

# Data Generative Model to Detect the Anomalies for IDS Imbalance CICIDS2017 Dataset

Azhari Shouni Barkah<sup>1,2</sup>, Siti Rahayu Selamat<sup>2</sup>, Zaheera Zainal Abidin<sup>2</sup>, Rizki Wahyudi<sup>1</sup>

<sup>1</sup> *Department of Informatics, Faculty of Computer Science, Universitas Amikom Purwokerto, Indonesia*

<sup>2</sup> *Fakulti Teknologi Maklumat dan Komunikasi (FTMK) Universiti Teknikal Malaysia Melaka*

**Abstract** – The system of intrusion detection dataset enables machine learning to recognize attack activity in the network. The intrusion, however, is naturally imbalanced, most of the traffic is normal traffic. Moreover, a certain attack is more popular than others. Therefore, the number of cases is highly imbalanced with the majority of attacks dominated by Distributed Denial of Services (DDoS), Denial of Service Hulk (DoS\_Hulk), and PortScan more than 4.5% of attacks data. While the minority attack such as DoS\_goldeneye, DoS\_slowloris, DoS\_slowhttpstest, Web Attacks, Infiltration, Bot, and Heartbleed was only recorded in less than 1% of attack data. We propose data generative model (DGM) using the Conditional Generative Adversarial Network (CGAN) to improve the class of minorities of the IDS dataset. In this study, we tested the performance of the Data Generative Model based on CGAN in the CICIDS2017 dataset. There are new attacks in this dataset, including Bot, Web attacks, Infiltration and Heartbleed. According to our experiments, the model successfully detect new attacks and improves the weighted f1-score by 99,92% compared to that of achievers by existing methods using the CICIDS2017 dataset.

**Keywords** – DGM, CGAN, imbalance data, IDS dataset.

DOI: 10.18421/TEM121-11

<https://doi.org/10.18421/TEM121-11>

**Corresponding author:** Azhari Shouni Barkah,  
*Department of Informatics, Faculty of Computer Science,  
Universitas Amikom Purwokerto, Indonesia  
Fakulti Teknologi Maklumat dan Komunikasi (FTMK)  
Universiti Teknikal Malaysia Melaka*


**Email:** [azhari@amikompurwokerto.ac.id](mailto:azhari@amikompurwokerto.ac.id)

*Received:* 11 October 2022.

*Revised:* 09 January 2023.

*Accepted:* 13 January 2023.

*Published:* 27 February 2023.

 © 2023. Azhari Shouni Barkah, Siti Rahayu Selamat, Zaheera Zainal Abidin, Rizki Wahyudi; published by UIKTEN. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDeriv 4.0 License.

The article is published with Open Access at <https://www.temjournal.com/>

## 1. Introduction

Network intrusion is naturally unbalanced [1],[2],[3],[4]. Most of the network traffic is normal, while a small proportion of it is attacked. Network attack techniques are evolving. However, some attacks are more popular than others due to their simplicity in carrying out and the effectiveness of the attacks themselves. Under these conditions, the attack dataset is also unbalanced [5]. This can be seen in many intrusion detection datasets. Due to evolving network attack techniques, it is almost impossible to establish general rules for detecting them. Implementing a rules-based network attack detector may fail to recognize new types of attacks [6]. Second, the rules-based approach requires network specialists to conduct very intensive observations and code new rules all the time to respond to new variations of attacks [7].

Therefore, the machine learning approach is seen as a superior solution to this issue.

Machine learning provides the ability for models to retrain whenever a new attack type is identified [8]. Once a new model is produced, new types of attacks can be recognized by the model. The problem is the availability of labelled data. If its data is provided, then the model can be produced. The downside of machine learning is its sensitivity to the amount or proportion of training set classes. Unbalanced datasets lead to low-quality models, which can easily recognize the majority class and do not have the ability to classify the class of minorities [9]. Because minority class representation in the test sample is restricted or sometimes it doesn't appear, the overall accuracy may still be high, but the memory of the minority class is low. This shows that a model fails to recognize the class of minorities [10]. Because many machine learning problems are inherently unbalanced, this problem may happen most of the time [11].

To overcome this problem, a data synthesis method was introduced to add the number of minority classes based on the nearest neighbor approach, which includes SMOTE the synthetic minority oversampling [12] technique.

The approach depends on the amount of neighbors considered to generate the sample of data. Besides that, this method has the drawback of causing an excessive amount of generalization or data overlapping. In addition to the problem of an overgeneralization, the proximity-neighbor-based approach (KNN) is aimed at addressing the imbalance of data on two classes. As a consequence, it is extremely challenging to apply in the actual world, where the majority of classes are represented by their minority classes.

Under these conditions, a generative model based on artificial neural networks is proposed to produce high-quality and better synthetic data than the old method based on proximity to neighbors. The model, in particular the network adversarial network (GAN) [13], helps to produce more realistic data. GAN is able to bypass the difficulty of approaching many probability calculations that are tough to resemble the distribution of actual training samples.

GAN makes a new sample of data that is similar enough to the existing sample that the presence of a minority class is no longer a reason why the machine learning model works well. However, Lin et al. [14] argue that when a GAN is used, the assaults that are produced cannot be differentiated from one another for some classes. This results in some types of attacks going undetected. Research [15] states that GAN has weaknesses because of the collapse mode, the training not being stable, and a lack of concern for the sample of class majority, which affects the limit of classification. Therefore, research [16],[17],[18] proposes CGAN, also known as Conditional Generative Adversarial Network, to resolve the issue.

