Few-Shot Constraint Enhancement Based on Generative Adversarial Networks

Xin Song[0000−0003−3209−3232]1,2, Yanqing Song[0009−0002−3995−6707]3,4,5, Jianguo Chen[0000−0003−4091−3464]6,4,5, and Long Chen[0000−0002−2535−4978]7,8,9

[1] School of Management, Hebei University, Baoding, China
[2] School of Cyber Security and Computer Science, Hebei University, Baoding, China
[3] School of Information Science and Technology, Beijing Forestry University, Beijing, China
[4] China Applied Sciences and Technologies Research, Shenyang, China
[5] Beijing Aerospace Information Sciences and Technologies, Beijing, China
[6] Institute of Geology,China Earthquke Administration, Beijing, China
[7] College of Information Science and Technology, Beijing University of Chemical Technology, Beijing, China
[8] College of Information Science and Technology, Beijing, China
[9] Beijing Iyuba Technology, Beijing, China

chen_long@buct.edu.cn

Abstract. A constrained theoretical model for Generative Adversarial Networks (GANs) is proposed. This model introduces a GAN structure and training process constrained by Directed Graphical Models (DGM) to mitigate issues such as overfitting, convergence difficulties, and mode collapse encountered during GAN training. It addresses the instability and quality concerns of generated samples. Subsequently, a static constraint method is put forward, utilizing the similarity of an interpretable measurement scale (EMS) alongside final classification metrics of generated data across various classifiers. This approach involves setting the topology of discriminators (D) and generators (G), where the EMS is used to measure the constraint strength, thereby mitigating overfitting in the generation process. Moreover, constraining label sharing features and weight updates significantly diminishes the likelihood of mode collapse by imposing appropriate restrictions on the use of label information during generation. This constrained approach to GANs effectively enhances sample quality and stability.

1 Introduction

Constraints are applied to GANs [1,2] to effectively enhance samples. In terms of constraining Generative Adversarial Network(GAN) structures, common methods include adding regularization terms or limiting the parameter range of networks. Specific network topology structures can also be used to constrain the connections between the generator and discriminator. In terms of constraining GAN loss functions, several methods have been proposed. For example, additional loss functions can be introduced to constrain the learning process of the generator and discriminator, or specific loss functions can be used to balance the diversity and authenticity of generated samples [1].

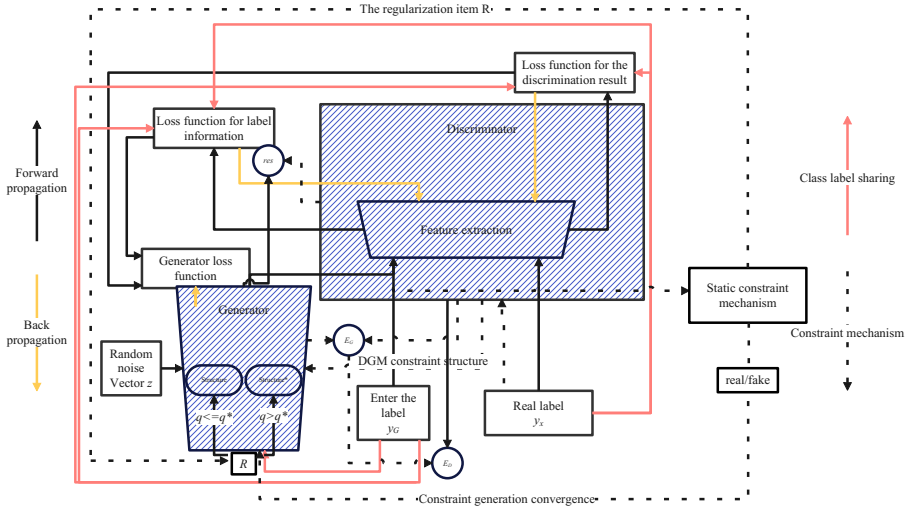

Fig. 1. Constrained Theory Model of Generative Adversarial Networks

However, there are still many shortcomings in practical training [1,2,3,4]. Training instability: The training process of GANs is often unstable and prone to mode collapse or mode collapse problems. One solution to address the challenges of training models with limited labeled data is few-shot data augmentation techniques [5,6,7]. This study proposes a constrained theoretical model for GANs to address the challenges in augmenting few-shot data, including generalization, overfitting, and transferability issues.

