A Fraudulent Data Simulation Method Based on Generative Adversarial Networks

Xiaoguo Wang, Ran Zhao and Yuanxiu Li

College of Electronic and Information Engineering, Tongji University, Shanghai, China
Email: 1730783@tongji.edu.cn

Abstract. In view of the positive and negative sample imbalance in the process of establishing anti-fraud rules and models of the third-party payment, this paper simulated the fraudulent transactions by means of generative adversarial networks. In the design of the model, the output of the discriminator is optimized where classification results are set to three categories; the loss function of the GAN model is changed to include two parts: the source loss function and the category loss function. The generated data and the real business data are mixed according to a certain ratio, to train a fraud detection model. Verify the generation effect of this simulation method by comparing the detection effects of different detection models.

1. Introduction

When training anti-fraud models, the problem is that fraudulent transaction samples is significantly less than normal transaction samples, which brings certain difficulties to the training of fraud detection models. Generative adversarial networks (GAN) is a deep learning network proposed by Ian Goodfellow et al of the University of Montreal[1]. The main problem solved is that it can generate new samples with similar features by learning without knowing the distribution of training samples. The use of GAN can amplify data of fraudulent transactions effectively, solve the problem of sample imbalance, and provide certain data support for the establishment of anti-fraud models. This paper will study the data simulation method of fraud transactions of third-party payment based on GAN.

2. Related Works

2.1. Generative Adversarial Networks

Generative adversarial networks (GAN) is a confrontational game model which consists of two sub-models: generator and discriminator. The generator uses a random vector input \( \mathbf{Z} \) to generate target data. The discriminator is a discriminant network that is used to determine whether input is real data or not. It’s a logistic regression model and the output \( D(\mathbf{X}) \) shows the probability that \( \mathbf{X} \) is real data. The goal of the generator is to "spoof" the discriminator by learning to generate data as similar as possible to real data. The goal of the discriminator is to distinguish the data produced by the generator and the data of real samples. It can be seen that these two models are a process of confrontational games. Through continuous optimization of these two sub-models, they can achieve a Nash equilibrium. Then the probability \( D(\mathbf{X}) \) of the discriminator to identify the generated data is 0.5, which means the real data and the generated data cannot be distinguished. At this time, the generator can be considered to learn the distribution of original data.
In order to make the generator to be able to learn the distribution of original data $P_{\text{data}}$, it is necessary to define the distribution of input noise $P_{z}$, and use the generator $G(Z, \theta_g)$ to represent the mapping of the original noise distribution to target data distribution, where $\theta_g$ represents the parameters of the generator. The discriminator $D(X)$ judges the probability that the input data $X$ comes from the distribution of the real data $P_{\text{data}}$. In the process of GAN training, the probability of identifying the data source is maximized by training the discriminator, and the recognition probability is minimized by training the generator. That is, the training is a problem of confrontational game of minimizing the maximization of the objective function:

$$ G^* = \min_G \max_D V(G, D) \quad (1) $$

where:

$$ V = E_{x \sim P_{\text{data}}} [\log D(x)] + E_{z \sim P_{z}} [\log(1 - D(G(z)))] \quad (2) $$

### 2.2. WGAN

The original GAN model has some problems such as difficulty in training and lack of diversity of generated samples, which leads to unsatisfactory effect of GAN in practical applications. Martin Arjovsky analyzed the problem of the original GAN: the better the discriminator is trained, the more serious the gradient disappearance problem is. In the process of GAN training, we always adopt the method of fixing one model and training another model. Fix the generator and optimize the discriminator, obtained by (2):

$$ \max_D E_{x \sim P_{\text{data}}} [\log(D(x))] + E_{z \sim P_{z}} [\log(1 - D(G(z)))] \quad (3) $$

Fix the discriminator and optimize the generator, obtained by (2):

$$ \min_G E_{z \sim P_{z}} [\log(1 - D(G(z)))] \quad (4) $$

When the accuracy of the discriminator is good, it will output a probability of 0 for all data generated, and does not have guiding effect on the generator. Martin Arjovsky pointed out that under the action of optimal discriminator, minimizing the loss of the generator is equivalent to minimizing the JS divergence between the generated data distribution $P_{\hat{g}}$ and the real data distribution $P_{\text{data}}$.