Based on previous research, Dlamini [16] proposed using a generative data model (DGM) to improve the class of minorities in intrusion detection based on anomalies. This model uses the CGAN, method, which is trained to produce a sample of minority class data synthesis using the Kullback–Leibler (KL) Divergence mechanism for the purpose of understanding the actual distribution of minority classes. Before training the model, the majority class was also reduced in size by using the undersampling technique. Experiments were done to compare how well the DGM worked with SMOTE, ADASYN, SMOTENN, and Borderline-SMOTE. These experiments used the NSL-KDD dataset and the UNSW-NB15 dataset. The achievement of DGM's performance obtained better and superior results by 1–5 percent. DGM also gives better results and is more consistent across all of the different classes of classification anomaly. In [17], the authors say that CGAN makes it easier to get a close approximation of the actual distribution of data and data made for the class of minority from different types of

unbalanced datasets. In their studies, which were done on a number of datasets with disparate levels of imbalance, they contrast CGAN's performance to that of a number of different oversampling methods, such as Random Oversampling, ADASYN, SMOTE, Cluster-SMOTE, and Borderline-SMOTE, and they found improvements that were worth pursuing. The authors [18] suggest putting restrictions on the CGAN's networks to limit how much freedom there is to converge. This makes it less likely that standard GAN will converge slowly or not at all, which can happen because it has a high degree of freedom.

According to research [19], many studies use the GAN framework to generate synthetic samples for cyberattack datasets, which are some of the most commonly used datasets, such as KDD99 [5], [20] and NSL-KDD [21], [22]. Researchers discovered that existing datasets are mostly old and unreliable, with some lacking traffic diversity and volume. Some issues do not reflect the current attack trend.

Among the recently released datasets, the UNSW-NB15 [23], [24] dataset reflects the latest attacks. However, it is not suitable as the dataset for this study because the types of attacks are less than those of the CICIDS2017 dataset and the item of attributes is slightly different. As a consequence, in this study, we used the CICIDS2017 dataset because they represent real-world attacks [25]. Sharafaldin [26] from the Canadian Institute of Cybercrime has created the CICIDS2017 dataset. It comprises typical data that is comparable to actual data as well as the most recent sorts of assaults. This dataset contains normal data in addition to 14 different types of attack. The percentage of the data that is almost identical to the actual network is over 80 percent, while less than one percent of the data belongs to a minority class. The CICIDS2017 dataset [27] has new attack types like Infiltration, Bot, Heartbleed, and Web attacks including Web Attack\_Brute\_Force, Web Attack\_SQL\_Injection, and Web Attack\_XSS.

This study talks about using a Data Generative Model (DGM) based on CGAN to generate new minority data on unbalanced datasets and detect new attacks with CICIDS2017 datasets.

## 2. Material and methods

In this study, the author thinks about applying a data generative model (DGM) to oversampling on the CICIDS2017 dataset and looking at how minority oversampling affects classification performance using CGAN integrated KL-Divergence with machine learning based on Decision Tree, Random Forest, Support Vector Machine (SVM), and Neural Network (MLP). A detailed explanation of material and methodology is in the section below.

2.1. Proposed model

Our study implementation of the Data Generative Model (DGM) is based on CGAN[28] integrated KL-Divergence and the popular machine learning approach. It is a very important approach to observe the effectiveness of DGM to detect new attacks in classification performance. This experiment using the CICIDS2017 dataset represents real-world attacks. Our experiment associated a multi-class evaluation process that employs confusion metrics, including f1-score, recall, precision, and weighted f1-score. Dos hulk, Port scan, DoS Golden Eye, DoS Slowloris, FTP Patator, DDoS, SSH Patator, DoS Slowhttpstest, Bot, Infiltration, Heartbleed, and Web attack including Web Attack Brute Force, Web Attack SQL Injection, Web Attack XSS, are among the 14 possible condition categories in the multi-class experiment. We compared four machine learning models: Decision Tree, Neural Network (MLP), Random Forest and SVM.

Then, they would be integrated into augmentation data based on CGAN integrated KL-Divergence. The GAN Model has been enhanced by CGAN. Below, Figure 1 illustrates the model presented for this study in detail.

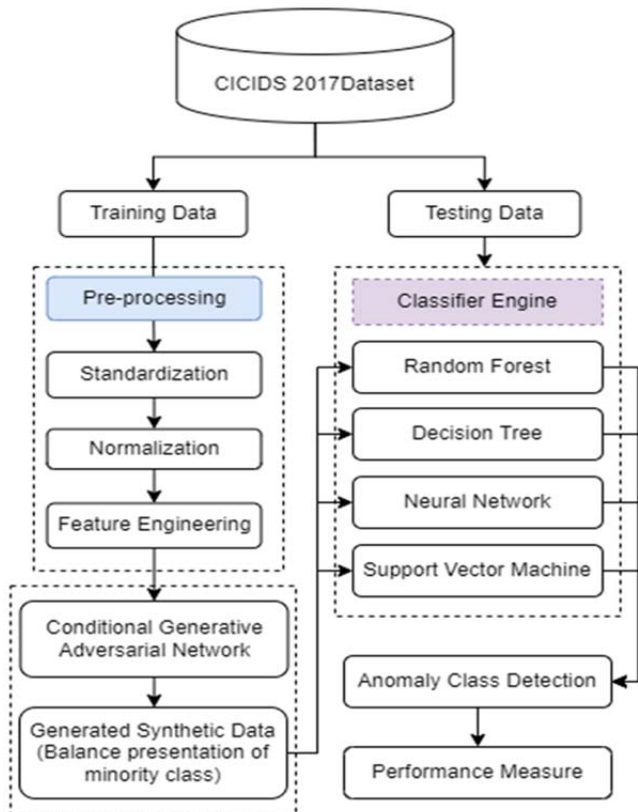


Figure 1. Detail process model and experiment scenarios

Phase 1 CICIDS2017 dataset description.

