Neural Computing and Applications (2021) 33:13635–13646
https://doi.org/10.1007/s00521-021-05993-w

ORIGINAL ARTICLE

DGM: a data generative model to improve minority class presence in anomaly detection domain

Gcinizwe Dlamini[1] · Muhammad Fahim[1]

Received: 25 April 2020 / Accepted: 31 March 2021 / Published online: 17 April 2021

Abstract
Anomaly detection is a process to identify abnormal behavior that does not confirm the normal behavior. The abnormal behavior clues are few because it appears rarely. To detect each abnormal behavior, the problem is transformed into a multi-class classification task where it lies into data imbalance representation. In data imbalance setting, minority classes are over-sampled to improve the performance of the classifier. Existing methods are unable to learn the distribution of the minority class and effects the performance of classifier. In this research, we introduced a data generative model (DGM) to improve the minority class presence in the anomaly detection domain. Our approach is based on a conditional generative adversarial network to generate synthetic samples for minority classes. It includes the KL-divergence to guide the model towards the true learning of minority class distribution. In this way, model learns the complex underlying data distribution and generates new samples. We performed experiments over the two benchmark datasets NSL-KDD and UNSW-NB15 that are publicly available to demonstrate the effectiveness of our approach. Furthermore, the comparative analysis with the existing approaches confirms the stability and superiority of our presented model.

Keywords Intrusion detection system · Data balancing · Generative adversarial network · Cybersecurity

1 Introduction

Anomalies are an integral part of almost every system including cellular networks, banking, aviation, healthcare, internet of things (IoT), and many more. Detection of these anomalies is crucial in helping to reduce loss in terms of money, time, and wastage of resources. In practice, the developed models are based on semi-supervised approaches which are trained over the normal data and any deviation is considered as as anomaly or outlier. Such approaches are relatively easy to develop because they consider the problem as a binary classification and any deviation from normal class is considered as anomaly [1–4]. Although anomaly labels are valuable that can transform the learning problem into a multi-class supervised learning domain. In this domain, a model can predict the anomaly as well as discriminate the class of anomaly. The learning ability of multi-class supervised learning model depends on the provided training examples. One of the lower performance causes is the presence of a few examples (i.e., minority class) of certain anomalous class. In cybersecurity scenarios, network intrusion detection data are scarce. Most of the attacks are rare that leads to minority class presence in training data, which affects the model learning [3].

To deal with the minority class presence, researchers over the years have proposed solutions that can be mainly grouped into two approaches. The first is the development of machine learning (ML) algorithms that reinforce the learning towards the minority class. This approach is much more expensive in terms of time and required deep domain knowledge. It mainly consists of ML model hyper-parameters tuning. The second and most popular way to deal with minority class presence is the modification of data by re-balancing the class distribution. This is mainly achieved through over-sampling, under-sampling, or hybrid methods. In this paper, we focus on dataset re-balancing as a solution to the class imbalance problem.

✉ Muhammad Fahim
m.fahim@innopolis.ru

Gcinizwe Dlamini
g.dlamini@innopolis.university

[1] Innopolis University, Innopolis, Russia

Random over-sampling minority class is one of the most common and simplest methods proposed to balance the dataset. In a random over-sampling method, minority class data points are randomly duplicated to balance the dataset distribution. The limitation of such method leads to over-fitting of data [5]. An alternative approach to solve this over fitting problem in random over-sampling by generating the minority class examples using interpolation mechanism based on nearest neighbour approach [6]. This approach is dependent on the number of neighbour considered for generating sample data. However, the state-of-the-art data generation methods such as synthetic minority oversampling technique (SMOTE) algorithm suffer from the problem of over generalization [3]. In addition to over generalization, k-NN-based approaches are primarily designed for two-class imbalance data which makes it hard to use in real world cases where multiple classes are present in the form of minority class [7].

In such scenarios, generative models play an important role to deal with the situation. Recent developments of neural network-based approaches achieve high-quality results over classical methods. Especially, generative adversarial networks (GAN)-based models are assisting to generate realistic data. Technically, GANs can bypass the difficulty of approximating many intractable probabilistic computations to approximate the distribution of real training examples [8]. GANs generate new data samples that are close enough to existing examples. Consequently, the minority class presence will no longer exist which leads to the high performance of machine learning models.

In this research, our DGM model solves this issue by generating a realistic sample of the minority classes in the dataset. Our model has the following contribution.

- We designed and developed a variant of generative adversarial networks know as a conditional generative adversarial network for anomaly detection domain. Our model holds a particular condition over each minor class to generate precise synthetic samples.
- We introduce KL divergence to ensure the model generates close enough data points that are similar to the original data space.
- We performed a thorough analysis over standard benchmark datasets and code is freely available for the research community.[1]

The structure of this paper is as follows: Sect. 2 outlines the related work on imbalance data learning and anomaly detection domain. In Sect. 3, the proposed model is presented in detail, while Sect. 4 presents the implementation details and obtained results followed by discussions. The paper is concluded in Sect. 6 with possible future directions.

2 Related work

The minority class presence problem faced by machine learning models exists from over decades, and research community has proposed both statistical and machine learning models to solve it. The simplest approaches are under-sampling [9] and random over-sampling [6], which have been proposed as a common method of balancing data. The limitation of under-sampling is that it can discard potentially useful information. In random over-sampling method, minority classes data points are randomly duplicated to balance the dataset distribution. The limitation of such method leads to over fitting of data [5].

