applied sciences

MDPI

Article

MCGAN: Modified Conditional Generative Adversarial Network (MCGAN) for Class Imbalance Problems in Network Intrusion Detection System

Kunda Suresh Babu and Yamarthi Narasimha Rao *

School of Computer Science and Engineering, VIT-AP University, Amaravathi 522237, India
* Correspondence: y.narasimharao@vitap.ac.in

Abstract: With developing technologies, network security is critical, predominantly active, and distributed ad hoc in networks. An intrusion detection system (IDS) plays a vital role in cyber security in detecting malicious activities in network traffic. However, class imbalance has triggered a challenging issue where many instances of some classes are more than others. Therefore, traditional classifiers suffer in classifying malicious activities and result in low robustness to unidentified glitches. This paper introduces a novel technique based on a modified conditional generative adversarial network (MCGAN) to address the class imbalance problem. The proposed MCGAN handles the class imbalance issue by generating oversamples to balance the minority and majority classes. Then, the Bi-LSTM technique is incorporated to classify the multi-class intrusion efficiently. This formulated model is experimented on using the NSL-KDD+ dataset with the aid of accuracy, precision, recall, FPR, and F-score to validate the efficacy of the proposed system. The simulation results of the proposed method are associated with other existing models. It achieved an accuracy of 95.16%, precision of 94.21%, FPR of 2.1%, and F1-score of 96.7% for the NSL-KDD+ dataset with 20 selected features.

Keywords: intrusion detection system; deep convolution generative adversarial network; class imbalance problem; NSL-KDD dataset; accuracy

check for updates

Citation: Babu, K.S.; Rao, Y.N. MCGAN: Modified Conditional Generative Adversarial Network (MCGAN) for Class Imbalance Problems in Network Intrusion Detection System. *Appl. Sci.* **2023**, *13*, 2576. https://doi.org/10.3390/app13042576

Academic Editor: Luis Javier García Villalba

Received: 11 January 2023
Revised: 1 February 2023
Accepted: 3 February 2023
Published: 16 February 2023

Copyright: © 2023 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

In recent years, the evolution of information technology and security protocols has increased exponentially in network traffic data [1]. Most computer applications are allied to cyberspace for efficient services of various applications such as browsing, social media, e-mails, etc. In addition, different security modules are invoked into network applications to tackle network intrusions. Network intrusions are unsolicited traffic behaviors that are prone to malicious attacks and are harmful to host networks. The hostile invasions are prone to various attacks, including denial-of-service (DoS) attacks, stealing user information by ID theft attacks, phishing attacks, etc. Therefore, it leads to the growth of security problems in cloud storage and leaks the confidentiality of users' data in a communal environment. The intruders execute these attacks with malicious nodes or malware to compromise the host system. Hence, the network security researchers introduced an intrusion detection system to handle anomalous networks [2].

IDS is recognized as one of the most powerful and promising techniques. It aids in perceiving threats and malicious actions by monitoring computer traffic information, and signals are raised when the threats are noticed. Generally, observing malicious activities is characterized into two processes: signature-based discovery and anomaly-based discovery. The signature-based detection method works like an antivirus application that compares the current task with historical task features. In contrast, the anomaly-based detection method works based on the comparison with regular traffic to process the decision. In the NSL-KDD dataset, the network attacks are characterized into four major divisions,

namely remote-to-local (R2L), denial-of-service (DoS), probe attack, and user-to-root (U2R) attacks [3].

Many network security researchers have recently incorporated learning models into intrusion detection systems to determine accurate network attacks. Learning models are utilized more because of their efficacy in processing large-dimension data, evolutionary-based learning capability, and automatic feature extraction. Most recent works used traditional machine learning models to handle the intrusion detection, namely support vector machine [4], XGBoost [5], naive Bayes [6], KNN [7], and random forest [8]. In addition, deep learning models have been utilized in several recent works, namely recurrent neural networks [9], multilayer perceptron [10], and convolutional neural networks [11]. They have proven their efficacy in detecting attacks with improved accuracy.

Nevertheless, the existing techniques have made thoughtful improvements, but the class-imbalanced information remains a challenging issue that hampers the performance of most IDS. The class imbalance arises when the standard trials are significantly higher than the number of intruder trials. Therefore, traditional activities dominate in real networks, thus lead to the misclassification of intrusion detection [12]. A quantitative indicator to determine the severity of class imbalance is the disparity ratio between the dominant and marginal classes. For illustration, the practical dataset of network incursion has nearly 22 lakhs of standard trials and only 5 lakhs of intrusion trials. Thus, the imbalance ratio is computed as 4.4:1. An observation is noticed that most of the samples belong to majority classes while abandoning the minority samples. The trials of minority classes are minimal and insufficient for those techniques. However, the methods are inadequate to learn from the minimal minority samples, and the outcomes are towards the majority. The misdetection of minority samples (intrusion) is much more critical than detecting the standard trials as an intrusion.

In this work, we introduced a novel learning technique, modified conditional generative adversarial network (MCGAN), to handle the class disparity issue, which generates adequate trials for minority classes. The MCGAN technique percolates the information to guarantee only generating minority classes to improve real intrusion discovery. A MCGAN-based IDS model is erected to handle the class imbalance issue and incorporates three strategies: feature extraction, CGAN, and deep neural network.

The main objective of this work is described as follows:

- A novel technique, namely the modified conditional generative adversarial network (MCGAN), is introduced to get rid of the class disparity issue.
- A linear correlation-based feature selection method was introduced to select the significant features, and the Bi-LSTM technique was used to classify the sub-class of intrusion.
- The proposed technique experimented on NSL-KDD+ datasets. We analyzed the efficiency of attack detection with measurable estimations.
- The outcome of the proposed technique is associated with traditional techniques under various modified features to validate the system's efficacy.

The next section, Section 2, discusses the existing methodology and its limitations. Section 3 illustrates the datasets and problem formulation. Section 4 discusses the proposed model's merits in handling the class imbalance problem and classification of multi-class intrusions. Section 5 describes the proposed model's experimental setup and analyzes the proposed model outcome with existing methods. Finally, Section 6 accomplishes the work with its imminent directions.