Experiments were conducted using the CICIDS2017 [26], which is the most popular dataset that is readily available to the public. When evaluating anomaly detection methods, these datasets are frequently used as standards because of their consistency. Based on the percentage that each class's samples make up in the overall dataset, we categorize each class as either belonging to the majority or the minority for the dataset. If a category has less than 10 per cent of the total population, it is placed in the minority category; otherwise, it is placed in the majority category.

Sharafaldin [26] from the Canadian Institute of Cybercrime has created the CICIDS2017 dataset. It comprises typical data that is comparable to actual data as well as the most recent sorts of assaults. This dataset contains normal data in addition to 14 different types of assault. The percentage of the data that is almost identical to the actual network is over 80 percent, while less than one percent of the data belongs to a minority class, which includes Bot, infiltration, Heartbleed, and web attacks including web\_attack\_brute\_force, web\_attack\_sql\_injection, and web\_attack\_xss, which are all examples of this minority class [27]. Table 1 shows the percentage distribution of the CICIDS2017 dataset class.

Phase 2 pre-processing of data. Preparing the dataset so that it can enter the next process is the main goal of preprocessing. Nominal data is converted into ordinal integers where numerical input and output variables are required by GAN to function properly. Standardization and normalization of data sets are utilized in machine learning algorithms.

We standardize each data sample to the unit standard and do not change the sample value to match the unit standard. Otherwise, it places the value in each column with a robust scalar [29].

Because the data is so uneven, feature engineering is a way to avoid the effect of outliers, which is a key part of the suggested strategy. It is the ideal option. In addition to this, the significant skewness of the data supports its standing as the best possible option. The median is removed from the data in each feature column, and the data is scaled based on the range between the first and third quartiles. On the basis of the Kendall's Spearman rank correlation coefficients and Pearson r, the feature columns that had a correlation of more than 95% or a constant value of more than 99.5% were eliminated [30]. As a component of the overall process of feature engineering, we got rid of each feature because they always had values of more than 99%.

Table 1. Distribution of training and testing sets CICIDS2017

Classes	Set of Training	Ratio	IR (%)	Set of Testing	Ratio	IR (%)	Class type
BENIGN	1589924	0.803189	80.318947	681396	0.803189	80.318920	Majority
DoS_Hulk	161087	0.081377	8.137709	69037	0.081377	8.137672	Majority
PortScan	111163	0.056157	5.615674	47641	0.056156	5.615639	Majority
DDoS	89618	0.045273	4.527275	38408	0.045273	4.527307	Majority
DoS_GoldenEye	7205	0.003640	0.363978	3088	0.003640	0.363995	Minority
FTPPatator	5554	0.002806	0.280574	2380	0.002805	0.280540	Minority
SSHPatator	4128	0.002085	0.208536	1769	0.002085	0.208519	Minority
DoS_slowloris	4057	0.002049	0.204949	1739	0.002050	0.204983	Minority
DoS_Slowhttptest	3849	0.001944	0.194442	1650	0.001945	0.194492	Minority
Bot	1369	0.000692	0.069158	587	0.000692	0.069192	Minority
Web_Attack_Brute_Force	1055	0.000533	0.053296	452	0.000533	0.053279	Minority
Web_Attack_XSS	456	0.000230	0.023036	196	0.000231	0.023103	Minority
Infiltration	25	0.000013	0.001263	11	0.000013	0.001297	Minority
Web_Attack_Sql_Injection	15	0.000008	0.000758	6	0.000007	0.000707	Minority
Heartbleed	8	0.000004	0.000404	3	0.000004	0.000354	Minority

IR=Imbalance Ratio

**Phase 3 conditional generative adversarial network (CGAN).** CGAN network architecture is made up of two feed-forward artificial neural networks. These networks are designed to mimic natural neural networks.

In the GAN configuration, a generator ( $G$ ) and a discriminator ( $D$ ) from a feed-forward neural network consisting of many layers. As suggested by [13], these two neural networks are taught to compete against one another so that they can successfully complete the goal of data production. The network  $G$  generator thought to be a nonlinear mapping function, and here's how to describe it:

$$G : (Z \times Y) \rightarrow X \quad (1)$$

where  $Z$  is a random noise vector that is produced by us using normal noise  $N(\mu, \sigma^2)$ . To obtain knowledge  $G$ , we take it for granted that we have access to underrepresented class pairings like as  $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), \dots, (x_n, y_n)\}$  with  $n$  in  $\in [1, N]$ . The sample  $G$  generator from previous distribution,  $p_z(z)$  where  $z \in Z$  and produce sample  $x$ . The produced sample  $x$  makes an effort to provide a sample that is as similar as possible to the class of minorities. The final purpose  $G$  is to understand how the data are spread out in the  $p_{data}(x, y)$  distribution of the data, and its definition includes the following:

$$D : (X \times Y) \rightarrow [0,1] \quad (2)$$

The data that is sent into the discriminator ( $D$ ), as well as the  $x$  class label  $y$ .