An alternative approach introduced by Chawla et al. [6] to solve this over-fitting problem in over-sampling by *k*-NN-based approach. They proposed synthetic minority over-sampling technique (SMOTE) that performed better than random over-sampling. The main idea is to generate minority class examples by interpolating between several minority class examples that lie together using nearest neighbor technique by considering *k* neighbors for every minority class. Abhishek et al. [10] used SMOTE to synthesize artificial minority class samples in anomaly detection domain. In their comprehensive study, they measured the effectiveness of SMOTE by comparing the performance of popular supervised learning methods, namely support vector machines (SVM), decision tree, random forest, neural networks and K-means. In the conclusion of their study, they noted that there exists a need for alternative approaches to improve anomaly detection over the currently available data.

After few years, SMOTE was found to be not fully addressing the problem of data imbalance and multiclass imbalance problem still exists [7]. Zhu et al. proposed a k-nearest neighbors (k-NN)-based synthetic minority oversampling algorithm, termed SMOM, to handle multi-class imbalance problems. Their approach mainly targeted class over generalization problem faced in balancing multiclass imbalance data. SMOM was evaluated over 27 multiclass imbalanced datasets with four different classifiers and appeared to superior than other compared over-sampling algorithms in terms of geometric mean, multi-area under curve (MAUC), and the recall on the smallest minority class

Choi et el. [11] came up with an unsupervised deep learning approach to solve the phenomenon of network intrusion data imbalance. They proposed an autoencoder-based approach for anomaly detection. The experiments are performed on NSL-KDD data and their approach

[1] https://github.com/Gci04/GANs_for_Network_Intrusion_Data.

outperformed SMOTE approach used in [10]. This unsupervised deep learning approach gave researchers a hint that its high time to explore more deeply deep learning models in the field of cyber security.

In this context, Douzas et al. [12] proposed a cGAN on 71 datasets with different imbalance ratios in their experiments. However, network intrusion data were not among the 71 datasets used for experimentation. Hung Ba [2] applied GANs in balancing credit card fraudulent transaction data. They utilized GAN, cGAN, Wasserstein GAN, conditional Wasserstein GAN. He also highlights that GAN frameworks produce better F1-score which is a result of more balanced values in precision and recall. With the aim of making network intrusion detection system robust, Lin et al. [13] proposed a GAN-based framework for generating network attacks. Their approach proved to be robust over NSL-KDD dataset and proved to generate network attacks. However, generated attacks are not distinguishable for specific class.

In light of the remarkable results brought by deep learning and generative models, we solve the problem by transforming the problem into neural architecture and design a data generative model (DGM). Our DGM can handle the minority multi class presence in the anomaly detection domain.

3 The proposed model

Lets' consider a multi-class supervised learning problem in the anomaly detection domain. The goal is to learn the model f from training examples $\mathcal{X}$ to predict the class label $\mathcal{Y}$ as:

$$f : \mathcal{X} \rightarrow \mathcal{Y} \tag{1}$$

For training, n observations are provided as a set of couples:

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

with $x_1 = \left(x_{k,1}, x_{k,1}, \ldots x_{k,d}\right)$ as d-dimensional vector. To obtain the high performance of machine learning model, the following constrain should be satisfied:

$$(Y)^{\text{minor}} \approx (Y)^{\text{major}} \tag{2}$$

We assume that $Y^{\text{minor}} \ll Y^{\text{major}}$ and $X \subset \mathcal{X}$ be a set of training examples defined as "minority" class along with $Y \subset \mathcal{Y}$. The required minority classes are:

$$(X, Y)^{\text{minor}} = \{x_i, y_i\}_{i=1}^{|X|} \tag{3}$$

From Eq. 3, the minority classes are extracted and supplied to generated model for training process. The block diagram is shown in Fig. 1.

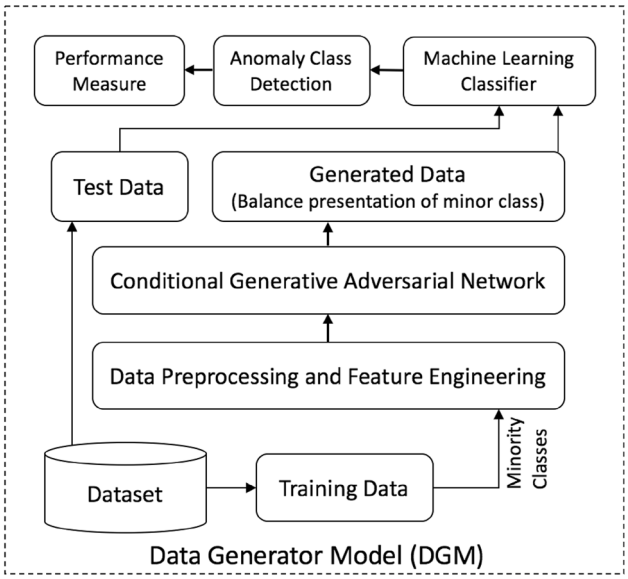

Fig. 1 The block diagram of proposed model (DGM)

It consists of the following four main components.

3.1 Data preprocessing and feature engineering

Data preprocessing is an essential part to curate the data for further processing. We convert the nominal data to ordinal integers. One main reason for converting nominal data to ordinal integers is that GAN requires numerical input and output variables. In machine learning models, standardization and normalization of a dataset are a common requirement; therefore, we normalized each data sample to unit norm. On top of normalizing, data samples to unit norm, we scale each column using robust scalar [14]. It fits as an optimal choice due to the high skewness of the data and serves as a way of avoiding outliers influence which is crucial in proposed approach. Each feature column removes the median and scales the data according to the inter-quartile range. On bases of Pearson r, Kendall τ and Spearman rank correlation those feature columns having more than 95% correlation or 99.5% constant value were removed [15]. As part of feature engineering, we removed all those features which have constant values more than 99%.