2 Constrained Theory of Generative Adversarial Networks

To address the issues of overfitting and mode collapse in GANs, various strategies such as constrained training based on structure and process, parameter optimization, the use of different loss functions, and regularization have been employed to improve the performance and stability of GANs. In this section, we propose a constrained theory model for GANs (Figure 1) which includes the following components: constrained training based on the DGM-GAN structure and process to address the instability of training and the quality of generated samples, static and dynamic constraint GANs which measure the strength of constraints using EMS to suppress overfitting, and the constraint of sharing label information in features and weight updates which effectively reduces the probability of mode collapse and solves the constraint convergence problem of GANs. The specific implementation of these components is described below.

Table 1. Parameter symbol lookup table

Symbols	Meaning	Symbols	Meaning
G	The set of parameters for the generator$G$	θ_q	The q point in the input probability space represents the node in $\Omega_{\Theta,\mathrm{q}_{\leftarrow}}$
D	The set of parameters for the discriminator$D$	θ_j	$\Omega_{\Theta,\mathrm{j}}$in node
E_G	Gloss function	σ	Smoothing parameters of the Dirac function δ
E_D	Dloss function	$\Omega_{\Theta,\mathrm{q}}$	The distribution probability in a region, which represents the number of nodes $\theta_j \in M_n$
f_j	The node weights of the neural network$D$	θ_G	Network parameters for the generator
$\widetilde{f}_j$	The node weights of the neural network$G$	θ_C	Network parameters for the classifier
q	Minimal normalized index of KMMD (index of the current iteration)	θ_D	Network parameters of the discriminator
q^*	Maximum normalized index of KMMD (currently optimized index in training)	z	Noise vector
λ	The strength of the constraint	m	The magnitude of the noise vector
N	Represents the total number of points	δ	Dirac delta function

2.1 Constrained Training Based on the DGM-GAN Structure and Process

In this section, we design a new neural network architecture and optimization strategy based on the DGM-GAN structure and process to generate adversarial training and effectively enhance the performance of the discriminator. The parameter symbols are defined in Table 1, and a directed graph model (DGM) [8] is established to analyze the structure and process of GANs. Different GAN models correspond to different loss functions, such as GAN [9] and WGAN [10]. In DGM, a latent variable res is defined, and the relationships between D↔res and G↔res are established by processing res. Based on the constraint relationships in DGM, formulas for training G and D are derived. By factorizing these formulas, the formulas for training G and D are further simplified. Finally, the learning function is obtained by adding constraints during the learning process. The specific details are elaborated below.

Abstracting GAN components into Hilbert spaces [11] enables computational operations, enhancing performance through structural constraints. Training involves mutual constraints between the generator (G) and discriminator (D) within this framework, aiming to prevent overfitting and improve data generation. The Directed Graph Model (DGM) integrates these constraints, facilitating a structured GAN process that more accurately mimics real data distributions. This approach is supported by theoretical analysis, demonstrating the effectiveness of such constraints in GAN training. The dynamic constraint GAN which constrains the training of GAN by Kernel MMD (KMMD) [12].

In summary, constraints are added during the training process to improve the performance of the generator and discriminator. The specific implementation is as follows:

First, a regularization term $R(\theta_G)$ is defined in this section to measure the constraint relationship between the generator G and discriminator D, where: ρ_{jj° represents the correlation coefficient between two network layers. This regularization term can be calculated using the EMS quantification method as follows:

$$R(\theta_G) = SR = \text{mean}\left(\sum_{i=0}^{n} \max \rho_{jj^\circ}\right) \tag{1}$$

The above constraint conditions are then integrated into the loss function of GAN. For the generator G, $L_{G,\text{original}}$ is the original generator loss function, and λ is a hyper-parameter that controls the strength of the constraint. The loss function is defined as $L_G = L_{G,\text{original}} + \lambda R(\theta_G)$

For the discriminator D, $L_{D,\text{original}}$ is the original discriminator loss function. The loss function is defined as $L_D = L_{D,\text{original}} + \lambda R(\theta_D)$

During the training process, gradient descent is used to optimize the new loss functions L_G and L_D. By adding constraint relationships in the loss function, this section guides the generator and discriminator to learn better parameters, thereby improving their performance. DGM constraints in GANs modulate G and D training, enhancing performance and mitigating overfitting, independent of loss function limitations.