The function of the discriminator is to assess if a particular data sample comes from a true class

distribution or whether it was created from  $p_z(z)$ . The discriminator's job is to recognize the sample  $x$ .

### Training of network

The networks of generator and discriminator start competing against one another in a way similar to a two-player minimax game while concurrently being taught [28].

The following is a definition of the value function  $V$ :

$$\min_G \min_D V(D, G) = E_D + E_G \quad (3)$$

In the event that a discriminator is involved, the expectation is computed as follows:

$$E_D = \mathbb{E}_{x,y \sim p_{data}(x,y)} [\log D(x|y)]$$

In the same way, the following formula is used to figure out the expectation for the generator:

$$E_G = \mathbb{E}_{z \sim p_z(Z), y \sim p(y)} [\log(1 - D(G(z, y)|y))]$$

The value function that is specified in Equation 3 is the following:

$$\begin{aligned} \min_G \min_D V(D, G) &= \mathbb{E}_{x,y \sim p_{data}(x,y)} [\log D(x|y)] \\ &+ \mathbb{E}_{z \sim p_z(Z), y \sim p(y)} [\log(1 - D(G(z, y)|y))] \end{aligned} \quad (4)$$

The first term in equation 4 is the discriminator, which depends on the class in the label of  $y$ . The term second is expectancy, which is based on two variables that are independent, which are  $z$  and  $y$ .

SGD, or "stochastic gradient descent," was utilized to reduce loss across networks of training. A discriminator and generator networks, for a given number of steps, work to achieve the lowest possible error rate of size  $m$  for mini batches. Based on Equation 4, the logistic cost functions are shown below so that we can figure out how much each neural network's weights should be changed:

$$J(\theta)_D = -\frac{1}{2m} + \left( \sum_{i=1}^m \log D(x_i|y_i) + \sum_{i=1}^m \log(1 - D(G(z_i, y_i)|y_i)) \right) \quad (5)$$

$$J(\theta)_D = -\frac{1}{m} \sum_{i=1}^m \log D(G(z_i, y_i)|y_i) \quad (6)$$

The cost functions for the discriminator are denoted by  $J_D$ , and the cost functions for the generator are denoted by  $J_G$ . In a perfect situation, the discriminator of the neural network's accuracy keeps getting better until it reaches 50%, and the KL-divergence [31] stays as low as possible. The fact that the samples of the data generated by the generator model in the KL-divergence tend to be close to nought shows that the samples of data made by the generator have the identical distribution as samples of real-class.

Accordingly, the discriminator isn't able to tell the difference between the original sample and the sample that was made. The KL-divergence may be found by using the following formula:

$$KL = (Distribution(p_{data}) || Distribution(p_z)) = \sum p_{data}(x) \log \left( \frac{p_{data}(x)}{p_z(x)} \right) \quad (7)$$

In Equation 7, the KL-divergence utilized for the network's stability convergence also serves as a signal for the generation of a sample for the minority class.  $p_z$  distribution is an estimate of the true distribution of training instances, denoted by the notation (i.e.,  $p_{data}$ ). In this instance, the distribution of  $p_{data}$  is known to us, and based on this information, we are inferring how it was distributed  $p_z$  in the proposed model when it is in the training phase of generator networks.

There is also an estimate of the maximum likelihood, which is defined as the minimum KL-divergence among the model and the generated distribution of data. The KL-divergence of the model's generated data samples is close to zero. This signifies the generator makes samples of data with the identical distribution as samples from the actual

class. Accordingly, that discriminator can't tell the difference between samples that are real and samples that are made. During the process's training phase, the distribution  $p_z$  is computed after each iteration of the process. As illustrated in Figure 2.

