15.

**Table 13.** KDD99 dataset II classification.

Algorithm	Accuracy	Precision	Recall	F1-Score
LSTM [49]	0.9651	0.9723	0.9742	0.9768
CNN	0.9655	0.9872	0.9764	0.9633
MLP	0.9544	0.9682	0.9671	0.9534
Our Approach	0.9981	0.9969	1.0	0.9953

**Table 14.** CIC-DDOS2019 dataset II classification.

Algorithm	Accuracy	Precision	Recall	F1-Score
LSTM [50]	0.8850	0.8810	0.8780	0.8700
RNN	0.9642	0.9133	0.9340	0.9586
MLP	0.9250	0.8440	0.9420	0.8900
Our Approach	0.9779	0.9382	0.9780	0.9418

**Table 15.** Classification of CIC-IDS2018 dataset II.

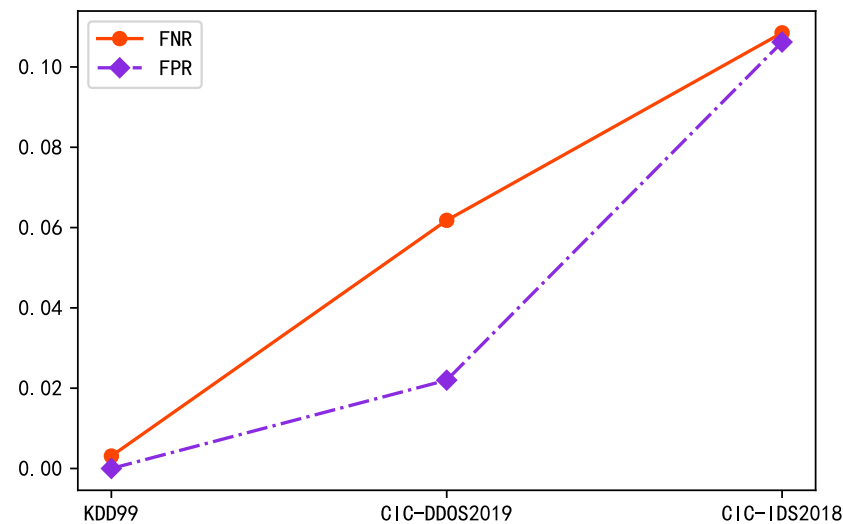
Algorithm	Accuracy	Precision	Recall	F1-Score
LSTM [47]	0.9265	0.7862	0.8971	0.8381
CNN	0.8297	0.6260	0.4852	0.5471
MLP	0.8867	0.8746	0.8912	0.8827
Our Approach	0.8925	0.8915	0.8938	0.8926

As indicated in Table 16, we conducted an analysis and summary of the performance metrics for various models on the SR-BH 2020 dataset. These models include the Two-phase MultiOutput CatBoost, Customized model CatBoost, Two-phase MultiOutput LightGBM, Single-phase Clas.Chain LightGBM, Single-phase Clas.Chain CatBoost, Customized model LightGBM, Single-phase Binary Relevance CatBoost, Two-phase Binary Relevance CatBoost, Single-phase Binary Relevance LightGBM, Two-phase Binary Relevance LightGBM, and our model. Our model demonstrated exceptional performance on the SR-BH 2020 dataset, exhibiting high levels of accuracy, precision, recall, and F1-score. This summary aids in assessing the suitability of these models for specific tasks and serves as a reference for selecting the best model.

This research used precision and recall to calculate the false-negative rate (FNR) and false-positive rate (FPR) of the model on the KDD99, CIC-IDS2018, and CIC-DDOS2019 datasets. Through detailed study, we found that the FNR and FPR of the model were close on the KDD99 and CIC-IDS2018 datasets, while the FPR was significantly higher than the FNR on the CIC-DDOS2019 dataset. However, they remained below 10%, and even below 0.01% on the KDD99 dataset. This series of experimental results shows that the FNR and FPR of the model are still maintained at a relatively low level, as shown in Figure 8.

