Research Article

Oversampling Imbalanced Data Based on Convergent WGAN for Network Threat Detection

Yanping Xu ¹, Xiaoyu Zhang,² Zhenliang Qiu ¹, Xia Zhang ¹, Jian Qiu,³
and Hua Zhang⁴

¹School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China

²Beijing Topsec Network Security Technology Co., Ltd., Hangzhou Filiale, Hangzhou, China

³Center for Undergraduate Education, Westlake University, Hangzhou, China

⁴School of Computer Science, Hangzhou Dianzi University, Hangzhou, China

Correspondence should be addressed to Zhenliang Qiu; zl_qiu@163.com

Received 15 July 2021; Accepted 15 October 2021; Published 5 November 2021

Academic Editor: Helena Rifà-Pous

Copyright © 2021 Yanping Xu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Class imbalance is a common problem in network threat detection. Oversampling the minority class is regarded as a popular countermeasure by generating enough new minority samples. Generative adversarial network (GAN) is a typical generative model that can generate any number of artificial minority samples, which are close to the real data. However, it is difficult to train GAN, and the Nash equilibrium is almost impossible to achieve. Therefore, in order to improve the training stability of GAN for oversampling to detect the network threat, a convergent WGAN-based oversampling model called convergent WGAN (CWGAN) is proposed in this paper. The training process of CWGAN contains multiple iterations. In each iteration, the training epochs of the discriminator are dynamic, which is determined by the convergence of discriminator loss function in the last two iterations. When the discriminator is trained to convergence, the generator will then be trained to generate new minority samples. The experiment results show that CWGAN not only improve the training stability of WGAN on the loss smoother and closer to 0 but also improve the performance of the minority class through oversampling, which means that CWGAN can improve the performance of network threat detection.

1. Introduction

In recent years, network attack and threat have gradually become two of the most important issues in the cyberspace. For example, in blockchain, big data, and IoT systems, the network attack occurs frequently, and global privacy leakage accidents caused by network threat are rising rapidly. It has caused great economic losses and undermining social stability [1–3]. Network threat detection based on machine learning and deep learning is one of the important ways to protect network security relying on massive data. However, the data sets collected through various ways are usually imbalanced. The number of the samples in each category varies significantly. In most cases, the benign category has a large number of samples, which is the so-called majority class. And the malicious category has much fewer samples, which is the

so-called minority class. For example, in network intrusion detection, most traffic issues are normal with very few abnormal ones [4]. In Android malware detection, the proportion of the malware APPs is relatively low [5]. In the network threat detection, if the malicious samples as minority class are misjudged and widely spread, it will bring big losses to users. Therefore, the correct and accurate detection of malicious data in imbalanced data sets is very important. Imbalanced data sets are so common that they bring many troubles to data mining and analysis [6]. However, when the number of samples of the minority class is very small, they are very likely to be discarded as noise data. It is very harmful for the training of classifier, thus reducing the accuracy of the minority class classification [7, 8]. Therefore, it is an urgent problem to strengthen the classification ability and improve the classification accuracy for the minority class.

So far, numerous methods and strategies have been proposed to solve the problem of imbalanced data, which are divided into two major categories, including data-level solution and algorithmic-level solution [9–11]. Data-level approaches focus on the data sets by undersampling the majority class, oversampling the minority class, or a combination of both to balance data sets. Undersampling is the key issue to reduce regular data from the majority class, which has a risk of losing important information. Oversampling is to generate some new data close to the minority class, which is popular due to using all the available information and significantly improving the overall performance of data classification. Synthetic minority oversampling technique (SMOTE) [12] is the most representative and popular oversampling algorithm, which generates synthetic minority examples based on the linear interpolation between each minority example and its nearest neighbors. Algorithmic-level approaches improve the algorithms to focus on the classification accuracy of the minority class, which can be divided into four types: cost-sensitive methods, kernel-based learning methods, active learning methods, and ensemble methods [10, 13]. There are a variety of common algorithmic-level methods, such as a Bayesian optimization algorithm, which maximizes Matthew's correlation coefficient by learning optimal weights for the positive and negative classes [14].

However, when the amount of data is big and the dimension is high, the traditional methods mentioned above are not so effective. But machine learning and deep learning techniques appear in time to make up for the shortcomings of the traditional methods [15]. Generative adversarial network (GAN) is a typical generative model, which is regarded as a potential solution for imbalanced data by generating new samples for the minority class [16, 17]. GAN is composed of two neural networks: generator and discriminator. The generator is used to generate complex and high-dimensional data close to the real data as far as possible to deceive the discriminator. The goal of the discriminator is to identify the real data and the false data generated by the generator as far as possible. Through multiple adversarial training, a generator with excellent performance is obtained [18]. However, GAN has some problems, such as unstable training, gradient disappearance, and mode collapse [19, 20]. In order to solve these problems, many variant algorithms have been proposed. Wasserstein GAN (WGAN) is one of the most typical algorithms [21]. WGAN uses Wasserstein distance instead of Jensen–Shannon (JS) divergence to make training stable. Most important of all, WGAN not only solves the unstable training problem of GAN and does not carefully balance the training of generator and discriminator but also solves the collapse mode problem and ensures the diversity of the generated samples. Another advantage of WGAN is that it can deal with the discrete data. Therefore, we apply WGAN as an oversampling method to generate the new minority samples to solve the imbalanced problem.

WGAN is still not perfect with some defects like GAN, such as training difficulty and slow convergence in the real experiment process. Therefore, we tend to construct the condition of alternate training between the generator and

discriminator. Inspired by the idea of letting the convergence of the discriminator guide the training stability of WGAN, we propose a convergent WGAN-based oversampling model called CWGAN, which contains a generator and a convergent discriminator. The generator is used to generate new samples close to the real ones. And the discriminator is used to distinguish the real minority samples and the new samples generated by the generator. In the training process of CWGAN, the generator and the discriminator are trained alternately many iterations. In each iteration, the epoch of the discriminator is dynamic, which is determined by the convergence of loss function in the last two iterations. When the discriminator is trained to convergence, the generator will be trained. Through the process, we can achieve the epochs of the discriminator in each iteration.

