PacketCGAN: Exploratory Study of Class Imbalance for Encrypted Traffic Classification Using CGAN

Pan Wang*, Shuhang Li*, Feng Ye†, Zixuan Wang* and Moxuan Zhang‡
*School of Modern Posts, Nanjing University of Posts & Telecommunications, Nanjing, China
†Department of Electrical & Computer Engineering, University of Dayton, Dayton, OH, USA
‡Schools of International Education, Jinling Institute of Technology, Nanjing, China
Email: *wangpan@njupt.edu.cn

***Abstract*—With the popularity of Deep Learning (DL), researchers have begun to apply DL to tackle with encrypted traffic classification problems. Although these methods can automatically extract traffic features to improve the ability of feature engineering of traditional methods like DPI, a large amount of data is still needed to learn the characteristics of various types of traffic. Therefore, the performance of classification model always significantly depends on the quality of datasets. Nonetheless, the building of datasets is a time-consuming and costly task, especially encrypted traffic. Apparently, it is often more difficult to collect a large amount of traffic samples of those unpopular applications than well-known ones, which often leads to the problem of class imbalance between major and minor encrypted applications in datasets. In this paper, we proposed a novel traffic data augmenting method called PacketCGAN using Conditional GAN, which can control the modes of data to be generated. PacketCGAN exploit the benefit of CGAN to generate specified samples with the input of applications' types as conditional and thereby achieve data balancing . As a proof of concept, three classical DL models including CNN were adopted to classify four types of encrypted traffic datasets augmented by Random Over Sampling (ROS), SMOTE(Synthetic Minority Over-sampling Techinique) , vanilla GAN and PacketCGAN respectively using two public datasets: ISCX2012 and USTC-TFC2016. The experimental evaluation results demonstrate that DL based encrypted traffic classifier over our new dataset augmented by PacketCGAN can achieve better performance than the other three in terms of encrypted traffic classification.**

***Index Terms*—encrypted traffic classification, data augmentation, Conditional Generative Adversarial Network, traffic identification, class imbalance**

I. Introduction

With the rapid development of network technology, the types and quantity of traffic data in cyberspace are increasing. Network traffic identification and classification is a crucial research task in the area of network management and security. It is the footstone of dynamic access control, network resources scheduling, content based billing, intrusion and malware detection etc. High efficient and accurate traffic classification is of great practical significance to provide service quality assurance, dynamic access control and abnormal network behaviors detection. With the widespread adoption of encryption techniques for internet, especially 5G and IoT applications, the growth of portion of encrypted traffic has dramatically posed a huge challenge for QoS, network management and security monitoring. Therefore, studies on encrypted traffic classification not only help to improve the fine-grained network resource allocation based on application, but also enhance security level of network and applications.

Traditionally, the evolution of encrypted traffic classification technology has gone through three stages: port matching, payload matching and flow statistical characteristics based classification methods. Port matching based method infers applications' types by assuming that most applications consistently use 'well known' TCP or UDP port numbers, however, the emergence of port camouflage, dynamic port, proprietary protocols with user-defined ports and tunneling technology makes these methods lose efficacy quickly. Payload matching based methods, namely, DPI (Deep Packet Inspection) technology cannot deal with encrypted traffic because of invisible packet content of encrypted traffic, in addition, it incurs high computational overhead and requires manual signatures maintenance [1–3]. As a result, in order to attempt to solve the aboved problems of encrypted traffic identification , flow-based methods emerged, which usually combine statistical or time series traffic features with Machine Learning (ML) algorithms, such as naive bayes(NB), support vector machine(SVM), decision tree, Random Forest(RF), k-nearest neighbor(KNN) [4–7]. Although classical machine learning approaches can solve many issues that port and payload based methods cannot solve, it still has some limitations, such as handcrafted traffic features driven by domain-expert, time-consuming, lack of ability of automation, rapidly outdated when compared to the evolution. Unlike most traditional ML algorithms, DL performs automatic feature extraction without human intervention, which undoubtedly makes it a highly desirable approach for traffic classification, especially encrypted traffic. Recent research work has demonstrated the superiority of DL methods in traffic classification [8], such as MLP [9], CNN [10–14], SAE [15], LSTM [16, 17].

However, due to the different popularity of various applications, the class imbalance problem of traffic samples often occurs when building traffic datasets. That is, the number of popular applications samples is much larger than others,

which always leads to the misclassifying problems of minor applications and thereby incurs deterioration of classifier performance. Imbalanced class distribution of a dataset has posed a serious challenge to most ML based classifiers which assume a relatively balanced distribution [18]. Network traffic classification is no exception due to the imbalanced property of network traffic data [19, 20], especially encrypted traffic. Therefore, it is very crucial to address such challenges of imbalanced class distribution of traffic datasets for network traffic classification. However, there are very few studies focusing on traffic data augmentation for traffic classification to overcome the limitation of class imbalance.

In this paper, we proposed traffic data augmentation method called PacketCGAN using Conditional GAN, one of a genre of GAN, which can control the modes of the data to be generated. As a generative model, PacketCGAN exploits the benefit of CGAN to generate synthesized traffic samples by learning the characteristics of the original traffic data. The synthesized data is then combined with the original (viz. real) data to build the new traffic dataset and thereby keep balance between major and minor classes of the dataset. As a proof of concept, three classical DL models like CNN were adopted and designed to classify four types of encrypted traffic datasets augmented by ROS, SMOTE, vanilla GAN and PacketCGAN respectively using two public datasets: ISCX2012 and USTC-TFC2016. The experimental evaluation results demonstrate that DL based encrypted traffic classifier over our new dataset augmented by PacketCGAN can achieve better performance than the other three in terms of encrypted traffic classification.

The rest of this paper is organized as follows. Section II introduces related works of traffic classification and some current methods for tackling with the problem of imbalanced class data. Section III describes the principles of GAN and CGAN. Section IV illustrates the methodology of PacketCGAN. The experimental results are provided and discussed in Section V. Section VI concludes our work and presents some future works.