2. Literature Survey

With the massive increase in network traffic data, learning-based network intrusion detection systems (NIDS) have been introduced to eradicate different unauthentic network actions and possible hidden threats [13]. However, various issues exist in the design and implementation of the learning-based NIDS. (1) Standard network intrusion discovery approaches frequently used several ways such as dimension reduction, compression, and

filtering approaches to reduce measurement noise to tackle the intricacy of dealing with large-scale, high-dimensional data facts. Consequently, extracting features for incursion behaviors will likely eliminate concealed but essential information. Hence, it might result in high false discovery rates. (2) For learning-based NIDS to accurately identify the aspects of intrusion activities, a significant volume of labelled data samples is often needed. The amount and quality of the labelled training set significantly impact how well the NIDS performs. Generating superior, significant training instances is challenging because we traditionally label the training samples manually, which is a time-consuming and error-prone method. (3) NIDS must react to intrusion behaviors in the real world to minimize the loss under an attack. For instance, overflow attacks are frequently concealed in network traffic that can get past the firewall. If such assaults are not quickly identified and stopped, the perpetrator may use them as a Launchpad to transmit a flood of negative posts to the intranet and leave a backdoor open in the breached system. (4) Class imbalance data lead to misclassification, which consists of minimal minority classes and huge majority classes. Hence, class imbalance requires optimization and parallelization architecture for a real-time intrusion detection system [14].

The deep learning model has attracted researchers and industry personnel to handle complex problems. It has been given significant importance in research on cyber security to improve the research quality [15]. Yin et al. [9] proposed an RNN-based IDS to classify multi-class intrusion on the NSL-KDD dataset. They tested the model by varying numerous hidden layers and learning factors. The model outcome in terms of accuracy is satisfactory. However, the multi-class classification, especially on U2R and Probe, is unsatisfactory. The author in [16] introduced a distributed approach using DBN and an ensemble multi-layer SVM model for large-scale NIDS. The DBN model was utilized to extract the features; the extracted features were then provided as input to the ensemble SVM model. The outcome was performed based on the majority voting technique. The model was validated on four different datasets and offers better accuracy in detecting abnormal behaviors.

Vinayakumar et al. [17] merged a scalable DNN model to address both HIDS and NIDS abnormal behavior. They utilized the Apache spark cluster tool for experimental purposes. In addition, six different datasets were used to validate the model's performance. The performance of the model is superior to other compared models. However, the author has not specified the class imbalance issue, and thereby it reduces the version of the model. Punam et al. [18] addressed NIDS issues using a Siamese neural network in which they concentrated on eradicating class imbalance issues. The model processes the oversampling and arbitrary under sampling mechanism to mitigate the problems. However, the model performance is not satisfactory in classifying the multi-class intrusions. Later, the same author in [19] improved the Siamese neural network to improve the classification accuracy. They incorporated the DL method to enrich the detection accuracy rate.

Gupta et al. [20] introduced the LIO-IDS framework to handle class imbalance issues and to improve classification accuracy. The model incorporates the LSTM technique to classify the multi-class intrusions efficiently. The experimentation was carried out on standard datasets and attained better accuracy than other compared models. Tang et al. [21] proposed a DNN-based IDS to secure the software-defined networking platform. They achieved an accuracy of 75% on the standard dataset of NSL-KDD. Later, Wang et al. [22] presented a hybrid CNN and LSTM model to extract the three-dimensional and time-based features. The model provides better outcomes in detecting the intrusion on the standard NSL-KDD dataset.

Based on the numerous studies in the literature [23–25], we observed that only a few works address the class imbalance problem. Hence, a promising model is required to knob the class disparity issue and improve the multi-class classification performance. This work introduces a novel network intrusion detection technique using modified deep learning to balance the imbalance issue and improve detection accuracy.

3. Preliminaries

3.1. Problem Definition

Let us consider Ψ as a feature set with the number of features as η, and assume (A_i, B_i) as samples, where $A_i \rightarrow \vartheta$ denotes actual network traffic trials and $B_i \rightarrow \gamma$ represents the actual labelled classes, and γ denotes the number of types. The objective of IDS is to acquire a classifier with $f : \vartheta \rightarrow \gamma$, which specifies an accurate depiction of the arriving network traffic. The motive of the attacker is to create unnoticeable attack data ρ which will be incorporated into the actual sample A_i and establish an attacker sample as A^*. Later, the sample will be classified as $A_i + \rho \neq B_i$. In this work, we formulated a novel framework with the aid of reinforced GAN to generate the sufficient A^* that aids ML/DL techniques to gain sufficient knowledge and equitize minority and majority classes. Furthermore, we also introduced a conditional GAN-based IDS system that strengthens the ML/DL approaches for the classification of attacker samples efficiently.

3.2. Dataset: NSL-KDD

The NSL-KDD dataset is considered as one of the standard benchmark datasets to validate the IDS model. This dataset is collected from a real-time network scenario, which consists of KDD train+ and test+ samples. In addition, it includes four major categories of malicious traffic data, namely root-to-local (R2L), probing (probe), denial-of-service (DoS), and user-to-root (U2R) attacks. This dataset contains 41 features, with 9discrete set features and 32 continuous set features. Based on the analysis of each element, these dataset features are classified into four significant sets, namely "content", "host-based traffic", "intrinsic", and "time-based traffic" [26].

4. Proposed Methodology

This section describes four processes: first is the data preprocessing technique to preprocess the data. Second is a modified CGAN technique introduced to handle the imbalance issues. Later, the feature selection technique is utilized to select the optimal features, and finally, the Bi-LSTM technique is used for efficient classification.

4.1. Data Preprocessing

The numeric adaptation and the regularization are used to preprocess facts earlier, being nourished into models since NSL-KDD has many feature types and ranges. There are three embedded non-numerical characteristics (protocol type, service, and flag). For instance, the three properties of "protocol type," TCP, UDP, and ICMP, will be transformed into one-hot vectors. The min-max regularization approach converts the unique numeric features of the data into the series of [0; 1] to remove the range influence among feature values in the input vectors. The mathematical formulation of the min-max regularization approach is presented as follows:

$$s = \frac{s - \beta_L}{\beta_U - \beta_L} \tag{1}$$

where β_L and β_U specify the maximal and minimal dimensional limits, respectively.