The proposed method includes the following advantages:

- (1) Compared with the WGAN, the proposed CWGAN constructs the condition of alternate training between the generator and discriminator to guide the training stability.
- (2) During the training process of CWGAN, the epochs of the discriminator are dynamic, which is determined by the convergence of loss function in the last two iterations.
- (3) CWGAN model can solve the imbalanced data problem through generating the new samples for the minority class to make the data set balance.
- (4) CWGAN not only improves the accuracy of the minority class but also enhances the accuracy of the majority class and the accuracy of the whole data set.

The remainder of this paper is organized as follows. Section 2 discusses the related work about GAN and imbalanced data and GAN training imbalanced. A description of the CWGAN model is provided in Section 3. Experimental results and analysis are presented in Section 4. The work is concluded in Section 5.

2. Related Work

2.1. GAN and Imbalanced Data. Class imbalance is a common issue in network space for classification tasks. Generating the minority samples is regarded as an effective way to solve the class imbalance problem, which is classified as oversampling. Random oversampling, synthetic minority oversampling technique (SMOTE) [12], and borderline SMOTE [16] are considered to be the best traditional oversampling algorithms. However, when the data have a high dimensional space, the performance reduces significantly. By comparison, GANs can generate artificial data close to the minority class when the data are high dimensional and data distributions are complex. So far, researchers have produced many modified versions of GANs.

Hao et al. [22] proposed an Annealing Genetic GAN (AGGAN) method to solve the class imbalance problem by generating data close to the minority class distribution based on the limited data samples. In the training process,

AGGAN used the mechanism of simulated annealing (SA) to update the training of GANs and avoided the local optimum to get the best generator, which could generate data close to the minority classes. Both theoretical analysis and experimental studies showed that AGGAN made the classes balance efficiently and effectively through reproducing the distributions.

Based on a conditional Wasserstein GAN, Engelmann and Lessmann [9] proposed an oversampling method called cWGAN-based model to balance the tabular data with numerical and categorical variables. In the cWGAN-based structure, the conditional distribution is estimated to sample the minority class explicitly. The loss function of the cWGAN-based model is added with an auxiliary classifier (AC) loss to encourage the generator to generate samples. The experiments showed that the cWGAN-based architecture could successfully estimate the data distribution and outperforms other oversampling methods.

Deepshikha and Naman [14] proposed a new GAN framework consisting of a generator network G , a discriminator network D , and a classifier network C , where the generator worked with the classifier to generate some samples in a min-max game with the discriminator to make class balance. In the framework, the generator could generate samples in the convex hull through competing with the classifier. Consequently, the generated samples appeared on the data boundary with better distribution, which could make the classifier easy to find the decision boundary of the whole data set with original imbalanced data and generated data. The experiments showed that the proposed method with an additional classifier achieved better performance than other state-of-the-art techniques on the image data sets.

Kim et al. [23] proposed a novel GAN-based model consisting of an autoencoder as the generator and two separate discriminators for anomaly detection on image data sets. Meanwhile, they proposed new loss functions consisting of patch loss and anomaly adversarial loss to optimize the GAN-based model and improve the robustness and performance for defect detection. There were eight loss functions: six losses for the generator, one for the normal discriminator, and another for the anomaly discriminator. The experiments were carried out on the benchmark data sets and real-world data sets. The experimental results showed that both of the performances of benchmark data sets and real-world data sets were state of the art.

In summary, since GAN was proposed by Goodfellow, it has been widely used to generate new class samples to solve the imbalanced problem. And the experimental results show that oversampling based on GAN can significantly improve the classification performance of the minority class. However, most researchers focus on solving the imbalance problem of image data set, and few on the numerical data. Actually, the image data are very different from the numerical data. Therefore, when solving the imbalanced problem of the network security data, it is necessary to adopt GAN-based methods to improve the detection performance of network threats.

2.2. GAN Training Imbalanced. GAN is one of the hottest deep learning models, but it is hard to train, and the Nash equilibrium is almost impossible to achieve between the discriminator and generator during the training. Training instability is thought as one of the defects [19, 20]. Many researchers have put forward their own ideas to solve the problem and design improved models suitable for specific tasks. New research articles about modified versions of GAN are published week by week, with the names such as 3D-GAN, BEGAN, and iGAN. Avinash Hindupur builds a website called “The GAN Zoo” [24] on GitHub and lists the new GAN models.

Many variants based on GAN have been proposed to improve the training stability and performance. According to the composition and loss, GAN models can be divided into two categories: architecture variants and loss variants. The former ones are to benefit for specific applications, and the latter ones are to improve the performance and enable more stable training [19]. The schemes belonging to loss variants are divided into three types: restricting loss function of the generator [25], restricting loss function of the discriminator [26, 27], and simultaneous restricting loss function of the generator [28, 29]. Our work focuses on the training stability to achieve approximately the Nash equilibrium by restricting loss function of the discriminator to improve the performance.

Heusel et al. [30] proposed a two time-scale update rule (TTUR) for training GAN models with stochastic gradient descent on loss functions. The TTUR defined different individual learning rates for both the discriminator and the generator and proved the learning rates in depth. Finally, the TTUR converged under a local Nash equilibrium. Experiments showed that the TTUR improved the performance of the generator on image data sets.

Brock et al. [31] studied the instabilities of the large-scale GANs. On one hand, two simple, general architectures with two to four parameters and eight times the batch size were introduced to improve scalability. On the other hand, the generator used orthogonal initialization to make it suitable for a simple “truncation trick,” which got the tradeoff between sample variety and fidelity by reducing the variance of generator inputs. The experimental results achieved natural images of multiple categories and improved the conditioning, demonstrably boosting performance of GANs.

Mescheder et al. [32] pointed out that unregularized GAN training was not always convergent locally and the Nash equilibrium led GAN hard to achieve training stability. They analyzed the convergence properties and discussed some regularization strategies, which made GAN training stable including adding instance noise and gradient penalties. In the experiment part, some gradient penalties were used to prove local convergence for the regularized GAN training dynamics.

Zadorozhnyy et al. [33] proposed a new family of discriminator loss functions consisting of real and fake data. According to the gradients of the loss, weights were computed adaptively to train a discriminator to obtain the stability. These methods can be used for any discriminator

model with a loss summed of the real and fake data. The experiment results showed that the methods were more effective on the image generation task.

Lee and Seok [34] proposed that it was essential to maintain balanced training between the discriminator and generator. They thought multiple updates and imbalanced learning rates could train balance. The two approaches directly trained the discriminator severer than the generator. However, the learning rates were dynamic according to the different GAN architecture, data sets, and tasks, which needed to be further studied to select the suitable number of multiple updates and values of learning rates.

