EN RU PROGRAM SYSTEMS: THEORY AND APPLICATIONS ISSN 2079-3316
Research Article HARDWARE AND SOFTWARE FOR DISTRIBUTED AND SUPERCOMPUTER SYSTEMS

UDC 519.683.1:681.513.7
10.25209/2079-3316-2024-15-4-97-110

Building robust malware detection through conditional Generative Adversarial Network-based data augmentation

Elshan **Baghirov**✉
Institute of Information Technology, Baku. Azerbaijan
✉*elsenbagirov1995@gmail.com*

Abstract. Malware detection is essential in cybersecurity, yet its accuracy is often compromised by class imbalance and limited labeled data. This study leverages conditional Generative Adversarial Networks (cGANs) to generate synthetic malware samples, addressing these challenges by augmenting the minority class.

The cGAN model generates realistic malware samples conditioned on class labels, balancing the dataset without altering the benign class. Applied to the CICMalDroid2020 dataset, the augmented data is used to train a LightGBM model, leading to improved detection accuracy, particularly for underrepresented malware classes.

The results demonstrate the efficacy of cGANs as a robust data augmentation tool, enhancing the performance and reliability of machine learning-based malware detection systems.

Key words and phrases: malware detection, Generative Adversarial Networks, machine learning, cybersecurity, data augmentation

For citation: Elshan Baghirov. *Building robust malware detection through conditional Generative Adversarial Network-based data augmentation*. Program Systems: Theory and Applications, 2024, **15**:4(63), pp. 97–110.
`https://psta.psiras.ru/read/psta2024_4_97-110.pdf`

Introduction

Malware refers to malicious software intended to disrupt operations, deny access to services, collect personal data without the user's consent, gain unauthorized access to system resources, or engage in other harmful activities. The rapid advancement of information technology has led to an exponential rise in malware, making it one of the primary threats to computer security [1–3].

Malware detection methods can be broadly categorized into two main types: signature-based detection and behavior-based detection [4]. Signature-based detection relies on known malware patterns or signatures, making it highly effective at identifying previously encountered threats.

However, it struggles to detect novel or evolving malware variants. On the other hand, behavior-based detection analyzes the actions and behaviors of programs to identify malicious intent, even in unknown or zero-day threats. While behavior-based methods provide more robust protection against new threats, they often come with higher computational costs and the potential for false positives [4].

As malware continues to evolve and adapt, traditional detection methods face significant limitations. Signature-based methods cannot keep up with the sheer volume of new malware variants, while behavior-based methods, although more adaptive, can be resource-intensive and prone to inaccuracies in complex environments.

Moreover, the challenge of class imbalance—where benign files vastly outnumber malware samples—further complicates the development of effective detection models. This imbalance can lead to poor generalization, causing machine learning models to misclassify malware or fail to detect it altogether.

To address these issues, recent advancements have explored machine learning and data augmentation techniques, particularly Generative Adversarial Networks (GANs), to enhance malware detection [4–6]. By generating synthetic malware samples, GANs help balance the dataset and improve model performance in detecting underrepresented malware classes.

Our study focuses on employing cGANs to augment malware datasets, addressing the class imbalance problem and enhancing the detection capabilities of machine learning models.

This paper presents the following key contributions:

- **cGAN for Malware Data Augmentation**: conditional GANs are applied to generate synthetic malware samples, addressing the class imbalance issue in malware detection datasets.
- **Comparative Evaluation**: a thorough evaluation of machine learning models on imbalanced and augmented datasets is conducted, highlighting the benefits of cGAN-generated data in improving model performance.
- **Reproducible Methodology**: a detailed, reproducible experimental design for applying cGANs in cybersecurity is outlined to improve detection accuracy and reliability.

These contributions help advance malware detection by addressing class imbalance and improving model performance against diverse threats.

The paper is organized as follows:

Section 1 reviews existing malware detection techniques and the application of GANs for data augmentation.

Section 2 provides background on GANs and cGANs, explaining their structure and usage in data generation tasks.

Section 3 covers the experiments and results, including a description of the dataset, environmental setup, experimental design, and detailed results of the experiments.

Section 4 concludes the paper, summarizing key findings and proposing future research directions.

1. Related works

The paper by Nguyen et al. [6] explores the use of GANs for image-based malware classification. The study compares the performance of GANs with various other machine learning models, such as support vector machines (SVM) and random forest (RF). They focus on using the GAN model to classify different malware families and evaluate its efficacy for adversarial attacks.