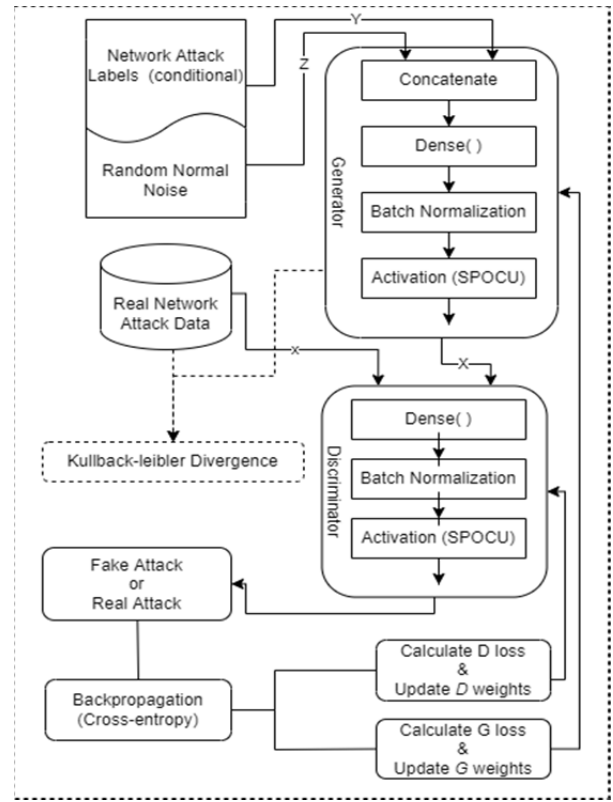


Figure 2. Conditional Network Adversarial Network (CGAN) Architecture

**Phase 4 data generation.** When the model's training is done, the network will be able to provide synthetic data for the supplementary classes. To effectively train a classifier using machine learning, it is important to specify the number of data samples created for each class for minorities. So that the dataset is balanced, we set up the generator so that it makes about the same number of samples from each class. This gets rid of the problem of the presence of underrepresented classes. Performance measures are used to figure out how useful and important the data generated is. Several different metrics, including recall, accuracy, F1-score, precision, and F1-score weighted, are used to judge the performance. Accuracy as a measure of performance isn't used because of something called the accuracy paradox [32].

**Phase 5 classifier engine.** The data generated from the model is analyzed for effectiveness with training the most prevalent models of machine-learning utilized by several classifiers. The decision tree classification selects the optimal features for the root node using the notion of information gain ratio and then constructs the entire tree [33],[34]. From the ensemble learning family, an algorithm of random

forest contains several decision trees that use the voting principle to label their inputs. The bootstrap aggregation technique is used to train different decision trees. This method is utilized to offer a distinct training subset for each decision tree. The goal is to reduce variation by changing how the tree learns so that at each split, a random subset of characteristics is chosen [35],[36]. Meanwhile, SVM, known as the support vector machines [37],[38], is used as a binary classification of the multi-class variant, which on decision boundary separation can accurately categorize data points. Moreover, analysing the performance of the data provided by the neural network is the optimal method for accomplishing complicated matters. It is believed that the multi-layer feed-forward neural network [39][40] can classify the class of anomalies.

**Phase 6 performance measure:** evaluation metrics standards are used including precision, f1-score, recall, and f1-score weighted. The confusion matrix value used for the calculation is computed as follows:

$$\text{Precision} = \frac{TP}{(TP + FP)} \quad (11)$$

$$\text{Recall} = \frac{TP}{(TP + FN)} \quad (12)$$

$$\text{F1 - score} = 2 \frac{\text{Precision} * \text{Recall}}{(\text{Precision} + \text{Recall})} \quad (13)$$

$$\begin{aligned} \text{Weighted F1 - score} \\ = \frac{\sum_{i=1}^k \text{Support}_i \cdot \text{F1}_i}{(\text{Total})} \end{aligned} \quad (14)$$

Where true positives are represented by TP, that reflects the frequency of the sample of anomalous with positive results. True negatives are represented by TN, that reflect the proportion of the sample of normal with negative results. FP represents false positives, which means how many aberrant samples were found to be positive.

FN represents false negatives, which means how many suspicious samples were found to be negative. Support is the amount of instances given a particular label i where F1i is the F1-score for that label. In addition, to ensure the classifiers and model of data generation that were proposed did not generate the results by change, k-fold cross-validation was utilized.

## 2.2. Definition of hyperparameter

In experiments, in the CICIDS2017 datasets, the recommended architecture for neural networks was modified. To avoid data-specific and time-consuming adjustments, hyperparameter adjustments for machine learning classifiers were made. Optimize hyperparameter selection to improve classification performance. Table 2 contains the optimal model hyperparameters.

Table 2. hyperparameters optimization of model

Parameter	CICIDS2017 (G)	CICIDS2017 (D)
Epochs	2000	2000
Drop out	-	-
Optimizer	SGD	SGD
Learning rate	0.0005	0.0005
Layer	4	4
Input of layer neurons	33	35
Layer 1 neurons	128	512
Layer 2 neurons	256	256
Layer 3 neurons	512	128
Layer 4 neurons	34	1
Output of layer neurons	35	1
Batch Dimensions	128	128
Activate function	ReLU	ReLU
Noise random	32	-

In Table 2, SGD is utilized to reduce loss. The number of training epochs was chosen by the convergence of the KL-divergence between actual and simulated data samples.

The activation function of the ReLU, which represents a rectified linear unit, is based on percolation. After network convergence, the model is capable of producing samples of the class of minorities for presentation purposes.

## 3. Results and discussions

We ran tests on the CICIDS2017 dataset to see how well the DGM model worked with real-world network traffic. The first generates a significant sample of the minority class to balance their percentages. In this experiment, an up-sampling strategy was applied to the dissimilar types of minority class attacks in the CICIDS2017, including Bot = 6000, Dos\_Goldeneye = 8000, DoS\_Slowhttptest = 5000, DoS\_slowloris = 6000, FTPPatator = 7000, Heartbleed = 100, Infiltration = 100, SSHPatator = 5500, Web\_Attack\_Brute\_Force = 1000, Web\_Attack\_Sql\_Injection = 100, and Web\_attack\_XSS = 500. Also, a down-sampling strategy was used for attacks like DDoS, DoS\_Hulk, and PortScan, which make up the majority of attacks. The DGM model is proven to improve machine learning performance, especially in the class of minorities, those are significant concerns in

unbalanced datasets. The results of the experimentation on the CICIDS2017 datasets with the evaluation metric including F1-score, precision, and recall can be observed in table 3. In addition, Cross-validation k-fold with  $k = 5$  was used for validation, which made certain that the suggested data-driven model and the classifiers' results were not just a matter of chance.