Sidheekh et al. [35] demonstrated that it was a problem to use duality gap to monitor the training progress of a GAN in the prevalent approach and proposed a dependable measure to estimate the duality gap based on the locally perturbed gradient descent, which could overcome the limitations of duality gap's capability. In the experiments, the researchers demonstrated their proposed method based on duality gap could influence the training process, such as tuning hyperparameters, a wide variety of GAN models and data sets. And the proposed method had ability to identify whether model convergence or divergence could measure the potential performance of GANs.

The training of GAN is an important and difficult problem. At present, many schemes have been put forward. The ideas of most algorithms are to balance the training of GAN by changing the loss of the generator and discriminator. Inspired by this, we can judge the convergence of the discriminator by calculating the loss trend of the discriminator in the training process and use it as the basis for determining the epochs of subsequent training.

3. Proposed Solution

Our proposed network threat detection model based on CWGAN is shown in Figure 1. The architecture consists of three parts: data segmentation, deep learning for oversampling, and shallow learning for classification. The details and process of the architecture are as follows:

- (1) Data segmentation: at first, the original imbalanced data sets are divided into the training data sets and testing data sets, respectively. Both the training data set and testing data set are imbalanced, including the majority class and the minority class. Then, the minority class is input to the deep learning model for oversampling.
- (2) Deep learning for oversampling: CWGAN works as a deep learning model, containing a generator and a convergent discriminator. The structures of generator and discriminator are composed of full connection layers. The generator is used to generate new data close to the real data. And the discriminator is used to distinguish the real minority data and the false data generated by the generator as far as possible.

Overall, the training process of CWGAN contains multiple iterations. In each iteration, the training epochs of the discriminator are dynamic, which is determined by the convergence of loss function in the last two iterations. In each iteration, the training process is divided into two stages. The first stage is to fix the discriminator to train the generator. At the beginning, the generator is relatively weak, and the generated data are easy to be identified by the discriminator. With the training keep on, the performance of the generator is improved. The second stage is to fix the generator to train the discriminator for distinguishing the real data and the generated false data. The loss function of the discriminator is computed between the real data and false data. Then, the convergence of the loss function is proved through the convexity and Lipschitzness continuity conditions. When the discriminator converges, the epochs of the loss of the previous iteration and that of the current iteration are computed, which are to determine the number of iterations in the next iteration. Therefore, the epoch is used as the index to balance the two iterations of convergence.

- (3) Shallow learning for classification: after the CWGAN generates the new minority samples for the minority class in the training data sets, the new minority samples and the original training data sets are fused to form the new balanced training data sets. Then, the new balanced training data sets are input to shallow learning models to train the model. Then, the imbalanced testing data sets are input to the trained shallow learning models to predict the labels. At the same time, the accuracy of the majority class in the testing data set as $\text{acc}+$, the accuracy of the minority class in the testing data set as $\text{acc}-$, and accuracy, precision, recall, $F1$, and G -means of the whole testing data set are computed to evaluate the performances.

3.1. Data Segmentation and Data Imbalanced. The original imbalanced data sets are divided into training data sets and testing data sets, respectively. Both the training data set and testing data set are imbalanced, including the majority class and the minority class. The data object we deal with is the training set.

Suppose the training set is $X = \{x_1, x_2, \dots, x_N\}$ and the size is N . The majority class is $X_d = \{x_{d1}, x_{d2}, \dots, x_{dN_+}\}$, and the size is N_+ . The minority class is $X_s = \{x_{s1}, x_{s2}, \dots, x_{sN_-}\}$, whose size is N_- . The imbalance ratio (IR) of the training set is computed based on the size of the majority class divided by the size of the minority class, which is defined as follows:

$$\text{IR} = \frac{N_+}{N_-}, \quad (1)$$

FIGURE 1: The architecture of the proposed network threat detection model.

IR indicates the degree of the imbalanced data sets. The higher the IR is, the more imbalanced the data set is, and the more data need to be generated. In order to achieve the balance of different class samples, it is necessary to generate $(N_+ - N_-)$ new samples for the minority class. $(N_+ - N_-)$ is close to $(IR - 1)N_-$, which can also determine the number of the generated samples.

3.2. Deep Learning Model Training. The WGAN model is built with a generator and a discriminator. The generator is used to generate new samples for the minority class. And the discriminator is used to distinguish the real minority samples and the generated samples. Suppose the input data of the generator are the noise data X_{Noise} , and the input data of the discriminator are the real minority data X_s and the false data generated by the generator are X_f .

The WGAN model uses full connection layers to construct the structure of the generator and discriminator. The loss function is computed based on Wasserstein distance, which is defined as

$$W(P_d, P_G) = \inf_{\gamma \sim \Pi(P_d, P_G)} \mathbb{E}_{(X_s, X_f) \sim \gamma} [\|X_s - X_f\|], \quad (2)$$

where P_G is the output data of the generator and P_d is the output data of the discriminator. $W(P_d, P_G)$ is the set of all possible joint distributions of the combination of P_G and P_d . For the joint distribution γ , the data set X_s and X_f conform to the distribution $(X_s, X_f) \sim \gamma$. $\|X_s - X_f\|$ represents the

distance between the samples. The expected value of sample distance under the joint distribution is $\mathbb{E}_{(X_s, X_f) \sim \gamma} [\|X_s - X_f\|]$, whose lower bound is defined as Wasserstein distance.

In order to solve the equation, the K -Lipschitz constraint is introduced, which is defined as

$$|f_w(X_s) - f_w(X_f)| \leq K * \|X_s - X_f\|, \quad (3)$$

where $f_w(\cdot)$ is the function of the neural network model and K is the Lipschitz constant.