2.2 Static and Dynamic Structural Random Constraints

EMS-imposed static and dynamic constraints on GAN mitigate overfitting through R-SGAN and R-DGAN frameworks, applying theoretical constraints to network structures [13]. Furthermore, the Dirac delta function limits the accuracy of numerical integration in the equation. A direct probability integration method is proposed by studying the distribution of the input probability space and the smoothing method of the Dirac delta δ function [14].

Generating Representative Points and Probability Allocation To propose a method for addressing the constraint problem in GANs, this section first considers a point selection method based on the Generalized F-Divergence (GF-divergence) [15]. This method partitions the input probability space into a set of non-overlapping representative regions based on Voronoi cells.

First, assume there is a generative adversarial network (GAN) with the generator *G* and discriminator *D* controlled by parameters θ_G and θ_D , respectively. The goal of this section is to find a method to constrain the training process of GAN in order to generate better samples in the input probability space. To achieve this, consider partitioning a set of non-overlapping representative regions in the input probability space $\Omega_{\Theta,q}$:

$$\Omega_{\Theta,q} = \{x \in (R)^n : |x - \theta_q| \leq |x - \theta_j|, \forall \theta_j \in M_n, j \neq q\} \tag{2}$$

These representative regions are used to constrain the generator during the GAN training process. Define a regularization term $R\left(\theta_G\right)$, which adjusts the distribution of generated samples in the input probability space θ_G based on the generator parameters . To achieve this, define the following regularization term:

$$R\left(\theta_G\right) = \sum_{q=1}^{Q} w_q \int_{\Omega_{\Theta,q}} D\left(G\left(z, \theta_G\right), \theta_D\right) dz \tag{3}$$

where Q is the number of representative regions, w_q is the weight of the q-th representative region, and z is the input noise for the generator. By adding the regularization term $R(\theta_G)$ to the objective function of the original GAN, with $\lambda > 0$ as a hyperparameter, the following optimization problem is obtained:

$$\begin{aligned} &\underset{\theta_G}{min}\underset{\theta_D}{max}E_{x\sim p_{data}(x)}\left(logD\left(x,\theta_D\right)\right] \\ &+E_{z\sim p_z(z)}\left(log\left(1-D\left(G\left(z,\theta_G\right),\theta_D\right)\right)\right]+\lambda R\left(\theta_G\right) \end{aligned} \tag{4}$$

The regularization term controls the impact of the regularization term on the GAN training process. By adding the regularization term $R(\theta_G)$, this section explores the generation of better samples by guiding the generator to produce representative regions in the input probability space, thereby imposing constraints on the generative adversarial network.

Smoothing the Integral Function to Improve Integration Accuracy In this section, Gaussian functions and probability density function mappings are used to compute the responses of random static and dynamic systems. Here, $P_q = \int_{\Omega_{\Theta,1}} p_\Theta(\theta)\, d\theta$ represents the allocation probability in region $\Omega_{\Theta,q}$, and $g(\theta_q)$ and $g(\theta_q, t)$ represent the mapping outputs of the random static and dynamic systems, respectively:

$$p_{Y(y,t)} = \sum_{q=1}^{N} \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\left(y-g\left(\theta_q,t\right)\right)^2}{2\sigma^2}} P_q \right\} \tag{5}$$

Combining the constraints of GAN, the output of the generator G is regarded as the response of a random static or dynamic system. The parameters θ_G of the generator G are treated as random parameters, and the partitioning of the input probability space is combined with the GAN training process. Here, $w_q = P_q / \sum_{i=1}^{N} P_i$ represents the normalized allocation probability. Under this condition, the following loss function is minimized:

$$min_{\theta_G,\theta_D}\left\{-E_{z\sim p_Z(z)}\left[logD\left(G\left(z,\theta_G\right),\theta_D\right)\right]+\lambda R\left(\theta_G\right)\right\} \tag{6}$$

where $\lambda > 0$ is a tunable regularization parameter that balances the quality of the generator's output with the constraint of the sample distribution in the input probability space. By incorporating the above loss function into the GAN training process, this section achieves constraints on the generator's output, thereby generating better samples in the input probability space.

