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

1st Shuhang Li
School of Modern Posts
Nanjing University
of Posts&Telecommunications
Nanjing, China
lish@runtrend.com.cn

2nd Pan Wang
School of Modern Posts
University of Nanjing University
of Posts and Telecommunications
Nanjing, China
wangpan@njupt.edu.cn

3rd Xiaokang Zhou
Faculty of Data Science
Shiga University Hikone, Japan
RIKEN Center for Advanced
Intelligence Project
Tokyo, Japan

4th ZiXuan Wang
School of Modern Posts
University of Nanjing University
of Posts and Telecommunications
Nanjing, China
wangzx@runtrend.com.cn

5th MoXuan Zhang
Schools of International Education
Jinling Institute of Technology
Nanjing, China
zhangmoxuan_7@126.com

***Abstract*—In the field of encrypted traffic identification, more and more researchers have begun to apply machine learning (ML), especially deep learning (DL) to traffic classification problems. Although these methods can automatically extract traffic features to overcome the difficulty of traditional classification methods like DPI in terms of feature engineering, a large amount of data is needed to learn the characteristics of various types of traffic. Therefore, the performance of classification model always significantly depends on the quality of the data set. The establishment of data sets is a time-consuming and laborious task. At the same time, it is often difficult to collect a large amount of traffic for some unpopular applications while traffic of hot applications is easy to access, which leads to the problem of data imbalances in data sets. In this paper, we proposed an unbalanced dataset solution based on FlowCGAN. As an instance of Generative Adversarial Nets(GAN), the method utilizes the advantage of GAN in data augmentation and generates new data samples by learning the characteristics of the traffic data, thereby achieving the purpose of balancing the data set. To verify the feasibility of this method, we use a classical deep learning model Convolutional Neural Networks(CNN) to classify the unbalanced data set, the random oversampled balanced data set, and the FlowCGAN balanced data set respectively. The experimental results show that the FlowCGAN method has better performance when compared with the other two data sets under the same conditions.**

***Index Terms*—encrypted traffic identification, deep learning, Conditional Generative Adversarial Nets, traffic classification, data balance, convolutional neural network**

I. Introduction

With the rapid development of network technology, the types and quantity of traffic data in cyberspace are increasing. The identification and classification of network traffic is an important research content in the field of network management and network security. The ISP needs to perform dynamic access control according to the proportion of various types of traffic in the bandwidth to provide a better user experience for the customer. Security regulators need to detect malicious traffic in network traffic in real-time to avoid serious losses. As user privacy awareness increases and various encryption technologies mature, many Internet applications begin to encrypt their traffic data. At the same time, with the rapid development of 5G and the Internet of Things and the Industrial Internet, the node interaction information has high-security requirements, and the proportion of encrypted traffic in the network is also increasing. The emergence of encrypted traffic poses a huge challenge to the fields of network security, QoS, and network resource scheduling.

Traditional traffic identification technologies include the following: port-based matching [1], payload-based [2], and flow feature statistics. The emergence of various proprietary protocols and VPN tunneling technologies has made port matching technology quickly ineffective. To ensure the security of the data, the payload of the encrypted traffic is invisible, so the method based on the payload cannot cope with the classification problem of the encrypted traffic. To solve the above problems, researchers began to try to combine machine learning algorithms with flow statistics or timing features for traffic identification and classification [3]–[6]. However, this method has certain defects: (1) it is easy to be limited by the sample size, and it falls into the local optimal solution, and the generalization ability is poor; (2) the classification effect is greatly affected by the feature design, which brings uncertainty to the classification effect; (3) The lack of ability to automatically learn traffic characteristics requires manual design of features, resulting in models outdated rapidly. The emergence of deep learning has solved the above problems well. Some scholars use deep learning models such as CNN, MLP, SAE, LSTM, etc. to perform automatic feature extraction without

human intervention [7]–[10]. The experimental results show that these methods are superior to traditional methods.

However, a large amount of data for feature learning is inseparable whether it is machine learning or deep learning. Due to the different heats of various applications, sample imbalances often occur when creating data sets. That is, the number of popular application samples is much larger than that of non-hot applications. During model training, the characteristics of small sample flows are easily masked by large samples, resulting in classification errors.