As shown in table 3, the DGM model has succeeded in increasing the efficiency of machine learning to detect IDS attacks, especially against minority-class attacks. The DGM model with Random Forest can achieve an average precision value of 98%, recall 97%, and f1-score of 97%. The DGM model with Decision Tree achieved an average precision value of 94%, recall of 95%, and f1-score

of 94%. The DGM model with MLP achieved an average precision value of 90%, recall of 88%, and f1-score of 87%. The DGM model with SVM achieved an average precision value of 85%, recall of 87%, and f1-score of 81%. Overall, the DGM model enables the improved detection of new attacks, especially for extreme class imbalances. But in the DGM model with MLP and SVM, there are several classes of attacks that are not detected properly. As in the SVM model, the Web\_Attack\_Brute\_Force was detected poorly. It could be visible from the low values of 14% recall and 24% f1-score. The slight values for 4% recall, 8% f1-score show how badly the MLP model found the Web\_Attack\_XSS attack. It means the classifiers correctly predicted are poorer because the recall value is lower than the precision.

Table 3. Classification results using CICIDS2017 Dataset

Class	RF			DT			MLP			SVM		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Bot	99%	100%	100%	100%	100%	100%	99%	100%	99%	96%	100%	98%
DDoS	100%	100%	100%	100%	100%	100%	100%	99%	100%	97%	99%	98%
DoS_GoldenEye	99%	100%	100%	99%	100%	100%	99%	99%	99%	98%	97%	97%
DoS_Hulk	100%	100%	100%	100%	100%	100%	100%	100%	100%	99%	98%	99%
DoS_Slowhttptest	100%	100%	100%	100%	100%	100%	92%	98%	95%	88%	98%	93%
DoS_slowloris	100%	100%	100%	100%	100%	100%	93%	93%	93%	92%	92%	92%
FTPPatator	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Heartbleed	100%	100%	100%	76%	100%	85%	100%	100%	100%	100%	100%	100%
Infiltration	100%	91%	95%	98%	91%	94%	78%	75%	75%	82%	85%	84%
PortScan	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
SSHPatator	100%	100%	100%	100%	100%	100%	98%	100%	99%	98%	100%	99%
Web_Attack_Brute_Force	92%	96%	96%	91%	93%	92%	69%	94%	80%	<b>99%</b>	<b>14%</b>	<b>24%</b>
Web_Attack_Sql_Injection	97%	83%	89%	70%	67%	68%	70%	67%	68%	3%	33%	6%
Web_Attack_XSS	90%	82%	86%	84%	83%	83%	<b>64%</b>	<b>4%</b>	<b>8%</b>	34%	98%	51%

To validate the effectiveness and performance of the DGM model with existing synthetic data methods. In this experiment we compared the DGM model with ADASYN [41], SMOTEENN [42], Boarderline-SMOTE [43] and SVMSMOTE [44]. The experiment was conducted by evaluating the k-fold cross-validation where  $k=5$  (5k-Fold) with the evaluation metric of F1-score weighted. Table 4 displays the accumulated results.

Table 4. The weighted F1-score comparison results using the CICIDS2017 dataset

Model	RF	DT	MLP	SVM
ADASYN	-	-	-	-
SMOTEENN	99.87%	99.83%	99.47%	98.64%
Borderline-SMOTE	99.90%	99.86%	99.47%	98.36%
SVMSMOTE	99.90%	99.87%	99.59%	98.53%
DGM	99.92%	99.90%	99.40%	98.52%

As shown in table 4, our experiment failed to generate new data synthesis when using the ADASYN method, and the results were not obtained for that method. The DGM model provides better performance than other methods when applied to the RF and DT classification models, but the performance decreases when using the MLP and SVM classification models. This may be seen from the results obtained in the RF model, with a F1-score weighted of 99.92%, and the DT model, with a F1-score weighted of 99.90%.

The DGM model has the lowest achievement compared to other models when applied to the MLP model, with an achievement of a F1-score weighted of 90.40%.

When the DGM model is applied to the SVM classification model, the performance is better than Borderline-SMOTE, with an F1-score of 98.52%. However, compared to SMOTEENN and SVM SMOTE, the DGM model has lower performance. Overall, the DGM model provides better performance compared to other models.

Referring to research [16], the DGM model showed a fairly good overall performance in detecting attacks on minority classes. This was demonstrated by the test results of the UNSW-NB15 classification model dataset, which included RF, DT, MLP, and SVM. But some attacks are not detected properly, such as analysis, backdoors, shellcode, and worm attacks. According to Dlamini [16], all classification models give poor results when detecting "analysis" attacks. This can be visible from the recall test score and f1-score obtained 0% results, which means that attacks are not detected at all. In backdoor attacks, recall scores and f1-scores were also below 4%, so it can be said that these attacks were also detected poorly by all of these classification models. Shellcode attacks on MLP and SVM models were also detected poorly, with recall scores and f1-scores below 1%. For worm attacks, the RF and MLP models still detected poor results with recall scores and f1-scores below 3%.

However, the DGM model against other minority class attacks is still detectable well, such as shellcode against the RF and DT models, while reconnaissance is detected well by all of these classification models. When compared with the test results with the CICIDS2017 dataset, the DGM model shows better results in detecting new attacks. This can be seen in table 3. The worst detected attacks are only web\_attack\_brute\_force attacks on the SVM model and web\_attack\_XSS against the MLP model. Overall, all attacks were detected properly by all classification models used during testing.