Static-Dynamic Constrained GAN In this section, a solution called the Static-Dynamic Constrained GAN is proposed, where the implicit generator f_{res} serves as a predefined network topology for G and D. The predefined topology captures specific network patterns in terms of layers and nodes. Therefore, by quantifying the constraint using the EMS, a higher EMS indicates higher similarity between the networks and stronger constraint from D to G. R-SGAN introduces Isomorphic, Axial, Self-Symmetric G-D structures, and a quantization method for GANs, integrating static and

dynamic constraints to refine the G-D relationship. First, define a regularization term $R(\theta_G)$ to measure the constraint relationship between the generator G and discriminator D. This regularization term can be calculated using the EMS [16] quantization constraint method as follows:

$$R(\theta_G) = SR = \text{mean}\left(\sum_{i=0}^{n} max\rho_{jj}\right) \tag{7}$$

$$f_{\text{rate}}(q, q^*) = \begin{cases} 0, & q \leq q^* \\ \alpha q^* + (\alpha + \lambda)(q - q^*), & q > q^* \end{cases} \tag{8}$$

where α and λ are two hyperparameters that control the variation of the dropout rate. Incorporate the above constraint into the loss function of GAN. For the generator G, the following loss function is defined:

$$L_G = L_{G,orig} + \beta R(\theta_G) + f_{rate}(q, q^*) \tag{9}$$

where $L_{G,orig}$ is the original generator loss function, and β is a hyperparameter that controls the weight of the regularization term. For the discriminator D, the original loss function remains unchanged:

$$L_D = L_{D,orig} \tag{10}$$

Optimization algorithms, such as gradient descent, minimize loss functions for generators and discriminators in static and dynamic constrained GAN training, enhancing sample quality, stability, and model generalization while mitigating overfitting.

2.3 Class Labels Share Characteristics and Weight Update Constraints

The combination of static and dynamic analysis through the random incentive parameter system leads to the proposal of a method called Class Labels Share Characteristics and Weight Update Constraints (CLSC-WUC) for generating static and dynamic constrained GANs. In GAN methods, incorporating class labels can effectively address the issue of overfitting in generated data by providing additional constraints. However, in CGAN [17,18], where label information is directly passed to the discriminator, the constraint effect is not satisfactory [19,20]. The CLSC-WUC method integrates G, D, and C modules, where C classifies inputs and imposes conditional constraints on G, enhancing sample generation with specific class characteristics. The discriminator and the classifier are mutually related through shared feature extraction information, as shown in Figure 1. The noise vector is represented as z, the input image is X, and y_X and y_G represent the real label and the label input to the generator, respectively. The label information only directly affects the generator and does not influence the discriminator. Specifically: Firstly, consider a semi-supervised learning scenario where a dataset contains labeled data (X, y_X) and unlabeled data X_u. Our goal is to train a GAN and a classifier C by utilizing this information to introduce class information into the GAN and improve the performance of the classifier. To achieve this goal, the framework of

Conditional Generative Adversarial Network (CGAN) is adopted, and the class information is incorporated into the optimization objective.

In the training process, constraints are introduced to achieve feature sharing and weight updating. To do this, the following strategy is adopted: in the forward propagation process, both the input image X and the generated image *G(z)* are fed into the discriminator D and the classifier C. In this way, the discriminator and classifier can share information from the feature extraction layer. In the backward propagation process, the parameters of the generator G, discriminator D, and classifier C are updated based on the loss functions L_D, L_G, and L_C. This enables weight updating during the training process. To handle unlabeled data in the semi-supervised learning scenario, only the contribution of labeled data (X, y_X) is considered when calculating the classifier loss function L_C. A mask matrix M_A is defined to indicate whether the corresponding true label can be obtained. Then, the mask matrix M_A is applied to the classifier loss function L_C as follows:

$$\begin{aligned} &\mathcal{L}_C(X, y_X, z, y_G; C, G) \\ &= E\left[\log P(C = y_X \mid X) M_A\right] + E\left\{\log P\left[C = y_G \mid G(z, y_G)\right]\right\} \end{aligned} \tag{11}$$

The CLSC-WUC method optimizes GANs and classifiers in semi-supervised learning by integrating class labels and shared feature information, and applying weight updating constraints, thus enhancing data quality and classifier accuracy.

3 Generation Experiment

APT few-shot attack data enhancement is achieved by pre-training GAN models with attack data, simulating attack-defense systems, and constructing APT data enhancers.