In this paper, we proposed an unbalanced dataset solution based on FlowCGAN, which utilizes the advantage of GAN in data augmentation, and generates a certain amount of minor class through feature learning to achieve the purpose of balancing the traffic data set. In this paper, the classical deep learning model CNN was used to classify the unbalanced data set, the random oversampled data set and the CGAN balanced data set to verify the feasibility of FlowCGAN data generation.

The subsequent chapters of this paper are organized as follows: Section II describes the related works about solving the unbalanced data set; Section III briefly describes the principles and network architecture of GAN and CGAN; Section IV elaborates on the methodology of FlowGAN, including data preprocessing, model architecture, and related algorithms; Section V describes the experimental environment and experimental results; Section VI provides conclusions about our work and an introduction to the future work.

II. Related works

In machine learning, scholars have done a lot of research on how to deal with unbalanced data sets [11]–[13]. The most common methods for dealing with unbalanced data are: collect more traffic data, modifying objective cost functions, resampling and generating artificial data [14]. The easiest way is to collect more small sample data, but this method often requires a lot of time, and the traffic data of some unpopular applications is difficult to collect. The main idea of modifying the objective cost function is to modify the sample weights, including increasing the weight of small samples and reducing the weight of large samples [15]. The difficulty of this approach is how to determine the weight of each sample properly. Resampling includes two methods: random oversampling and random undersampling [16]. Random undersampling randomly culling of some of the data in major class, while random oversampling randomly copying some data from the minor class to achieve the purpose of expanding the sample. However, this method does not generate new features in nature, it is only an extension of the number of minor class. The classic method of generating artificial data is the smote oversampling method, which constructs new sample data by randomly sampling the attribute values, not just the copying of the samples [17]. However, it is possible to generate data that does not exist in reality and thereby destroy the linear relationship of the original features.

III. Generative Adversarial Nets

A. GAN

A classic GAN network consists of two parts, the generator G and the discriminator D.GAN is unsupervised learning. The role of the generator is to simulate noise into real data by learning the characteristic distribution of real data. The discriminator aims at determining whether the sample is real data or data 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 data and the generated data, and the output $D(x;\theta_d)$ indicates the probability of whether the sample data inputted is real [18]. During the training process, G and D play a mini-max game until D can't judge whether the sample data is real, which means that the two networks reach the Nash balance. 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 (1), $P_{data}(x)$ represents the true data distribution. When training D, the goal is to judge $G(z)$ as the true probability $D(G(z))$ as small as possible and to judge the true data x as the true probability $D\,(x)$ as large as possible. When training G, the goal is to make $D(G(z))$ as large 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

When compared with other complex generation models, GAN's game competition makes it possible to generate data without pre-modeling. It only needs to sample the distribution to approximate the real data. This is the biggest feature of GAN. This approach also has the disadvantage that it is too free and the output of the generator cannot be controlled by humans. For example, when training the Mnist dataset, the number output by the generator may be any number from 0-9. In order to solve this problem, Montreal proposed Conditional Generative Adversarial Nets CGAN [19]. The method guides the data generation process by adding a constraint condition y to both D and G. This simple improvement has proven to be very effective and widely used. Fig. 1 shows the network structure of CGAN.

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

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

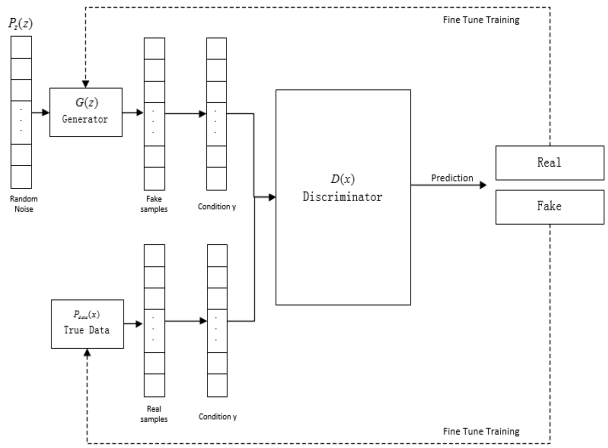


Fig. 1. The network structure of CGAN.