II. Related works

A. ML and DL based approach of Traffic Classification

Different from port and payload matching methods, ML based classification methods always use payload-independant parameters such as packet length, inter-arrival time and flow duration to circumvent the problems of encrypted content and user's privacy [22]. Many work was carried out using ML algorithms during the last decades. In general, there are two learning strategies used: one is the supervised methods like decision tree, SVM and Naive Bayes, the other is unsupervised approaches like k-means and PCA [23]. Nevertheless, many drawbacks hindered ML based methods widely applied to traffic classification, such as handcrafted traffic features driven by domain-expert, time-consuming, unsuited to automation, rapidly outdated when compared to the evolution. Unlike most traditional ML algorithms, Deep Learning performs automatic feature extraction without human intervention, which undoubtedly makes it a highly desirable approach for traffic classification, especially encrypted traffic. Recent research work has demonstrated the superiority of DL methods in traffic classification [8–17, 24]. The workflow of DL based classfication usually consists of three steps. First, model inputs are defined and designed according to some principles, such as raw packets, PCAP files or flow statistics features. Second, models and algorithms are elaborately chosen according to models' characteristics and aim of the classifier. Finally, the DL classifier is trained to automatically extract the features of traffic.

B. Traditional methods for handling imbalanced data

In general, there are three methods for dealing with class imbalance problem: ***Modifying the objective cost function, Sampling and Generating artificial data*** [20]. The approach of ***modifying objective cost function*** alleviates the problem by means of weighting differently the data samples in minor and major classes, which gives higher score on the minor samples to penalize more intensely on miss-classifiying of the sample in the minor class. Sampling methods include two different ways of ***under-sampling*** and ***over-sampling***, which is to reduce the size of major class by removing some major data samples and raise the ones in the minor class, respectively. Random under sampling (RUS) and Random over sampling (ROS) are two main methods of under-sampling and over-sampling [25]. RUS randomly removes some instances in major class, accordingly, ROS generates some copies of samples of the minor class. However, overfitting problem is always the main drawback of ROS due to generating same copies from the minor class. A classical method for generating artificial data is Synthetic Minority Over-sampling Techinique (SMOTE) in which minority samples are generated by synthetic samples rather than copies [26].

C. The application of GAN and other DL techniques in generating traffic data samples

Due to the great success of GAN applying in images, computer vision and NLP etc., this innovative technique has been already applied to network security recently. A few current studies have shown that GAN has been applied in IDS and Malware detection to generate adversarial attacks to deceive and evade the detection systems [27, 28] and thus effectively improve the performance of malware detection or IDS [27–31]. Correspondingly, as for traffic classification, some researchers have introduced some approaches based on GAN to generate the traffic samples to overcome the imbalaced limitation of network data. In [32], the authors proposed a novel method called auxiliary classifier GAN (AC-GAN) to generate synthesized traffic samples for balancing between the minor and major classes over a well-known traffic dataset NIMS. The AC-GAN took both a random noise and a class label as input in order to generate the samples of the input class label accordingly. The experimental results has shown that their proposed method achieved better performance compared to other methods like SMOTE. However, the NIMS dataset was only composed of SSH and non-SSH two classes.

In [33], the authors proposed a novel data augmentation approach based on the use of Long Short Term Memory (LSTM) network to learn the traffic flow patterns and Kernel Density Estimation (KDE) for replicating the sequence of packets in a flow for classes with less population. The results have shown that this method can improve the performance of DL algorithms over augmented datasets.

III. Generative Adversarial Networks

A. GAN

As an unsupervised learning model, a classic GAN network consists of two parts, the generator G and the discriminator D. The role of the generator is to take random noise as input by learning the characteristic distribution of real data. The discriminator aims at determining whether the data is real or generated by G. The generator G simulates the feature distribution P_g of the real data by the prior distribution $P_z(z)$. The input of the discriminator is the real and generated data, correspondingly, the output $D(x)$ indicates the probability of whether the input data is real or not [34]. During the training process, G and D play a two-player mini-max game until D can't judge whether the sample data is real, which means that the two networks reach the Nash Equilibrium. The objective function of GAN can be expressed by (1):

$$\begin{aligned}\min_{\mathrm{G}}\max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] \\ +\mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]\end{aligned} \tag{1}$$

In Equation (1), $P_{data}(x)$ represents distribution of the real data. When training D, the goal is to optimize the probability of TRUE $D(G(z))$ as small as possible and the probability of TRUE $D(x)$ of the real data x as much as possible. When training G, the goal is to make $D(G(z))$ as much as possible. From (1), we can calculate the optimal discriminator as (2). As can be seen from (2) below, when $P_{data}(x) = P_z(z)$, it means that D cannot distinguish whether the sample is true or false, D and G reach the Nash Equilibrium, and the discriminator output is 0.5.

$$D(x) = \frac{P_{data}(x)}{P_{data}(x) + P_z(z)} \tag{2}$$

B. CGAN

In GAN, there is no control over modes of the data to be generated. The conditional GAN changes that by adding the constraint condition y as an additional parameter to the generator G and hopes that the corresponding images are generated. For example, in MNIST, the digit generated by GAN may be either any digit from 0-9 instead of specified one or always output the same digit. Fig. 1 shows the network structure of CGAN.