The approximate solution of formula (2) becomes

$$K * W(P_d, P_G) = \max_{w: |f_w|_L \leq K} \mathbb{E}_{x \sim P_d} [f_w(X_f, X_s)] - \mathbb{E}_{x \sim P_G} [f_w(X_f)]. \quad (4)$$

The parameter K makes the gradient K times larger but does not affect the direction of the gradient. Formula (4) can be expressed as a discriminator network with parameter w and the last layer is not a nonlinear active layer. Then, formula (4) takes the upper boundary of all functions $f_w(\cdot)$, which satisfies the K -Lipschitz restriction. Then, the Wasserstein distance is converted to

$$\mathcal{L} = \mathbb{E}_{x \sim P_d} [f_w(X_f, X_s)] - \mathbb{E}_{x \sim P_G} [f_w(X_f)]. \quad (5)$$

Next, the generator is to approximately minimize the Wasserstein distance, which is equivalent to minimize $\mathcal{L}$ in formula (5). Considering that the first term of formula (5) is

independent of the generator, we can get the discriminator loss and generator loss of WGAN in formulas (6) and (7).

$$\mathcal{L}_D = \mathbb{E}_{x \sim P_G} [f_w(X_f)] - \mathbb{E}_{x \sim P_d} [f_w(X_f, X_s)], \quad (6)$$

$$\mathcal{L}_G = -\mathbb{E}_{x \sim P_G} [f_w(X_f)]. \quad (7)$$

Formula (6) is the inverse of formula (5), and formula (6) can indicate the training process. The smaller the value of formula (6) is, the smaller the Wasserstein distance between the real data and the generated data is, and the better WGAN training is.

3.3. Model Training Convergence. After the model is established, the model enters into the iterative process of training. The whole iterative process contains many iterations, and each iteration contains many epochs. In each iteration, the generator and the discriminator are trained alternately. The training process is the process of reducing the discriminator loss and the generator loss. In our work, we design a convergence rule, when the generator is fixed, the number of training epochs of the discriminator is not fixed, which is determined dynamically according to the convergence of the loss function.

Assuming that the distribution of the generated data is similar to that of the real data, in one iteration of training process, the discriminator's recognition ability is

continuously enhanced and eventually converges. The convergent discriminator can correctly classify the generated data and the real data.

When the discriminator converges, the epochs in the next iteration are determined by the change times of the discriminator loss between the previous iteration and the current iteration. Therefore, we use the change times of the discriminator loss as the index to balance the training of the discriminator in the next iteration.

3.3.1. Convergence Proof of Cost Function. Suppose the WGAN model trains S iterations. In each iteration, the generator trains one epoch, while the discriminator trains T epochs. In t_s -th epoch in s -th ($s = 0, 1, \dots, S$) iteration, suppose the loss function of the discriminator is $\mathcal{L}_D^{(s, t_s)}(f_w(x))$, which is convex and K -Lipschitzness. The updating of the discriminator loss function is

$$x^{t_s+1} = x^{t_s} + \alpha^{(s, t_s)} * \nabla \mathcal{L}_D^{(s, t_s)}, \quad (8)$$

where $x \in [X_f, X_s]$, and $\alpha^{(s, t_s)}$ is the stride. Suppose x^* is the optimal solution of $\mathcal{L}_D^{(s, t_s)}(f_w(x))$, which is received at the t_s -th iteration. $\mathcal{L}_D^*$ is the optimal value of $\mathcal{L}_D^{(s, t_s)}(f_w(x))$ and expressed as $\mathcal{L}_D^{(s, t_s)}$. K is the Lipschitzness constant, and the value is 1. Therefore, we can get

$$\begin{aligned} \|x^{t_s+1} - x^*\|^2 &= \|x^{t_s} - x^*\|^2 - 2\alpha^{(s, t_s)} \langle \nabla \mathcal{L}_D^{(s, t_s)}, x^{t_s} - x^* \rangle + \left(\alpha^{(s, t_s)}\right)^2 \|\nabla \mathcal{L}_D^{(s, t_s)}\|^2 \\ &\leq \|x^{t_s} - x^*\|^2 - 2\alpha^{(s, t_s)} \left(\mathcal{L}_D^{(s, t_s)} - \mathcal{L}_D^*\right) + \left(\alpha^{(s, t_s)}\right)^2 \|\nabla \mathcal{L}_D^{(s, t_s)}\|^2 \\ &= \|x^{t_s} - x^*\|^2 - \frac{\left(\mathcal{L}_D^{(s, t_s)} - \mathcal{L}_D^*\right)^2}{\|\mathcal{L}_D^{(s, t_s)}\|^2} \\ &\leq \|x^{t_s} - x^*\|^2 - \frac{\left(\mathcal{L}_D^{(s, t_s)} - \mathcal{L}_D^*\right)^2}{K^2}. \end{aligned} \quad (9)$$

In formula (9), the first inequality satisfies the convexity of functions and the second inequality is based on the

Lipschitzness continuity condition. By accumulating formula (9), the result is obtained as

$$\begin{aligned} \frac{1}{K^2} \sum \left(\mathcal{L}_D^{(s, t_s)}(f_w(x)) - \mathcal{L}_D^*\right)^2 &\leq \sum_{t_s=0}^{T_s} \|x^{(s, t_s)} - x^*\|^2 - \|x^{(s, t_s)+1} - x^*\|^2 \\ &= \|x^0 - x^*\|^2 - \|x^{(s, t_s)+1} - x^*\|^2. \end{aligned} \quad (10)$$

Multiply both sides of formula (10) by K^2 to get

$$\sum \left(\mathcal{L}_D^{(s,t_s)}(f_w(x)) - \mathcal{L}^* \right)^2 \leq K \|x^0 - x^*\|^2. \quad (11)$$

According to the right bounded, we can get

$$\mathcal{L}_D^{(s,t_s)}(f_w(x)) \longrightarrow \mathcal{L}_{\mathcal{D}}^*, \quad k \longrightarrow \infty. \quad (12)$$

Under the condition of $\mathcal{L}_{D,\text{Best}}^{(s,t_s)} = \min_{t_s} (\mathcal{L}_D^{(s,t_s)})$, we can get

$$N_+ \left(\mathcal{L}_{D,\text{Best}}^{(s,t_s)} - \mathcal{L}_{\mathcal{D}}^* \right) \leq \sum_{k=0}^m \left(\mathcal{L}_D^{(s,t_s)} \longrightarrow \mathcal{L}_{\mathcal{D}}^* \right) \leq \|x^0 - x^*\|^2. \quad (13)$$

In this paper, the data are high dimensional. We use the Wasserstein distance to measure the distance between generated data and real data. Suppose W_k represents the Wasserstein distance between generated data and real data when the discriminator is convergent at the k -th iteration. The formula (13) is transformed to

$$\mathcal{L}_D^{(s,t_s)} - \mathcal{L}_{\mathcal{D}}^* \leq \frac{K * W_k}{\sqrt{N_+}}. \quad (14)$$