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, and updating the parameters according to the output result;
- Generate random noise $P_z(z)$, which is fed into generator G together with label y in the above step, and G generates analog data.
- Feed the analog data and the label y generated in the above step into the discriminator, and G adjusts the parameter according to the output result of D.
- Repeat the above steps until G and D reach the Nash equilibrium

IV. The Methodology of FlowGAN

A. Data preprocessing

The captured network traffic data is often saved in PCAP or PCAPNG format when framing the data set. As the PCAP file format specification shows that the byte information of the data traffic type is saved in hexadecimal. When converted to decimal, the hexadecimal ranges from 0 to 255, which corresponds to the pixel range of the single-channel grayscale image. Therefore, traffic classification can refer to many deep learning methods for image recognition. However, PCAP packets can not directly be used for model training and need to be pre-processed.

Data preprocessing consists of three steps: filtering, specification length, and normalization. Fig. 3(a) is a flow chart of data preprocessing, which is detailed as follows:

- Load the entire PCAP file. Filter the first 24 bytes of the file header. This part only contains file information and does not help with traffic classification.
- While cycle through the group information the traffic, grouping information is first filtered. Since the network environment is not guaranteed to be pure during the process of capturing traffic, it is necessary to filter some useless data packets, such as APR and DHCP.
- Specification the length of the filtered packets. In the deep learning process, the model converts the input into a matrix for calculation, which requires the length of the data input to be consistent. The length of collected packet information tends to be different in each group, so it is necessary to truncate the long packet and padding the short packet. The specified packet length used in this paper is 1480, and the processed packet forms a Packet Byte Matrix (PBM) as an input to the deep learning model [8].
- Data normalized. Limiting the pre-processed data to the range [0,1] can improve the accuracy and convergence speed of the model.
- Data marked. Marke up the traffic information according to the application classification after processing.

B. Unbalanced data set processing

In this paper, two methods were used to process the unbalanced dataset. The first one is a random oversampling method. The principle of this method is to randomly sample minor class to supplement the sample size. As this method is simple, the new data is only a copy of the original data, which will lead to the model learning wrong features easily, and even the over-fitting phenomenon. The second method is to generate minor sample data using a FlowCGAN generator that reaches the Nash balance. The data generated in this way is very close to the real data, and new sample features are introduced. The data balancing process is shown in Fig. 3(c). It includes the following processes: reading the data set and sample tags obtained in Section 4.1 above and obtaining the number of different samples; The number of each application that we set to train the FlowCGAN is 10,000. For major class, 10000 data is randomly sampled, and 4000 data for the minor class; Load the trained FlowCGAN model to generate 6000 small sample data by G, and merge with the data in the previous step to get a balanced data set.

C. Algorithm Description

In the FlowCGAN designed in this paper, both G and D have a three-layer structure including an input layer, an output layer, and a hidden layer, as is shown in Fig. 2 below. CGAN's training steps mainly include the following three steps:

Step one, train the discriminator. You need to fix the parameters of G while training D. The data for the input layer is derived from the PBM packet byte matrix obtained in Section IV-A above, with a dimension of 1495. The hidden layer consists of 128 neurons, updating the weight of the input data and adding a bias:

$$D_{h1} = relu(input \cdot W_1 + b_1) \tag{4}$$

The 'input' in (4) is composed of the real data x and the sample label y. The hidden layer uses $ReLU$ as the activation function. The output layer has only one neuron, and the data from the hidden layer is weighted and offset to obtain the probability that if the sample is true:

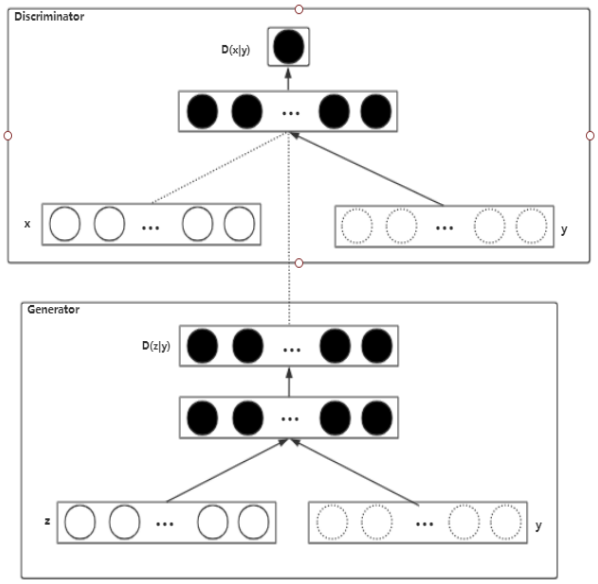


Fig. 2. The model architecture of CGAN.

$$d\_out = D_{h1} \cdot W_2 + b_2 \tag{5}$$

The model uses the Adam optimization algorithm as the optimizer to update weights W_1, W_2 and offsets b_1, b_2. 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 identical to the value of data dimension:1480,m equals to the value of mini_batch.