Since there can be no non-negligible overlap between $P_{\hat{g}}$ and $P_{\text{data}}$, the JS divergence is a fixed value of log2, causing gradient disappear. Furthermore, in the case of a fixed discriminator, the generator's loss function is equivalent to $\text{KL}(P_{\hat{g}}||P_{\text{data}}) - 2JS(P_{\hat{g}}||P_{\text{data}})$. Since KL divergence is an asymmetric measurement function, that is, the penalty function has different degrees of punishment for different samples, the generator will tend to generate repeated safe samples and reduce the generation of different samples, resulting in poor sample diversity.

In order to solve these problems, Martin Arjovsky proposed the Wasserstein distance to solve the problem of measuring the distance between two distributions.

$$ W(P_{\text{data}}, P_{\hat{g}}) = \inf_{\gamma \sim \Pi(P_{\text{data}}, P_{\hat{g}})} E_{(x,y) \sim \gamma} [||x - y||] \quad (5) $$

where $\Pi(P_{\text{data}}, P_{\hat{g}})$ represents a collection of all possible joint distributions of $P_{\hat{g}}$ and $P_{\text{data}}$.
The advantage of the Wasserstein distance compared to the KL divergence and the JS divergence is that even if there is no overlap between two distributions, the Wasserstein distance can measure the difference. Because the function values of KL divergence and JS divergence have some mutation points, and the Wasserstein distance function is smooth and no mutation occurs.

3. Model Design

Aiming at solving the problem of sample imbalance in the training process of fraud detection models of third-party payment, the method of amplifying the number of negative samples can be used. For this reason, a data enhancement model SGAN is proposed in this paper based on GAN. It is hoped that by using this model, we can learn the distribution of fraudulent and suspicious transactions, and simulate the target data with similar features, then solve problem of the imbalance of positive and negative samples, at last improve the detection rate of fraud detection models. The SGAN model is designed as follows:

3.1. Generator

The target data of the generator is the transaction records of third-party payment, which have low dimensions, and have no relationship between these dimensions. Convolution layer in neural networks is not suitable for the generator. The fully-connected layer can approximately achieve all functions, thus the fully connected layer is used as the infrastructure of this sub-model.

The \( i \)-th layer neurons:

\[ O_i = \text{relu}(W_i \ast O_{i-1} + b_i) \]  

where \( O_i, W_i, b_i \) represents the output, weight, bias of the \( i \)-th layer.

3.2. Discriminator

The goal of the discriminator is to identify whether the input data is real or generated. So it can be considered as a fraud detection model. The data determined to be 0 by this sub-model is generated data, and the data determined as 1 is real. Base on this, the discriminator is optimized to increase the type of classification, which determines that the input data sources are of three types: 0 means generated data, 1 means real but fraud data, and 2 means real and normal data. This sub-model is equivalent to a semi-supervised classification model. The generated data is unlabeled, and real data has the label “real”\(^4\). This classification model is used instead of original discriminant model. The classification model receives input, for the K-classification problem, the softmax function is used to give the K+1-classification probability. The front K dimension represents the original classification probability, and the k+1 dimension represents the probability that the input is generated data.