#### 4. Conclusion

According to the experimental result in table 3, Synthetics oversampling consistently provides a better recognition rate for the minority class. This happened in GAN framework oversampling for this dataset under study. It is expected behavior because the algorithms learn the minority class with more data so that the model can easily capture the pattern. Based on comparison results, a GAN framework balanced dataset has been demonstrated to have a better recognition rate for the minority class.

For further research, it is necessary to investigate the performance improvement of the proposed model by optimizing the hyper-parameters and validating the new synthesis data generated from the proposed model with a statistical approach.

#### Acknowledgements

*The authors would like to thank Universitas AMIKOM Purwokerto for the financial support and the Faculty of Information and Communications Technology, Universiti Teknikal Malaysia Melaka (UTeM) for their assistance in this research.*

#### References

- [1]. Sridhar, S., & Kalaivani, A. (2020). A Survey on Methodologies for Handling Imbalance Problem in Multiclass Classification. *Advances in Intelligent Systems and Computing*, 775–790. [https://doi.org/10.1007/978-981-15-5029-4\\_67](https://doi.org/10.1007/978-981-15-5029-4_67)
- [2]. Liu, X., Li, T., Zhang, R., Wu, D., Liu, Y., & Yang, Z. (2021). A GAN and Feature Selection-Based Oversampling Technique for Intrusion Detection. *Security and Communication Networks*, 2021, 1–15. <https://doi.org/10.1155/2021/9947059>
- [3]. Ding, H., Chen, L., Dong, L., Fu, Z., & Cui, X. (2022). Imbalanced data classification: A KNN and generative adversarial networks-based hybrid approach for intrusion detection. *Future Generation Computer Systems*, 131, 240–254. <https://doi.org/10.1016/j.future.2022.01.026>
- [4]. Dina, A.S., Siddique, A. B., & Manivannan, B.(2022). Effect of Balancing Data Using Synthetic Data on the Performance of Machine Learning Classifiers for Intrusion Detection in Computer Networks. *IEEE Access*, 10(96731-96747). doi: 10.1109/ACCESS.2022.3205337.
- [5]. Divekar, A., Parekh, M., Savla, V., Mishra, R., & Shirole, M. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*. <https://doi.org/10.1109/cccc.2018.8586840>



- [6]. Singh, V. K., & Govindarasu, M. (2021). A Cyber-Physical Anomaly Detection for Wide-Area Protection Using Machine Learning. *IEEE Transactions on Smart Grid*, 12(4), 3514–3526. <https://doi.org/10.1109/tsg.2021.3066316>
- [7]. Nguyen, T. C. H., Le-Nguyen, M. K., Le, D. T., Nguyen, V. H., Tôn, L. P., & Nguyen-An, K. (2022). Improving Web Application Firewalls with Automatic Language Detection. *SN Computer Science*, 3(6). <https://doi.org/10.1007/s42979-022-01327-2>
- [8]. Wang, J., Pan, J., AlQerm, I., & Liu, Y. (2021). Def-IDS: An Ensemble Defense Mechanism Against Adversarial Attacks for Deep Learning-based Network Intrusion Detection. In *2021 International Conference on Computer Communications and Networks (ICCCN)*. <https://doi.org/10.1109/icccn52240.2021.9522215>
- [9]. Wei, C., Sohn, K., Mellina, C., Yuille, A., & Yang, F. (2021). CReST: A Class-Rebalancing Self-Training Framework for Imbalanced Semi-Supervised Learning. In *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. <https://doi.org/10.1109/cvpr46437.2021.01071>
- [10]. Arif, A., Javaid, N., Aldegheishem, A., & Alrajeh, N. (2021). Big data analytics for identifying electricity theft using machine learning approaches in microgrids for smart communities. *Concurrency and Computation: Practice and Experience*, 33(17). <https://doi.org/10.1002/cpe.6316>
- [11]. Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1). <https://doi.org/10.1186/s40537-019-0192-5>
- [12]. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321–357. <https://doi.org/10.1613/jair.953>
- [13]. Ian J. G. *et al.*, (2014). *Generative Adversarial Networks*. *Adv Neural Inf Process Syst*, 2672–2680.
- [14]. Lin, Z., Shi, Y., & Xue, Z. (2022b). IDSGAN: Generative Adversarial Networks for Attack Generation Against Intrusion Detection. *Advances in Knowledge Discovery and Data Mining*, 13282, 79–91. [https://doi.org/10.1007/978-3-031-05981-0\\_7](https://doi.org/10.1007/978-3-031-05981-0_7)
- [15]. Zheng, M., Li, T., Zhu, R., Tang, Y., Tang, M., Lin, L., & Ma, Z. (2020). Conditional Wasserstein generative adversarial network-gradient penalty-based approach to alleviating imbalanced data classification. *Information Sciences*, 512, 1009–1023. <https://doi.org/10.1016/j.ins.2019.10.014>
- [16]. Dlamini, G., & Fahim, M. (2021). DGM: a data generative model to improve minority class presence in anomaly detection domain. *Neural Computing and Applications*, 33(20), 13635–13646. <https://doi.org/10.1007/s00521-021-05993-w>
- [17]. Douzas, G., & Bacao, F. (2018). Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems With Applications*, 91, 464–471. <https://doi.org/10.1016/j.eswa.2017.09.030>
- [18]. Ye, J., Fang, Y., & Ma, J. (2019). Intrusion Detection Model Based on Conditional Generative Adversarial Networks. In *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*. <https://doi.org/10.1145/3377713.3377807>
- [19]. Vu, L., Bui, C. T., & Nguyen, Q. U. (2017). A Deep Learning Based Method for Handling Imbalanced Problem in Network Traffic Classification. In *Proceedings of the Eighth International Symposium on Information and Communication Technology*. <https://doi.org/10.1145/3155133.3155175>
- [20]. UCI KDD Archive. (1999). *KDD Cup 1999 Data*. Retrieved from: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html> [accessed: 03 August 2022].
- [21]. University of New Brunswick. *Nsl-kdd dataset*. Retrieved from: <https://www.unb.ca/cic/datasets/nsl.html> [accessed: 15 September 2022].
- [22]. Tavallae, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. <https://doi.org/10.1109/cisda.2009.5356528>
- [23]. UNSW Sydney. *Unsw-nb15 dataset description*. Retrieved from: <https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/> [accessed: 24 September 2022]
- [24]. Moustafa, N., & Slay, J. (2015). 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)*. <https://doi.org/10.1109/milcis.2015.7348942>
- [25]. Maseer, Z. K., Yusof, R., Bahaman, N., Mostafa, S. A., & Foozy, C. F. M. (2021). Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. *IEEE Access*, 9, 22351–22370. <https://doi.org/10.1109/access.2021.3056614>
- [26]. Sharafaldin, I., Habibi Lashkari, A., & Ghorbani, A. A. (2019). A Detailed Analysis of the CICIDS2017 Data Set. *Communications in Computer and Information Science*, 172–188. [https://doi.org/10.1007/978-3-030-25109-3\\_9](https://doi.org/10.1007/978-3-030-25109-3_9)
- [27]. Sharafaldin, I., Habibi Lashkari, A., & Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In the *Proceedings of the 4th International Conference on Information Systems Security and Privacy*. <https://doi.org/10.5220/0006639801080116>
- [28]. Mirza, M. and Osindero, S. (2014). *Conditional Generative Adversarial Nets*.arXiv. Retrieved from: <https://doi.org/10.48550/arXiv.1411.1784> [accessed: 28 September 2022].
- [29]. Pedregosa, F., *et al.*, (2011). Scikit-learn: machine learning in python. *The Journal of machine Learning research*, 12, 2825–2830. <https://dl.acm.org/doi/10.5555/1953048.2078195>