The principle, structure and training process of CGAN are similar to GAN. The cost function is slightly different as is shown in (3):

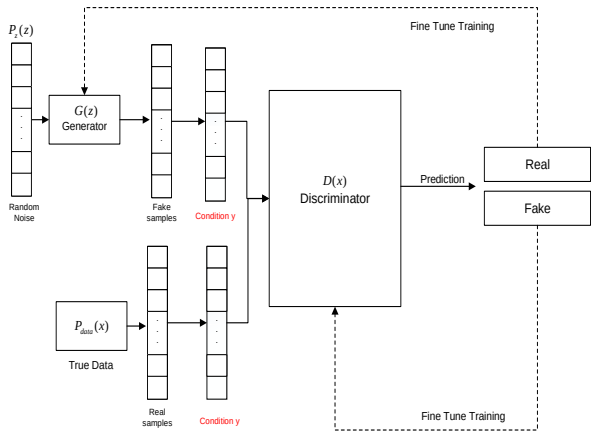


Fig. 1: The network structure of CGAN.

$$\begin{aligned}\min_{\mathrm{G}}\max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)] \\ +\mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z|y)))]\end{aligned} \tag{3}$$

As shown in Fig. 1, CGAN's training process includes the following steps:

- Sampling the real data to obtain $P_{data}(x)$, obtaining the label y corresponding to the sampling data $P_{data}(x)$, feed $P_{data}(x)$ and y into the discriminator D, then updating the parameters according to the output results;
- Generating random noise $P_z(z)$, which is then fed into generator G together with label y in the above step, and G generates synthesized data.
- Feeding the synthesized data and the label y generated in the above step into the discriminator D, and G will optimize the parameters according to the output result of D.
- Repeat the above steps until G and D reach the Nash equilibrium.

IV. The Methodology of PacketCGAN

A. The Workflow of PacketCGAN Based Encrypted Traffic Classification

The workflow of PacketCGAN based encrypted traffic classification is shown in Fig. 2, in which there are three phases: packet data preprocessing, class balancing using PacketCGAN and traffic classifier training.

1) Phase 1: Generally speaking, packet data preprocessing is the fundermental task of DL modelling. As we all know, the captured network traffic data is often saved in PCAP or PCAPNG [35] format, in which packet bytes of traffic data are saved in hexadecimal. The bytes converted to decimal range from 0 to 255, which resemble the pixel range of the single-channel grayscale image. Apparently, traffic classification can refer to DL methods of image recognition. However, packets from PCAP files can not directly be used for model training and need to be pre-processed. Two public datasets used in this paper are ISCX2012 [36] and USTC-TFC2016 [12] as shown

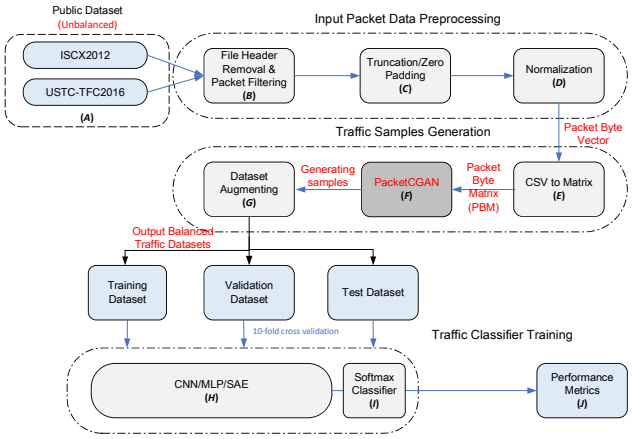


Fig. 2: The workflow of PacketCGAN.

in (A) of Fig. 2, which are both composed of PCAP files. We will describe the two datasets in Section. V-A1.

Packet data preprocessing consists of three steps as follows:

- File header removal and packet filtering as shown in (B) of Fig. 2. The first 24 bytes of the PCAP file header need to be removed which only contains file information and does not help with traffic classification. Filtering useless packets in PCAP file for classification is also needed, such as APR and DHCP packets etc., which is irrelevant to specific traffic types.
- Truncation or zero padding as shown in (C) of Fig. 2. During the training of DL model, the input of which needs to be converted into a vector or matrix, which requires the length of the data input to be fixed. Nevertheless, the length of captured packet in dataset tends to be different, so it is necessary to truncate the long packets or padding those short ones based on a fixed and predefined length value [37], which is 1480 in this paper.
- Data normalization as shown in (D) of Fig. 2. Normalizing the pre-processed traffic data to the range [0,1] can improve the accuracy and enhance convergence speed of the model training.

After all above mentioned operations, each processed packet will be formated to a Packet Byte Vector (PBV) [9] which can be stored in csv format.

2) Phase 2: Data balancing is the most critical problem in this phase. It is essential to read the vectors of PBV from csv files to build the matrix of PBM as the input of PacketCGAN as shown in (E) of Fig. 2, which will be illustrated in detail in Section. IV-B. Synthesized packet samples will be generated by PacketCGAN in PBM format and then combined with the samples of original dataset to build the new balanced traffic dataset as shown in (E) and (F) of Fig. 2.

3) Phase 3: Traffic Classifier Training: In this paper, we adopted five DL traffic classifiers to evaluate the performance of datasets augmented by PacketCGAN, which are MLP/CNN/SAE from [9] and 1D-CNN/2D-CNN from [11, 12], respectively. The new balanced traffic datasets will always be devided into three parts: training, testing and validation datasets. Finally, classifier related performance metrics will be evaluated to indirectly verify the quality of datasets generated by different data augmentation methods as shown in (H,I,J) of Fig. 2.

B. The Model Architecture and Algorithm Description of PacketCGAN

1) The Description of Model Architecture: The model of our proposed PacketCGAN is shown in Fig. 3, in which both G and D adopted MLP as the basic architecture. Random noise $P_z(z)$ as the input of model with Normal Distribution are N*100-dimentional vectors, which will be fed into G with $G(z)$ distribution. Different from vanilla GAN, the label of traffic/applications types with one-hot encoding will also be fed into our PacketCGAN as conditional c. The synthesized samples as the output of G are N*1480 vectors, which will be compose of the matrix of PBM and fed into D with $D(x)$ distribution eventually. Meanwhile, PBM from original unbalanced datasets with $P_{data}(x)$ distribution will be fed into D with the same 1480-dimentional to compete to reach Nash equilibrium with fine-tune training.

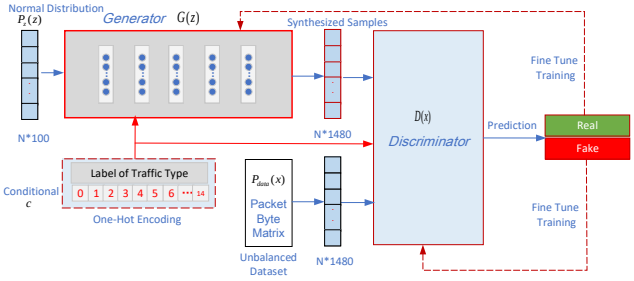


Fig. 3: The Architecture of PacketCGAN.