3.2 Conditional generative adversarial network

The designed of our network is comprised on two feed-forward artificial neural networks as shown in Fig. 2.

These two feed-forward artificial neural networks are known as Generator (G) and Discriminator (D) in GAN setting. To accomplish the task of data generation, these two neural networks are trained in an an adversarial manner as proposed by [16]. The generator network G is a

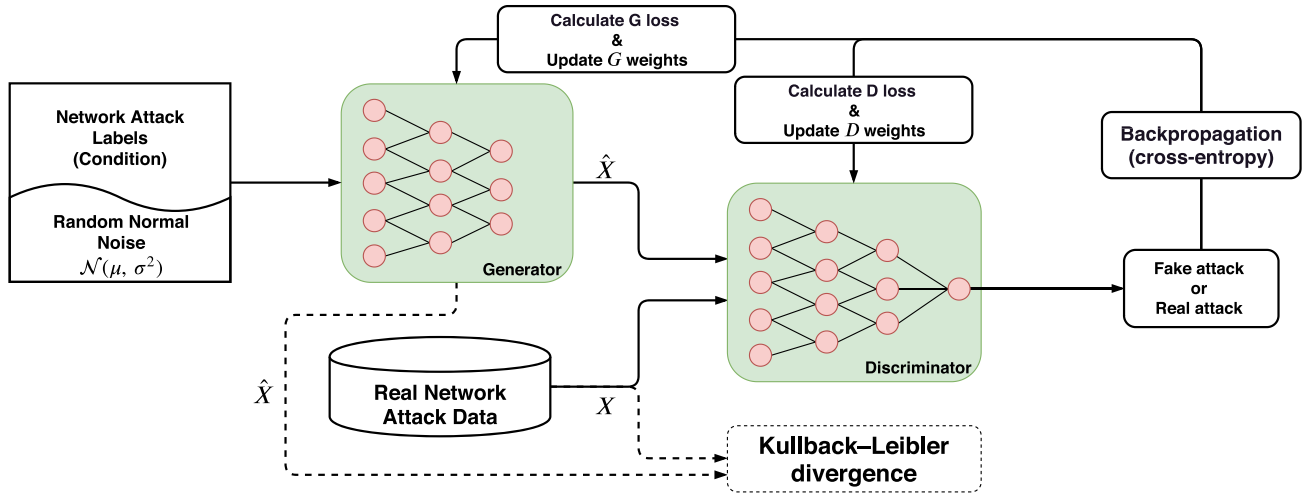

Fig. 2 A detailed model architecture based on conditional generative adversarial network

multi-layer feed-forward neural network, which is considered as nonlinear mapping function and defined as:

$$G : (Z \times Y) \rightarrow X \tag{4}$$

where Z is random noise vector that we generate through normal noise $\mathcal{N}(\mu, \sigma^2)$. To learn G, we assume that we have access to minority class pairs such as $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)\}$ with n in $\in [1, N]$. The generator G samples form prior distribution $p_z(z)$ where $z \in Z$ and generate sample x. The generated sample x tries to generate sample which may close enough to minority class. The ultimate goal of G is to learn the data distribution $p_{\text{data}}(x, y)$ empirically and produce data sample belonging to target minority class. The discriminator network D is a multi-layer feed-forward neural network is defined as:

$$D : (X \times Y) \rightarrow [0, 1] \tag{5}$$

The input to the discriminator (D) is x and class label y. The role of discriminator is to recognize the sample x either the data sample belongs to real class distribution or generated from $p_z(z)$.

3.2.1 Network training

The two networks (i.e, generator and discriminator) compete in a two-player minimax game manner and trained simultaneously [8]. The value function V is defined as follows:

$$\min_G \max_D V(D, G) = E_D + E_G \tag{6}$$

In case of discriminator, the expectation is calculated as:

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

Similarly, in case of generator, the expectation is calculated as:

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

The value function defined in Eq. 6 becomes:

$$\begin{aligned} \min_G \max_D & V(D, G) \\ &= \mathop{\mathbb{E}}_{x,y \sim p_{\text{data}}(x,y)} [\log D(x|y)] \\ &\quad + \mathop{\mathbb{E}}_{z \sim p_z(z), y \sim p(y)} [\log(1 - D(G(z, y)|y))] \end{aligned} \tag{7}$$

In Eq. 7, first term is discriminator which is conditioned over the class label y and second term is expectation over two independent variables (i.e., z and y). The stochastic gradient descent (SGD) approach is used to minimize the loss over training networks. For s number of steps, the discriminator and generator networks minimize the error over the mini batches of size m. To calculate how much to update the neural network weights for each network, the following logistic cost functions are based on Eq. 7:

$$\begin{aligned} J(\theta)_D = -\frac{1}{2m} \Big(& \sum_{i=1}^{m} \log D(x_i|y_i) \\ & + \sum_{i=1}^{m} \log \left(1 - D(G(z_i, y_i)|y_i)\right) \Big) \end{aligned} \tag{8}$$

$$J(\theta)_G = -\frac{1}{m} \sum_{i=1}^{m} \log D(G(z_i, y_i)|y_i). \tag{9}$$

where J_D is cost functions for discriminator and J_G cost functions for generator. In an ideal situation, the accuracy of the discriminator neural network converges to 50% and the KL-divergence [17]. The data samples from the generator model converge to zero in KL-divergence, which means that the generator produces data samples having same distribution as real class samples. Thus, the

discriminator can no longer differentiate between real and generated samples. The KL-divergence is calculated as follows:

$$\begin{aligned} &\mathrm{KL}(\mathrm{Distribution}(P_{\mathrm{data}})||\mathrm{Distribution}(P_z)) \\ &= \sum p_{\mathrm{data}_i}(x) \log\left(\frac{p_{\mathrm{data}}(x)}{p_{z_i}(x)}\right) \end{aligned} \tag{10}$$