The results show that GANs are effective at generating realistic images for classification but are less useful for adversarial attacks, as the generated images are easily distinguishable from real ones. The paper also highlights the strengths of auxiliary-classifier GANs (AC-GANs) in achieving competitive classification results when compared with other machine learning techniques.

Li et al. [7] introduce GMADV, a framework for Android malware classification and adversarial training, addressing challenges posed by anti-reverse engineering techniques and limited malware sample diversity. GMADV converts APK files into RGB Markov images and utilizes a VGG13 model for feature extraction and classification. Additionally, a GMM-GAN is employed to increase malware variant diversity by blending content and style features.

Experimental results show significant improvements in classification performance, with F1-scores exceeding 95%, demonstrating the framework's effectiveness in generating diverse malware variants and enhancing adversarial training.

Reilly et al. [5] explore the use of deep learning, specifically GANs, to enhance the robustness of image-based malware classification models against adversarial attacks. The study evaluates two image conversion techniques — byteplot and space-filling curves — using a ResNet-50 model to classify malware samples.

The results show that models trained with GAN-generated data exhibit significantly better resistance to adversarial attacks compared to those trained without GANs. The paper underscores how adversarial training, particularly with GANs, improves classification performance and robustness in malware detection tasks, highlighting the potential of GANs to bolster the security and accuracy of such systems despite certain limitations.

The provided papers present several limitations in the use of GANs for malware detection and classification. In the GMADV framework, while diverse malware variants are generated, the method still produces samples with similarities to the originals, limiting the creation of entirely novel behaviors, and the GMM-GAN may struggle with scalability for larger datasets.

The image-based malware classification model trained with GANs, as discussed by Li et al. [7], shows improved robustness against adversarial attacks but is limited to image-based transformations, leaving potential gaps in detecting more dynamic malware types, and the results are specific to the ResNet-50 model. Nguyen et al.'s work [6] also faces challenges, particularly with the computational expense and instability of GANs during training on large datasets, and the generated samples, while useful for classification, are less effective in adversarial settings where real and synthetic data are more easily distinguished.

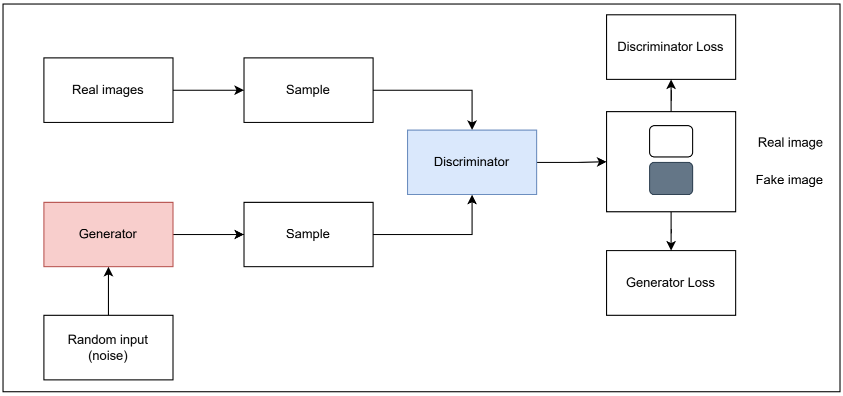

Figure 1. General structure of GANs

These studies highlight the constraints of GAN-based approaches in scaling, sample diversity, and generalization across various malware types.

2. Background

2.1. Generative Adversarial Networks

First introduced by Ian Goodfellow and colleagues in 2014, GANs have swiftly evolved into a pivotal technique within machine learning and artificial intelligence [8]. Originally created for image generation tasks, GANs feature two neural networks in competition: a generator, which produces data resembling real-world examples, and a discriminator, which works to differentiate between real and generated data. This adversarial dynamic pushes both networks to improve iteratively.

The overall structure of GANs is depicted in Figure 1. Since their inception, GANs have been applied to a wide range of areas, such as data augmentation, artistic generation, and more recently, in cybersecurity for generating synthetic datasets that strengthen machine learning models against threats like malware. Despite their versatility, training GANs can be difficult, with issues like mode collapse and high computational demands being common challenges [9]. Nevertheless, their ability to generate highly realistic synthetic data marks GANs as a transformative tool in modern machine learning.