The trading accounts are extracted for all fraudulent transaction records in data set, which are referred to as fraudulent accounts \( A_f \). Then we need to select all transaction records \( T_{a-A_f} \) for the fraudulent accounts from the original data set. For each record \( T_a \), the transaction serial number \( T_{id} \), time \( T_{time} \), amount \( T_{amt} \), source account \( T_{src} \) and destination account \( T_{dst} \) are used as initial feature. In addition, in order to be able to discriminate the legitimacy of a transaction, the historical information of the transaction account is also required as a criterion. The historical feature of the user, such as the frequency, the average transaction amount etc, are extracted, because whether the transaction is a fraudulent transaction or not is determined according to whether it conforms to the recently trading habits of this user. The transaction frequency \( T_{frq, tw} \) of trading account \( T_{src} \) in specific time window before \( T_{time} \), as well as the maximum \( T_{max, tw} \) or minimum amount \( T_{min, tw} \), the average amount \( T_{avg, tw} \) and some other statistics are used as trading feature of the account, called the supplementary feature. The initial feature and the supplementary feature are combined as transaction features into the discriminator, and then train discriminator to identify the source of input data.

Input:

- Initial transaction feature: \( [T_{id}, T_{time}, T_{amt}, T_{src}, T_{dst}] \)
- Supplementary transaction feature: \( [T_{max, tw}, T_{min, tw}, T_{avg, tw}, T_{frq, tw}] \) for \( tw \) in \( twl \)

where \( twl \) means time window list.
3.3. Loss Function

For the improved model SGAN, the loss function is modified into following two parts: the source loss function of data $L_s$ and the category loss function of data $L_t$ \[^{[5]}\]. In the process of training the generator, the source loss function $L_s$ needs to be minimized, so that it can generate data as similar as possible to real data to deceive discriminator, and the source of data cannot be recognized. In the process of training the discriminator, it is necessary to maximize the source loss function $L_s$ and minimize the category loss function $L_t$, so that it can distinguish the source of the input and determine the tag of real data.

Source loss function $L_s$:

$$L_s = -E_{X \sim p_g}[\ln(k + 1|x)] - E_{X \sim p_{data}}[\ln(1 – p(k + 1|x))]$$ \hspace{1cm} (7)

Category loss function $L_t$:

$$L_t = -E_{X \sim p_g}[\ln(p(k + 1|x))] - E_{X \sim p_{data}}[\ln(p(y|x))]$$ \hspace{1cm} (8)

where $-E_{X \sim p_{data}}[\ln(p(y|x))]$ represents the information entropy of real transactions which are discriminated as their real category (fraud, normal), $-E_{X \sim p_g}[\ln(p(k + 1|x))]$ represents the information entropy of generated transactions which are discriminated as fake data, and $-E_{X \sim p_{data}}[\ln(1 – p(k + 1|x))]$ represents the information entropy of real transaction which are discriminated to real.

When train the generator, the target is:

$$\min_G L_s = \min_G -E_{X \sim p_g}[\ln(p(k + 1|x))]$$ \hspace{1cm} (9)

When train the discriminator, the target is:

$$\max_D L_s + \min_D L_t = \min_D L_t - L_s = \min_D E_{X \sim p_{data}}[\ln(1 – p(k + 1|x))] - E_{X \sim p_{data}}[\ln(p(y|x))]$$ \hspace{1cm} (10)
Table 1. The algorithm of SGAN.

| Steps:                                                                 |
|-----------------------------------------------------------------------|
| Random initialize parameters of generator and discriminator           |
| For $t=1, 2, ..., T$ do                                               |
|  \quad \text{Random sampling from real data distribution } P_{\text{data}} \text{ to obtain } Q \text{ samples } X_r |
|  \quad \text{Random sampling from noise distribution } P_z \text{ to obtain } Q \text{ noise } Z                   |
|  \quad \text{Input noise } Z \text{ into the generator to generate } Q \text{ samples } X_g                      |
|  \quad \text{Input real samples } X_r \text{ and generated samples } X_g \text{ into the discriminator, and update parameters } \theta_d \text{ in a way that reduces the loss function(10)} |
|  \quad \text{For } i=1, 2, ..., K do                                |
|  \quad \quad \text{Random sampling from noise distribution } P_z \text{ to obtain } Q \text{ noise } Z           |
|  \quad \quad \text{Input noise } Z \text{ into the generator, and update parameters } \theta_c \text{ in a way that reduces the loss function(9)} |
|  \quad \text{End for}                                              |
|  \text{End for}                                                      |

