A Data Simulation Method of Bank Fraud Transaction Based on Flow-Based Generative Model

Xiaoguo Wang and Yin Zhang
Department of Computer Science, School of Electronic and Information Engineering, Tongji University, Zhixin Building, 4800 Cao’an Road, Jiading District, Shanghai, China
Email: 1930790@tongji.edu.cn

Abstract. In order to solve the problem of category imbalance caused by the shortage of bank fraud transaction data, this paper proposes a bank fraud transaction data simulation method based on flow-based generative model. On the basis of the flow-based generative model and the real fraudulent transaction data of the bank, this method designs a generative model suitable for the bank data, and learns the distribution of the real data through the generators $G$ and $G^{-1}$. The experimental results show that mixing the generated simulated data and real business data in a certain proportion to train the fraud detection model can improve the detection effect of the model to a certain extent.

1. Introduction
E-banking is an important part in banking business. The fraudulent transactions carried out by using electronic banking business loopholes or taking technical means have seriously damaged the property security of banks and customers. Fraud detection through technical means is faced with the problem that the amount of fraudulent transaction data is much smaller than the amount of normal transaction data, and the number of positive and negative samples is seriously unbalanced, which seriously affects the authenticity and effectiveness of detection. Establishing a data simulation model to generate new samples for the category with a small number of samples (minorities) is an effective method to solve the problem of data imbalance. Fraudulent transactions in electronic banks have their unique distribution characteristics. The flow-based generative model can simulate its probability distribution and obtain hidden layers with rich semantic attributes, and then through sampling, simulated fraud data can be obtained. In this paper, a simulation method of e-bank fraudulent transaction data based on flow-based generative model is studied.

2. Related Work
2.1. Generative Model
Generative model is a kind of important model in probability statistics and machine learning. It refers to a series of models for randomly generating observable data [1]. Constructing a generative model is to fit a given data sample with a known probability model, to obtain the distribution $q_{\theta}(x)$. Discrete distribution can be written out directly; Continuous distribution can be expressed by simple distribution (such as Gaussian distribution) and integral. As shown in equation (1) [2], $q(z)$ is generally a standard Gaussian distribution, $q(x|z)$ is a conditional Gaussian distribution or Dirac distribution.
\[ q(x) = \int q(z)q(x|z)\,dz \] (1)

Using the idea of maximum likelihood to maximize the objective function, the true data distribution \( \hat{p}(x) \) can be obtained, as shown in equation (2).

\[ E_{x \sim \hat{p}(x)}[\log q(x)] \] (2)

2.2. Flow-Based Generative Model

The flow-based generative model is different from the previous generative models (autoregressive models, VAE and GAN, etc.), it directly faces the probability calculation of the generative model, and its conversion is reversible. Assuming there are two different distributions of \( A \) and \( B \), it finds a duplex path between the distributions of \( A \) and \( B \). To achieve the conversion between \( A \) and \( B \) distributions, there are currently three major developments in flow-based generative models: NICE [3] realizes the reversible solution from \( A \) distribution to Gaussian distribution; RealNVP [4] implements the reversible solution from \( A \) distribution to conditional non-Gaussian distribution; GLOW [5] realizes the reversible solution from \( A \) distribution to \( B \) distribution.

In the flow-based generative model, generally \( q(x|z) \) is Dirac Distribution \( \delta(x - g(z)) \), and \( g(z) \) is reversible, \( x = g(z) \iff z = f(x) \), \( q(z) \) is the standard Gaussian Distribution.

It can be obtained by performing integral transformation \( z = f(x) \) on formula (1) (\( D \) is the dimension of \( z \)):

\[ q(x) = \frac{1}{2\pi^D/2} e^{-\frac{1}{2}\|f(x)\|^2} \left| \text{det} \left( \frac{\partial f}{\partial x} \right) \right| \] (3)

Using the idea of maximum likelihood, we can get the optimization goal:

\[ \log q(x) = -\frac{D}{2} \log (2\pi) - \frac{1}{2} \|f(x)\|^2 + \log \left| \text{det} \left( \frac{\partial f}{\partial x} \right) \right| \] (4)

When \( f \) is reversible and easy to invert, and the corresponding Jacobian Determinant is easy to calculate, after solving this optimization goal, randomly sample a \( z \), then generate a sample through the inverse \( g \) of \( f \) to get the generative model.