In Eq. 10, the KL-divergence is used for the convergence of the network stability and provide a signal to generate minority class samples. The distribution of P_z is the approximation of the real distribution of training examples (i.e., P_{data}). In our case, the distribution of P_{data} is known to us and we are inferring the distribution P_z of generator network in proposed model during the training phase. Furthermore, it is a maximum likelihood estimation as minimizing the KL-divergence between the generated data distribution and the model. The data samples from the generator of DGM converge to zero in KL-divergence, which means that the generator produces data samples having same distribution as real class samples. Thus, the discriminator can no longer differentiate between real and generated samples. The distribution P_z is calculated after each iteration during the training phase. The detailed algorithm is presented in algorithm 1.

Algorithm 1 Training mechanism of conditional GAN based on minibatch stochastic gradient descent.

- **Input:** $k = 1$: the number of steps applied to discriminator, $\alpha = 0.0005$: learning rate, $n = 128$: mini batch size, θ_D and θ_G: initialize the network parameters
- **Output:** Trained DGM

for number of training iterations **do**
  for k steps **do**
  - Sample of m noise samples $\{\boldsymbol{z}_{(1)}, \ldots, \boldsymbol{z}_{(m)}\}$ from noise prior $p_z(\boldsymbol{z})$.
  - Sample of m examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ from $p_{\mathrm{data}}(\boldsymbol{x}, \boldsymbol{y})$.
  - Update the discriminator D by ascending its stochastic gradient $(J(\theta)_D)$:

$$J(\theta)_D = -\frac{1}{2m}\Big(\sum_{i=1}^{m} logD(x_i|y_i) + \sum_{i=1}^{m} log(1 - D(G(z_i, y_i)|y_i))\Big).$$

  end for
  - Sample minibatch of m noise samples $\{\boldsymbol{z}_{(1)}, \ldots, \boldsymbol{z}_{(m)}\}$ from noise prior $p_z(\boldsymbol{z})$.
  - Update the generator G by descending its stochastic gradient $(J(\theta)_G)$:

$$J(\theta)_G = -\frac{1}{m}\sum_{i=1}^{m} logD(G(z_i, y_i)|y_i).$$

end for
Note: the number of training iterations are determined by KL-divergence

3.3 Data generation

After the training of DGM model, the network is able to generate the synthetic data for minor classes Sect. 3. The decision on how many data samples to generate for specific minority class in order to induce machine learning classifier is critical. To balance the dataset, we generate number of samples for each class by conditioning the generator and eliminate the problem of minority class presence. The effectiveness and impact of the generated data are measures in terms of performance measures. The performance is measured in terms of precision, recall, F1-score, and weighted F1-score. We avoid using accuracy as a performance measure because of the accuracy paradox [18].

3.4 Machine learning classifier

To analyze the effectiveness of generated data by DGM, we trained most commonly used machine learning models from different classifier families. The decision tree classifier is based on C4.5 algorithm. It recursively uses the concept of information gain ratio to select the best features for the root node and then builds the whole tree [19]. A random forest algorithm is considered from ensemble learning family. It is made up of many decision trees and assigns a label to its input using the "voting" principle. To train the different decision trees, the bootstrap aggregating technique is applied to provide different training subsets to each decision trees with the goal of reducing the variance by having tree learning algorithm modified to select a random subset of features for each split [20]. Similarly, support vector machine (SVM) [21] is also utilized as a multi-class variant of binary classification. It is based on separability of decision boundaries and has ability to classify the data points successfully. Furthermore, an

Table 1 Train and test set distribution (NSL-KDD)

Class	Training set	Percentage (%)	Test set	Percentage (%)	Class-type
Normal	67,342	53.46	9710	43.07	Major
DoS	45,927	36.46	7457	33.08	Major
Probe	11,656	9.25	2421	10.74	Minor
R2L	995	0.79	2754	12.22	Minor
U2R	52	0.041	200	0.89	Minor

Table 2 Train and test set distribution (UNSW-NB15)

Class	Training set	Percentage (%)	Test set	Percentage (%)	Class-type
Analysis	2000	1.14	677	0.82	Minor
Backdoor	1746	0.99	583	0.71	Minor
DoS	12,264	6.99	4089	4.97	Minor
Exploits	33,393	19.04	11,132	13.52	Major
Fuzzers	18,184	10.37	6062	7.36	Minor
Generic	40,000	22.81	18,871	22.92	Major
Normal	56,000	31.94	37,000	44.94	Major
Reconnaissance	10,491	5.98	3496	4.25	Minor
Shell code	1133	0.65	378	0.46	Minor
Worms	130	0.07	44	0.05	Minor

artificial neural network is considered to evaluate the performance of the generated data. It is regarded as best choice for solving complex problems. The multi-layer feed-forward neural network [22] is considered to classify the anomaly classes.

4 Implementation details and results

This section describes the experimental details and obtained results of DGM. We implemented our approach using open-source python neural networks library Keras having Tensorflow 2.x as a back-end [23]. The experiments were carried out on an Intel core i5 CPU with 8 GB RAM. The Python library sklearn (version 0.21.2) is used for training and testing the model [24]. Furthermore, setting a benchmark performance for classifiers, we used default training parameters set by sklearn Python library. The details are as follow.