4.2. Modified Conditional Generative Adversarial Network (MCGAN)

A generative adversarial network (GAN)is a deep learning technique that mimics the game theory concept of two-human zero-sum games and is utilized to handle large-scale real-time complex information. This technique is used to oversample the available data by balancing minority and majority classes [27]. GAN incorporated two neural networks, namely generator ($\mathfrak{g}$) and discriminator ($\mathfrak{D}$). Let $\mathfrak{g}$ be used to analyze the distribution of actual input samples $S = \{s_1, s_2, \ldots, s_n\}$ and create a new set of data samples, and $\mathfrak{D}$ be considered as a binary classifier utilized to specify whether the s is original or engendered data (z). The classified outcome reverts to g and D to eradicate the weight loss. The process repeats until D has successfully classified original and engendered data samples. The

learning method is a mini-max game to obtain a Nash equilibrium among $\mathfrak{g}$ and $\mathfrak{D}$. The optimization function for GAN is represented as

$$\min_{\mathfrak{g}} \max_{\mathfrak{D}} V(\mathfrak{g}, \mathfrak{D}) = \mathbb{E}_{s\sim p(s)}\Big[\log \mathfrak{D}(s)] + \mathbb{E}_{z\sim p(z)}[\log(1-\mathfrak{D}(\mathfrak{g}(z))] \tag{2}$$

where $p(s)$ specifies the dispersal of actual data instances, the method $\mathfrak{g}(z)$ implies the noise information (z) to the search space, and $\mathfrak{D}(s)$ denotes the probability that instance s is actual data. Moreover, the $\mathfrak{D}(s)$ samples must have more data than $\mathfrak{D}(\mathfrak{g}(z))$ to differentiate the actual and generated data. The traditional GAN technique has the drawback of mode-collapse occurrence; the outcome leads to only one class instead of giving importance to the entire distribution. This problem may occur when the actual sample distribution is multi-modal.

To handle the above issue, we proposed the conditional generative adversarial network (CGAN) that is a modified technique of the GAN Model, in which the categorical data and noise information are clubbed together with the actual samples as the input to the g and D with loss strategy. CGAN is effective in learning together with the existing distribution samples.

$$\min_{\mathfrak{g}} \max_{\mathfrak{D}} V(\mathfrak{g}, \mathfrak{D}) = \mathbb{E}_{s\sim p(s)}\Big[\log \mathfrak{D}(s|x)] + \mathbb{E}_{z\sim p(z)}[\log(1-\mathfrak{D}(\mathfrak{g}(z|x))] \tag{3}$$

where x denotes the class details and the other parameters as specified in Equation (1). The working process of CGAN is presented in Figure 1. Generally, GAN and CGAN create a set of instances and minimize the class disparity issues. Nevertheless, their practice of the Jensen Shannon distributer involves overlay between the scattering of actual and produced cases, which is unreal or unacceptable. The $\mathfrak{D}$ is trained to be optimal; hence, it may lead to the prototypical downfall and wipe out gradient issues [28].

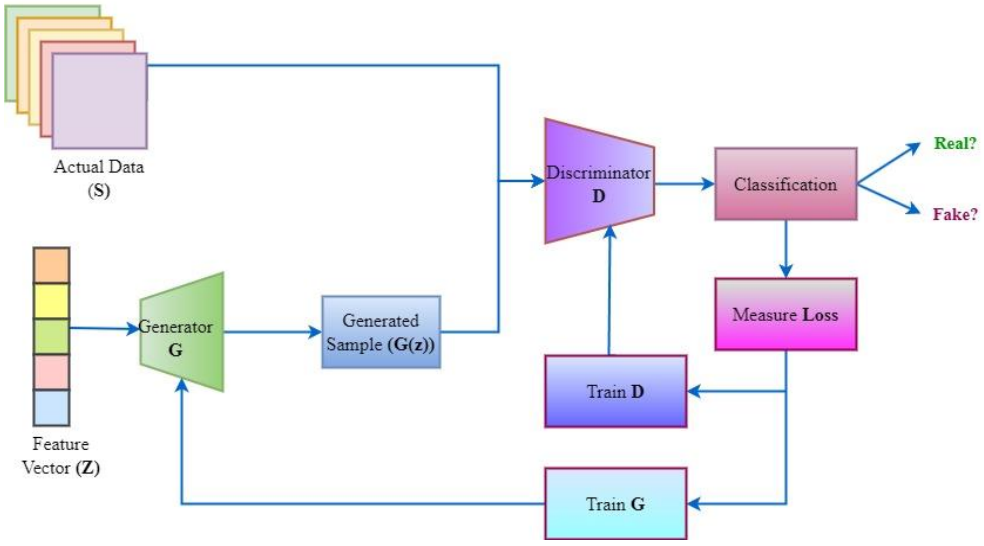


Figure 1. The architecture of conditional generative adversarial network.

In this work, we modified CGAN by merging the Lipschitz boundary and Wasserstein distance to handle the above issues. These incorporated techniques address the $\mathfrak{D}$, z vector and mapping labels to $\mathfrak{g}$. Here, we utilize $\mathfrak{D}$ to determine the actual and created instances s. If $\mathfrak{D}$ fails to distinguish the genuine and generated s, then we fine-tune the $\mathfrak{g}$ and train the $\mathfrak{D}$ and vice versa. We replicate the process until the loss rate is minimized at about 0.5. The proposed model can create the data of a quantified pattern to balance the imbalance

dataset by eradicating the waning gradients that affect the $\mathfrak{D}$ in the training process. The fitness function of MCGAN is presented below.

$$V(\mathfrak{g},\ \mathfrak{D}) = \max_{\mathfrak{D}}\left\{\mathbb{E}_{s\sim p(s)}[\mathfrak{D}(s|x)] - \mathbb{E}_{s\sim p(g)}[\mathfrak{D}(s|x)] - \varphi\mathbb{E}_{s\sim p(\omega)}[||\nabla_s\mathfrak{D}(s|x)||-1]^2\right\} \quad (4)$$

where φ denotes an artificial factor, $||\nabla_s\mathfrak{D}(s|x)||$ specifies the computing pattern for s in $\mathfrak{D}(s)$, and $s \sim p(\omega)$ determines the centre location of the line-linking facts on $p(\omega)$ and $p(\mathfrak{g})$.