2) The Description of Model Algorithm: In general, PacketCGAN's training mainly includes the following three steps: the training of discriminator D, generator G and fine-tune optimization. During the training of D in step 1, one needs to fix the parameters of G while training D. We adopt the simple MLP to build the D model, in which there are three layers: the input layer, a hidden layer and the output layer. The data for the input layer is derived from the N*1480 matrix of PBM aforementioned in Section IV-A1 with the distribution of $P_{data}(x)$. The hidden layer consists of 128 neurons with the notation of D_{h1}.

$$D_{h1} = ReLU(input_D \cdot W_1^D + b_1^D) \tag{4}$$

The $input_D$ in (4) is composed of the real data x with $P_{data}(x)$ and the traffic/applications types label y in one-hot encoding, such as 0 for SSH, 1 for YouTube. The hidden layer D_{h1} adopts $ReLU$ as the activation function. The output layer has only one neuron referring to real or fake sample as following Equation: 5:

$$D_{out} = D_{h1} \cdot W_2^D + b_2^D \tag{5}$$

The model uses the Adam optimization algorithm as the optimizer to update weights W_1^D, W_2^D and bias b_1^D, b_2^D. The overall training process of the discriminator is shown in Algorithm. 1, in which $X_\tau = \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mk} \end{pmatrix}$,$y_\tau = \begin{pmatrix} y_{11} \\ \vdots \\ y_{m1} \end{pmatrix}$,k is 1480, which is the data dimension. m is mini_batch referring to a group of packets from dataset.

Algorithm 1 The training of Discriminator of PacketCGAN

Input: real data X_τ and label y_τ from Section IV-B2, τ is the number of iterations
Output: the probability of the output data being real or false

Initializing the relevant parameters, e represents the training cycle: epoches.
for τ in e **do**
  for each batch of m input data **do**
    concatenate X_τ with y_τ:
    Compute the output using Equation. (4);
    Compute the output using Equation. (5);
    Output discriminating results according to Equation. (5);
    Optimize the loss function
    Update weights and bias;
  end for
end for

Step 2 is to train the generator G. The parameters of D need to be fixed like step 1. The input layer of G consists of 115 neurons, which includes stochastic noise $P_z(z)$ with 100 neurons and sample label y with 15 neurons of applications types. The hidden layer also has 128 neurons to update the weight and bias of the input data:

$$G_{h1} = ReLU(input_G \cdot W_1^G + b_1^G) \tag{6}$$

The 'input' in (7) is made up of random noise $P_z(z)$ and the sample label y with $ReLu$ as the activation function. The output layer contains 1480 neurons with the same dimension of the $P_{data}(x)$. The computed result will be activated by the sigmoid function for discrimination of real or fake.

$$G\_out = sigmoid(G_{h1} \cdot W_2^D + b_2^D) \tag{7}$$

The optimization function is the same as the discriminator. The generator training algorithm is shown in Algorithm. 2. The sample label y in Algorithm. 2 should be consistent with y in Algorithm. 1. $Z_\tau = \begin{pmatrix} z_{11} & \cdots & z_{1n} \\ \vdots & \ddots & \vdots \\ z_{m1} & \cdots & z_{mn} \end{pmatrix}$,$n = 100$.

Algorithm 2 The training of Generator of PacketCGAN

Input: random noise Z_τ, and label y_τ, τ is the number of iterations
Output: synthesized data g_τ

Set the relevant initialization parameters, e represents the training cycle: epoches.
for τ in e **do**
  for each batch of m input data **do**
    concatenant Z_τ with y_τ:
    Compute the output using Equation. (6);
    Compute the output using Equation. (7);
    Output generation data according to Equation. (7);
    Optimal the loss function
    Update weights and bias;
  end for
end for

V. Evaluation and Experimental results

A. Experimental Settings

1) Dataset for Evaluation: Two public datasets used in this paper are ISCX2012 [36] and USTC-TFC2016 [12], which are both composed of PCAP files. The full name of 'ISCX2012' dataset is 'ISCX VPNnonVPN traffic dataset', which contains many encryption applications or protocols such as HTTPS, SFTP, Facebook, Hangouts, etc. In addition to these applications encapsulated in regular session, some VPN tunnel's applications were also captured in this dataset. 15 applications were chosen to build the original dataset as shown in Table I. Apparently, the original chosen dataset happens to the problem of class imbalance, in which the majority class such as Netflix accounting for 25.13%, while the minor class ICQ account for only 2.05%. 10 applications were chosen from USTC-TFC2016 as shown in Table II. Balanced dataset made by different generative methods are all 10000 samples per applications.

TABLE I: Description of the chosen datasets from ISCX.

Application	Security Protocol	Unbalanced dataset		Balanced dataset	
		Quantity	Percentage	Quantity	Percentage
AIM	HTTPS	4869	2.356%	10000	6.67%
Email-Client	SSL	4417	2.137%	10000	6.67%
Facebook	HTTPS	5527	2.674%	10000	6.67%
Gmail	HTTPS	7329	3.546%	10000	6.67%
Hangout	HTTPS	7587	3.671%	10000	6.67%
ICQ	HTTPS	4243	2.053%	10000	6.67%
Netflix	HTTPS	51932	25.126%	10000	6.67%
SCP	SSH	15390	7.446%	10000	6.67%
SFTP	SSH	4729	2.287%	10000	6.67%
Skype	proprietary	4607	2.229%	10000	6.67%
Spotify	proprietary	14442	6.987%	10000	6.67%
torTwitter	proprietary	14654	7.089%	10000	6.67%
Vimeo	HTTPS	18755	9.074%	10000	6.67%
voipbuster	proprietary	35469	17.161%	10000	6.67%
Youtube	HTTPS	12738	6.163%	10000	6.67%
TOTAL		**206688**	**100%**	**150000**	**100%**

2) Configurations of the Computing Platform: The experimental environmental parameters of this paper are shown in Table III. The performance evaluations are conducted using a Dell R730 server with an Intel I7-7600U CPU 2.8 GHz, 16 GB RAM and an external GPU (Nvidia GeForce GTX

TABLE II: Description of the chosen datasets from USTC-TFC2016.