### 4. Experiments

#### 4.1. Data Set

The original data set uses the real transactions of cooperative bank's payment platform. These transactions start from January 01, 2015 to April 30, 2017, which contains 135,670,635 transaction records, but including only 40 marked fraudulent transactions. It can be seen that in the original data set, the ratio of fraud samples to normal samples reached 1:3400000, and the positive and negative samples were extremely unbalanced.

Due to the special time nature of the transaction records, the training set and the test set are divided according to time instead of randomly dividing the data set. In the data set provided by the bank, the transactions of 2015 and 2016 are divided into training data, including 28 fraudulent transactions, and normal transactions of these two years are downsampled to obtain about 200,000 normal samples; use transactions from January, 2017 to April, 2017 as test data, which contains 12 fraudulent transactions, and downsample the normal transaction data in January to obtain about 1000,000 samples. The data set used by the fraud detection model is as follows:

Table 2. Experiment dataset

| Dataset   | Time       | Num of normal data | Num of fraudulent data |
|-----------|------------|--------------------|------------------------|
| Train set | 2015.01-2016.12 | 202155            | 28                     |
| Test set  | 2017.01    | 994666             | 12                     |
| All       | 1196821    |                    | 40                     |

#### 4.2. Data Preprocessing

Some fields of the transaction records have high null-value ratio or do not contain useful information, so we need to extract valid information from these transactions. The relevant fields are extracted as shown in Table 3.
### Table 3. Useful fields of transactions

| Fields           | Description         |
|------------------|---------------------|
| Evt_id           | Event id            |
| Trans_type       | Type of transaction|
| Merchat_cd       | Code of merchant    |
| Order_commit_dt  | Date when commit the order |
| Order_commit_tm  | Time when commit the order |
| Acct_id_type     | Type of account     |
| Trans_acct_id    | Account id          |
| Trans_amt        | Amount of transaction |

In addition, the fields, order_commit_dt and order_commit_tm, are also processed and merged into timestamps.

#### 4.3. Experiment Parameters

The network is designed based on the GAN algorithm. The generator and the discriminator all adopt 6 fully connected layers, and each layer contains 16 neural nodes. Using the loss function described in Section 3.3, the learning rate is set to multiple values. In order to balance the generator and the discriminator during training, update the generator once and the discriminator K times in one iteration, and K is set to 10. The time window for supplemental features was selected for 24 hours, 7 days, and 30 days, respectively.

The GAN structure is shown as Table 4.

### Table 4. Network structure

| Generator                                      | Discriminator                          |
|------------------------------------------------|----------------------------------------|
| Fully-connected layer, 5 neural nodes, LRELU   | Fully-connected layer, 17 neural nodes, LRELU |
| Fully-connected layer, 16 neural nodes, LRELU  | Fully-connected layer, 16 neural nodes, LRELU |
| Fully-connected layer, 16 neural nodes, LRELU  | Fully-connected layer, 16 neural nodes, LRELU |
| Fully-connected layer, 16 neural nodes, LRELU  | Fully-connected layer, 16 neural nodes, LRELU |
| Fully-connected layer, 16 neural nodes, LRELU  | Fully-connected layer, 16 neural nodes, LRELU |
| Fully-connected layer, 16 neural nodes, LRELU  | Fully-connected layer, 3 neural nodes, SOFTMAX |