4.3. Feature Selection

The feature selection process is a vital method to handle high-dimensional data and reduce computational complexity. This process selects the significant features from various elements in the problem datasets. In this work, we utilized the Nadam optimizer in the neural network to extract the components [29]. Later, we used a linear correlation-based feature selection technique which computes the correlation distance between two arbitrary feature vectors. In addition, it also calculates linear correlations between the features. For instance, if feature X with value a, class Y with value b are specified as arbitrary vectors. Then, the correlation C among the vectors is mathematically formulated as below.

$$C(X,Y) = \mathbb{C} = \frac{\sum_{i=1}^{N}(a_i-\overline{a})\left(b_i-\overline{b}\right)}{\sqrt{\left[\sum_{i=1}^{N}(a_i-\overline{a})^2\sum_{i=1}^{N}\left(b_i-\overline{b}\right)^2\right]}} \quad (5)$$

where $\overline{a}$ and $\overline{b}$ are denoted as predicted values of X and Y. C is equal to zero if X and Y are linearly independent or else one if they are wholly dependent. The proposed feature selection technique utilizes the linear correlation coefficient to select the significant features that minimize the computational complexity.

4.4. Bidirectional Long Short-Term Memory (Bi-LSTM) Technique

LSTM is a variant of recurrent neural networks that uses a gating technique to study long-term dependencies. It eradicates the issue of disappearing gradient. At the same time, it performs the training of the generic recurrent neural network (RNN). The LSTM technique uses different switch gates that trigger them to circumvent units and, as an outcome, memory for long short-term steps [30]. In this work, we utilized bidirectional LSTM in which the first LSTM was applied directly to input samples, whereas another LSTM was applied to the replica of the input samples. This aids the network in learning more data information, thereby improving classification accuracy.

Further, it takes the original input to the initial layer and the reversed imitation of input samples provided to the replicated layer. This work eradicates the issue of vanishing gradients in generic RNNs. The training of Bi-LSTM processed based on all earlier and forthcoming data resides within a specified time sequence. In addition, Bi-LSTM processes the input samples in two significant ways with the aid of a forward hidden layer and a backward hidden layer. The mathematical method of forward and backward hidden layers is specified as follows.

$$\overrightarrow{h}_k = \mathcal{H}\left(\psi_{\underset{ah}{\rightarrow}}s_k + \psi_{\underset{hh}{\rightarrow\rightarrow}}\overrightarrow{h}_{k-1} + \beta_{\underset{h}{\rightarrow}}\right) \quad (6)$$

$$\overleftarrow{h}_k = \mathcal{H}\left(\psi_{\underset{ah}{\leftarrow}}s_k + \psi_{\underset{hh}{\leftarrow\leftarrow}}\overleftarrow{h}_{k+1} + \beta_{\underset{h}{\leftarrow}}\right) \quad (7)$$

$$b_t = \psi_{\underset{yh}{\rightarrow}}\overrightarrow{h}_k + \psi_{\underset{yh}{\leftarrow}}\overleftarrow{h}_k + \beta_b \quad (8)$$

where $\psi_{\underset{ah}{\rightarrow}}$ and $\psi_{\underset{ah}{\leftarrow}}$ denote the forward and backward hidden input weight values; $\beta_{\underset{h}{\rightarrow}}$ and $\beta_{\underset{h}{\leftarrow}}$ specify the bias values of forward and backward hidden layers; and $\mathcal{H}$ denotes

the hidden layer, respectively. The detailed working process of the proposed model is offered in Algorithm 1. The proposed architecture of the multi-class Bi-LSTM is illustrated in Figure 2.

Algorithm 1: Training procedure for Bi-LSTM model

1. Input dataset (NSL-KDD$^+$)
2. **For** Samples in Training and Testing sets, **do**
    a. Extract the features (a)
    b. Extract Labels (y)
3. **End For**
4. **For** features in α **do**
    a. **If** a = Nonnumeric, **then**
        i. Using Keras Library to encode the features
    b. **End if**
5. **End for**
6. Scale the features using Equation (1)
7. **For** $k = 1 : n$ **do**
    a. Initialize $k = 10$
    b. Divide training samples into k-sectors
    c. Load Bi-LSTM model
    d. Fit model with $k - 1$
    e. Validate model with rest of kth sectors
    f. Repeat until all k-sectors are used as validation samples
8. **End for**
9. Test model on Test sets (NSL-KDD$^+$)

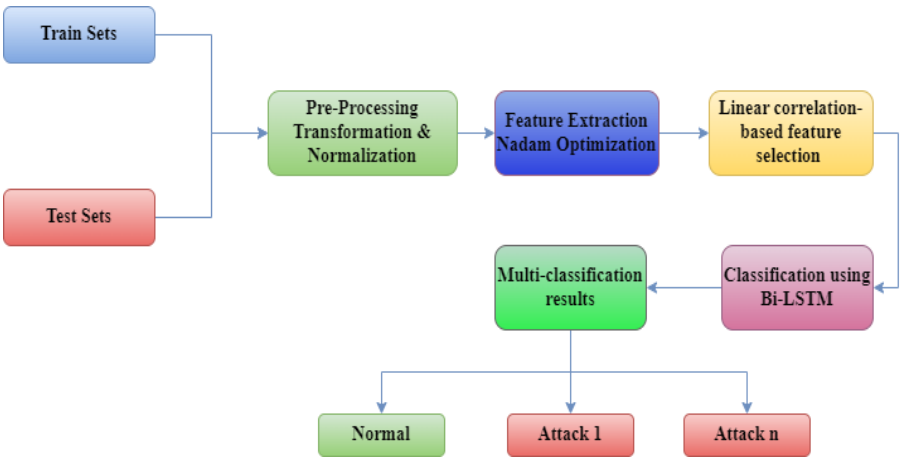


Figure 2. Proposed Model Multi-class Bi-LSTM architecture.

5. Experimentation and Result Analysis

In this section, we specify the experimental setup and examine the performance evaluation of the model. On the other hand, we examined enough experiments to validate the efficacy of the data augmentation, and an improved number of attribute reduction models are discussed in Section 3. Furthermore, we associated the formulated outcome with other existing models.