Application	Security Protocol	Unbalanced dataset		Balanced dataset	
		Quantity	Percentage	Quantity	Percentage
BitTorrent	SSL	7535	8.10%	10000	10.00%
Facetime	SSL	2990	3.22%	10000	10.00%
FTP	FTP	11506	12.37%	10000	10.00%
Gmail	HTTPS	11477	12.34%	10000	10.00%
MySQL	MySQL	11385	12.24%	10000	10.00%
Outlook	SSL	7467	8.03%	10000	10.00%
Skype	proprietary	6028	6.48%	10000	10.00%
SMB	SMB	11543	12.41%	10000	10.00%
Weibo	HTTPS	11510	12.38%	10000	10.00%
WorldOfWarcraft	SSL	11559	12.43%	10000	10.00%
TOTAL		**93000**	**100%**	**100000**	**100%**

1050TI). The software platform for deep learning is built on Keras library with Tensorflow (GPU-based version 1.13.1) as the back-end support.

3) Description of deep learning based network traffic classifier: To verify the feasibility and evaluate the performance of the PacketCGAN algorithm, we adopted three classical DL models to classify the traffic over the datasets synthesized by different generative methods, which is MLP, CNN and SAE, respectively. The detail of these three models can be find in our previous works [9].

TABLE III: Experimental Environment Parameters

Category	Parameters
GPU	Nvidia GPU(GeForce GTX 1050Ti)
Operating System	Win 10
Deep learning platform	TensorFlow 1.13.1 + Keras 1.0.7
CUDA Version	9.0
CuDNN Version	7.6.0

4) Performance Metrics for Classification: The performance metrics used for evaluations of network traffic classifiers are *Precision*, *Recall* and F_1 *score*.

- ***Precision***: precision r_p is the ratio of *true positives* n_T^P over the sum of n_T^P and *false positives* n_F^P. In the proposed classification methods, precision is the percentage of packets that are properly attributed to the targeted application.

$$r_p = \frac{n_T^P}{n_T^P + n_F^P}. \tag{8}$$

- ***Recall***: recall r_c is the ratio of n_T^P over the sum of n_T^P and *false negatives* n_F^N or the percentage of packets in an application class that are correctly identified.

$$r_c = \frac{n_T^P}{n_T^P + n_F^N}. \tag{9}$$

- F_1***-score***: the F_1 score r_f is a widely-used metric in information retrieval and classification that considers both precision and recall as follows:

$$r_f = \frac{2r_p \cdot r_c}{r_p + r_c}. \tag{10}$$

B. Data Augmenting using PacketCGAN

1) Data Augmenting Methods handling the problem of class imbalance: In this paper, four methods were used to address the problem of the unbalanced dataset for the purpose of comparism. The first one is random oversampling method (ROS) mentioned in Section II-B . The essence of this method is to randomly copy some samples of minor class to supplement the dataset. Since the simple method is only a copy of the original data, it will lead to some wrong features be learned by the model easily and even over-fitting. The second method is SMOTE in which minority samples are generated by synthetic samples rather than copies. The third one is vanilla GAN with a generator and discriminator based on MLP. The last method is to generate minor sample data using our PacketCGAN generator. The dataset balanced by different methods is described in V-A1.

2) The Architecture of PacketCGAN: In our experiments, a fully connected MLP network was adopted to design generators and discriminators, as shown in Table IV and V. The generator's input is a 100-dimensional vector generated by random Gaussian noise with additional 15 applications types for ISCX2012 or 10 for USTC-TFC2016 datasets. The next three hidden layers have 128, 256, 512 neurons, and 1480 output neurons with additional 15 for ISCX2012 or 10 for USTC-TFC2016 datasets. Therefore, input of discriminator network is 1495 or 1490 dimension vectors mixed by real traffic data or generated data. Three hidden layers and output layers are both activated by LeakyReLU.

TABLE IV: The Architecture of Generator Network of PacketCGAN

Layer(type)	Output Shape (ISCX)	Output Shape (USTC)
dense_1(Dense)	115	110
LeakyReLU	115	110
dense_2(Dense)	256	256
LeakyReLU	256	256
dense_3(Dense)	512	512
LeakyReLU	512	512
dense_4(Dense)	1495	1490
LeakyReLU	1495	1490

3) The Training of PacketCGAN: Table VI shows the parameters of the optimizer, loss function, epoches, and mini_batch used in the deep training model training process.

Figure 4 shows the trend of loss of generator and discriminator during the PacketCGAN training process. It can be seen that PacketCGAN does not improve the unstability characteristics of GAN. The loss of G and D always fluctuates within a range instead of convergence. Nevertheless, it is a big

TABLE V: The Architecture of Discriminator Network of PacketCGAN

Layer(type)	Output Shape (ISCX)	Output Shape (USTC)
dense_6(Dense)	1495	1490
LeakyReLU	1495	1490
Dropout	1495	1490
dense_7(Dense)	512	512
LeakyReLU	512	512
Dropout	512	512
dense_8(Dense)	256	256
LeakyReLU	256	256
Dropout	256	256
dense_9(Dense)	128	128
LeakyReLU	128	128
Dropout	128	128
dense_10(Dense)	1	1