3.1 Experiment Objective

The objective of this experiment is to generate sample files containing UNSW-NB15 [20] and KDD Cup 1999 datasets [21].

3.2 Network Structure Construction

As text is a discrete data, the generator is difficult to update using gradient descent, i.e., it is difficult to adjust the generator's parameters using backpropagation algorithm. Additionally, since the generator can only generate partial text, the discriminator is unable to accurately evaluate incomplete sequences, i.e., the discriminator is unable to accurately determine whether the text is real or not when only partial text is available. Therefore, the basic framework will use Seq-GAN. The reward will come from the discriminator's score for the sequence. The framework diagram is shown in Figure 2.

The Seq-GAN framework employs a reinforcement learning approach, with the generator as the agent and the discriminator as the environment, enhancing text generation through LSTM and TextCNN models for sequence evaluation and action generation.

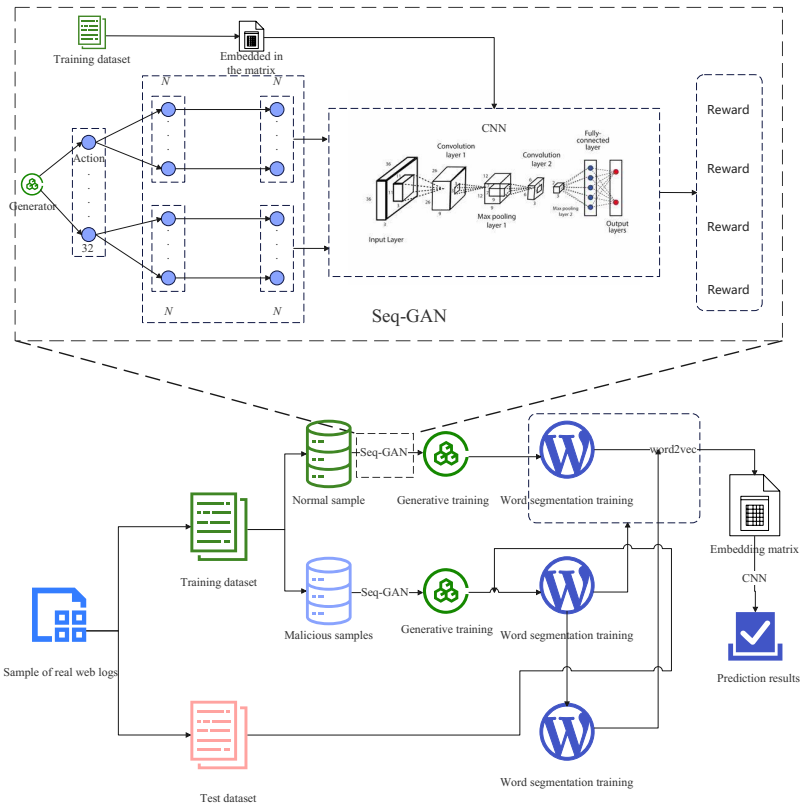

Fig. 2. Few-shot Reinforcement Learning

3.3 Training Process

By deploying the Directed Graph Model (DGM), the structure and process of GAN are analyzed, and a constraint function is defined to ensure that the relationship between the generator and the discriminator satisfies an affine transformation. The specific implementation methods are summarized as follows: First, let's define the generator *G*, which takes random noise z as input and generates a sample *x'*. The discriminator D takes x as input (which can be either a real or generated sample) and outputs the probability value *D(x)*, indicating the likelihood of x being a real sample.The objective is to minimize this loss function L_D. To add the constraint in the training process, we introduce a latent variable *res*. We aim to establish an affine transformation relationship between D and G, i.e., $D\left(x^{'}\right) = a*G\left(z\right)+b$, where a and b are learnable parameters. Using SGD or other algorithms for iterative updates in GAN training with Hilbert space constraints enhances sample quality while suppressing overfitting and mode collapse.