5.1. Experimental Setup

In this work, all experiments were carried out in Intel® core™ i5-8250U processor @1.60GHz 1.80 GHz, 8GB RAM, with the Windows ten operating system. The coding and simulation of the model were executed in python 3.8, and PyTorch 2.0 and the sklearn library

were utilized. For the testing and validation, samples were taken from NSL-KDD+ datasets, described in Section 3.1. The proposed model outcome is endorsed with a stratified K-fold cross-validation approach with k fixed as 10. In addition, the proposed model outcome is associated with other existing models such as AE-CGAN-RF [28], LSSVM-IDS [31], RNN-IDS [9], and SSAE-LSTM [32], which were applied to the balanced NSL-KDD+ dataset.

5.2. Performance Metrics

The standard evaluation metrics such as true positive (TP), true negative (TN), false positive (FP), and false negative (FN) were utilized to validate the efficacy of the classification. In addition, false-positive rate (FPR), precision (Ψ), recall (Y), specificity (ϑ), accuracy (Φ), and F1-score were utilized to compare the efficacy of the formulated approach with other compared models. The mathematical formulation of the performance metrics is described as follows.

$$\Psi = \frac{TP}{TP + FP} \tag{9}$$

$$\Upsilon = \frac{TP}{TP + FN} \tag{10}$$

$$\vartheta = \frac{TN}{FP + TN} \tag{11}$$

$$FPR = \frac{FP}{FP + TN} \tag{12}$$

$$\Phi = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

$$F1_{Score} = 2 \times \frac{\Psi \times \Upsilon}{\Psi + \Upsilon} \tag{14}$$

If the outcomes of Φ, Υ, Ψ, ϑ, and $F1_{Score}$ are high, then the outcome of the proposed model is improved. On the other hand, the FPR value must be less to ensure enriching the classification quality.

5.3. Result Analysis

We conducted experiments on a modified NSL-KDD+ dataset balanced by the MC-GAN approach. In addition, we created the NSL-KDD+ and NSL-KDD+20 datasets that include all 41 features and 20 selected features. To validate the outcome of the proposed model, we utilized the datasets in training (80%) and testing (20%) samples. The proposed model uses the linear correlation feature selection approach to diminish the features and select significant characteristics for training purposes. In this work, we set 20 features that aid the model in learning from the selected low-dimensional parts to improve the classification outcome of the classifiers.

Table 1 provides the class-wise outcome achieved by the proposed model on NSL-KDD+ and NSL-KDD+20 datasets. In the NSL-KDD+ dataset, the F1-score outcome for the typical class was 95.78%, with a DoS of 94.31%, probe of 84.87%, R2L of 94.57%, and U2R of 81.45%. The false-positive rate (FPR) outcome for the typical class was 4.57%, with a DoS of 0.87%, probe of 2.14%, R2L of 0.51%, and U2R of 0.69%. As we specified earlier in Section 5.2, FPR should be less to ensure that the proposed model achieved a better outcome. Further, we applied the proposed model on the selected 20 significant features termed NSL-KDD+20. In the NSL-KDD+20 dataset, the F1-score result for the typical class was 96.91%, with a DoS of 94.87%, probe of 85.74%, R2L of 95.71%, and U2R of 82.97%. The false-positive rate (FPR) outcome for the typical class was 4.14%, with a DoS of 0.74%, probe of 2.45%, R2L of 0.47%, and U2R of 0.71%. Based on the analysis of both datasets, we noticed that the proposed model on the NSL-KDD+20 dataset provides a better outcome than the NSL-KDD+ dataset. This outcome is achieved due to formal learning from samples by the proposed model. Figures 3–6 show the precision, recall, specificity, and F1-score of different classes on NSL-KDD+ and NSL-KDD+20 datasets.

Table 1. The outcome of the proposed model for multi-class classification.

Dataset with Features	Class	Precision	Recall	FPR	Specificity	F1-Score
NSL-KDD+	Normal	92.04%	93.45%	4.57%	96.47%	95.78%
	DoS	93.51%	92.74%	0.87%	97.54%	94.31%
	Probe	91.78%	90.27%	2.14%	98.12%	84.87%
	R2L	90.47%	94.12%	0.51%	95.97%	94.57%
	U2R	88.54%	84.54%	0.69%	89.78%	81.45%
NSL-KDD+20	Normal	93.80%	94.12%	4.14%	96.89%	96.91%
	DoS	94.21%	91.54%	0.74%	98.57%	94.87%
	Probe	92.54%	89.71%	2.45%	98.90%	85.74%
	R2L	89.12%	93.74%	0.47%	94.78%	95.71%
	U2R	86.87%	85.78%	0.71%	88.74%	82.67%

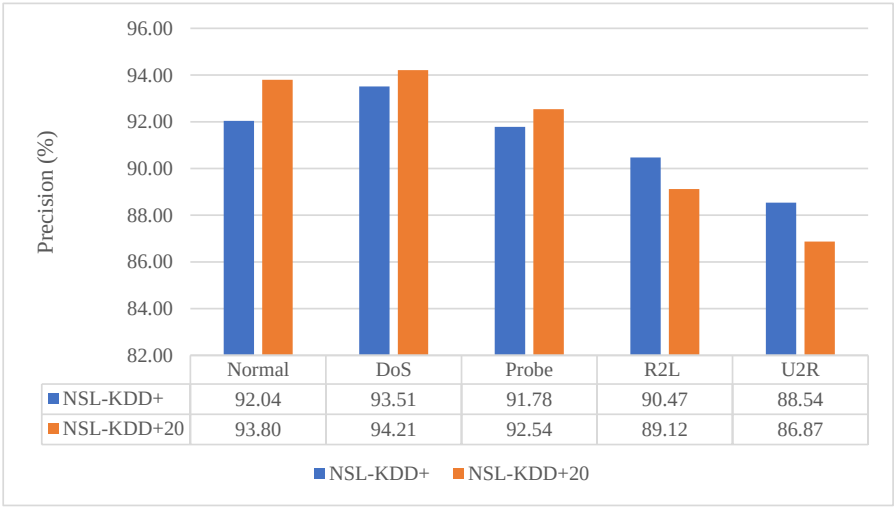


Figure 3. Precision rate of different classes on NSL-KDD+ and NSL-KDD+20 datasets.