TABLE VI: Training Parameters of Packet CGAN

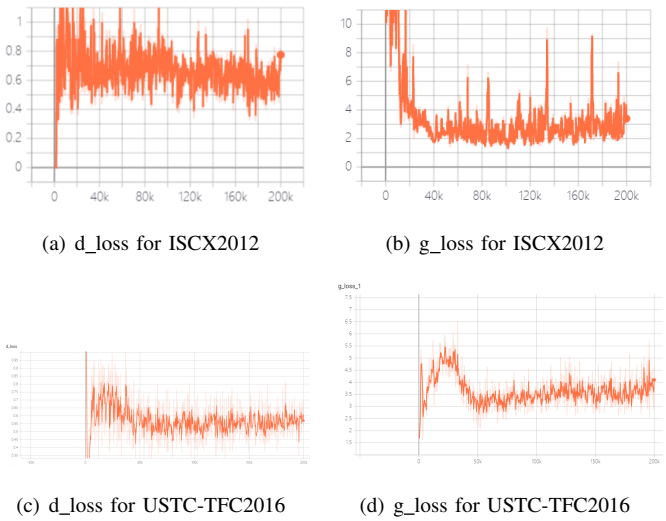
Training Parameters	Generator	Discriminator
Optimizer	Adma	Adma
Loss Function	Cross-Entropy	Cross-Entropy
Epoches	200000	200000
Mini_batch	64	64

improvement that only one CGAN model needs to be trained when using PacketCGAN compared with vanilla GAN. One can generate any samples with the additional input of the applications types as conditional c, while vanilla GAN can only generate one type of samples at a time, that means we can control the output of the PacketCGAN generator so that we can easily generate specified minor classes samples needed to be augmented to handle the problem of class balance.

Fig. 5 and 6 shows the Confusion Matrices of the CNN-based encrypted traffic identification model based on 8 datasets augmented by four different generative methods, which are ROS/SMOTE/GAN/CGAN over ISCX2012 and USTC-TFC2016, respectively. The elements on the diagonal of Confusion Matrix refer to the correct ones of classification and all the others are mis-classified. It can be clearly seen that the false positive rate of minor class in Fig 5(d) and 6(d) is lower than ROS as shown in Fig 5(a) and 6(a), SMOTE as shown in Fig 5(b) and 6(b) and GAN as shown in Fig 5(c) and 6(c).

Fig 8 shows the performance metrics of 8 datasets more clearly. As we can see from Fig 8 that PacketCGAN greatly outperforms the other three methods for the minor class like gmail, facebook.

From Table VII, we can see the detail of the evaluation performance metrics of classification over the datasets augmented by different methods more clearly. For simplicity, only



(a) d_loss for ISCX2012 (b) g_loss for ISCX2012

(c) d_loss for USTC-TFC2016 (d) g_loss for USTC-TFC2016

Fig. 4: Loss of PacketCGAN

the performance metrics of ISCX2012 datasets are presented here as an example. The performance statistics of classification based on all four methods are shown in Table VIII, apparently, our proposed PacketCGAN has achieved better performance than the other three methods.

VI. Conclusion and Future Work

In this paper, we proposed a CGAN-based traffic data augmenting method called PacketCGAN to solve the class imbalance problem in the dataset. After the specified sample data is generated by the generator of PacketCGAN, the synthesized data is combined with the original data to build a new balanced dataset. We use MLP/CNN/SAE models to verify the classification performance over different datasets. The experimental results show that the balanced dataset augmented by FlowCGAN can achieve better performance than the others including ROS, SMOTE and GAN. In the future, we will further study other types of GAN applications in traffic classification and try to solve the problems of unstable training and mode collapse.

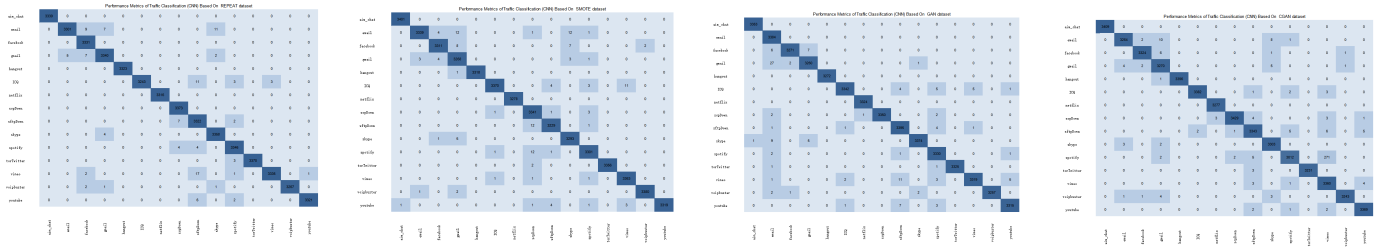
Acknowledgment

This work was supported by the National Science Foundation of China (61972211)

Reference

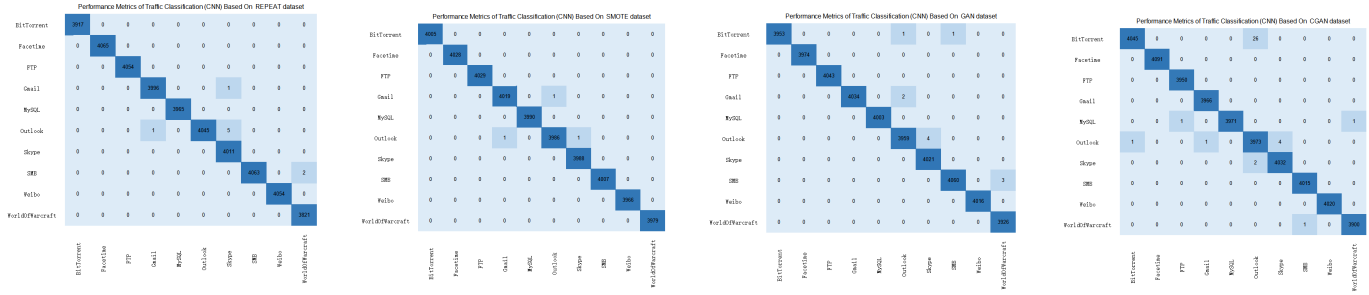
[1] M. Finsterbusch, C. Richter, E. Rocha, J. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 1135–1156, Second 2014.

[2] P. Wang, F. Ye, and X. Chen, "A smart home gateway platform for data collection and awareness," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 87–93, Sep. 2018.