4.1 Dataset description

We performed experiments over two most common and publicly available datasets NSL-KDD [25] and UNSW-NB15 [26]. These datasets are considered as standard benchmarks to evaluate the anomaly detection models. For each dataset, we assign each class to either majority or minority based on the class samples percentage in the whole dataset. A class is assigned to minor category if its less than 10%; otherwise, it is assigned to majority.

4.1.1 NSL-KDD

This dataset is refined version of KDD-99, which do not contains any duplicate instances and considered as benchmark dataset to evaluate the anomaly detection models [25]. The dataset retains the characteristic of minor class presence that may affect the performance of the machine learning model. After data exploration, we found the problem of high skeweness and high correlation between features inherited from KDD-99. It contains twenty features and four types of anomalous attacks which are a summarized in Table 1. The detailed information about the dataset can be found in [27].

Table 1 presents the five classes, where first is normal data and denial of service (DoS), probe, remote to local (R2L), user to root (U2R) are considered as anomalies. In case of DoS, authentic users are unable to use the service over the network, probe tries to get the information about the target hose, R2L tries to gain the access on the local machine, U2R tries to get the rights of super user. Furthermore, Table 1 illustrates NSL-KDD train and test data distribution where it is also evident that it is imbalanced for multi-class. NSL-KDD data exhibit class overlap and have multi-minority characteristic. Class overlapping is one of the main reasons to make multiple classes representation

Table 3 The optimal hyperparameters of DGM

Parameter	NSL-G	NSL-D	UNSW-G	UNSW-D
Ephocs	2000	2000	2000	2000
Dropout	0.2	–	–	–
Optimizer	sgd∗	sgd∗	sgd∗	sgd∗
Learning rate	0.0005	0.0005	0.0005	0.0005
Layers	4	5	4	4
Input layer neurons	33	26	33	35
Layer 1 neurons	27	108	128	512
Layer 2 neurons	81	81	256	256
Layer 3 neurons	108	54	512	128
Layer 4 neurons	25	27	34	1
Layer 5 neurons	–	1	–	–
Output layer neurons	26	1	35	1
Batch size	128	128	128	128
Activation	relu∗∗	relu∗∗	spocu∗ ∗ ∗	spocu∗ ∗ ∗
Random noise	32	–	32	–

∗Stochastic gradient descent, ∗∗rectified linearunit, ∗∗∗scaled polynomial constant unit

complex in imbalance dataset. We presented this details in Appendix (i.e., Sect. 6).

4.1.2 UNSW-NB15

A recent dataset UNSW-NB15 was created by the cyber range laboratory of the Australian centre for cyber security [28]. They captured the data by using the tool tcpdump. It contains real network traffic of normal patterns and synthetic anomalies of network traffic. The detailed information about dataset can be found in [26]. Table 2 outlines the data distributions of the classes in the dataset.

In Table 2, more than 50% of classes belong to minor class presentation, which may affect the performance of the

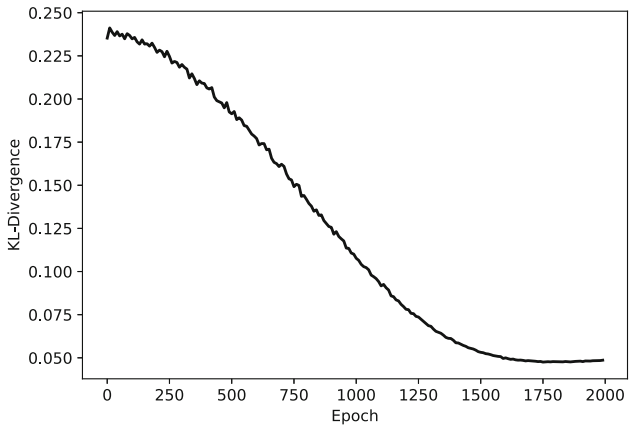

Fig. 3 KL-divergence during the training phase of NSL-KDD

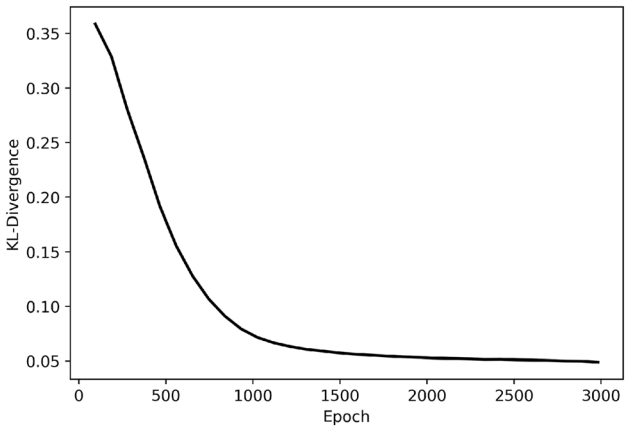

Fig. 4 KL-divergence during the training phase of UNSW-NB15

machine learning model. Description of the each class (newtwork intrusion type) an be found in [26]. In UNSW-NB15 dataset, classes overlap presented in appendix Fig. 5 and 6 is another main challenge faced by machine learning models in minority class detection. Despite this challenge, still our DGM works well to generate the data, which consequently improve the performance of the classifier.

4.2 Performance metric

The four standard performance metrics are used in this paper and are namely: precision, recall, F1-score, and weighted F1-score. They are calculated using values of a confusion matrix and computed as:

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

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

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

$$\text{Weighted } F1\text{−score} = \frac{\sum_{i=1}^{K} \text{Support}_i \cdot F1_i}{\text{Total}} \tag{14}$$

where TN is the number of true negative, TP is the number of true positives, FP is the number of false positives, FN is the number of false negatives, support is the number of samples with a given label i and $F1_i$ is F1 score of a given label i. Furthermore, we used K-fold cross-validation (i.e., $K = 5$) to make sure that the classifiers and proposed data generated model did not produce the results by chance.