Algorithm 1 Discriminator of FlowCGAN training

Input: real data X_τ and label y_τ from Section IV-B,τ is the number of iterations
Output: Scores of the input data d_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**
    concat X_τ with y_τ:
    Compute the output using Equation. (4);
    Compute the output using Equation. (5);
    Output distinguishing results according to Equation. (5);
    Optimal the loss function
    Update weights and bias;
  end for
end for

Step two, train the generator G.The parameters of D need to be fixed like step one. The input layer of G consists of 115 neurons, and the input data includes stochastic noise $P_z(z)$ and sample label y. The hidden layer also has 128 neurons to update the weight and offset of the input data:

$$G_{h1} = relu(input \cdot W_1 + b_1) \tag{6}$$

The 'input' in (7) is made up of random noise $P_z(z)$ and the sample label y, and $ReLu$ is also used as the activation function. The output layer contains 1480 neurons, weighting and biasing the output of the hidden layer. The computed result will be activated by the sigmoid function.

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

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 Generator of FlowCGAN training

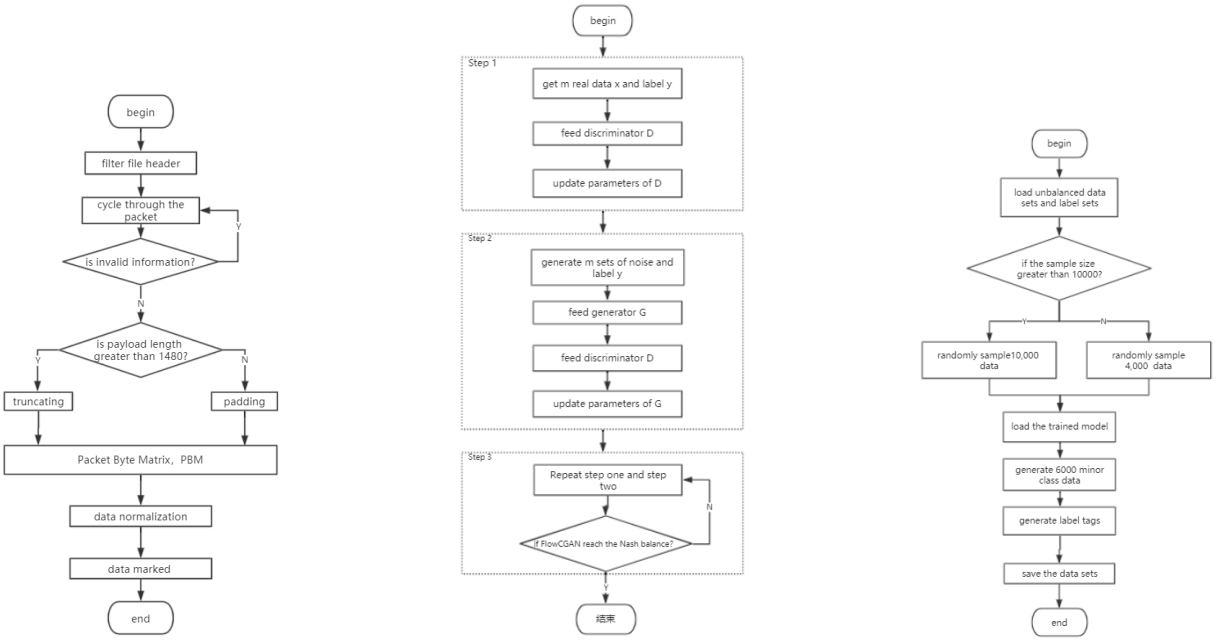
Input: random noise Z_τ, and label y_τ,τ is the number of iterations
Output: generation data g_τ