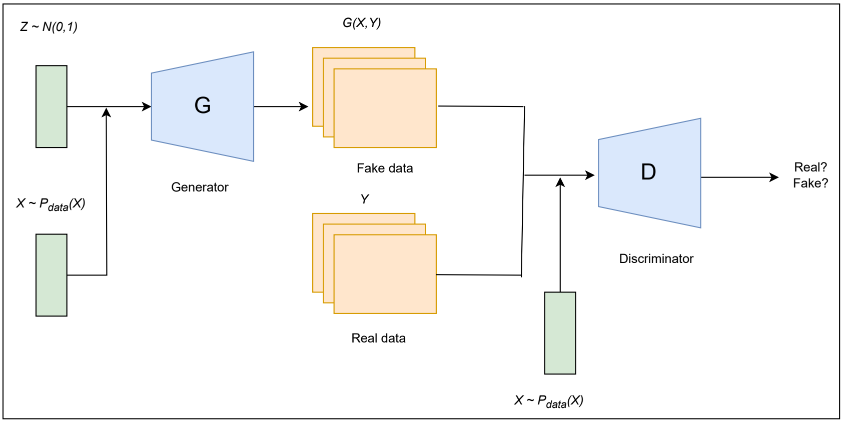

FIGURE 2. The architecture of cGAN [11]

2.2. Conditional Generative Adversarial Networks

CGANs represent a refined architecture of the standard Generative Adversarial Network, designed to generate synthetic data closely aligned with conditioned constraints. This architecture incorporates both a generator and a discriminator, functioning through an adversarial process to improve the realism and specificity of generated outputs [10, 11].

The interaction between G and D is depicted in Figure 2. Here, the generator aims to fool the discriminator by creating data indistinguishable from real data, while the discriminator learns to better differentiate real data from forgeries. This adversarial training not only enhances the generator's ability to mimic real data but also sharpens the discriminator's analytical skills.

The generator, denoted by G, takes as input a noise vector Z, typically sampled from a standard normal distribution, along with conditional labels Y. These labels ensure that the generated data, $G(X, Z)$, conforms to specific attributes required by the application, such as class characteristics in classification tasks. This approach allows the generator to produce not only realistic but also conditionally relevant data.

Simultaneously, the discriminator, D, evaluates the authenticity of both the data produced by G and the actual data samples from the training set. Each set of data, real or synthetic, is presented to D along with its corresponding labels Y, which helps D make more informed judgments.

Table 1. Original CICMalDroid2020 dataset size with categories

Category	**Count**
Adware	1,253
Banking	2,100
SMS malware	3,904
Riskware	2,546
Benign	1,795

The discriminator's task is to distinguish between 'real' and 'fake' data, effectively identifying the synthetic outputs of the generator.

CGANs are particularly valuable for applications requiring data augmentation in imbalanced datasets, significantly contributing to the diversity and volume of training data. This makes them ideal for improving model performance and robustness in various domains, including cybersecurity and healthcare.

3. Experiments and results

3.1. Dataset description

The experiments in this study utilize the CICMalDroid2020 dataset [12, 13], which contains a balanced distribution of benign and malware samples. Distribution of file categories in this dataset has been shown in Table 1. However, to simulate a more challenging, imbalanced scenario that is common in real-world malware detection, we intentionally reduced the number of malware samples to 10% of their original count, ensuring that the percentage distribution of different malware categories was maintained to preserve sample diversity. This resulted in 980 malware samples and 1,795 benign samples. Using cGAN, we generated additional synthetic malware samples to increase the total malware count to 1,795, thereby balancing the dataset while preserving the diversity of malware categories for improved detection performance.

3.2. Experimental design

In this study, the cGAN was employed to address class imbalance by generating synthetic malware samples. The cGAN architecture used for malware data augmentation is outlined in Algorithm 1. This algorithm consists of several key steps, which involve training both the generator and discriminator in a competitive setting.