**Table 16.** Metrics in the SR-BH 2020 dataset.

Method	Accuracy	Precision	Recall	F1-Score
Two-phase MultiOutput CatBoost [51]	0.88445	0.89557	0.88829	0.88912
Customized model CatBoost	0.88436	0.88863	0.88790	0.88501
Two-phase MultiOutput LightGBM	0.88095	0.89137	0.88641	0.88615
Single-phase Clas.Chain LightGBM	0.87224	0.87610	0.87360	0.87227
Single-phase Clas.Chain CatBoost	0.87213	0.87876	0.87343	0.87171
Customized model LightGBM	0.85888	0.86108	0.86270	0.85874
Single-phase Binary Relevance CatBoost	0.84939	0.90279	0.85734	0.87221
Two-phase Binary Relevance CatBoost	0.85201	0.90515	0.85508	0.87680
Single-phase Binary Relevance LightGBM	0.84419	0.89927	0.85112	0.87204
Two-phase Binary Relevance LightGBM	0.84782	0.90075	0.85049	0.87216
Ours	0.95150	0.91689	0.99300	0.95343

**Figure 8.** FNR and FPR of the dataset.

## 5. Discussion and Conclusions

Web3.0's extensive connectivity and data transmission provide potential attack channels for hackers, leading to serious issues such as personal privacy leakage, data breaches, and system crashes. Therefore, ensuring the security and privacy protection of Web3.0 is crucial. Intrusion detection systems enhance the security of Web3.0 by analyzing data sources such as network traffic and system logs to detect these attacks.

This study presents an intrusion detection model based on CycleGAN, which utilizes the CycleGAN network as its foundation and employs deep neural network (DNN) structured generators and discriminators. The interaction between the generators and discriminators endows the generators with powerful data augmentation capabilities, while the discriminators exhibit excellent detection abilities. The robust data augmentation capabilities of these two generator models partially address the scarcity of certain anomalous traffic data available for analysis by IDS, thereby enhancing the system's ability to document instances of malicious behavior. Simultaneously, the outstanding detection abilities of the two discriminators also partially address issues, such as inadequate feature extraction and inaccurate system classification results, typically encountered in traditional IDSs. The intrusion detection system based on CycleGAN makes significant contributions to handling the complexities of feature-rich, nonlinear traffic data in the Web3.0 environment.

To validate the detection performance of this approach, comprehensive performance evaluations were conducted on four widely used intrusion detection benchmark datasets: KDD99, CIC-IDS2018, CIC-DDoS2019, and SR-BH 2020 datasets. The results clearly demonstrate that the proposed deep learning intrusion detection method is feasible and holds practical application potential.

Future research can be directed towards several key areas. First of all, there is scope to improve model performance by exploring advanced deep learning techniques and algorithms. This exploration aims to improve the accuracy and efficiency of intrusion



detection systems. Secondly, it is crucial to extend the applicability of the model to different fields, such as industrial control systems and intelligent transportation systems. This extension expands the scope of the model and increases its usefulness in different industries. Finally, rigorous verification and practical application are essential. Working with IoT device manufacturers to deploy the model in a real IoT environment will validate its effectiveness and reliability in practical settings.

This study highlights the prospects of deep learning technology in the field of network security, encouraging further research and application to continuously enhance the level of network security and protect the digital world.

**Author Contributions:** Methodology, M.F.; Software, Y.W.; Formal analysis, Z.Y.; Investigation, Z.X.; Data curation, X.L.; Writing—review & editing, H.W.; Supervision, Z.K.; Funding acquisition, L.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by Fundamental Research Funds for the Central Universities, the University of International Relations (3262024T01, 3262024T25, 3262024T29), and the Teaching Reform and Innovation Project, University of International Relations (2023030, 2023029).