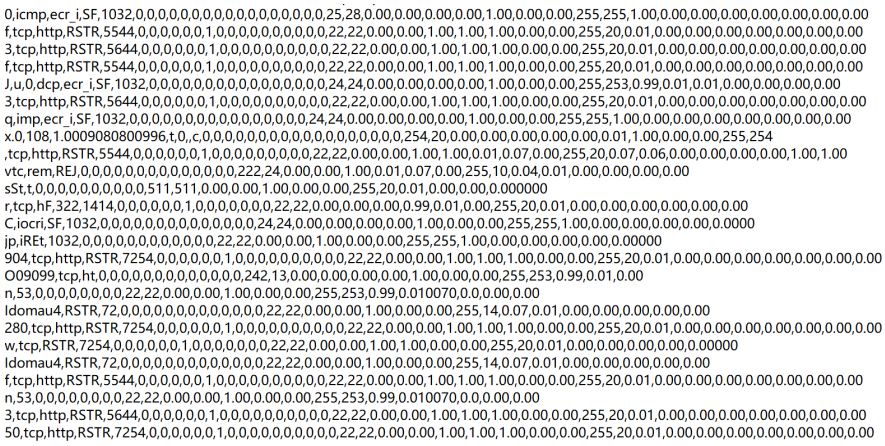

Fig. 3. Partial results of security log text generation based on Seq-GAN.

3.4 Generate effects

The generated text is judged using the Target LSTM network with the same loss function as the unsupervised above. Effect section log text generation effect such as Figure 3.

3.5 Generation Results

The generated text is evaluated using the Target LSTM network, with the same unsupervised loss function as mentioned above. The log text generation results are shown in Figure 3.

3.6 Experimental Conclusion

By using Seq-GAN to generate corresponding log texts, the problem of limited samples can be effectively addressed. Similarly, it is also possible to generate data such as binary files and network traffic parameters, thus enhancing the few-shot capability in network security. This approach has significant implications for threat detection, various attack-defense scenarios, and cryptographic algorithms.

4 Few-shot APT Data Augmentation Learning Modeling

Iterative training of discriminative and generative models, with weight adjustments in the D network, generates additional APT attack data, enhancing few-shot learning against limited samples.

The Wasserstein distance can be expressed as:

$$W = \frac{1}{K} \sup_{\|f\|_{\mathrm{L}} \leq K} E_{x \sim P_{\mathrm{rad}}}[f(x)] - E_{x \sim P_{\mathrm{g}}}[f(x)] \tag{12}$$

Table 2. Prameter symbol lookup table

Symbols	Meaning	Symbols	Meaning
P_{r}	Distribution of real data	P_{g}	The distribution of the generated data
$f_{\omega}(x)$	A collection of functions that are parameters that are limited by the scope of parameters $f\omega(-c, c]$	K	The upper limit of Lipschitz's constant
$E_{x\sim P_{\mathrm{rad}}}$	Expectations from which to sample P_{rad}	$\|f\|_{\mathrm{L}}$	The Lipschitz norm of the function f

To constrain the parameter range of function ω and ensure f_{ω} that satisfies Lipschitz continuity while only slightly fluctuating in local regions, we can define the parameter. We can derive the loss functions for the generator *G* and the discriminator *D:*

$$L(G) = -E_{x\sim P_{\mathrm{g}}}[f_{\omega}(x)] \tag{13}$$

$$L(D) = E_{x\sim P_{\mathrm{g}}}[f_{\omega}(x)] - E_{x\sim P_{\mathrm{r}}}[f_{\omega}(x)] \tag{14}$$

By maximizing $L(G)$ and minimizing $L(D)$, we can train the generator and the discriminator to generate more APT attack data and improve their ability to discriminate between real and generated data. The few-shot APT data augmentation learning process can be defined as follows:

(1) Initialize the parameters θ_G and θ_D for the generator $G(z;\theta_G)$ and the discriminator $D(x;\theta_D)$.

(2) For each training iteration $t = 1, 2, \ldots, T$:Randomly sample a mini-batch of training samples $\{(x_i, y_i)\}_{i=1}^{n}$ from the training set *T* , where $.n \ll N$

Randomly sample a mini-batch of noise samples $\{z_i\}_{i=1}^{n}$from the noise space z. Generate a batch of additional samples $\{x^{'}{}_i = G(z_i;\theta_G)\}_{i=1}^{n}$using the generator $G(z;\theta_G)$. Update the discriminator parameters θ_D to maximize the following objective function:$max_{\theta_D}\frac{1}{n}\sum_{i=1}^{n}\left[logD(x_i;\theta_D) + log\left(1 - D\left(x^{'}{}_i;\theta_D\right)\right)\right]$

(3) By optimizing the generator and discriminator parameters, we obtain a generator that can generate additional samples.