Set the relevant initialization parameters,e represents the training cycle: epoches.
for τ in e **do**
  for each batch of m input data **do**
    concat 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. Experimental result

A. Experimental environmental

The experimental environmental parameters of this paper are shown in Table I. To verify the feasibility of the FlowCGAN algorithm, we set up an unbalanced data set, a random oversampling balanced data set, and a FlowCGAN balanced data set for comparison. The classification of the three data sets was verified by the CNN convolutional neural network, the most basic deep learning model. As shown in Fig. 4, we design a simple CNN model for encrypted traffic identification. The model uses 1 input layer, 1480 neurons; 3 convolution layers and pooling layer, composed of 128 convolution kernels, activated by ReLU; a fully connected layer of 128 neurons and an output layer, 15 neurons, and the activation function is Softmax for classification. Table II shows the parameters of the optimizer, loss function, epoches, and mini_batch used in the deep training model training process.



(a) Data Preprocessing (b) FlowCGAN Model Training (c) Unbalanced Dataset Processing

Fig. 3. The methodology of FlowGAN

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 1480, 1)	0
conv1d_1 (Conv1D)	(None, 1476, 128)	768
max_pooling1d_1 (MaxPooling1	(None, 295, 128)	0
conv1d_2 (Conv1D)	(None, 291, 128)	82048
max_pooling1d_2 (MaxPooling1	(None, 58, 128)	0
conv1d_3 (Conv1D)	(None, 54, 128)	82048
max_pooling1d_3 (MaxPooling1	(None, 1, 128)	0
flatten_1 (Flatten)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 15)	1935

Total params: 183,311
Trainable params: 183,311
Non-trainable params: 0

Fig. 4. Model architecture of CNN.

TABLE I
EXPERIMENTAL ENVIRONMENT PARAMETERS

Category	Parameters
GPU	Nvidia GPU(GeForce GTX 1660Ti)
Operating System	Win 10
Deep learning platform	TensorFlow 1.13.1 + Keras 1.0.7
CUDA Version	9.0
CuDNN Version	7.6.0

TABLE II
TRAINING PARAMETERS

Training Parameters	Generator	Discriminator	CNN
Optimizer	Adma	Adma	rmsprop
Loss Function	Cross-Entropy	Cross-Entropy	Cross-Entropy
Epoches	200000	200000	50
Mini_batch	64	64	256

B. Dataset for train

The dataset for evaluation is selected from the "ISCX VPNnonVPN traffic dataset" [20]. The dataset contains many encryption applications, protocols such as HTTPS, SFTP, Facebook, Hangouts, etc. A regular session and a session over VPN were captured in this dataset. 15 applications were chosen to make up the training samples for CGAN as Table V shows. The majority classes such as NetFlix accounting for 25.13% in the selected dataset, while the minimal sample size ICQ account for only 2.05%. The number of traffic per application in the balanced data set is 10,000.

C. Performance metrics

In order to evaluate the performance of the model, we used the following three indicators: Precision, Recall, and F1-Score. FP(False Positive) means that the traffic of non-category C

TABLE III

DESCRIPTION OF THE CHOSEN DATASETS.

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%**

(C refers to a specific category) is classified into category C; TN(True Negative) means that streams of non-category C are classified into non-category C. FN(False Negative) means that traffic belonging to category C is classified as non-category C; TP(True Positive) refers to traffic belonging to category C and is classified into category C.F1-Score is the weighted harmonic average of the accuracy rate and the recall rate [21], which is used to comprehensively reflect the overall indicator.

- Precision:

$$Precision = \frac{TP}{TP + FN} \tag{8}$$

- Recall:

$$Recall = \frac{TP}{TP + FP} \tag{9}$$

- F1-Score:

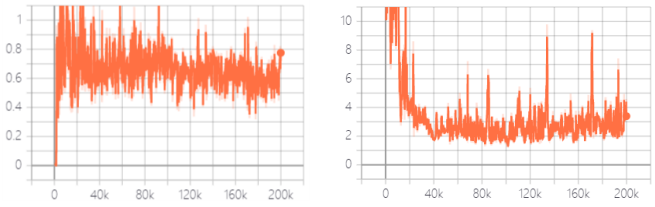
$$F1 - Score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{10}$$

D. Analysis of experimental results

Figure 5 shows the trend of loss of generator and discriminator during the FlowCGAN training process. It can be seen that CGAN does not change the instability characteristics of GAN. The loss of G and D always fluctuates within a range instead of reaching a convergence value.