- [30]. Lewis-Beck, M., A. Bryman, and T. Futing Liao, (2012). *The SAGE Encyclopedia of Social Science Research Methods*. London: Sage Publications. doi: 10.4135/9781412950589.
- [31]. Kullback, S., & Leibler, R. A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1), 79–86. <https://doi.org/10.1214/aoms/1177729694>
- [32]. He, H., & Y. Ma. (2013). *Imbalanced Learning*. New York: Wiley. doi: 10.1002/9781118646106.
- [33]. Salzberg, S. L. (1994). C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. *Machine Learning*, 16(3), 235–240. <https://doi.org/10.1007/bf00993309>
- [34]. Karatas, G., Demir, O., & Sahingoz, O. K. (2020). Increasing the Performance of Machine Learning-Based IDSs on an Imbalanced and Up-to-Date Dataset. *IEEE Access*, 8, 32150–32162. <https://doi.org/10.1109/access.2020.2973219>
- [35]. Tin Kam Ho, (1995). Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1, 278–282. doi: 10.1109/ICDAR.1995.598994.
- [36]. Lee, J., & Park, K. (2019). GAN-based imbalanced data intrusion detection system. *Personal and Ubiquitous Computing*, 25(1), 121–128. <https://doi.org/10.1007/s00779-019-01332-y>
- [37]. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. <https://doi.org/10.1007/bf00994018>
- [38]. Alqarni, A. A., & El-Alfy, E. S. M. (2022). Improving Intrusion Detection for Imbalanced Network Traffic using Generative Deep Learning. *International Journal of Advanced Computer Science and Applications*, 13(4). <https://doi.org/10.14569/ijacsa.2022.01304109>
- [39]. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer Series in Statistics. <https://doi.org/10.1007/978-0-387-84858-7>
- [40]. Huang, S., & Lei, K. (2020). IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks. *Ad Hoc Networks*, 105, 102177. <https://doi.org/10.1016/j.adhoc.2020.102177>
- [41]. Haibo He, Yang Bai, Garcia, E. A., & Shutao Li. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* <https://doi.org/10.1109/ijcnn.2008.4633969>
- [42]. Batista, G. E. A. P. A., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1), 20–29. <https://doi.org/10.1145/1007730.1007735>
- [43]. Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *Lecture Notes in Computer Science*, 878–887. [https://doi.org/10.1007/11538059\\_91](https://doi.org/10.1007/11538059_91)
- [44]. Akbani, R., Kwek, S., & Japkowicz, N. (2004b). *Applying Support Vector Machines to Imbalanced Datasets*. *Machine Learning: ECML 2004*, 39–50. [https://doi.org/10.1007/978-3-540-30115-8\\_7](https://doi.org/10.1007/978-3-540-30115-8_7)