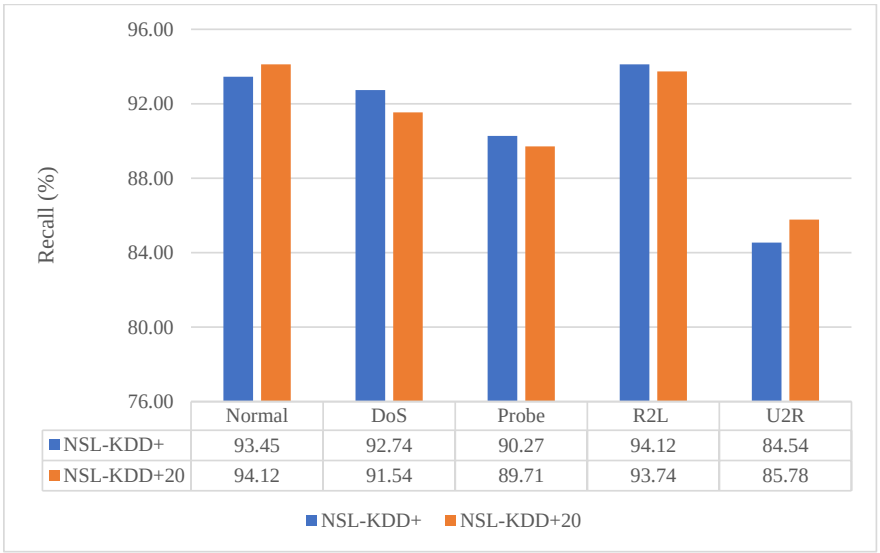


Figure 4. Recall rate of different classes on NSL-KDD+ and NSL-KDD+20 datasets.

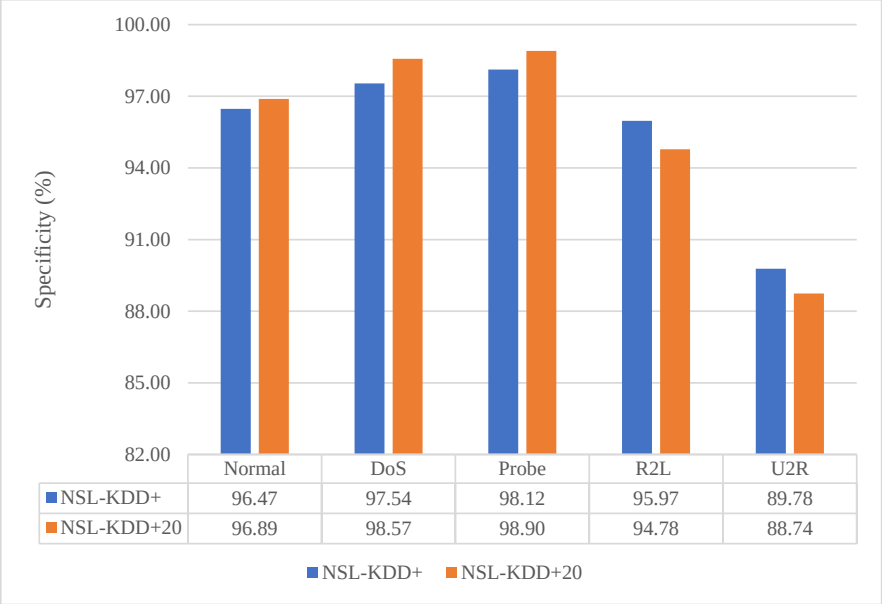


Figure 5. Specificity rate of different classes on NSL-KDD+ and NSL-KDD+20 datasets.

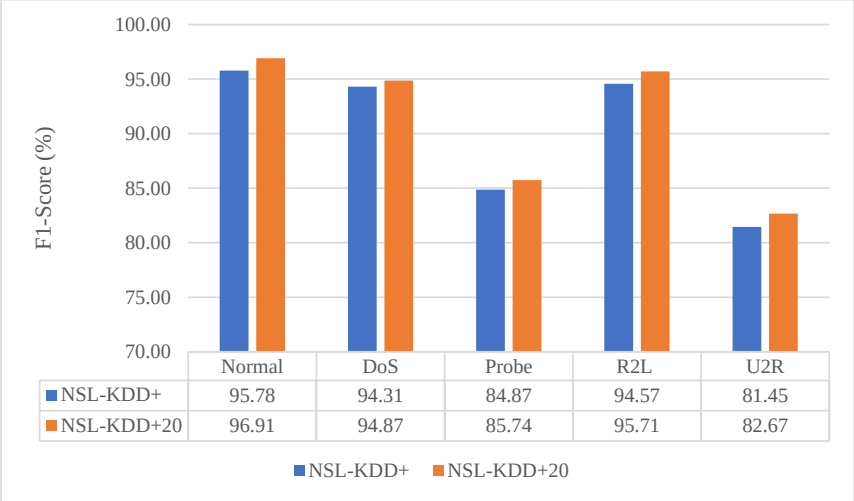


Figure 6. F1-score of different classes on NSL-KDD+ and NSL-KDD+20 datasets.

5.4. Comparative Analysis of Proposed Model

To highlight the efficacy of the proposed model, the outcome of the model associated with the other existing approaches that include LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM is presented. The result of the proposed model deliberates higher accuracy than other compared models such as LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM. Figure 7 provides the accuracy of the proposed approach and other existing approaches on NSL-KDD+ and NSL-KDD+20 datasets. The LSSVM-IDS model offers an accuracy of 53.21% and 55.86% on NSL-KDD+ and NSL-KDD+20 datasets. The performance of LSSSVM-IDS shows poor performance compared to the AE-CGAN-RF approach. In addition, AE-CGAN-RF offers 64.56% and 67.12% accuracy on NSL-KDD+ and NSL-KDD+20 datasets. However, the outcome of AE-CGAN-RF provides an unsatisfactory performance. Consequently, the RNN-IDS and SSAE-LSTM approaches provide satisfactory accuracy outcomes of 81.42% and 84.93% and 85.98% and 88.79% on the NSL-KDD+ and NSL-KDD+20 datasets. Simultaneously, the formulated approach offers higher accuracies of 91.76% and 95.16% on NSL-KDD+ and NSL-KDD+20 datasets.