3.3.2. Calculation of Discriminator Training Epochs. Under the conditions of formulas (12)–(14), the minimum loss of convergence in the $s-1$ iteration is $\mathcal{L}_D^{(s,t_{s-1})}$, and the minimum loss of convergence in the s iteration is $\mathcal{L}_D^{(s,t_s)}$. So the training epoch number of the $s+1$ iteration discriminator is defined as

$$t_{s+1} = \begin{cases} \left\lfloor \frac{\mathcal{L}_D^{(s,t_{s-1})}}{\mathcal{L}_D^{(s,t_s)}} \right\rfloor, & \frac{\mathcal{L}_D^{(s,t_{s-1})}}{\mathcal{L}_D^{(s,t_s)}} \geq 1, \\ \left\lfloor \frac{\mathcal{L}_D^{(s,t_{s-1})}}{\mathcal{L}_D^{(s,t_s)} + 1} \right\rfloor + 1, & 0 < \frac{\mathcal{L}_D^{(s,t_{s-1})}}{\mathcal{L}_D^{(s,t_s)}} \leq 1, \end{cases} \quad (15)$$

where $\lfloor \cdot \rfloor$ is the rounded-down value of the data.

That is to say, the epoch number of discriminator training in each iteration depends on the ratio of the minimum loss of the previous two iterations of discriminator training convergence, which is rounded down. If the ratio is rounded down to 0, the epoch of discriminator training is 1.

Moreover, it is noted that there are many math symbols in this section. So, we make a new symbol table for all used math symbols in Table 1.

3.4. Shallow Machine Learning Classifier. Shallow machine learning has good performance and high efficiency to learn and analyze the data [36]. There are many shallow machine learning classifiers, e.g., NB, RF, and LR. Based on our previous experimental results and analysis of the existing literature, we find that SVM has some advantages, such as the stability for maintaining good classification performance, the ability of dealing with the noise and outlier data, and effectiveness for handling the nonlinear and high-dimensional data. Therefore, we choose SVM as the classifier in our experiment.

Shallow machine learning consists of two stages: training stage and testing stage. In the training stage, the generated samples based on CWGAN combined with the original training data set are input to the shallow machine learning classifier to train and obtain the optimal model structure. In the testing stage, the imbalanced testing data set is input to the trained shallow machine learning model to get the labels of the predicted testing data.

In the experiment, the true labels of the testing data set have been known, so the performance of the shallow machine learning models, such as accuracy, precision, recall, and F1, can be obtained by comparing the true labels with the predicted labels and calculating the confusion matrix. In this work, the accuracy of the majority class in the testing data set as acc+, the accuracy of the minority class in the testing data set as acc-, and accuracy, precision, recall, F1, and G-means of the whole testing data set are computed to evaluate the performances.

The confusion matrix for binary classification includes four index items, such as true positive (TP), false negative (FN), false positive (FP), and true negative (TN). The accuracy of the minority class in the testing data set as acc- is computed on the minority class:

$$\text{acc-} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}. \quad (16)$$

The accuracy of the majority class in the testing data set as acc+ is computed on the majority class:

$$\text{acc+} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}. \quad (17)$$

Accuracy, precision, recall, F1, and G-means of the whole testing data set are computed:

$$\begin{aligned} \text{accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}, \\ \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ F1 &= \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \\ &= \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FN} + \text{FP}} \quad (\beta = 1). \end{aligned} \quad (18)$$

4. Experiments

4.1. Data Set. The data sets are downloaded from the website of Canadian Institute for Cybersecurity, which is based at the University of New Brunswick in Fredericton [37]. There are many security data collected on the website, which are used by companies, research centers, and universities. In our experiment, we download the URL data set, which contains some different types of URLs. And 79 features are selected from the original data, such as URL, domain, path, file name,

TABLE 1: The symbols are used in the formulas from (1) to (15) in Section 3.3.

No.	Math symbols	Meaning of the math symbols
1	X	The training set
2	N	The size of the training set X
3	X_d	The majority class of the training set
4	N_+	The size of the majority class in the training set
5	X_s	The minority class of the training set
6	N_-	The size of the minority class in the training set
7	IR	The imbalance ratio of the imbalanced data sets
8	R	The oversampling rate of the minority class
9	X_f	The false data generated by the generator of GAN
10	P_G	The output data of the generator
11	P_d	The output data of the discriminator
12	$W(P_d, P_G)$	The set of all possible joint distributions of the combination of P_G and P_d
13	γ	One joint distribution of all possible joint distributions of the combination of P_G and P_d
14	$f_w(\cdot)$	The function of neural network model
15	K	The Lipschitz constant
16	$\mathcal{L}$	The Wasserstein distance
17	S, s	The iterations in the training process of CWGAN
18	T, t	The training epoch in each iteration
19	t_s	The training epoch in s iteration
20	$\mathcal{L}_D^{(s, t_s)}(f_w(x))$	The loss function of the discriminator
21	$\alpha^{(s, t_s)}$	The training stride in the training process of CWGAN
22	x^*	The optimal solution of $\mathcal{L}_D^{(s, t_s)}(f_w(x))$
23	$\mathcal{L}_D^*$	The optimal value of $\mathcal{L}_D^{(s, t_s)}(f_w(x))$
24	W_k	The Wasserstein distance between generated data and real data when the discriminator is convergent at the k -th iteration.

and argument [38]. Then, we just extract some samples from the data set to form some imbalanced data sets, ranking from low to high according to IR. The details of the data sets are shown in Table 2.

Benign URLs. They are collected from Alexa top websites. First, the domains are crawled by a Heritrix web crawler. Then, around half a million unique URLs are extracted from the domains. Later, the URLs are filtered by VirusTotal from the extracted URLs.

Malware URLs. They link to the malicious websites and are owned to DNS-BH, which maintains a list of the malicious sites.

Defacement URLs. They are fraudulent or hidden URLs ranked by Alexa and linked to the malicious Web pages.

Spam URLs. They are collected from the publicly available WEBSpam-UK2007 data set.

In the experiments, the original data set is divided into the training set and the testing set according to the ratio of 7 : 3. The training set is used to train the model, and the testing set is used to validate and evaluate the model.