#### 4.4. Experiment Result and Analysis

Run the above model on a Centos 7.2 virtual machine with 4 CPUs and 16G memory, and installed python 3.6.5. and Tensorflow 1.12.0. In the experiment, the models were run under different learning rates and batch sizes, and loss functions of different models were obtained. The learning rate selects 3e-5 and 5e-5 respectively, and iterates 10000 times. The values of loss function of the generator and the discriminator are shown in figure 3,4, which show that the model with a learning rate of 5e-5 converges faster and the final value of loss function is relatively lower. In addition, the Batch size is selected from 6, 8 and 10 for comparison. The figure 5 and 6 is obtained. It can be seen that the convergence speed is almost the same in these three experiments, and the value of loss function of the discriminator with a Batch size of 10 is lower.
Figure 3. Generator loss of different learning rate

Figure 4. Discriminator loss of different learning rate

Figure 5. Generator loss of different batch size

Figure 6. Discriminator loss of different batch size

The goal of the proposed model is to solve the problem of unbalanced normal samples and fraudulent samples in the fraud detection models. In order to verify the effectiveness of the GAN model, the fraud data generated by the model is added to the original data set, and the detection effect of the model obtained by the different data set training on the fraud data is compared. The fraud detection model uses a commonly used random forest model [6]. The random forest model was used as the criterion for judging. In the experiment, the simulated fraud data was trained in the random forest model according to the ratio of 1:1 and 1:2, and then the obtained model was tested on the test set. The fraud sample identification accuracy (Sensitivity), normal sample recognition accuracy (Specificity), G-mean and recall rate (Recall) were used as evaluation indicators of the model. The table 6 shows that by merging the simulated samples into the training set, the recognition accuracy rate of the fraud samples can be effectively improved, and the G-mean value can reach 0.4070 as the input ratio increases.

Table 5. Confusion matrix

|                      | Predict to normal | Predict to fraud |
|----------------------|-------------------|------------------|
| Real mark is normal  | TP                | TN               |
| Real mark is fraud   | FP                | FN               |
Table 6. The result of fraud detection model

| Input ratio | Original training set | Augmented training set |
|-------------|-----------------------|------------------------|
|             | 1:1                   | 1:2                    |
| Sensitivity | 0.0833                | 0.125                  |
| Specificity | 0.9972                | 0.9816                 |
| G-mean      | 0.2882                | 0.3403                 |
| Recall      | 0.08333               | 0.125                  |

5. Conclusion
This paper proposes a data simulation algorithm based on generative adversarial networks, which is used to simulate fraudulent transactions. On the basis of GAN, the model increases the classification results of the discriminator: simulated data, real but fraud data and real and normal data. When verifying the effect of the algorithm, the discriminator of GAN is first used to discriminate the simulation data. In addition, the simulation data is mixed with the real data to train a valid fraud detection model which is approved by the bank. Then use this detection model to capture fraudulent transactions in the real transactions of the bank. Experiments show that the method of simulating data using our data enhancement algorithm is effective. In the future, the simulation methods of more attribute fields including the necessary attributes will be further studied.

6. References
[1] Goodfellow I J, Pouget-Abadie J and Mirza M, et al. 2014 Generative adversarial nets. *International Conference on Neural Information Processing Systems*. pp 2672-80.
[2] Arjovsky M and Bottou L 2017 Towards Principled Methods for Training Generative Adversarial Networks. Stat.
[3] Arjovsky M, Chintala S and Bottou L 2017 Wasserstein generative adversarial networks. *34th International Conference on Machine Learning* pp 298-321.
[4] Salimans T, Goodfellow I and Zaremba W, et al 2016 Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems* pp 2234-42.
[5] Xu J Q, Hong L P, Zhu H B and Zhao H 2018 Generative Adversarial Networks for the Classification of Lung Nodules Malignant. *Journal of Northeastern University(Natural Science)* pp 1556-61
[6] Bauder R A and Khoshgoftaar T M 2018 Medicare fraud detection using random forest with class imbalanced big data. *IEEE 19th International Conference on Information Reuse and Integration for Data Science* pp 80-87.