**Data Availability Statement:** Previously published articles were used to support this study and these prior studies, and datasets are cited at the relevant places within this article. The link to the datasets and the code for this paper are publicly available at the code address: <https://github.com/poshangcun13/CycleGAN-based-intrusion-detection.git>.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Dimitris, M.; Nikos, P.; John, A.; Baicun, W.; Lihui, W. Human centric platforms for personalized value creation in metaverse. *J. Manuf. Syst.* **2022**, *65*, 653–659.
2. Sean, Y.; Max, L. Web3.0 Data Infrastructure: Challenges and Opportunities. *IEEE Netw.* **2023**, *37*, 4–5.
3. Tidjon, L.N.; Frappier, M.; Mammari, A. Intrusion detection systems: A cross-domain overview. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3639–3681. [CrossRef]
4. De Filippi, P.; Mannan, M.; Reijers, W. The a legality of blockchain technology. *Policy Soc.* **2022**, *41*, 358–372. [CrossRef]
5. Shi, W.; Pallis, G.; Xu, Z. Edge computing [scanning the issue]. *Proc. IEEE* **2019**, *107*, 1474–1481. [CrossRef]
6. Dara, S.; Zargar, S.T.; Muralidhara, V. Towards privacy preserving threat intelligence. *J. Inf. Secur. Appl.* **2018**, *38*, 28–39. [CrossRef]
7. Cirillo, S.; Desiato, D.; Scalera, M.; Solimando, G. A Visual Privacy Tool to Help Users in Preserving Social Network Data. In Proceedings of the IS-EUD 2023: 9th International Symposium on End-User Development, Cagliari, Italy, 6–8 June 2023.
8. Michelena, Á.; Aveleira-Mata, J.; Jove, E.; Alaiz-Moretón, H.; Quintián, H.; Calvo-Rolle, J.L. Development of an Intelligent Classifier Model for Denial of Service Attack Detection. *Int. J. Interact. Multimed. Artif. Intell.* **2023**, *8*, 33. [CrossRef]
9. Li, K.; Cheng, L.; Teng, C.I. Voluntary sharing and mandatory provision: Private information disclosure on social networking sites. *Inf. Process. Manag.* **2020**, *57*, 102128. [CrossRef]
10. Cerruto, F.; Cirillo, S.; Desiato, D.; Gambardella, S.M.; Polese, G. Social network data analysis to highlight privacy threats in sharing data. *J. Big Data* **2022**, *9*, 19. [CrossRef]
11. Li, B.; Hu, W.; Qu, X.; Li, Y. A Novel Multi-Attack IDS Framework for Intelligent Connected Terminals Based on Over-the-Air Signature Updates. *Electronics* **2023**, *12*, 2267. [CrossRef]
12. Xuan, C.D.; Huong, D.; Nguyen, T. A novel intelligent cognitive computing-based APT malware detection for Endpoint systems. *J. Intell. Fuzzy Syst.* **2022**, *43*, 3527–3547. [CrossRef]
13. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
14. Available online: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html> (accessed on 25 March 2024).
15. Available online: <https://www.unb.ca/cic/datasets/ids-2018.html> (accessed on 25 March 2024).
16. Available online: <https://www.unb.ca/cic/datasets/ddos-2019.html> (accessed on 25 March 2024).
17. Available online: <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/OGOIXX> (accessed on 25 March 2024).
18. Rosenbaltt, F. *The Perceptron—A Perceiving and Recognizing Automation*; Cornell Aeronautical Laboratory: Buffalo, NY, USA, 1957.
19. Minsky, M.; Papert, S. An introduction to computational geometry. *Camb. Trass. HIT* **1969**, *479*, 104.
20. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
21. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]

22. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
23. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
24. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
25. RM, S.P.; Maddikunta, P.K.R.; Parimala, M.; Koppu, S.; Gadekallu, T.R.; Chowdhary, C.L.; Alazab, M. An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. *Comput. Commun.* **2020**, *160*, 139–149.
26. Creswell, A.; White, T.; Dumoulin, V.; Arulkumaran, K.; Sengupta, B.; Bharath, A.A. Generative adversarial networks: An overview. *IEEE Signal Process. Mag.* **2018**, *35*, 53–65. [\[CrossRef\]](#)
27. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
28. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to discover cross-domain relations with generative adversarial networks. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 1857–1865.
29. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [\[CrossRef\]](#)
30. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
31. Anderson, J.P. *Computer Security Threat Monitoring and Surveillance*; Technical Report; James P. Anderson Company: Kent, OH, USA, 1980.
32. Denning, D.E. An intrusion-detection model. *IEEE Trans. Softw. Eng.* **1987**, *SE-13*, 222–232. [\[CrossRef\]](#)
33. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [\[CrossRef\]](#)
34. Muhammad, G.; Hossain, M.S.; Garg, S. Stacked autoencoder-based intrusion detection system to combat financial fraudulent. *IEEE Internet Things J.* **2020**, *10*, 2071–2078. [\[CrossRef\]](#)
35. Yang, Y.; Zheng, K.; Wu, B.; Yang, Y.; Wang, X. Network intrusion detection based on supervised adversarial variational auto-encoder with regularization. *IEEE Access* **2020**, *8*, 42169–42184. [\[CrossRef\]](#)
36. Khare, N.; Devan, P.; Chowdhary, C.L.; Bhattacharya, S.; Singh, G.; Singh, S.; Yoon, B. Smo-dnn: Spider monkey optimization and deep neural network hybrid classifier model for intrusion detection. *Electronics* **2020**, *9*, 692. [\[CrossRef\]](#)
37. Tang, C.; Luktarhan, N.; Zhao, Y. SAAE-DNN: Deep learning method on intrusion detection. *Symmetry* **2020**, *12*, 1695. [\[CrossRef\]](#)
38. Al-Fawa'reh, M.; Al-Fayoumi, M.; Nashwan, S.; Fraihat, S. Cyber threat intelligence using PCA-DNN model to detect abnormal network behavior. *Egypt. Inform. J.* **2022**, *23*, 173–185. [\[CrossRef\]](#)
39. Thakkar, A.; Lohiya, R. Analyzing fusion of regularization techniques in the deep learning-based intrusion detection system. *Int. J. Intell. Syst.* **2021**, *36*, 7340–7388. [\[CrossRef\]](#)
40. Rao, K.N.; Rao, K.V.; Pvgd, P.R. A hybrid intrusion detection system based on sparse autoencoder and deep neural network. *Comput. Commun.* **2021**, *180*, 77–88.
41. Balamurugan, E.; Mehbodniya, A.; Kariri, E.; Yadav, K.; Kumar, A.; Haq, M.A. Network optimization using defender system in cloud computing security based intrusion detection system with game theory deep neural network (IDSGT-DNN). *Pattern Recognit. Lett.* **2022**, *156*, 142–151. [\[CrossRef\]](#)
42. Thakkar, A.; Lohiya, R. Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Inf. Fusion* **2023**, *90*, 353–363. [\[CrossRef\]](#)
43. Thakkar, A.; Lohiya, R. Attack classification of imbalanced intrusion data for IoT network using ensemble learning-based deep neural network. *IEEE Internet Things J.* **2023**, *10*, 11888–11895. [\[CrossRef\]](#)
44. Dunmore, A.; Jang-Jaccard, J.; Sabrina, F.; Kwak, J. A Comprehensive Survey of Generative Adversarial Networks (GANs) in Cybersecurity Intrusion Detection. *IEEE Access* **2023**, *11*, 76071–76094. [\[CrossRef\]](#)
45. Kumar, V.; Sinha, D. Synthetic attack data generation model applying generative adversarial network for intrusion detection. *Comput. Secur.* **2023**, *125*, 103054. [\[CrossRef\]](#)
46. Zhang, Y.; Liu, Q. On IoT intrusion detection based on data augmentation for enhancing learning on unbalanced samples. *Future Gener. Comput. Syst.* **2022**, *133*, 213–227. [\[CrossRef\]](#)
47. Bingu, R.; Jothilakshmi, S.; Srinivasu, N. An intelligent multiclass deep classifier-based intrusion detection system for cloud environment. *Concurr. Comput. Pract. Exp.* **2023**, *35*, e7840. [\[CrossRef\]](#)
48. Zhou, F.; Du, X.; Li, W.; Lu, Z.; Wu, J. NIDD: An intelligent network intrusion detection model for nursing homes. *J. Cloud Comput.* **2022**, *11*, 91. [\[CrossRef\]](#)
49. Staudemeyer, R.C. Applying long short-term memory recurrent neural networks to intrusion detection. *S. Afr. Comput. J.* **2015**, *56*, 136–154. [\[CrossRef\]](#)

50. Sayed, M.I.; Sayem, I.M.; Saha, S.; Haque, A. A Multi-Classifer for DDoS Attacks Using Stacking Ensemble Deep Neural Network. In Proceedings of the 2022 International Wireless Communications and Mobile Computing (IWCMC), Dubrovnik, Croatia, 30 May–3 June 2022; pp. 1125–1130.
51. Riera, T.S.; Higuera, J.R.B.; Higuera, J.B.; Herraiz, J.J.M.; Montalvo, J.A.S. A new multi-label dataset for Web attacks CAPEC classification using machine learning techniques. *Comput. Secur.* **2022**, *120*, 102788. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.