Algorithm 1. cGAN for Malware Data Augmentation

```
 1: Input: Real dataset $X_{real}$ with class labels $y_{real}$, latent space dimension
    $z$, total number of epochs $epochs$, batch size $m$
 2: Output: Trained generator $G$, trained discriminator $D$, generated
    synthetic malware samples $X_{fake}$
 3: Initialize generator $G$ and discriminator $D$ with random weights
 4: Set optimizer and hyperparameters
 5: for each epoch $t$ from 1 to $epochs$ do
 6:     Step 1: Train the Discriminator $D$:
 7:     Sample a batch of real data $X_{real}$ and labels $y_{real}$
 8:     Sample a batch of random noise vectors $z$ and malware labels $y_{fake}$
 9:     Generate synthetic samples $X_{fake} = G(z, y_{fake})$
10:     Compute discriminator loss:
            $L_{real} = D(X_{real}, y_{real}), \quad L_{fake} = D(X_{fake}, y_{fake})$
11:     Update $D$ by minimizing combined loss:
            $L_D = \frac{1}{2}(L_{real} + L_{fake})$
12:     Step 2: Train the Generator $G$:
13:     Sample new batch of random noise vectors $z$ and malware labels
    $y_{fake}$
14:     Generate synthetic samples $X_{fake} = G(z, y_{fake})$
15:     Compute generator loss:
            $L_G = D(G(z, y_{fake}), y_{fake})$
16:     Update $G$ by maximizing $L_G$
17: end for
18: Step 3: Generate new malware samples:
19: Sample random noise $z$ and malware labels $y_{fake} = 1$
20: Generate synthetic malware samples $X_{synthetic} = G(z, y_{fake})$
21: Combine $X_{synthetic}$ with the original dataset
```

The process begins by initializing the generator G and the discriminator D with random weights. In each epoch, the discriminator is trained first by sampling batches of real data, along with the corresponding class labels, and comparing them with synthetic data generated by the generator.

The discriminator's loss is calculated based on its ability to distinguish between real and fake samples, and the model is updated by minimizing the combined loss from both real and synthetic data, as outlined in Step 1 of Algorithm 1.

Following this, the generator is trained to improve the realism of the synthetic data. It does this by generating new samples from random noise vectors, conditioned on malware labels, and receiving feedback from the discriminator. The generator's loss is calculated based on how well it can "fool" the discriminator into classifying synthetic data as real, and it is updated by maximizing its loss function, as described in Step 2.

The final step, Step 3 of Algorithm 1, involves generating new synthetic malware samples by sampling random noise and malware labels. These newly generated samples are combined with the original dataset to balance the distribution of benign and malware samples.

The iterative adversarial process between the generator and discriminator allows the model to refine the quality of the synthetic malware samples over multiple epochs, ultimately creating a balanced dataset that enhances the performance of the machine learning models trained on it. This approach effectively addresses the issue of class imbalance in malware detection tasks, enabling models to better detect minority class malware instances.

3.3. Enviromental setup

The experiments in this study were conducted on the Kaggle platform, leveraging its computational resources for efficient model training and evaluation. Various configurations were used, including CPU (4 cores, 30 GB RAM) for data processing, and GPU (Nvidia Tesla P100 or T4) with up to 2 GPUs, 4 CPU cores, and 29 GB RAM for deep learning tasks. The TPU 1VM setup, with 96 CPU cores and 330 GB RAM, provided the highest computational power, ensuring efficient handling of large-scale deep learning workloads and producing reliable results.

3.4. Results and discussions

The initial experiments conducted on the imbalanced dataset and the subsequent trials on the augmented dataset using LightGBM and other machine learning models yielded significant insights, presented

Table 2. Comparison of the models on the imbalanced dataset

Model	Accuracy	Precision	Recall	F1-Score
LightGBM	0.9153	0.8958	0.8643	0.8798
Random Forest	0.9189	0.9096	0.8593	0.8837
KNN	0.8559	0.8083	0.7839	0.7959
Logistic Regression	0.7532	0.7460	0.4724	0.5785

Table 3. Comparison of the models on the augmented dataset

Model	Accuracy	Precision	Recall	F1-Score
LightGBM	0.9568	0.9619	0.9541	0.9579
Random Forest	0.9513	0.9666	0.9378	0.9520
KNN	0.8955	0.9041	0.8919	0.8980
Logistic Regression	0.8078	0.8816	0.7243	0.7953

Table 4. Confusion matrix for the imbalanced dataset using LightGBM

		Predicted	
		Negative	**Positive**
Actual	**Negative**	335	21
	Positive	16	183

Table 5. Confusion matrix for the augmented dataset using LightGBM

		Predicted	
		Negative	**Positive**
Actual	**Negative**	335	13
	Positive	16	354

in Table 2 and Table 3. Corresponding confusion matrices for LightGBM's performance on both datasets are detailed in Table 4 and Table 5.