3. Simulation Algorithm of Bank Fraud Transaction Data Based on Flow-Based Generative Model

Aiming at the imbalance of sample data faced by the training of e-banking transaction fraud detection model, this paper uses a data simulation method to generate new samples by simulating fraud transactions and increase the number of fraud samples. This paper proposes a bank fraud transaction data simulation algorithm based on the flow-based generative model. With the help of the Jacobian Matrix, a reversible function is learned, and the simple distribution is gradually restored to a complex real data distribution through the sequential reversible function transformation. The framework of the flow-based generative model is shown in figure 1.

![Figure 1. Framework of flow-based generative model.](image-url)

3.1. Algorithm Principle

Define the generator \( G \) as a network model that defines a probability distribution \( P_G \). Assuming an input variable \( z \), the output after the input network \( G \) is \( x = G(z) \). Suppose \( z \) follows a very simple distribution, such as the Standard Normal Distribution, denoted by \( \pi(z) \). After passing through the network \( G \), the generated \( x \) may obey a very complex distribution, that is \( P_G(x) \). Finally, what we need to get in this paper is a generator network model \( G \), in which the pumped \( P_G(x) \) defined by \( G \) is as close as possible to the \( P_{data}(x) \) of real data.
To this end, combined with the idea of maximum likelihood. If m sample points \( \{x^1, x^2, x^3, \ldots, x^m\} \) are sampled from them, then the training goal of this paper is shown in equation (5), which is to maximize the similarity of \( P_G(x) \) and \( P_{data}(x) \), equivalent to minimizing the KL divergence of the two, as shown in formula (6).

\[
G^* = \arg \max_G \Sigma_{i=1}^m \log P_G(x^i) \tag{5}
\]

\[
G^* \approx \arg \min_G \text{KL}(P_{data}||P_G) \tag{6}
\]

According to the variable substitution theorem [6], suppose there is a distribution \( \pi(z) \), and \( x = f(z) \), then this \( x \) also forms a distribution \( p(x) \). For an one-dimensional distribution, we can write out \( p(x')|\Delta x = \pi(z')|\Delta z \), assuming that \( \Delta \) is extremely small, we can get:

\[
p(x') = \pi(z') \left| \frac{dz}{dx} \right|
\]

For a two-dimensional distribution, we can write out:

\[
p(x') \left| \begin{vmatrix}
\Delta x_1 \\
\Delta x_2
\end{vmatrix} \right| = \pi(z') \Delta x_1 \Delta x_2
\]

Assuming that \( \Delta \) is extremely small, we have:

\[
p(x') \left| \begin{vmatrix}
\frac{\partial x_1}{\partial z_1} & \frac{\partial x_1}{\partial z_2} \\
\frac{\partial x_2}{\partial z_1} & \frac{\partial x_2}{\partial z_2}
\end{vmatrix} \right| = \pi(z')
\]

Combining the knowledge of Jacobian matrix, we can get:

\[
p(x')|\det(f_{ij})| = \pi(z')
\]

Thus, equation (7) can be obtained by combining equation (5).

\[
P_G(x^i) = \pi(z^i)|\det(I_{G^{-1}})| \tag{7}
\]

where \( z^i \) is the result of bring \( x^i \) into \( G^{-1}z^i = G^{-1}(x^i) \).

Taking the logarithm of equation (7), we can get:

\[
\log P_G(x^i) = \log \pi(G^{-1}(x^i)) + \log |\det(I_{G^{-1}})| \tag{8}
\]

To maximize it, the design requirements of the generator network model \( G \): \( \det(I_G) \) is easy to calculate; \( G \) is reversible and easy to reverse.

3.2. Model Design

3.2.1. Flow. Because the requirement that \( \det(I_G) \) should be easy to calculate, and the network \( G \) should be easy to inverse, the ability of \( G \) is limited. The flow-based generative model uses a combination of multiple \( G \), just like running water. Such a process becomes a ‘flow’, which greatly improves the ability of the network. Suppose that \( K G \)'s is superimposed, and the resulting distribution \( P_K(x^l) \) is shown in formula (9).

\[
P_K(x^l) = \pi(z^l) \left| \begin{vmatrix}
\det(I_{G^{-1}}) \\
\ldots \\
\det(I_{G^{-1}})
\end{vmatrix} \right|
\]

Taking the logarithm of formula (9), we can get:

\[
\log P_K(x^l) = \log \pi(z^l) + \Sigma_{k=1}^K \log |\det(I_{G^{-1}})| \tag{10}
\]

where \( z^l \) is the result of \( x^l \) continuously taking \( G_k^{-1} \), \( \ldots, G_1^{-1}, z^l = G_1^{-1}(\ldots G_k^{-1}(x^l)) \).