(4) Use the training set $T$and the generated samples to train a classifier by minimizing the following loss function: $min_{\theta_C}\frac{1}{N+n}\sum_{i=1}^{N+n}L(y_i, C(x_i;\theta_C)) + \lambda R(C)$, $R(C)$ here is a regularization term and $\lambda > 0$ is the regularization parameter.

(5) Use the trained classifier $C(x;\theta_C^*)$ for prediction and generalization.

In conclusion, the few-shot APT data augmentation process based on GANs aims to achieve a Nash equilibrium state where the generated adversarial samples and the defense strategies reach their optimal points. At this state, the generated adversarial samples can be considered as the best possible attacks, and the defense strategies can be considered as the most effective countermeasures.

Table 3. Sample data

Data	Positive sample	Negative samples
Original training set	17725	339200
Test the dataset	17718	339207
Synthetic samples	300000	20000
Enhance datasets	317736	359189

Table 4. Generation validity verification

Dataset processing method	Accuracy	F1-Score
Unaugmented	0.9839	0.9850
SMOTE	0.9920	0.9494
ROS	0.9883	0.9877
GAN	0.9845	0.9846
AC-GAN	**0.9989**	0.9543
DCGAN	0.9936	**0.9932**
GADCN	0.9922	0.9922

5 Result Analysis

In the context of effectiveness evaluation, the NIMS(Network Information Management and Security Group) [22] and USTC-TFC2016 dataset [23] are divided into training and testing sets (as shown in Table 3). The NIMS dataset consists of a total of 35,443 positive flow samples and 678,407 negative flow samples, with a ratio of approximately 0.0522 between positive and negative samples. Table 3 displays the number of generated synthetic samples, as well as the training, testing, and augmentation datasets.

Afterwards, the training set is used to train the few-shot augmentation model. Once the training process is completed, the trained generator (G) is used to generate synthetic samples with specific labels. Therefore, the trained G network can generate augmented samples to address the issue of data imbalance in the network dataset. The generator (post-training) is used to generate new synthetic samples, aiming to address the data imbalance in the few-shot training set by generating more synthetic samples as augmentation for positive samples and fewer synthetic samples for negative samples. The AC-GAN augmented dataset is generated using the AC-GAN method, while the SMOTE-SVM augmented dataset is generated using the SMOTE-SVM method, which is a comparative method for artificially synthesizing network traffic data [24].

Finally, the synthesized samples are combined with the training dataset to form the augmented dataset. The results in this section indicate that the GAN-generated augmented network traffic dataset, when compared to the original imbalanced dataset and the artificially synthesized SMOTE dataset, improves the performance of supervised learning classification (Table 4).

6 Conclusion

For addressing the challenge of augmenting few-shot data effectively, this article proposes a constrained theoretical model for Generative Adversarial Networks (GANs).

It incorporates a Directed Graphical Model (DGM)-based structure and process-constrained training to tackle training instability and enhance the quality of generated samples. A novel static constraint method is introduced, leveraging the similarity in an Interpretable Measurement Scale (EMS) and final classification metrics of generated data by imposing topological constraints on discriminators (D) and generators (G) across various classifiers. The EMS quantifies the constraint's strength, effectively mitigating overfitting during generation. Additionally, constraints on label feature sharing and weight updates refine the utilization of label information in the generation process, significantly lowering the risk of mode collapse. Consequently, this paper presents a GAN-based constrained augmentation approach tailored for few-shot data scenarios.

Acknowledgements

Xin Song, Yanqing Song, Jianguo Chen and Long Chen contributed equally to this work. This study was supported by the National Science Fund for Distinguished Young Scholars of China (62225303), the Fundamental Research Funds for the Central Universities (ZY2411), the National Science Fund of China (62303038), and China Postdoctoral Science Foundation (2023M740202).