4.3 Hyper-parameter details of model

The proposed architecture of neural networks was fixed in both NSL-KDD and UNSW-NB15 experiments. For an easy adaptation of our approach by any machine learning

Table 4 Classification performance on NSL-KDD dataset

Class	Classifier	Original data			DGM		
		P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
DoS	Random forest	90	84	86	94	86	90
	Decision tree	93	78	85	90	80	84
	Neural network	95	83	88	96	83	89
	SVM	96	84	89	93	90	91
Probe	Random forest	44	86	57	44	88	59
	Decision tree	42	80	56	47	87	61
	Neural network	43	88	58	43	86	57
	SVM	40	88	55	45	90	60
R2L	Random forest	77	26	38	69	27	39
	Decision tree	56	39	46	50	25	33
	Neural network	58	28	38	62	30	40
	SVM	32	09	14	88	26	40
U2R	Random forest	43	09	15	47	17	24
	Decision tree	19	23	21	20	22	21
	Neural network	65	24	36	41	23	29
	SVM	40	25	31	18	10	12

practitioner, we refrain from tuning hyper-parameters for machine learning classifiers which can be time-consuming and data specific. The hyper-parameters were tuned on the preference to yield the higher classification performance. The optimal hyper-parameters of the the model are described in Table 3.

In Table 3, stochastic gradient descent that is used to reduce the loss. The total number of training epochs was determined by convergence of Kullback–Leibler divergence of the real and synthetic data samples (i.e., 2000 epochs). Scaled polynomial constant unit (SPOCU) activation function is percolation-based and it is aimed at overcoming challenges faced by relu activation function [29]. Figrues 3 and 4 show the KL-divergence during the training stage.

After convergence of the network, the model has the ability to generate the minor class samples for sufficient presentation. The details about the experiments are given below.

4.4 NSL-KDD experiment

To train the model, we reduce the majority class by undersampling using SMOTE. In this case, only DoS class is undersampled to 25,925. To minimize information loss in random sampling, we used Pearsons chi-square test [30] and Kolmogorov-Smirnov test [31]. To measure how well a random sample represents the original training dataset, a 95% was set as confidence level both on Pearsons chi-square test and Kolmogorov-Smirnov test. We generate the data for three minor classes (i.e., Probe, R2L and U2R to 14,000, 13,995 and 10,062, respectively). To evaluate our generated data for minor classes, machine learning models are trained and performance is evaluated on original as well as data generated by DGM. Paying more attention to the minor classes as outlined in Table 1, we see that machine learning models trained on generated data from DGM have significantly better performance compared to those trained with original dataset. A detailed performance evaluation is presented in Table 4.

4.5 UNSW-NB15 experiment

To evaluate DGM on modern network traffic, we conducted experiments over UNSW-NB15. We generate significant samples for minor classes to balance its' presentation. In this experiment, Backdoor, DoS, Fuzzers, Reconnaissance, Shellcode and Worms minority classes were upsampled by 6000, 6000, 3000, 300, 2000, 5000, 2000 samples, respectively. The DGM proved to be improving performance of machine learning models, especially on the least classes which are the main concern in imbalanced datasets. Table 5 presents the class wise precision, recall and F1-score for original data and generated data from DGM. In Table 5, P, R and F stand for precision, recall and F1-score. Classes overlap in UNSW-NB15 is one of the main challenges faced by machine learning models in minority class detection in the dataset. UNSW-NB15 classes overlap is presented in appendix Figs. 5 and 6.

Table 5 Classification performance on UNSW-NB15

Class	Classifier	Original data			DGM		
		P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
Analysis	Random forest	0	0	0	0	0	0
	Decision tree	2	4	2	2	4	2
	Neural network	0	0	0	0	0	0
	SVM	0	0	0	0	0	0
Backdoor	Random forest	2	4	3	1	3	2
	Decision tree	3	13	5	5	2	8
	Neural network	3	25	1	35	0	1
	SVM	4	5	5	56	1	2
DoS	Random forest	41	5	8	30	47	37
	Decision tree	23	14	17	24	30	27
	Neural network	1	81	1	33	49	34
	SVM	32	58	41	32	64	42
Exploits	Random forest	59	80	68	77	60	67
	Decision tree	63	64	64	73	54	62
	Neural network	2	49	58	60	45	50
	SVM	89	3	6	67	24	35
Fuzzers	Random forest	63	73	68	62	75	68
	Decision tree	71	60	65	70	62	65
	Neural network	15	45	54	43	71	51
	SVM	31	58	41	36	74	48
Generic	Random forest	100	96	97	100	96	98
	Decision tree	99	97	98	99	96	98
	Neural network	0	100	98	99	96	98
	SVM	96	96	96	99	96	98
Reconnaissance	Random forest	84	62	70	81	64	72
	Decision tree	80	71	75	78	72	75
	Neural network	0	72	49	67	38	48
	SVM	69	38	49	68	37	48
Shell code	Random forest	55	29	38	51	33	40
	Decision tree	33	46	38	31	47	38
	Neural network	0	39	18	23	2	4
	SVM	7	24	11	38	10	16
Worms	Random forest	0	0	0	10	1	2
	Decision tree	16	20	15	17	21	17
	Neural network	1	0	0	0	0	0
	SVM	1	61	2	2	18	3