3.2.2. Coupled Layer. In the specific design of the structure, first consider the case of one \( G \). According to the observation equation (8), in fact, the model trains \( G^{-1} \), and \( G^{-1} \) is inverted to get the
generative model $G$ to generate simulation data. The right side of the equation of equation (8) is obtained by the sum of $\log(\pi(G^{-1}(x)))$ and $\log(\det(\mathbf{J}_{G^{-1}}))$. Among them, because $\pi$ is a standard Gaussian distribution, it is known that when $z^{1} = G^{-1}(x^{1})$ is 0, $\log(\pi(G^{-1}(x)))$ will get the maximum value. But in this case $\mathbf{I}_{G^{-1}}$ will be a 0 matrix. That means, $\det(\mathbf{I}_{G^{-1}})$ will be 0 and $\log(\det(\mathbf{I}_{G^{-1}}))$ will be $-\infty$. Therefore, when finding the maximum, you need to consider the influence of both $\log(\pi(G^{-1}(x)))$ and $\log(\det(\mathbf{I}_{G^{-1}}))$.

The structure used here is called the coupling layer. The design of the network $G$ starts from the requirement that the $\det(\mathbf{J}_{G})$ is easy to calculate. According to the fact that the determinant of a triangular matrix is equal to the product of diagonal elements, if the Jacobian matrix of $G$ is a triangular matrix, the value of its determinant can be easily calculated. The $z$ dimension of the $D$ dimension is divided into two parts on average, the first $d$ dimension of $z$ is a group $\{z^{1}, z^{2}, ..., z^{d}\}$, and the last $d$ dimension of $z$ is a group $\{z^{d+1}, z^{d+2}, ..., z^{D}\}$.

Then take the following transformation:

\[
\begin{align*}
x_{\text{isd}} &= z_{\text{isd}} \\
\beta_{i>d} &= F(z_{\text{isd}}) \\
\gamma_{i>d} &= H(z_{\text{isd}}) \\
x_{i>d} &= \beta_{i} z_{i} + \gamma_{i}
\end{align*}
\]  

(11)

Among them, $F$ and $H$ can be arbitrarily complex functions.

The method of finding the inverse of the coupling layer is as shown in equation (12), which satisfies the condition of easy inversion.

\[
\begin{align*}
z_{\text{isd}} &= x_{i} \\
z_{i>d} &= \frac{x_{i} - \gamma_{i}}{\beta_{i}}
\end{align*}
\]  

(12)

When the matrix is divided into blocks, it is easy to find the value of the determinant of $G$’s Jacobian matrix as shown in equation (13).

\[
\det(\mathbf{J}_{G}) = \frac{\partial x_{d+1}}{\partial z_{d+1}} \frac{\partial x_{d+2}}{\partial z_{d+2}} ... \frac{\partial x_{D}}{\partial z_{D}} = \beta_{d+1} \beta_{d+2} ... \beta_{D}
\]  

(13)

3.2.3. Superimposed Interlaced Scrambling Layer. Combine these coupling layers to form a complete $G$. The first part of transformation (11) is trivial, so the whole transformation cannot reach very strong nonlinearity. In order to obtain an extraordinary transformation, we can consider randomly disrupting or reversing the order of the input dimensions before each additive coupling, or simply exchanging the positions of the two parts directly, so that the information can be fully mixed, as shown in equation (14).

\[
\begin{align*}
x^{(1)}_{\text{isd}} &= z_{\text{isd}} \\
\beta^{(1)}_{i>d} &= F_{1}(z_{\text{isd}}) \\
\gamma^{(1)}_{i>d} &= H_{1}(z_{\text{isd}}) \\
x^{(1)}_{i>d} &= \beta^{(1)}_{i} z_{i} + \gamma^{(1)}_{i}
\end{align*}
\]  

\[
\begin{align*}
x^{(2)}_{\text{isd}} &= z_{\text{isd}} \\
\beta^{(2)}_{i>d} &= F_{2}(x^{(1)}_{i>d}) \\
\gamma^{(2)}_{i>d} &= H_{2}(x^{(1)}_{i>d}) \\
x^{(2)}_{i>d} &= \beta^{(2)}_{i} x^{(1)}_{i} + \gamma^{(2)}_{i}
\end{align*}
\]  

\[
\begin{align*}
x^{(3)}_{\text{isd}} &= x^{(2)}_{\text{isd}} \\
\beta^{(3)}_{i>d} &= F_{3}(x^{(2)}_{i>d}) \\
\gamma^{(3)}_{i>d} &= H_{3}(x^{(2)}_{i>d}) \\
x^{(3)}_{i>d} &= \beta^{(3)}_{i} x^{(2)}_{i} + \gamma^{(3)}_{i}
\end{align*}
\]  