(a) ROS (b) SMOTE (c) GAN (d) PacketCGAN

Fig. 5: Confusion Matrices of CNN-based encrypted traffic identification method on ISCX2012 dataset



(a) ROS (b) SMOTE (c) GAN (d) PacketCGAN

Fig. 6: Confusion Matrices of CNN-based encrypted traffic identification method on USTC-TFC2016

[3] P. Wang, X. Chen, F. Ye, and Z. Sun, "A smart automated signature extraction scheme for mobile phone number in human-centered smart home systems," *IEEE Access*, vol. 6, pp. 30 483–30 490, 2018.

[4] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and future directions in traffic classification," *IEEE Network*, vol. 26, no. 1, pp. 35–40, January 2012.

[5] G. Sun, Y. Xue, Y. Dong, D. Wang, and C. Li, "An novel hybrid method for effectively classifying encrypted traffic," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, Dec 2010, pp. 1–5.

[6] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, "A survey of methods for encrypted traffic classification and analysis," *Netw.*, vol. 25, no. 5, pp. 355–374, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1002/nem.1901

[7] D. J. Arndt and A. N. Zincir-Heywood, "A comparison of three machine learning techniques for encrypted network traffic analysis," in *2011 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, April 2011, pp. 107–114.

[8] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescap, "Mobile encrypted traffic classification using deep learning," in *2018 Network Traffic Measurement and Analysis Conference (TMA)*, June 2018, pp. 1–8.

[9] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep learning based encrypted network traffic classification in sdn home gateway," *IEEE Access*, vol. 6, pp. 55 380–55 391, 2018.

[10] M. J. S. M. S. Mohammad Lotfollahi, Ramin Shirali Hossein Zade, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Available from http://www.arxiv.org*, 2017.

[11] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pp. 43–48, 2017.

[12] and, , and and, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, Jan 2017, pp. 712–717.

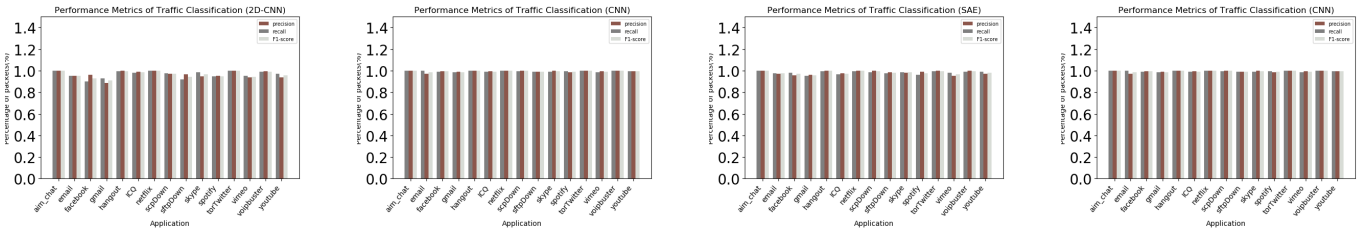
[13] Z. Chen, K. He, J. Li, and Y. Geng, "Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks," in *2017 IEEE International Conference on Big Data (Big Data)*, Dec 2017, pp. 1271–1276.

[14] X. Chen, J. Yu, F. Ye, and P. Wang, "A hierarchical approach to encrypted data packet classification in smart home gateways," in *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, Aug 2018, pp. 41–45.

[15] Z.Wang, "The application of deep learning on traffic identification," *Available from http://www.blackhat.com*, 2015.

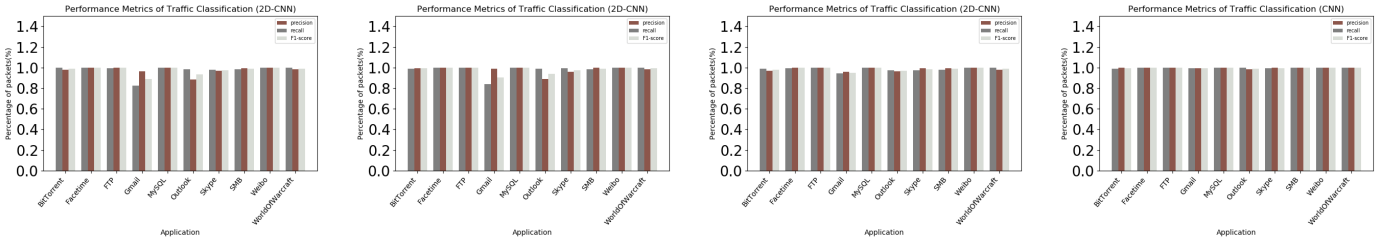
[16] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[17] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu,



(a) ROS (b) PacketCGAN (c) GAN (d) PacketCGAN

Fig. 7: Performance metrics of DL-based encrypted traffic identification method on ISCX



(a) ROS (b) SMOTE (c) GAN (d) PacketCGAN

Fig. 8: Performance metrics of DL-based encrypted traffic identification method on USTC-TFC2016

"Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2018.

[18] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, Oct. 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=1293951.1293954

[19] L. Vu, C. T. Bui, and Q. U. Nguyen, "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*, ser. SoICT 2017. New York, NY, USA: ACM, 2017, pp. 333–339. [Online]. Available: http://doi.acm.org/10.1145/3155133.3155175

[20] L. Vu, D. Van Tra, and Q. U. Nguyen, "Learning from imbalanced data for encrypted traffic identification problem," in *Proceedings of the Seventh Symposium on Information and Communication Technology*, ser. SoICT '16. New York, NY, USA: ACM, 2016, pp. 147–152. [Online]. Available: http://doi.acm.org/10.1145/3011077.3011132

[21] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: The databoost-im approach," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 30–39, Jun. 2004. [Online]. Available: http://doi.acm.org/10.1145/1007730.1007736

[22] D. C. Sicker, P. Ohm, and D. Grunwald, "Legal issues surrounding monitoring during network research," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '07. New York, NY, USA: ACM, 2007, pp. 141–148. [Online]. Available: http://doi.acm.org/10.1145/1298306.1298307

[23] T. T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, Fourth 2008.