5 Comparison analysis

To validate DGM is effective, we implemented SMOTE [10] which is based on k-nearest neighbours algorithm, ADASYN [32] which is based on k-nearest neighbors and non-uniform weight assignment to minority data points, SMOTEENN [33] which is a modified version of SMOTE using edited nearest neighbours, Borderline-SMOTE [34] which is variant of the original SMOTE, whereby borderline samples are used to generate new synthetic samples. The comparison with the existing methods shows that purposed DGM outperforms. The results are presented in Tables 6 and 7 for both benchmark datasets two activtation functions (i.e., relu and spocu). It is evident that our approach in general contributes to minimize the problem of minor classes presentation that is faced by machine learning models. In case of NSL-KDD (i.e., Table 6), the model achieved 7% more accurate results as compared to the original dataset setting using relu activation function. The existing minority class data generation methods improved the results, while our model outperforms. The DGM has enough capabilities to generate the minority class samples

Table 6 A comparison of weighted F1-scores for NSL-KDD

Model	SVM	Decision tree	Random forest	Neural networks
Original	66	70	69	71
SMOTE	67	69	71	69
ADASYN	67	69	70	67
SMOTEENN	67	68	71	71
Borderline-SMOTE	69	67	69	68
DGM-SPOCU	71	66	69	70
DGM-RELU	**73**	68	**72**	**71**

Bold present the high result obtained durig the comparision

Table 7 A comparison of weighted F1-scores for UNSW-NB15

Model	SVM	Decision tree	Random forest	Neural networks
Original	55	73	73	66
SMOTE	57	72	74	59
ADASYN	47	71	72	57
SMOTEENN	54	62	61	56
Borderline-SMOTE	50	71	71	57
DGM-SPOCU	**63**	**73**	**75**	**66**
DGM-RELU	57	73	75	67

Bold present the high result obtained durig the comparision

by explicit learning the distribution using KL-divergence mechanism. The obtained results in Table 6 depicts that artificial neural network-based model outperforms without handling the minority class, while existing methods reduced the performance in case of decision tree, random forest and neural network. In contrast, DGM provides better results and consistent for all multi-class anomaly classification except decision tree. We examined the root cause of no performance improvement which comes from the original data itself and challenge induced by class overlapping. Furthermore, in certain scenarios of anomaly detection domain, few limitations are posed on machine learning model to be lightweight in terms of memory as well as interpretation. In such scenarios, the DGM can generate minority class data at the pre-processing step of the machine learning pipeline that can help the simple machine learning models to perform better as compared to the existing methods for data generation. For instance, in case of SVM our model performs significantly better as compared to other data generation techniques. The quality of data generation is observed in case of ADASYN, SMOTEENN, and Borderline-SMOTE for benchmark datasets (i.e., UNSW-NB15 and NSL-KDD dataset). The synthetics data reduced the performance of the machine learning models, while DGM consistently improves the results. Our reported results are based on k-fold cross-validation ($k = 5$) experimental setting that confirms the obtained performance is not by chance. In both benchmark datasets, the performance of machine learning model is better over the minority class generation of DGM model.

6 Conclusion

In anomaly detection domain, available dataset is small and most of the time imbalanced due to the presence of minority classes. It effects the performance of machine learning models. In this research, we proposed a data generative model to generate minor classes data in anomaly detection domain. Our generated model can conditioned on the minority class to sample the data. Consequently, it improved machine learning classifiers performances. We performed experiments on two publicly available benchmark datasets (i.e., NSL-KDD and UNSW-NB15). We compared the DGM with existing statistical approach SMOTE and reported that DGM performed better. In our future work, we plan to investigate the quality of data generated by generative models with respect to the quantity of real data.

At the moment, our approach is focused on cyber-security domain and its extension to other domains required re-training of the model. In our future work, the developed DGM will be applied to healthcare domains to detect the anomalies.

Appendix

To better understand the minor classes distribution with respect to each other, we also reported the visualization of the data using t-distributed stochastic neighbor embedding