Fig 6 shows the classification confusion matrices of three datasets based on the CNN-based encrypted traffic identification model, in which the elements on the diagonal representing the correct classification, and all other elements are misjudged. It can be clearly seen that the false positive rate of minor class in Fig 6(a) is higher than Fig 6(b) and Fig 6(c).

Fig 7 shows the performance indicators of the three data sets more clearly. As we can see from Fig 7 that FlowCGAN performs best in the minor class except for ICQ. The performance metrics of ICQ in the unbalanced dataset were aways higher better than the two balanced datasets.



(a) d loss (b) g loss

Fig. 5. Loss of FlowCGAN

TABLE IV

PERFORMANCE OF CNN-BASED TRAFFIC CLASSIFIER.

data augmenting methods	Accuracy	Precision	Recall	F1-Score
results of unbalanced dataset	0.9897	0.9759	0.9775	0.9766
results of oversampling dataset	0.9889	0.9892	0.9889	0.989
results of CGAN dataset	0.9951	0.9951	0.9951	0.9951

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a CGAN-based traffic data enhancement method called FlowCGAN to solve the problem of unbalanced sample size in the dataset. After the specified sample data is generated by the generator, The generated data is combined with the original data to construct a new balanced dataset. We use CNN to verify the classification effect of different data sets. The experimental results show that the balanced data set generated by FlowCGAN has better performance. In the future, we will further study other types of GAN applications in traffic classification and try to solve the problem of unstable training.

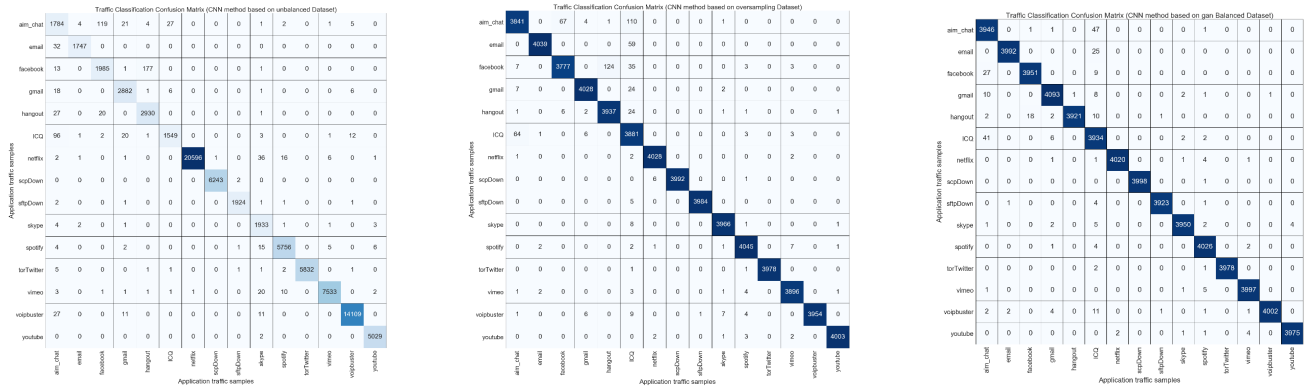
ACKNOWLEDGMENT

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

REFERENCES

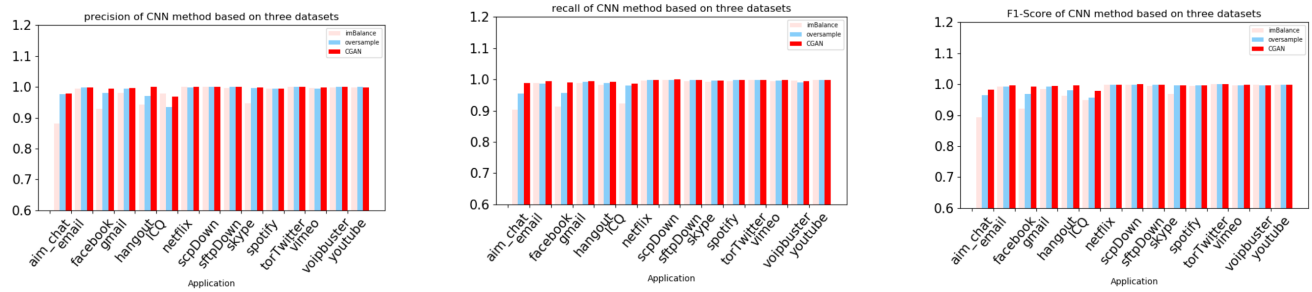
[1] Service Name and Transport Protocol Port Number Registry, *Available from https://www.iana.org/assignments/service-names-port-numbers*.