(14)

3.2.4. Scaling Layer. At this point, $G^{-1}$ is obtained, and $G$ can be obtained by inversion. Because the design of network $G$ requires it to be reversible, the dimensions of the input random variable $z$ and the
output sample \( \mathbf{x} \) must be the same. When we specify \( \mathbf{z} \) as the standard Gaussian Distribution, it is distributed throughout the D-dimensional space, and \( \mathbf{D} \) is also the size of \( \mathbf{x} \). Although \( \mathbf{x} \) has D-dimension, it may not really spread throughout the D-dimensional space. In order to solve the problem of dimension waste here, a scale transformation layer is used, and a scale transformation is made for the features of each dimension encoded at last, namely \( \mathbf{z} = \mathbf{s} \otimes \mathbf{x}_i^{(n)} \), where \( \mathbf{s} = \{s_1, s_2, ..., s_D\} \) is also a parameter vector to be optimized (each element is non-negative). This \( \mathbf{s} \) vector can identify the importance of the dimension (the smaller the dimension is, the more important it is, the larger the dimension is, the less important the dimension is, and the proximity can be ignored), which plays the role of compressing the manifold.

The Jacobian Determinant of this scale transformation layer is no longer 1, and it can be calculated that its Jacobian Matrix is the diagonal matrix \( \text{diag}(s) \), so the value of its determinant is \( \prod_i s_i \), so according to equation (8), combined with the formula of standard Gaussian Distribution, the logarithmic likelihood can be obtained:

\[
\log P_G(x) \sim - \frac{1}{2} \| \mathbf{s} \otimes \mathbf{G}^{-1}(x) \|^2 + \sum_i \log s_i
\]  

(15)

This scale transformation can identify the importance of features. Let the prior distribution of \( \mathbf{z} \) be a standard Gaussian distribution, that is, each variance is 1. The variance of the prior distribution is also used as the training parameter, so that the variance after the training is large and small, and the smaller the variance, the larger the "dispersion" of the feature is. If the variance is 0, then the feature is always the mean of 0. The distribution of this dimension collapses to a point, so this means that the manifold is reduced by one dimension.

If the normal distribution with variance is written:

\[
P(z) = \frac{1}{(2\pi)^{D/2} \prod_i \sigma_i} \exp\left( -\frac{1}{2} \sum_i \frac{z_i^2}{\sigma_i^2} \right)
\]

(16)

If we plug the stream model \( z = \mathbf{G}^{-1}(x) \) into the upper equation (16) and take the logarithm, we can get:

\[
\log P_G(x) \sim - \frac{1}{2} \sum_i \frac{(\mathbf{G}^{-1})_i^2(x)}{\sigma_i^2} - \sum_i \log \sigma_i
\]

(17)

Contrast equation (15), there is \( s_i = 1/\sigma_i \). Therefore, the scale transformation layer is equivalent to taking the variance (standard deviation) of the prior distribution as a training parameter. If the variance is small enough, it can be considered that the manifold represented by this dimension collapses to a point, so that the dimension of the overall manifold is reduced by 1. It plays a role in reducing the dimension to a certain extent.

3.3. Model Realization

The bank fraud transaction data simulation model based on flow-based generative model is a kind of flow-based generative model. The specific implementation details are as follows: The model in this paper consists of multiple additive coupling layers, each of which is shown in formula (11), and its inverse is equation (12). Before coupling, the input dimensions need to be reversed so that the information is fully mixed. The last layer needs to add a scale transformation layer, and the loss function is the inverse of equation (15).

The additive coupling layer needs to divide the input into two parts. The model in this paper uses interleaved partitioning. The subscript is even number as the first part, and the subscript is odd number as the second part, in which any complex function \( F \) and \( H \) use multiple fully connected, a total of 4 additive coupling layers are coupled.