In the comparative analysis of machine learning models trained on both imbalanced and augmented datasets, LightGBM consistently emerged as the standout performer. Initially, on the imbalanced dataset, LightGBM demonstrated a strong capability with an accuracy of 91.53%, precision of 89.58%, and an F1-score of 87.98%. This performance indicates a robust ability to handle class imbalances effectively, particularly in identifying the minority class malware samples.

Upon augmenting the dataset to mitigate class imbalance, LightGBM's performance further excelled, leading to even higher metrics with an accuracy of 95.68%, precision of 96.19%, and an outstanding F1-score of 95.79%. The most notable improvement was observed in its recall, which reached 95.41%, underscoring the model's enhanced ability to correctly identify almost all malware samples in the augmented dataset.

4. Conclusions and future works

In this study, we utilized cGAN to address the challenge of class imbalance in malware detection. By intentionally reducing the number of malware samples to 10% of their original count, we simulated a real-world imbalanced scenario. Using cGAN, we augmented the minority class and balanced the dataset.

The results showed that after balancing the dataset, the LightGBM model achieved a high accuracy of 95.68%, with improvements in precision (96.19%), recall (95.41%), and F1-score (95.79%). These metrics indicate that LightGBM is highly effective when paired with cGAN-generated synthetic data, outperforming other models in terms of detecting underrepresented malware samples. This demonstrates the strength of LightGBM in handling large-scale, imbalanced datasets when supplemented with generative techniques like cGAN.

There are several key directions for future research based on the results of this study. First, optimizing the models with more advanced data augmentation techniques, such as Variational Autoencoders (VAEs) or Generative Pretrained Transformers (GPT), could further enhance the detection of complex malware patterns. Additionally, implementing these augmented models in real-time malware detection systems would provide valuable insights into their performance in dynamic and evolving environments.

Another important area is cross-dataset validation, where the models could be tested across different malware datasets to assess their generalizability and robustness in varied attack scenarios. Furthermore, investigating hybrid approaches, such as combining multiple machine learning models or ensemble methods, could further improve detection accuracy while reducing false positives and negatives.

Finally, it is essential to evaluate the resilience of these models against adversarial attacks, ensuring that the augmented data not only improves detection rates but also enhances the security and reliability of the entire detection system. Addressing these areas will contribute to the advancement of more robust and effective malware detection frameworks.

References

[1] D. O. Won, Y. N. Jang, S. W. Lee. "PlausMal-GAN: Plausible malware training based on generative adversarial networks for analogous zero-day malware detection", *IEEE Transactions on Emerging Topics in Computing*, **11**:1 (2023), pp. 82–94. doi ↑98

[2] E. Baghirov. "A comprehensive investigation into robust malware detection with explainable AI", *Cyber Security and Applications*, **3** (December 2025), id. 100072. doi ↑98

[3] E. Baghirov. "Techniques of malware detection: Research review", *2021 IEEE 15th International Conference on Application of Information and Communication Technologies*, AICT 2021 (13–15 October 2021, Baku, Azerbaijan), IEEE, 2021, ISBN 978-1-6654-3641-0/21, pp. 1–6. doi ↑98

[4] S. Jang, S. Li, Y. Sung. "Generative adversarial network for global image-based local image to improve malware classification using convolutional neural network", *Applied Sciences*, **10**:21 (2020), id. 7585, 14 pp. doi ↑98

[5] C. Reilly, S. O Shaughnessy, C. Thorpe. "Robustness of image-based malware classification models trained with generative adversarial networks", *EICC'23: Proceedings of the 2023 European Interdisciplinary Cybersecurity Conference* (14–15 June 2023, Stavanger, Norway), ACM, New York, 2023, ISBN 978-1-4503-9829-9, pp. 92–99. doi ↑98, 100

[6] H. Nguyen, F. Di Troia, G. Ishigaki, M. Stamp. "Generative adversarial networks and image-based malware classification", *Journal of Computer Virology and Hacking Techniques*, **19** (2023), pp. 579–595. doi ↑98, 99, 100

[7] S. Li, Z. Tang, H. Li, J. Zhang, H. Wang, J. Wang. "GMADV: An android malware variant generation and classification adversarial training framework", *Journal of Information Security and Applications*, **84** (August 2024), id. 103800. doi ↑100