[2] J. Sherry, C. Lan, R. A. Popa, and S. Ratnasamy, "Blindbox: Deep packet inspection over encrypted traffic," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM '15. New York, NY, USA: ACM, 2015, pp. 213–226. [Online]. Available: http://doi.acm.org/10.1145/2785956.2787502



(a) confusion matrix of unbalanced dataset (b) confusion matrix of oversampling dataset (c) confusion matrix of FlowCGAN dataset

Fig. 6. Confusion matrices of CNN-based encrypted traffic identification method



(a) performance metrics of the unbalanced dataset (b) performance metrics of oversampling dataset (c) performance metrics of FlowCGAN dataset

Fig. 7. Performance metrics of CNN-based encrypted traffic identification method

[3] 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.

[4] 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

[5] M. Shen, M. Wei, L. Zhu, and M. Wang, "Classification of encrypted traffic with second-order markov chains and application attribute bigrams," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1830–1843, Aug 2017.

[6] M. Finsterbusch, C. Richter, E. Rocha, J. A. 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.

[7] 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.

[8] C. X. Wang Pan, "Encrypted traffic identification method based on stacked automatic encoder," 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] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, Jan 2017, pp. 712–717.

[11] R. Longadge and S. Dongre, "Class Imbalance Problem in Data Mining Review," *arXiv e-prints*, p. arXiv:1305.1707, May 2013.

[12] C. Chen and L. Breiman, "Using random forest to learn imbalanced data," *University of California, Berkeley*, 01 2004.

[13] H. M. Nguyen, E. W. Cooper, and K. Kamei, "Borderline over-sampling for imbalanced data classification," *Int. J. Knowl. Eng. Soft Data Paradigm.*, vol. 3, no. 1, pp. 4–21, Apr. 2011. [Online]. Available: http://dx.doi.org/10.1504/IJKESDP.2011.039875

[14] 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

[15] Z. Qin, C. Zhang, T. Wang, and S. Zhang, "Cost sensitive classification in data mining," in *Proceedings of the 6th International Conference on Advanced Data Mining and Applications: Part I*, ser. ADMA'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 1–11.

TABLE V

DESCRIPTION OF EXPERIMENTAL RESULTS

Application	Unbalanced dataset			oversampling Balanced dataset			CGAN Balanced dataset		
	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	0.9792	0.9875	0.9833
Email-Client	0.9949	0.9893	0.9921	0.9985	0.9866	0.9925	0.9992	0.9938	0.9965
Facebook	0.9287	0.913	0.9208	0.9808	0.9571	0.9688	0.9952	0.991	0.9931
Gmail	0.9803	0.988	0.9841	0.9939	0.9919	0.9929	0.9959	0.9944	0.9951
Hangout	0.9421	0.9827	0.962	0.9712	0.9881	0.9796	0.9997	0.9917	0.9957
ICQ	0.9778	0.9225	0.9494	0.9343	0.9807	0.9569	0.969	0.9872	0.978
Netflix	0.9999	0.9965	0.9982	0.9983	0.9978	0.998	0.9995	0.998	0.9988
SCP	0.9997	0.9989	0.9993	1	0.9982	0.9991	1	0.9997	0.9999
SFTP	0.9963	0.9942	0.9952	0.9995	0.998	0.9988	0.9995	0.9985	0.999
Skype	0.9473	0.992	0.9692	0.996	0.996	0.996	0.9982	0.9965	0.9973
Spotify	0.995	0.9935	0.9942	0.9952	0.9975	0.9964	0.9951	0.9983	0.9967
torTwitter	1	0.9993	0.9997	1	0.9992	0.9996	1	0.9992	0.9996
Vimeo	0.9973	0.9951	0.9962	0.9948	0.9973	0.9961	0.998	0.9983	0.9981
voipbuster	0.9991	0.9965	0.9978	1	0.9909	0.9954	0.9998	0.9945	0.9971
Youtube	0.9975	0.9986	0.998	0.9997	0.9982	0.999	0.999	0.998	0.9985
Average	**0.9759**	**0.9775**	**0.9766**	**0.9892**	**0.9889**	**0.989**	**0.9951**	**0.9951**	**0.9951**