For the input, the model in this paper compresses the original data to between 0 and 1, and then adds a uniform distribution of noise \([-0.01,0]\) to the input. The addition of noise can effectively
prevent overfitting and improve the quality of the generated data. This is also a measure to alleviate the problem of dimensional waste.

4. Experiments and Results

4.1. Data Situation

In this paper, the real fraudulent transaction data of the cooperative bank is used as the original data, and it is used as the input of the data set in the flow-based generative model. The data set is based on individual mobile banking history transactions from January 2016 to August 2016. Because of the time specificity of the transaction records, the training set and the test set are divided according to the transaction time, with July 1, 2016 as the boundary.

4.2. Experimental Design

4.2.1. Data Pre-processing. Some fields in the transaction record have high null rate or do not contain useful information, so it is necessary to extract the effective information from the transaction information and extract the valid fields.

The different types of fields of the valid fields of the transaction data are processed appropriately. In this paper, the field types are divided into numeric, time and enumerated types. For the time-type field, the conversion time-stamp processing method is adopted; for the enumeration-type field, the Hash mapping processing method is adopted; finally, in order to ensure the fast and correct convergence of the generated model, the normalization processing is adopted for all fields, the specific fields and types are shown in table 1.

| Field Name           | Meaning                | Type                  |
|----------------------|------------------------|-----------------------|
| ACCOUNTNUMBER        | Payer account number   | Enumeration type      |
| PAYEACCOUNTNUMBER    | Payee account number   | Enumeration type      |
| TXTIME               | Time of transaction    | Time type             |
| AMOUNT               | Amount of transaction  | Numeric type          |

4.2.2. Model Parameter Configuration. In this paper, in the flow-based generative model, five hidden layers are set up in the fully connected layer, and each layer contains 100 neural nodes. Using the loss function described in Section 3.3, several groups of values are set for the training and learning rate. The model eventually needs to sample random numbers from the standard Gaussian distribution to generate samples. In the specific implementation, the annealing parameter of the model is selected as 0.75, and the quality of the generated model is optimal at this time.

4.3. Experimental Results and Analysis

4.3.1. Analysis of Experimental Results. Run the above model on Centos 7.2 operating system, configure 4-core CPU and 16G memory, and install python 3.6.5 and keras2.2.4. In the experiment, the models are run under different learning rate and batch size, and the loss functions of different models are obtained. The results of many experiments show that the model with the learning rate of $e^{-3}$ converges faster and the final loss function value is relatively low, and the model with the Batch size of 64 converges lower when the convergence speed is basically the same, as shown in figure 2.
4.3.2. Comparative Experimental Design. In order to evaluate the simulation effect of the flow-based generative model on bank fraudulent transaction data, using the fraudulent transaction detection model, the comparative experiments are designed as follows:

The data set includes the original data set and the enhanced data set.

The original data set selects all fraudulent transaction data from January 2016 to June 2016 as the training set.

The training set of the enhanced data set is: to add the fraud simulation data generated by the flow-based generative model to the training set of the original data set.

The test set of the original data set and the enhanced data set are the same, both are: all fraudulent transaction data from July 2016 to August 2016.

The training method is a fraud detection model training method that has been confirmed by the cooperative bank and has a good effect of fraud detection, that is, the Adacost+ Random Forest Joint Fraud Detection Model to train the fraud detection model.

In the contrast experiment, the training sets of the original data set and the enhanced data set are put into the fraud transaction detection model for training, and the two fraud detection models are called model 1st and model 2nd respectively. The features and parameters selected by training model 1st and model 2nd are consistent, and fraud detection experiments are carried out with the same test set. The fraud detection experimental results are compared, and the evaluation of the effect of simulation data generation is obtained.

4.3.3. Evaluation Index. In the real bank transaction scenario, the proportion of normal transaction is as high as 99%, the proportion of normal transaction and fraudulent transaction is very unbalanced, and the cost of a few types of samples is much higher than that of most types [7], therefore, the classification of unbalanced data sets tends to pay more attention to a few classes, that is, the classification results of fraud data, and the simple use of overall Accuracy to measure the effect of the model lacks reference value. Therefore, based on the confusion matrix (table 2), this paper selects the fraud sample recognition accuracy (Sensitivity), normal sample recognition accuracy (Specificity), G-mean, F1 Score and other indicators to evaluate the quality of the simulation data.