All experiments are preformed in JetBrains PyCharm 2017 with the Python 3.6 interpreter on a laptop with Intel CORE i5-6200U 2.3 GHz with 8 GB RAM running on the Windows 10 OS.

4.2. Experiment Results and Discussion

4.2.1. Performance of Original Imbalanced Data Set. In the first part of the experiments, the performances of the original

imbalanced data sets are computed. First, the original imbalanced data sets are divided proportionally into the training data sets and testing data sets, respectively. Both the training data set and testing data set are imbalanced. Second, the training data set is input to SVM to train the model. SVM is viewed as a traditional and effective classifier. Third, the testing data set is input to the trained SVM to predict the labels. At the same time, the accuracy of the majority class in the testing data set as acc+, the accuracy of the minority class in the testing data set as acc-, and accuracy, precision, recall, F1, and G-means of the whole testing data set are computed to evaluate the performances, which are shown in Table 3. In addition, it should be noted that the recorded results are the average of 5 experiments.

According to the performances of original imbalanced data sets in Table 3, we can find some interesting results. First, with the increase of IR, the acc+ values representing the accuracy of the majority class are increasing. By comparison, the acc- values representing the accuracy of the minority class are decreasing. These results are consistent with the conclusion, which is that the boundaries of the classifiers tend to be the majority class and lead to the classification error of the minority class. Second, with the increase of IR, accuracy, precision, recall, and F1 of the whole testing data sets are increasing, and also G-means of the whole testing data sets are increasing. That is because accuracy, precision, recall, and F1 represent the overall performance of the testing data sets. The accuracy of the majority class can ensure the overall accuracy. The growth trend of acc- is the same as that of G-means. Both of them decrease with the decrease of IR. So, G-means can represent the performance of the minority class. Furthermore, we can obtain

TABLE 2: Details of the data sets.

Data sets	Component	Number of features	Number of samples (normal/abnormal)	IR
D1	Benign/malware	79	2709/1500	1.81
D2	Benign/defacement	79	2709/1300	2.08
D3	Benign/malware	79	2709/1000	2.71
D4	Benign/defacement	79	2709/800	3.39
D5	Benign/defacement	79	2709/500	5.42
D6	Benign/spam	79	2709/300	9.03
D7	Benign/spam	79	2709/140	19.35
D8	Benign/defacement	79	2709/54	50.17

TABLE 3: The performances of original imbalanced data sets.

Data sets	IR	Acc+	Acc−	Accuracy	Precision	Recall	F1	G-means
D1	1.81	0.981	0.809	0.919	0.921	0.981	0.945	0.866
D2	2.08	0.994	0.778	0.924	0.904	0.994	0.947	0.878
D3	2.71	0.982	0.828	0.940	0.941	0.982	0.960	0.900
D4	3.39	0.997	0.709	0.932	0.922	0.997	0.958	0.836
D5	5.42	0.997	0.656	0.944	0.943	0.997	0.968	0.777
D6	9.03	1.000	0.323	0.932	0.930	1.000	0.964	0.560
D7	19.35	1.000	0.171	0.959	0.959	1.000	0.979	0.404
D8	50.17	1.000	0.109	0.984	0.984	1.000	0.992	0.289

TABLE 4: The performances based on WGAN.

Data sets	The number of the new samples	Acc−	Acc+	Accuracy	Precision	Recall	F1	G-means
D1	1209	0.825	0.978	0.924	0.926	0.978	0.948	0.878
D2	1409	0.808	0.983	0.926	0.915	0.983	0.948	0.891
D3	1709	0.859	0.980	0.947	0.951	0.980	0.965	0.916
D4	1909	0.766	0.989	0.938	0.936	0.989	0.962	0.867
D5	2209	0.726	0.990	0.949	0.953	0.990	0.971	0.837
D6	2409	0.340	0.883	0.829	0.923	0.883	0.902	0.545
D7	2569	0.293	0.951	0.919	0.963	0.951	0.956	0.520
D8	2655	0.573	0.992	0.984	0.992	0.992	0.992	0.729

TABLE 5: The performances based on CWGAN.

Data sets	The number of the new samples	Acc−	Acc+	Accuracy	Precision	Recall	F1	G-means
D1	1209	0.881	0.975	0.942	0.942	0.975	0.957	0.923
D2	1409	0.802	0.984	0.925	0.913	0.984	0.947	0.887
D3	1709	0.885	0.979	0.954	0.960	0.979	0.969	0.930
D4	1909	0.772	0.979	0.932	0.937	0.979	0.957	0.866
D5	2209	0.736	0.982	0.944	0.954	0.982	0.968	0.839
D6	2409	0.680	0.937	0.912	0.963	0.937	0.950	0.798
D7	2569	0.643	0.922	0.908	0.980	0.922	0.950	0.765
D8	2655	0.573	0.994	0.986	0.992	0.994	0.993	0.719

some conclusions. On one hand, the overall performance of the whole data sets cannot reflect the performance of the imbalanced data. On the other hand, when the IR is very high, the accuracy of imbalanced data is very low.

In the field of cyberspace security, network threat data are usually viewed as the minority class. If the minority data are misclassified so that achieve a low accuracy, the consequences can be serious. Therefore, it is necessary to solve the problem of the imbalanced data set. This is the goal of our work. In the following parts, we will introduce the results of our proposed method to generate the sufficient new minority samples to solve the imbalanced problem and improve the accuracy of the minority class.

4.2.2. Comparison between WGAN and CWGAN.

WGAN can generate the new data based on the original data set to solve the problem of the imbalanced data set. In this work, we propose CWGAN to improve the training stability to improve the performance of WGAN. In this part, the performances of WGAN and CWGAN are computed and compared. First, the original imbalanced data sets are divided proportionally into training data sets and testing data sets. The number of the needed new minority samples of the training data sets equals the number of the majority class minus the number of the minority class. Second, WGAN and CWGAN work to generate the new minority samples for the training data sets. Then, the new minority samples and

FIGURE 2: The discriminator loss of WGAN and CWGAN on different data sets. (a) Discriminator loss on D1. (b) Discriminator loss on D2. (c) Discriminator loss on D3. (d) Discriminator loss on D4. (e) Discriminator loss on D5. (f) Discriminator loss on D6. (g) Discriminator loss on D7. (h) Discriminator loss on D8.

FIGURE 3: The training convergence of CWGAN.