[Online]. Available: http://dl.acm.org/citation.cfm?id=1947599.1947601

[16] 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

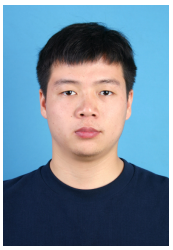
[17] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622407.1622416

[18] 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.

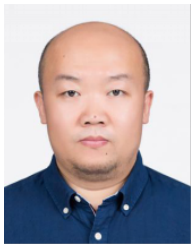
[19] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," *arXiv e-prints*, p. arXiv:1411.1784, Nov 2014.

[20] 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.

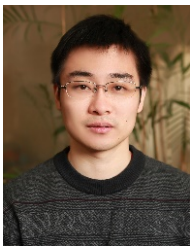
[21] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: Myths, caveats, and the best practices," in *Proceedings of the 2008 ACM CoNEXT Conference*, ser. CoNEXT '08. New York, NY, USA: ACM, 2008, pp. 11:1–11:12. [Online]. Available: http://doi.acm.org/10.1145/1544012.1544023



ShuHang Li was graduated from Jiangsu University of Science and Technology,Zhenjiang ,China, in 2013.He is currently pursuing a master's degree at Nanjing University of Posts & Telecommunications,Nanjing China,under the direction of Professor Wang. His research direction is encrypted traffic identification,and he also interested in Deep Packet Inspection and applications.(email:lish@runtrend.com.cn)



Pan Wang (M'18) received the BS degree from the Department of Communication Engineering, Nanjing University of Posts & Telecommunications, Nanjing, China, in 2001, and the PhD degree in Electrical & Computer Engineering from Nanjing University of Posts & Telecommunications, Nanjing, China, in 2013. He is currently an Associate Professor in the School of Modern Posts, Nanjing University of Posts & Telecommunications, Nanjing, China. His research interests include cyber security and communication network security, network measurements, Quality of Service, Deep Packet Inspection, SDN, big data analytics and applications. From 2017 to 2018, he was a visiting scholar of University of Dayton (UD) in the Department of Electrical and Computer Engineering.(email:wangpan@njupt.edu.cn)



Xiaokang Zhou (M12) received the Ph.D. degree in human sciences from Waseda University, Japan, in 2014. From 2012 to 2015, he was a research associate with the Department of Human Informatics and Cognitive Sciences, Faculty of Human Sciences, Waseda University, Japan. From 2016, he has been a lecturer with the Faculty of Data Science, Shiga University, Japan. He also works as a visiting researcher in the RIKEN Center for Advanced Intelligence Project (AIP), RIKEN, Japan, from 2017. Dr. Zhou has been engaged in interdisciplinary research works in the fields of computer science and engineering, information systems, and social and human informatics. His recent research interests include ubiquitous and social computing, big data mining and analytics, machine learning, behavior and cognitive informatics, cyber-physical-social-system, cyber intelligence and cyber-enabled applications. Dr. Zhou is a member of the IEEE CS, and ACM, USA, IPSJ, Japan, and JSAI, Japan.

ZiXuan Wang was born in Nanjing, Jiangsu, China ,in 1994 . He obtained a bachelor's degree from Tongda College of Nanjing University of Posts and Telecommunications in 2017, He is currently pursuing a master's degree in logistics engineering at Nanjing University of Posts and Telecommunications, under the direction of Professor Wang. His research interests include encrypted traffic identification and data balancing.(email:wangzx@runtrend.com.cn)

Moxuan Zhang was born on April 7, 1998. In Zhenjiang, Jiangsu Province, China. In June 2016,Graduated from High School Affiliated To Nanjing Normal University. Studied at Jinling Institute of Technology, majoring in software engineering from September 2016. (Email: zhangmoxuan_7@126.com)