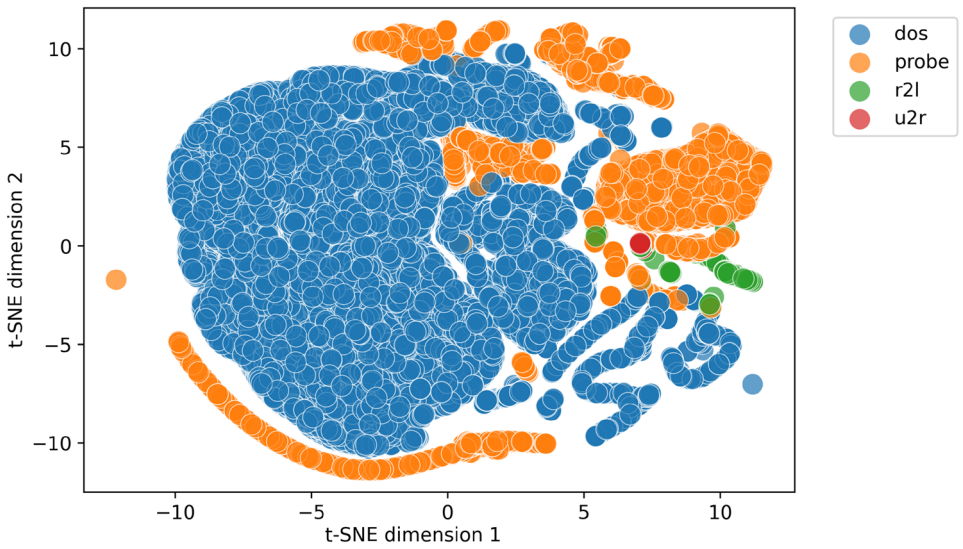

Fig. 5 NSL-KDD all attacks t-SNE visualisation

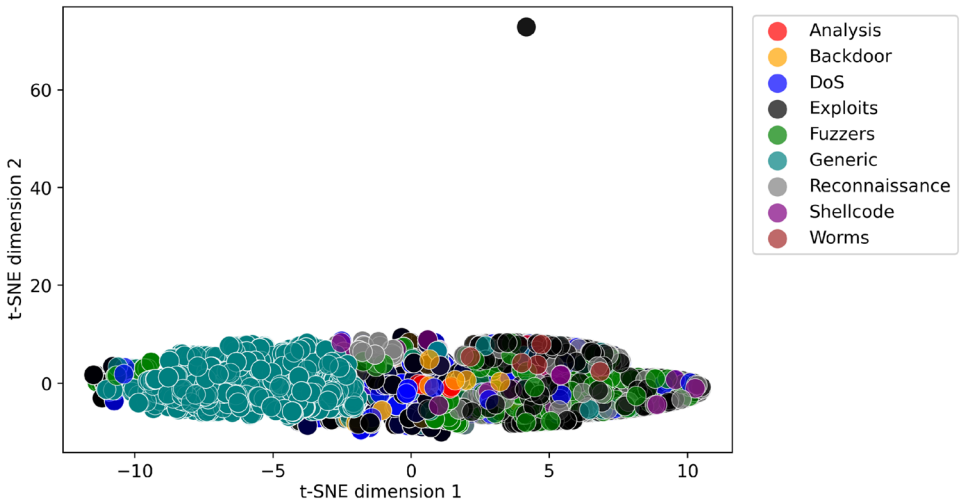

Fig. 6 UNSW-NB15 all attacks t-SNE visualisation

(t-SNE) [35]. The visualization for each class can be seen on Figs. 5 and 6.

In Figs. 5 and 6, it presents the 2D representation of the classes. In case of NSL-KDD dataset, the classes are overlapped with minor region. While in case of UNSW-NB15, the visualization depicts that classes are overlapped as the number of classes increase. It shows a complex task to learn the data distribution for each minority class. In this scenario, our model DGM performs well to generate the synthetic data as compared to the existing methods.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Abdi L, Hashemi S (2015) To combat multi-class imbalanced problems by means of over-sampling techniques. IEEE Trans Knowl Data Eng 28(1):238–251
2. Ba H (2019) Improving detection of credit card fraudulent transactions using generative adversarial networks. arXiv:1907.03355
3. Wang S, Yao X (2012) Multiclass imbalance problems: analysis and potential solutions. IEEE Trans Syst Man Cybern Part B (Cybern) 42(4):1119–1130
4. Xiaolong XU, Wen CHEN, Yanfei SUN (2019) Over-sampling algorithm for imbalanced data classification. J Syst Eng Electr 30(6):1182–1191
5. He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
6. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357

7. Zhu T, Lin Y, Liu Y (2017) Synthetic minority oversampling technique for multiclass imbalance problems. Pattern Recogn 72:327–340
8. Mirza M, Osindero S (2014) Conditional generative adversarial nets. arXiv:1411.1784
9. Hasanin T, Khoshgoftaar T (2018) The effects of random undersampling with simulated class imbalance for big data. In: 2018 IEEE international conference on information reuse and integration (IRI). IEEE, pp 70–79
10. 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). IEEE, pp 1–8
11. Choi H, Kim M, Lee G, Kim W (2019) Unsupervised learning approach for network intrusion detection system using autoencoders. J Supercomput 75(9):1–25
12. Douzas G, Bacao F (2018) Effective data generation for imbalanced learning using conditional generative adversarial networks. Expert Syst Appl 91:464–471
13. Lin Z, Shi Y, Xue Z (2018) Idsgan: generative adversarial networks for attack generation against intrusion detection. arXiv:1809.02077
14. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
15. Lewis-Beck M, Bryman AE, Liao TF (2003) The Sage encyclopedia of social science research methods. Sage Publications, London
16. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems. pp 2672–2680
17. Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat 22(1):79–86
18. He H, Ma Y (2013) Imbalanced learning: foundations, algorithms, and applications. Wiley, New York
19. Quinlan JR (2014) C4. 5: programs for machine learning. Elsevier, New York
20. Ho TK (1995) Random decision forests. In: Proceedings of 3rd international conference on document analysis and recognition, vol 1. IEEE, pp 278–282
21. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
22. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction. Springer, Berlin
23. Chollet F et al. (2018) Keras: the python deep learning library. Astrophysics Source Code Library
24. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
25. Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Accessed 9 Apr 2020
26. 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). IEEE, pp 1–6
27. Nsl-kdd dataset description. https://www.unb.ca/cic/datasets/nsl.html. Accessed 9 Apr 2020
28. Unsw-nb15 dataset description. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. Accessed 9 Apr 2020
29. Kisel'ák J, Lu Y, Švihra J, Szépe P, Stehlík M (2020) "spocu": scaled polynomial constant unit activation function. Neural Comput Appl 33(8):1–17
30. McHugh ML (2013) The chi-square test of independence. Biochemia medica: Biochemia medica 23(2):143–149
31. Conover William J, (1980) Practical nonparametric statistics
32. He H, Bai Y, Garcia EA, Li S (2008) Adasyn: adaptive synthetic sampling approach for imbalanced learning. In: IEEE international joint conference on neural networks (IEEE world congress on computational intelligence). IEEE, pp 1322–1328
33. Batista GEAPA, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29
34. Han H, Wang W-Y, Mao B-H (2005) Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: International conference on intelligent computing. Springer, pp 878–887
35. van der Maaten L, Hinton G (2008) Visualizing data using t-sne. J Mach Learn Res 9:2579–2605