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. "Generative adversarial nets", *Advances in Neural Information Processing Systems 27*, NIPS 2014 (8–13 December 2014, Montreal, Canada), 2014, ISBN 9781510800410, 9 pp. URL ↑101

[9] R. Yuwana, F. Fauziah, A. Heryana, D. Krisnandi, R. S. Kusumo, H. F. Pardede. "Data augmentation using adversarial networks for tea diseases detection", *Jurnal Elektronika dan Telekomunikasi*, **20**:1 (2020), pp. 29–35. doi ↑101

[10] M. Mirza, S. Osindero. *Conditional generative adversarial nets*, 2014, 7 pp. arXiv 1411.1784~[cs.LG] ↑102

[11] M. Iwayama, S. Wu, C. Liu, R. Yoshida. “Functional output regression for machine learning in materials science”, *Journal of chemical information and modeling*, **62**:20 (2022), pp. 4837–4851. ↑102

[12] S. Mahdavifar, A. F. A. Kadir, R. Fatemi, D. Alhadidi, A. A. Ghorbani. “Dynamic android malware category classification using semi-supervised deep learning”, *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress*, DASC/PiCom/CBDCom/CyberSciTech (17–22 August 2020, Calgary, AB, Canada), 2020, ISBN 978-1-7281-6609-4, pp. 515–522. ↑103

[13] S. Mahdavifar, D. Alhadidi, A. A. Ghorbani. “Effective and efficient hybrid android malware classification using pseudo-label stacked auto-encoder”, *Journal of Network and Systems Management*, **30** (2022), id. 22, 34 pp. ↑103

Received	05.12.2024;
approved after reviewing	07.12.2024;
accepted for publication	07.12.2024;
published online	10.12.2024.

Recommended by *prof. S. M. Abramov*

Information about the author:

Elshan Baghirov

Elshan Baghirov is a PhD candidate at the Institute of Information Technology, Baku, Azerbaijan. His research focuses on machine learning, and cybersecurity, with a particular emphasis on malware detection. He is also a data scientist at Kapital Bank OJSC, where he applies advanced analytics and predictive modeling to real-world business problems. His scientific interests include artificial intelligence, deep learning, and cybersecurity.

0000-0002-5940-4136

e-mail: ***elsenbagirov1995@gmail.com***

The author declare no conflicts of interests.

RU EN **ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ** ISSN 2079-3316
научная статья ПРОГРАММНОЕ И АППАРАТНОЕ ОБЕСПЕЧЕНИЕ РАСПРЕДЕЛЕННЫХ И СУПЕРКОМПЬЮТЕРНЫХ СИСТЕМ

УДК 519.683.1:681.513.7
doi 10.25209/2079-3316-2024-15-4-97-110

Построение надёжной системы обнаружения вредоносного ПО с использованием генеративных состязательных сетей для увеличения данных

Эльшан **Багиров**
Институт Информационных Технологий, Баку, Азербайджан
✉*elsenbagirov1995@gmail.com*

Аннотация. Обнаружение вредоносного ПО является важным аспектом кибербезопасности, однако его точность часто снижается из-за дисбаланса классов и ограниченного количества размеченных данных. В данном исследовании используются генеративные состязательные сети с условными данными (conditional Generative Adversarial Networks, cGAN) для генерации синтетических образцов вредоносного ПО, что позволяет решить эти проблемы за счёт увеличения объёма данных в классе меньшинства.

Модель cGAN генерирует реалистичные образцы вредоносного ПО, основываясь на метках классов, балансируя набор данных без изменения класса безопасных образцов. Применённый к набору данных CICMalDroid2020, увеличенный объём данных используется для обучения модели LightGBM, что приводит к повышению точности обнаружения, особенно для слабо представленных классов вредоносного ПО.

Результаты демонстрируют эффективность использования cGAN в качестве надёжного инструмента для увеличения объёма данных, что улучшает производительность и надёжность систем обнаружения вредоносного ПО на основе машинного обучения.

Ключевые слова и фразы: Обнаружение вредоносного ПО, Генеративные Состязательные Сети, Машинное Обучение, Кибербезопасность, Увеличение Объёма Данных

Для цитирования: Багиров Э. *Построение надёжной системы обнаружения вредоносного ПО с использованием генеративных состязательных сетей для увеличения данных* // Программные системы: теория и приложения. 2024. **Т. 15**. № 4(63). С. 97–110. `https://psta.psiras.ru/read/psta2024_4_97-110.pdf`