References

1. Zhang, J., Zhai, W.: Blind attention geometric restraint neural network for single image dynamic/defocus deblurring. IEEE Transactions on Neural Networks and Learning Systems pp. 1–14 (2022), 0.1109/TNNLS.2022.3151099
2. Sun, B., Wu, Z., Feng, Q.: Small Sample Reliability Assessment With Online Time-Series Data Based on a Worm Wasserstein Generative Adversarial Network Learning Method. IEEE Transactions on Industrial Informatics **19**(2), 1207–1216 (2022)
3. Striuk, O.S., Kondratenko, Y.: Generative Adversarial Networks in Cybersecurity: Analysis and Response. In: and others (ed.) Artificial Intelligence in Control and Decision-making Systems . pp. 373–388. Springer (April 2023)
4. Huang, Z., Li, W., Wang, Y.: Multi-level noise-aware network for low-dose CT imaging implemented with constrained cycle Wasserstein generative adversarial networks. Artificial Intelligence in Medicine **143**, 102609–102609 (2023)
5. Song, Y., Wang, T., Mondal, S.K., Sahoo, J.P.: A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. ACM Computing Surveys **55**(13), 1–40 (2023)
6. Guo, H., Zhang, X., Wang, Y.: Few-Shot Malware Traffic Classification Method Using Network Traffic and Meta Transfer Learning. 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall) pp. 1–5 (2022)
7. Rong, C., Gou, G., Hou, C.: UMVD-FSL: unseen malware variants detection using few-shot learning. In: and others (ed.) 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (July 2021)
8. Park, G.: Large-scale directed graphical models learning (2016), https://www.semanticscholar.org/paper/Large-scale-directed-graphical-models-learning-Park/428afe0c020dfdc2f3792657905b8bc275f6163a
9. Cai, Z., Xiong, Z., Xu, H.: Generative adversarial networks: A survey toward private and secure applications. ACM Computing Surveys (CSUR) **2021**, 1–38 (2021). https://doi.org/10.1145/3459992

10. Chakrabarti, S., Yiming, H., Li, T.: Quantum wasserstein generative adversarial networks. Advances in Neural Information Processing Systems pp. 32–32 (2019)
11. Ghojogh, B., Ghodsi, A., Karray, F., Crowley, M.: Reproducing Kernel Hilbert Space, Mercer's Theorem, Eigenfunctions, Nystr" om Method, and Use of Kernels in Machine Learning: Tutorial and Survey (June 2021)
12. Fiore, U., Santis, D., Perla, A., F: Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. Information Sciences **479**, 448–455 (2019)
13. Li, X., Chen, G., Cui, H., Yang, D.: Direct probability integral method for static and dynamic reliability analysis of structures with complicated performance functions. Computer Methods in Applied Mechanics and Engineering **374**, 113583–113583 (2021)
14. Chen, G., Yang, D.: Direct probability integral method for stochastic response analysis of static and dynamic structural systems. Computer Methods in Applied Mechanics and Engineering **357**, 112612–112612 (2019)
15. Chen, J., Yang, J., Li, J.: A GF-discrepancy for point selection in stochastic seismic response analysis of structures with uncertain parameters. Structural Safety **59**, 20–31 (2016)
16. Tian, Y., Jiang, T., Gong, Q., Morcos, A.: Luck matters: Understanding training dynamics of deep relu networks (May 2019)
17. Zhou, G., Fan, Y., Shi, J.: Conditional Generative Adversarial Networks for Domain Transfer: A Survey. Applied Sciences (16), 8350–8350 (2022)
18. Sajun, A.R., Zualkernan, I.: Survey on Implementations of Generative Adversarial Networks for Semi-Supervised Learning. Applied Sciences (3), 1718–1718 (2022)
19. Li, Y., Li, B., Gao, Z., Wang, J.: Antimode collapse generative adversarial networks. Journal of Electronic Imaging **28**(2) (2019)
20. Moustafa, N., Slay, J.: UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 military communications and information systems conference (MilCIS). pp. 1–6 (2015)
21. Luz, A., Olaoye, G.: Comparative Analysis of UNSW-NB15 and NSL-KDD Datasets (2024)
22. Vu, L., Bui, C.T., Nguyen, Q.: A Deep Learning Based Method for Handling Imbalanced Problem in Network Traffic Classification. In: and others (ed.) Proceedings of the 8th International Symposium on Information and Communication Technology. pp. 333–339 (December 2017)
23. Wang, W., Lu, D.: A traffic dataset called "USTC-TFC2016" (2016), https://github.com/yungshenglu/USTC-TFC2016
24. Vu, L., Tra, D.V., Nguyen, Q.: Learning from imbalanced data for encrypted traffic identification problem. In: and others (ed.) Proceedings of the 7th Symposium on Information and Communication Technology. pp. 147–152 (2016)