FIGURE 4: Performance comparison between the oversampling methods. (a) acc_- . (b) G-means. (c) acc_+ . (d) Accuracy.

the training data sets are fused to form the new balanced training data sets. Third, the new balanced training data sets are input to SVM to train the model. Then, the testing data sets are input to the trained SVM to predict the labels. At the same time, $\text{acc}+$ and $\text{acc}-$, accuracy, precision, recall, $F1$, and G-means are computed to evaluate the performances, which are shown in Tables 4 and 5. In addition, it should be noted that the numbers of training iterations of WGAN and CWGAN both are 1000. The recorded results are the average of 5 experiments.

The performances in Tables 4 and 5 are computed based on the output of the trained SVM on the balanced training data sets, which are fused with the new generated minority samples based on WGAN and CWGAN. On the whole, comparing the results in Tables 3–5, we can conclude that $\text{acc}-$ and G-means in Tables 4 and 5 increase obviously compared with that in Table 3; $\text{acc}+$, accuracy, precision, recall, and $F1$ in Tables 4 and 5 decrease a little compared with that in Table 3. That is because WGAN and CWGAN generate some new generated minority samples to make the original data sets balance, which train SVM to achieve a model that focuses on the minority class. Comparing the results in Tables 4 and 5, $\text{acc}-$ and G-means in Table 5 based on CWGAN are higher than that in Table 4 based on WGAN, especially when IR is high; $\text{acc}+$, accuracy, precision, recall, and $F1$ of some data sets in Table 5 are a little higher than that in Table 4. Therefore, we can conclude that the performances based on CWGAN generating the new minority samples outperform those based on WGAN.

4.2.3. Discriminator Loss of WGAN and CWGAN. During the training stage, the discriminator loss is achieved based on the cross entropy loss function to compare the probability that the predicted labels of the generated data are close to that of the real data. The smaller the discriminator loss, the closer the generated data to the real data. In this section, we study the trend of the discriminator loss of WGAN and CWGAN on different data sets. The epoch of WGAN in each iteration is fixed to 5. The epoch of CWGAN in each iteration is computed when the model is trained to convergence in one iteration. The numbers of the training iterations of WGAN and CWGAN are 1000. The loss of each iteration in one experiment is recorded and plotted in Figure 2.

Figures 2(a)–2(h) shows the discriminator loss curves of WGAN and CWGAN on D1 to D8. These curves show that the discriminator loss of CWGAN is smaller and more stable than that of WGAN with the increase of the iterations, and the values of the discriminator loss of CWGAN are very close 0. The discriminator loss measures the similarity between the generated data and the real data. The results reflect that the generated data based on CWGAN are closer to the real data than that based on WGAN. Therefore, combined with the experimental results in Section 4.2.2, we can ensure that the performance of CWGAN outperforms that of WGAN.

4.2.4. Training Convergence of CWGAN. The training of WGAN with a generator and a discriminator is imbalanced. To make the training process trend to Nash equilibrium, we fix the training epoch of the generator and compute dynamically the training epoch of the discriminator according to the convergence of the training loss. In each iteration, the value of the training epoch of the discriminator is determined by the ratio of training loss in the last two iterations. In this part, we compute the training epoch of the discriminator in each iteration on different data sets. The results are plotted in Figure 3. In addition, it should be noted that the number of the training iterations of CWGAN is 1000. The recorded results are from one of some experiments.

Figure 3 shows the training epoch of the discriminator in each iteration on different data sets. We divide the training process into two stages. In the early stage, the values of the training epoch change frequently and are unstable. In the later stage, the values of the training epoch are stable. In addition, in most cases, when the training process converges, the higher the IR, the more samples generated by CWGAN, and the higher the training epoch. Such as the training epoch of D8 is highest, whose IR is highest.

4.2.5. Comparison between the Oversampling Methods. At present, there are many oversampling methods, such as VAE and SMOTE. In this section, we compare the performance between the oversampling methods. The experimental process of the oversampling methods is similar to that in Section 4.2.2. First, the oversampling methods, including VAE, SMOTE, WGAN, and CGWGAN, are applied to generate the minority examples and combined with the original training data sets to train SVM. Second, the testing data sets are input to the trained SVM to predict the labels. At the same time, $\text{acc}+$, $\text{acc}-$, accuracy, and G-means are computed to evaluate the performances, which are plotted in Figure 4. In addition, it should be noted that the numbers of training iterations of WGAN and CWGAN both are 1000. The recorded results are the average of 5 experiments.

In Figure 4(a), the accuracy of the minority data $\text{acc}-$ is highest on the same data sets based on CWGAN. Especially, on D6, D7, and D8, whose IR is higher, the accuracy on most data sets are improved most after CWGAN generating the minority samples to make the imbalanced data sets balance. In Figure 4(b), the comparison on G-means is similar to that of $\text{acc}-$. G-means on most data sets based on CWGAN are highest. We can conclude that the new generated minority samples based on CWGAN can improve the performance of the minority class most. In Figures 4(c) and 4(d), in most cases, the accuracy of the majority class $\text{acc}+$ and the accuracy of the whole data are close to those of the original data, which means that CWGAN cannot weaken the accuracy of the majority class and the whole data.

5. Conclusion

WGAN as a generative model can solve the imbalanced data problem through generating new samples for the minority

class. However, it may suffer mode collapse and the training cannot converge sometimes. In this paper, we improved the training process to realize the training stability of WGAN by proposing a convergent WGAN-based oversampling model called CWGAN. In each iteration of the training process, the epoch of the generator is fixed, but the epoch of the discriminator is dynamic, which is determined by the convergence of discriminator loss function in the last two iterations. The experiment results showed that CWGAN not only improves the training stability of WGAN on the loss smooth and more close to 0 but also improves the performance of the minority class through oversampling and ensure the accuracy of the majority class and the whole data. Compared with other oversampling methods, it can be concluded that CWGAN performs better and enables stable training.

Data Availability

The data in experiments are downloaded from the public websites. The links are cited in [37]. In addition, the datasets are also available from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper was supported by the National Natural Science Foundation of Zhejiang Province (nos. LY20F020012 and LQ19F020008) and the National Natural Science Foundation of China (no. 61802094).