4.3.4. Results and Analysis. The training sets of the original data set and the enhanced data set are respectively put into the fraud transaction detection model for training, and the characteristics and parameters selected by model 1st and model 2nd are consistent during training, and fraud detection is carried out with the same test set, and the results are compared. The verification results of the simulation data generation effect based on the flow-based generative model are shown in table 3.
Table 2. Confusion matrix.

| True value   | Predicted value   |
|--------------|-------------------|
|              | Fraudulent transaction | Normal transaction |
| Fraudulent transaction | TP                | FN                |
| Normal transaction   | FP                | TN                |

Accuracy rate:

\[ OA = \frac{TN + TP}{TP + FP + FN + TN} \]

Fraud sample recognition accuracy rate:

\[ Sensitivity = \frac{TP}{TP + FN} \]

Non-fraud sample recognition accuracy rate:

\[ Specificity = \frac{TN}{TN + FP} \]

G-mean:

\[ G - mean = \sqrt{Specificity \times Sensitivity} \]

Recall:

\[ Recall = \frac{TP}{TP + FN} \]

Precision:

\[ Precision = \frac{TP}{TP + FP} \]

F1 Score:

\[ F1\ Score = \frac{2 \times Recall \times Precision}{Recall + Precision} \]

Table 3. Evaluation comparison of Adacost+ random forest joint model.

| Evaluation index | Original data set | Enhanced data set |
|------------------|-------------------|------------------|
| OA               | 0.89247           | 0.82857          |
| Sensitivity      | 0.66667           | 0.83333          |
| Specificity      | 0.89247           | 0.82857          |
| G – mean         | 0.77135           | 0.83095          |
| Recall           | 0.66667           | 0.83333          |
| Precision        | 0.00005           | 0.00003          |
| F1 Score         | 0.00010           | 0.00008          |

According to the experimental results: after the simulated fraud data generated by the flow-based generative model is mixed with real data into training, the performance of the fraud detection model is improved. Among them, the correct rate of fraud sample identification is increased by 0.25 times, the G-mean is 1.07 times that of the original, and the recall rate is increased by 0.25 times. The main reason is that the fraudulent data in the original data is seriously insufficient, the category distribution of the data set is extremely uneven, and the isolated samples may be submerged by noise. The flow-based generative model uses a reversible function to extract the information of a small number of samples and generate simulation data near the real data, which reasonably and effectively improves
the unbalanced degree of sample distribution and enriches the potential fraud characteristics of the data. As a result, the detection ability of the model is improved.

5. Conclusion
In order to solve the problem of category imbalance caused by the shortage of bank fraud transaction data, this paper proposes a bank fraud transaction data simulation method based on flow-based generative model. Based on the flow-based generative model and the characteristics of bank transaction data, this method learns the distribution of real data through generators $G$ and $G^{-1}$. $G^{-1}$ encodes the input $x$ as a hidden variable $z$, and through continuous learning, makes $z$ obey the standard Gaussian distribution. Since $G^{-1}$ is reversible, after $G^{-1}$ training is completed, $G$ can be obtained at the same time to complete the construction of the generative model.

In this paper, we first take the real data as samples to generate the simulation data that meet the convergence conditions and the expected effect, and then mix the real data and put it into the anti-fraud model to verify the effect. The experimental results show that the fraud detection ability of the model with augmented data is better than that of the model with unmixed simulated data in terms of fraud sample identification accuracy, G-mean, recall and other indicators. In the next stage, we will expand the attribute fields and make an in-depth study on the simulation methods of different attribute fields to further improve the quality of the simulation data.

References
[1] Guo L, Zhao J and Shi Z 2010 Image classification method based on the fusion of generative model and discrimination method Acta Electronica Sinica (05) 1141-1145.
[2] Su J 2018 The basic concept and implementation of NICE: flow model of slender https://spaces.ac.cn/archives/5776.
[3] Dinh L, Krueger D and Bengio Y 2015 NICE: non-linear independent components estimation 3rd International Conference on Learning Representations (ICLR 2015) (San Diego, CA, USA).
[4] Dinh L, Sohl-Dickstein J and Bengio S 2016 Density estimation using real NVP arXiv preprint arXiv:1605.08803.
[5] Diederik P K and Prafulla D 2018 Glow: Generative flow with invertible 1x1 convolutions arXiv preprint arXiv:1807.03039.
[6] Pouso R 2010 Change of variable in Riemann integration revisited arXiv preprint arXiv:1010.0554v1.
[7] Menardi G and Torelli N 2014 Training and assessing classification rules with imbalanced data Data Mining and Knowledge Discovery (1) 92-122.