[24] D. Li, Y. Zhu, and W. Lin, "Traffic identification of mobile apps based on variational autoencoder network," in *2017 13th International Conference on Computational Intelligence and Security (CIS)*, Dec 2017, pp. 287–291.

[25] N. Japkowicz, "Learning from imbalanced data sets: A comparison of various strategies." AAAI Press, 2000, pp. 10–15.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622407.1622416

[27] Z. Lin, Y. Shi, and Z. Xue, "Idsgan: Generative adversarial networks for attack generation against intrusion detection," *CoRR*, vol. abs/1809.02077, 2018.

[28] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on gan," *CoRR*, vol. abs/1702.05983, 2017.

[29] J.-Y. Kim, S.-J. Bu, and S.-B. Cho, "Malware detection using deep transferred generative adversarial networks," 10 2017, pp. 556–564.

[30] M. Salem, S. Taheri, and J. S. Yuan, "Anomaly generation using generative adversarial networks in host based intrusion detection," *CoRR*, vol. abs/1812.04697, 2018.

[31] M. Rigaki and S. Garcia, "Bringing a gan to a knife-fight: Adapting malware communication to avoid detection," in *2018 IEEE Security and*

TABLE VII: Description of Experimental results on different generative methods for ISCX2012

Application	Unbalanced dataset			ROS dataset			PacketCGAN dataset			GAN dataset			PacketCGAN dataset		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
AIM	0.8823	0.9027	0.8924	0.9757	0.9557	0.9656	0.9532	0.9386	0.9642	0.9989	0.9989	0.9989	0.9792	0.9875	0.9833
Email-Client	0.9949	0.9893	0.9921	0.9985	0.9866	0.9925	0.9541	0.9627	0.9558	0.9821	0.9584	0.9701	0.9792	0.9875	0.9833
Facebook	0.9287	0.913	0.9208	0.9808	0.9571	0.9688	0.9531	0.9639	0.9579	0.9469	0.9667	0.9567	0.9792	0.9875	0.9833
Gmail	0.9803	0.988	0.9841	0.9939	0.9919	0.9929	0.9426	0.9578	0.9512	0.9417	0.9436	0.9427	0.9792	0.9875	0.9833
Hangout	0.9421	0.9827	0.962	0.9712	0.9881	0.9796	0.9578	0.9621	0.9601	0.9669	0.9881	0.9774	0.9792	0.9875	0.9833
ICQ	0.9778	0.9225	0.9494	0.9343	0.9807	0.9569	0.9731	0.9639	0.9681	0.9731	0.9778	0.9754	0.9792	0.9875	0.9833
Netflix	0.9999	0.9965	0.9982	0.9983	0.9978	0.998	0.9781	0.9651	0.9688	0.998	0.9998	0.9998	0.9792	0.9875	0.9833
SCP	0.9997	0.9989	0.9993	0.9343	0.9982	0.9991	1	0.9679	0.9736	0.9891	0.9872	0.9882	0.9792	0.9875	0.9833
SFTP	0.9963	0.9942	0.9952	0.9995	0.998	0.9988	0.9841	0.9837	0.9826	0.9833	0.9652	0.9742	0.9792	0.9875	0.9833
Skype	0.9473	0.992	0.9692	0.996	0.996	0.996	0.9881	0.9871	0.9873	0.9787	0.9787	0.9787	0.9792	0.9875	0.9833
Spotify	0.995	0.9935	0.9942	0.9952	0.9975	0.9964	0.9731	0.9652	0.9663	0.9768	0.9671	0.9719	0.9792	0.9875	0.9833
torTwitter	0.9343	0.9993	0.9997	0.9343	0.9992	0.9996	0.9831	0.9846	0.9841	0.9908	0.9998	0.9954	0.9792	0.9875	0.9833
Vimeo	0.9973	0.9951	0.9962	0.9948	0.9973	0.9961	0.9656	0.9631	0.9642	0.9718	0.9564	0.9641	0.9792	0.9875	0.9833
voipbuster	0.9991	0.9965	0.9978	0.9343	0.9909	0.9954	0.9639	0.9745	0.9721	0.9887	0.9826	0.9856	0.9792	0.9875	0.9833
Youtube	0.9975	0.9986	0.998	0.9997	0.9982	0.999	0.968	0.9662	0.9731	0.9613	0.9807	0.9709	0.9792	0.9875	0.9833
Average	**0.9759**	**0.9775**	**0.9766**	**0.9892**	**0.9889**	**0.9891**	**0.9751**	**0.9789**	**0.9771**	**0.9766**	**0.9767**	**0.9766**	**0.9936**	**0.9958**	**0.9947**

TABLE VIII: Classification Performance Statistics.

data augmenting methods	Accuracy	Precision	Recall	F1-Score
unbalanced dataset	0.9797	0.9759	0.9775	0.9766
ROS dataset	0.9889	0.9892	0.9889	0.9891
SMOTE dataset	0.9769	0.9751	0.9789	0.971
GAN dataset	0.9766	0.9766	0.9767	0.9766
PacketCGAN dataset	0.9951	0.9936	0.9958	0.9947

Privacy Workshops (SPW), May 2018, pp. 70–75.

[32] L. Vu, C. Thanh Bui, and U. Nguyen, "A deep learning based method for handling imbalanced problem in network traffic classification," 12 2017, pp. 333–339.

[33] R. Hasibi, M. Shokri, and M. Dehghan, "Augmentation scheme for dealing with imbalanced network traffic classification using deep learning," *ArXiv*, vol. abs/1901.00204, 2019.

[34] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *arXiv e-prints*, p. arXiv:1406.2661, Jun 2014.

[35] PCAP Next Generation Dump File Format, *Available from http://www.tcpdump.org/pcap/pcap.html*.

[36] A. Habibi Lashkari, G. Draper Gil, M. Mamun, and A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related features," 02 2016.

[37] C. X. Wang Pan, "Encrypted traffic identification method based on stacked automatic encoder," 2018, pp. 1–8.

[38] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 2016, pp. 407–414.