References

- [1] T. Ling, L. Chong, X. Jingming, and C. Jun, "Application of self-organizing feature map neural network based on K-means clustering in network intrusion detection," *Computers, Materials & Continua*, vol. 61, no. 1, pp. 275–288, 2019.
- [2] S. Zou, H. Sun, G. Xu, and R. Quan, "Ensemble strategy for insider threat detection from user activity logs," *Computers, Materials & Continua*, vol. 65, no. 2, pp. 1321–1334, 2020.
- [3] J. Zhang, S. Zhong, T. Wang, H. C. Chao, and J. Wang, "Blockchain-based systems and applications: a survey," *Journal of Internet Technology*, vol. 21, no. 1, pp. 1–14, 2020.
- [4] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems," *International Journal of Engineering & Technology*, vol. 7, no. 3, pp. 479–482, 2018.
- [5] Y. Xu, C. Wu, K. Zheng, X. Niu, and Y. Yang, "Fuzzy-synthetic minority oversampling technique: oversampling based on fuzzy set theory for Android malware detection in imbalanced datasets," *International Journal of Distributed Sensor Networks*, vol. 13, no. 4, pp. 1–15, 2017.
- [6] M. Luo, K. Wang, Z. Cai, A. Liu, Y. Li, and C. C. Fong, "Using imbalanced triangle synthetic data for machine learning anomaly detection," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 15–26, 2019.
- [7] J. Błaszczyński and J. Stefanowski, "Neighbourhood sampling in bagging for imbalanced data," *Neurocomputing*, vol. 150, pp. 529–542, 2015.
- [8] M. Bekkar and T. A. Alitouche, "Imbalanced data learning approaches review," *International Journal of Data Mining & Knowledge Management Process*, vol. 3, no. 4, pp. 15–33, 2013.
- [9] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," *Expert Systems with Applications*, vol. 174, Article ID 114582, 2021.
- [10] H. H. Haibo and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
- [11] Y. Yang, Q. Zhao, L. Ruan, Z. Gao, Y. Huo, and X. Qiu, "Oversampling methods combined clustering and data cleaning for imbalanced network data," *Intelligent Automation & Soft Computing*, vol. 26, no. 5, pp. 1139–1155, 2020.
- [12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [13] M. Antonelli, P. Ducange, and F. Marcelloni, "An experimental study on evolutionary fuzzy classifiers designed for managing imbalanced datasets," *Neurocomputing*, vol. 146, pp. 125–136, 2014.
- [14] K. Deepshikha and A. Naman, "Removing Class Imbalance Using Polarity-GAN: An Uncertainty Sampling Approach," 2020, <https://arxiv.org/abs/2012.04937>.
- [15] J. Wang, Y. Yang, T. Wang, R. Sherratt, and J. Zhang, "Big data service architecture: a survey," *Journal of Internet Technology*, vol. 21, no. 2, pp. 393–405, 2020.
- [16] M. Quintana, S. Schiavon, and K. W. Tham, "Balancing thermal comfort datasets: we GAN, but should we," in *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pp. 120–129, NY, USA, November 2020.
- [17] I. Goodfellow, J. A. Pouget, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [18] X. Li, Y. Liang, M. Zhao, and Y. Jiang, "Few-shot learning with generative adversarial networks based on woa13 data[J]," *Computers, Materials & Continua*, vol. 58, no. 2, pp. 1073–1085, 2019.
- [19] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks in computer vision," *ACM Computing Surveys*, vol. 54, no. 2, pp. 1–38, 2021.
- [20] J. Gui, Z. Sun, Y. Wen et al., "A Review on Generative Adversarial Networks: Algorithms, Theory, and applications," 2020, <https://www.arxiv-vanity.com/papers/2001.06937>.
- [21] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein Gan," pp. 1–32, 2017, <https://arxiv.org/abs/1701.07875>.
- [22] J. Hao, C. Wang, H. Zhang et al., "Annealing Genetic GAN for Minority Oversampling," 2020, <https://arxiv.org/abs/2008.01967>.
- [23] J. Kim, K. Jeong, H. Choi, and K. Seo, "GAN-based anomaly detection in imbalance problems," in *Proceedings of the Computer Vision - ECCV 2020 Workshops*, pp. 128–145, Glasgow, UK, August 2020.
- [24] <https://github.com/hindupuravinash/the-gan-zoo>.
- [25] T. Chen, M. Lucic, N. Houlsby et al., "On Self Modulation for Generative Adversarial networks," 2018, <https://arxiv.org/abs/1810.01365>.
- [26] I. Gulrajani, F. Ahmed, M. Arjovsky et al., "Improved Training of Wasserstein gans," 2017, <https://arxiv.org/abs/1803.01541>.
- [27] G. J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *International Journal of Computer Vision*, vol. 128, no. 5, pp. 1118–1140, 2020.

- [28] T. Che, Y. Li, A. P. Jacob et al., "Mode Regularized Generative Adversarial networks," 2016, <https://arxiv.org/abs/1612.02136>.
- [29] X. Mao, Q. Li, H. Xie et al., "Least Squares Generative Adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, Venice Italy, October 2017.
- [30] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proceedings of the Advances in Neural Information Processing Systems*, Red Hook USA, December 2017.
- [31] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image synthesis," 2018, <https://arxiv.org/abs/1809.11096>.
- [32] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for gans do actually converge?" in *Proceedings of the International Conference on Machine Learning*, pp. 3481–3490, PMLR, Stockholm, Sweden, July 2018.
- [33] V. Zadorozhnyy, Q. Cheng, and Q. Ye, "Adaptive weighted discriminator for training generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4781–4790, Los Alamitos, CA, USA, June 2021.
- [34] M. Lee and J. Seok, "Regularization Methods for Generative Adversarial Networks: An Overview of Recent studies," 2020, <https://arxiv.org/abs/2005.09165>.
- [35] S. Sidheekh, A. Aimen, V. Madan et al., "On Duality Gap as a Measure for Monitoring GAN Training," 2020, <https://arxiv.org/abs/2012.06723>.
- [36] X. Yu, Z. Tian, J. Qiu, S. Su, and X. Yan, "An intrusion detection algorithm based on feature graph," *Computers, Materials & Continua*, vol. 61, no. 1, pp. 255–274, 2019.
- [37] <https://www.unb.ca/cic/datasets/url-2016.html>.
- [38] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting Malicious Urls Using Lexical analysis," in *Proceedings of the International Conference on Network and System Security*, pp. 467–482, Tianjin, China, October 2021.