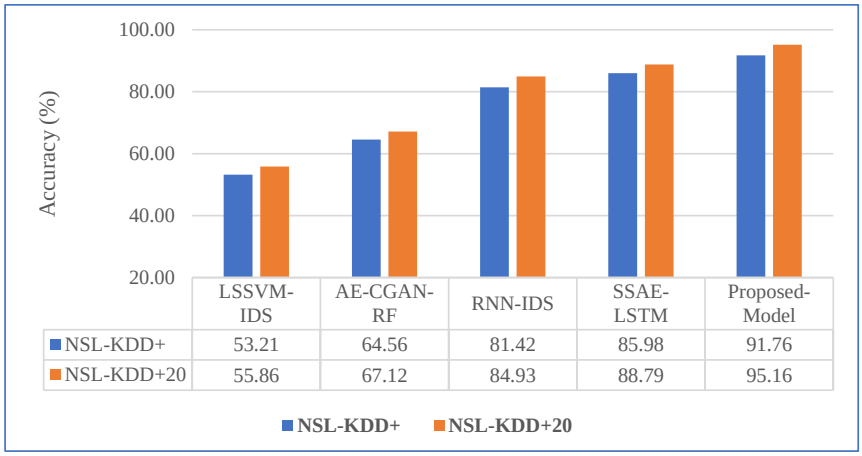


Figure 7. The performance of the proposed model with other compared models on NSL-KDD+ and NSL-KDD+20 datasets.

To highlight the efficacy of the proposed model, the outcome of the model associated with the other existing approaches that include LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM is presented. The result of the proposed model deliberates higher accuracy than other compared models such as LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM. Figure 7 provides the accuracy of the proposed approach and other existing approaches on NSL-KDD+ and NSL-KDD+20 datasets. The LSSVM-IDS model offers an accuracy of 53.21% and 55.86% on NSL-KDD+ and NSL-KDD+20 datasets. The performance of LSSSVM-IDS shows poor performance compared to the AE-CGAN-RF approach. In addition, AE-CGAN-RF offers 64.56% and 67.12% accuracy on NSL-KDD+ and NSL-KDD+20 datasets. However, the outcome of AE-CGAN-RF provides an unsatisfactory performance. Consequently, the RNN-IDS and SSAE-LSTM approaches provide satisfactory accuracy outcomes of 81.42% and 84.93% and 85.98% and 88.79% on the NSL-KDD+ and NSL-KDD+20 datasets. Simultaneously, the formulated approach offers higher accuracies of 91.76% and 95.16% on NSL-KDD+ and NSL-KDD+20 datasets.

Table 2 compares the overall performance of the proposed model to that of other comparable models on the NSL-KDD+ and NSL-KDD+20 datasets. To ensure the effectiveness of the proposed model, the accuracy, recall, FAR, and F1-score of LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM were measured. When compared to previous models, the suggested proposed model has a higher accuracy and recall rate. Furthermore, it is obvious that the suggested model achieves 1.85% and 1.06% false alarm rates for the NSL-KDD+ and NSL-KDD+20 datasets, respectively, while the comparison model fails to reach higher FPR outcomes. On the other hand, the suggested model's F1-score outperforms the NSL-KDD+ and NSL-KDD+20 datasets.

Table 2. Overall performance comparison.

Algorithms	NSL-KDD+				NSL-KDD+20			
	Precision	**Recall**	**FPR**	**F1-Score**	**Precision**	**Recall**	**FPR**	**F1-Score**
LSSVM-IDS	52.74%	52.89%	8.78%	53.17%	55.78%	55.91%	7.14%	55.88%
AE-CGAN-RF	64.94%	65.06%	5.17%	64.62%	67.23%	67.98%	4.97%	67.57%
RNN-IDS	82.14%	82.77%	3.47%	81.76%	85.04%	85.87%	3.01%	84.97%
SSAE-LSTM	84.17%	86.16%	2.74%	84.87%	88.54%	89.03%	2.14%	88.67%
Proposed model	91.94%	92.05%	1.85%	91.88%	95.42%	96.07%	1.06%	95.78%

Based on the comparative result analysis, we conclude that the proposed model identifies various classes of known and unknown assaults by boosting the learning accuracy of low-dimensional characteristics. The proposed model's overall performance expresses

by outperforming other models in terms of accuracy. Furthermore, the suggested model may be integrated into a real-time intrusion detection system to increase detection speed and accuracy.

6. Conclusions

Class imbalance is considered a severe issue that might lead to poor detection accuracy in network intrusion detection systems. An efficient detection approach is necessary to eradicate the class imbalance problem. This work introduces a modified conditional generative adversarial network (MCGAN) to handle the imbalance issues. MCGAN generates a good set of samples to eradicate the class imbalance problem. In addition, the Nadam optimizer for feature extraction and linear-correlation-based feature selection were utilized to select the significant features. Later, the Bi-LSTM approach was used to classify the attacks according to a different set of classes. The experimentation was carried out on NSL-KDD+ datasets with balanced data samples and NSL-KDD+20 with 20 selected features. The proposed model was applied to those selected datasets, and the model's performance was measured using standard performance metrics such as precision, recall, accuracy, specificity, false-positive rate, and F1-score. The outcome of the proposed model was compared with other state-of-art approaches that include LSSVM-IDS, AE-CGAN-RF, RNN-IDS, and SSAE-LSTM. The overall accuracy of the proposed model on NSL-KDD+ achieved 91.76% and 95.16% on the NSL-KDD+20 datasets. Given that the proposed model achieved better accuracy than other compared models, further, this work can be extended by incorporating a meta-heuristic algorithm to choose the optimal features and to improve the model's accuracy by reducing computational complexity.

Author Contributions: The authors confirm responsibility for the following: study conception and design: K.S.B. and Y.N.R.; data collection: K.S.B.; analysis and interpretation of result: K.S.B.; investigation: Y.N.R.; manuscript preparation: K.S.B.; supervision: Y.N.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the first author upon reasonable request.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Scarfone, K.; Mell, P.M. *Guide to Intrusion Detection and Prevention Systems (IDPS)*; U.S. Department of Commerce: Washington, DC, USA, 2007. [CrossRef]
2. Salo, F.; Nassif, A.B.; Essex, A. Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection. *Comput. Networks* **2019**, *148*, 164–175.
3. Revathi, S.; Malathi, A. A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *Int. J. Eng. Res. Technol.* **2013**, *2*, 1848–1853.
4. Gu, J.; Wang, L.; Wang, H.; Wang, S. A novel approach to intrusion detection using SVM ensemble with feature augmentation. *Comput. Secur.* **2019**, *86*, 53–62. [CrossRef]
5. Dhaliwal, S.S.; Nahid, A.-A.; Abbas, R. Effective Intrusion Detection System Using XGBoost. *Information* **2018**, *9*, 149. [CrossRef]
6. Sharmila, B.S.; Nagapadma, R. Intrusion detection system using Naive Bayes algorithm. In Proceedings of the 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Bangalore, India, 15 November 2019; pp. 1–4.
7. Rao, B.B.; Swathi, K. Fast kNN Classifiers for Network Intrusion Detection System. *Indian J. Sci. Technol.* **2017**, *10*, 1–10. [CrossRef]
8. Jabbar, M.; Aluvalu, R. RFAODE: A Novel Ensemble Intrusion Detection System. *Procedia Comput. Sci.* **2017**, *115*, 226–234. [CrossRef]
9. Yin, C.; Zhu, Y.; Fei, J.; He, X. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access* **2017**, *5*, 21954–21961. [CrossRef]

10. Shettar, P.; Kachavimath, A.V.; Mulla, M.M.; Hanchinmani, G. Intrusion detection system using MLP and chaotic neural networks. In Proceedings of the 2021 International Conference on Computer Communication and Informatics (ICCCI), Virtual, 27 January 2021; pp. 1–4.
11. Sun, P.; Liu, P.; Li, Q.; Liu, C.; Lu, X.; Hao, R.; Chen, J. DL-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System. *Secur. Commun. Networks* **2020**, *2020*, 1–11. [CrossRef]
12. Rodda, S.; Erothi, U.S.R. Class imbalance problem in the Network Intrusion Detection Systems. *IEEE* **2016**, *775*, 2685–2688. [CrossRef]
13. Ahmad, Z.; Shahid Khan, A.; Wai Shiang, C.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4150. [CrossRef]
14. Yang, J.; Li, T.; Liang, G.; He, W.; Zhao, Y. A Simple Recurrent Unit Model Based Intrusion Detection System With DCGAN. *IEEE Access* **2019**, *7*, 83286–83296. [CrossRef]
15. Wang, X.; Zhao, Y.; Pourpanah, F. Recent advances in deep learning. *Int. J. Mach. Learn. Cybernetics.* **2020**, *11*, 747–750. [CrossRef]
16. Marir, N.; Wang, H.; Feng, G.; Li, B.; Jia, M. Distributed Abnormal Behavior Detection Approach Based on Deep Belief Network and Ensemble SVM Using Spark. *IEEE Access* **2018**, *6*, 59657–59671. [CrossRef]
17. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; Al-Nemrat, A.; Venkatraman, S. Deep learning approach for intelligent intrusion detection system. *Ieee Access* **2019**, *7*, 41525–41550. [CrossRef]
18. Bedi, P.; Gupta, N.; Jindal, V. Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network. *Procedia Comput. Sci.* **2020**, *171*, 780–789. [CrossRef]
19. Bedi, P.; Gupta, N.; Jindal, V. I-SiamIDS: An improved Siam-IDS for handling class imbalance in network-based intrusion detection systems. *Appl. Intelligence* **2021**, *51*, 1133–1151. [CrossRef]
20. Gupta, N.; Jindal, V.; Bedi, P. LIO-IDS: Handling class imbalance using LSTM and improved one-vs-one technique in intrusion detection system. *Comput. Networks* **2021**, *192*, 108076. [CrossRef]
21. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.; Ghogho, M. Deep learning approach for network intrusion detection in software defined networking. In Proceedings of the 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 26 October 2016; pp. 258–263.
22. Wang, W.; Sheng, Y.; Wang, J.; Zeng, X.; Ye, X.; Huang, Y.; Zhu, M. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection. *IEEE Access* **2017**, *6*, 1792–1806. [CrossRef]
23. Ngueajio, M.K.; Washington, G.; Rawat, D.B.; Ngueabou, Y. Intrusion Detection Systems Using Support Vector Machines on the KDDCUP'99 and NSL-KDD Datasets: A Comprehensive Survey. In Proceedings of the 2022 Intelligent Systems Conference (IntelliSys), Amsterdam, The Netherlands, 2–3 September 2021; Volume 2, pp. 609–629.
24. Devarakonda, A.; Sharma, N.; Saha, P.; Ramya, S. Network intrusion detection: A comparative study of four classifi-ers using the NSL-KDD and KDD'99 datasets. *J. Phys. Conf. Ser.* **2022**, *2161*, 012043.
25. Kilincer, I.F.; Ertam, F.; Sengur, A. A comprehensive intrusion detection framework using boosting algorithms. *Comput. Electr. Eng.* **2022**, *100*, 107869. [CrossRef]
26. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE symposium on computational intelligence for security and defense applications, Ottawa, ON, Canada, 8 July 2009; pp. 1–6.
27. Zhang, G.; Wang, X.; Li, R.; Song, Y.; He, J.; Lai, J. Network intrusion detection based on conditional Wasserstein generative adversarial network and cost-sensitive stacked autoencoder. *IEEE Access* **2020**, *8*, 190431–190447.
28. Lee, J.; Park, K. AE-CGAN Model based High Performance Network Intrusion Detection System. *Appl. Sci.* **2019**, *9*, 4221. [CrossRef]
29. Murugan, P.; Durairaj, S. Regularization and optimization strategies in deep convolutional neural network. *arXiv* **2017**, arXiv:1712.04711.
30. Staudemeyer, R.C.; Morris, E.R. Understanding LSTM—A tutorial into long short-term memory recurrent neural networks. *arXiv* **2019**, arXiv:1909.09586.
31. Ambusaidi, M.A.; Xiangjian, H.; Priyadarsi, N.; Zhiyuan, T. Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Trans. Comput.* **2016**, *65*, 2986–2998. [CrossRef]
32. Lin, Y.; Wang, J.; Tu, Y.; Chen, L.; Dou, Z. Time-Related Network Intrusion Detection Model: A Deep Learning Method. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019. [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.