



A Transformer-based Model Integrated with Feature Selection for Credit Card Fraud Detection

Miao Yuan

Southwest Jiaotong University

ABSTRACT

Credit card fraud has grabbed a lot of attention in data science domain in the past several years. Most previous works tackled this problem by using machine learning methods equipped with hand-crafted features. In this paper, we proposed a novel multi-stage method combining both feature selection by a machine learning method, as well as a transformer-based model as a classifier. Concretely, we firstly extracted the top-k important features from the original dataset, where they are further used to train the self-designed deep learning model. Experimental results shown that our model can achieve better results compared to the baseline models. Since we also incorporated transformer layer into our deep learning model, we also tested its performance and the results proves the effectiveness.

CCS CONCEPTS

• Security and privacy; • Human and societal aspects of security and privacy; • Economics of security and privacy → Computing methodologies; • Machine learning; • Machine learning approaches; • Classification and regression trees;

KEYWORDS

Feature selection, Transformer, Deep learning, Credit card fraud

ACM Reference Format:

Miao Yuan. 2022. A Transformer-based Model Integrated with Feature Selection for Credit Card Fraud Detection. In *2022 7th International Conference on Machine Learning Technologies (ICMLT) (ICMLT 2022)*, March 11–13, 2022, Rome, Italy. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3529399.3529429>

1 INTRODUCTION

Credit card fraud increased dramatically in the past decade, as credit card became the most popular paying method. Fraudsters pretend to be the normal user and use their credit cards to make transactions, which cost billions of dollars every year.

The traditional way to detect fraudulent transactions is to use machine learning [1], such as decision tree [2], SVM [3]. However, due to the large scale of the dataset, machine learning methods is not able to achieve a promising result with respect to a high dimensional feature space. Although machine learning methods

seems to gain the high accuracy in terms of the fully-sized dataset where fraudulent cases count for less than 1% in total, they still suffer from the data imbalance problem, which means the model still achieves a high accuracy even if the method predicts all cases as not fraudulent. As a result, most of the previous attempts [4-7] to detect credit fraud use deep learning models. Some use Long short term memory network (LSTM) [1] to solve the problem, the others also use convolutional neural network [2] and Bayesian neural network [3]. In this paper, these aforementioned models are selected as our baselines, and compared with the proposed model on a well-studied dataset derived from Kaggle competition [8].

In this project, our proposed approach is a combination of machine learning algorithms and deep learning methods. In the first part, features with low importance will be filtered out by Random Forest regressor. While in the second part, we trained a deep learning method based on Transformer structure.

We tested our approach on a public data set which contains 492 fraud samples out of 284,807 transactions. The accuracy matrix and ROC curve was derived to evaluate the performance of our model and baselines.

2 RELATED WORK

[6] propose a method using decision tree which combines Luhn's algorithm and Hunt's algorithm. Their method will check whether the billing address is the same as shipping address, the card number is valid, and finally take outlier detection.

[7] build a neural networks with built-in time and memory components such as Long Short-term memory. In this paper, they conduct a sensitivity analysis of model parameters with regard to performance in fraud detection. They also present a framework for parameter tuning of Deep Learning models for credit card fraud detection.

The dataset is imbalanced with little fraudulent transactions, [9] propose a method using denoising auto encoder and oversampling to address imbalanced dataset. Their experiment shows that their method have a better accuracy than fully connected neural network.

3 METHOD

3.1 Overview

In our task, we use a pipeline approach for model construction. As shown in Figure 1, our model is divided into two parts. The first part is a feature filter based on a machine learning algorithm, which serves to select the most important features in the data for training, while the less important features are filtered out to reduce the noisy features of the model; the second part is a deep learning based neural network, in which we use a feature extractor based on the Transformer structure for the data optimization and learning, and a binary classifier is introduced in the last layer of this network for prediction whether the data is fraudulent or not.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ICMLT 2022, March 11–13, 2022, Rome, Italy

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9574-8/22/03...\$15.00

<https://doi.org/10.1145/3529399.3529429>

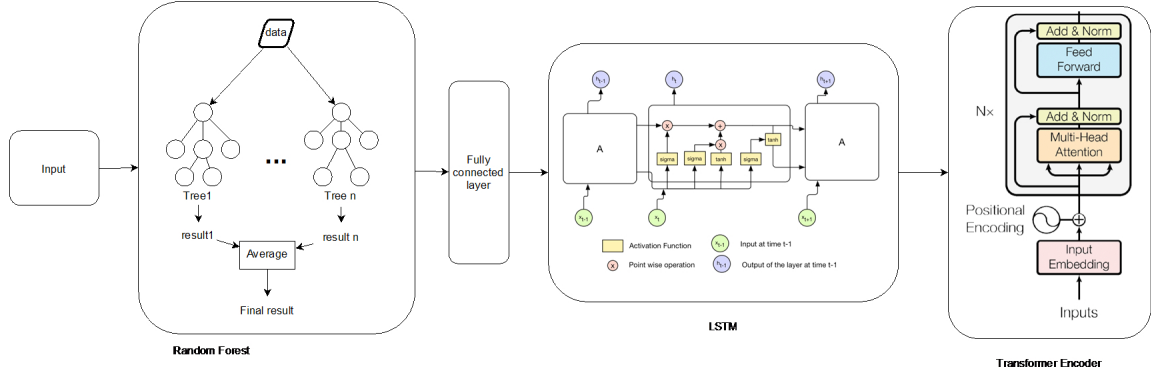


Figure 1: structure overview

3.2 Feature filter layer

In this section, we perform feature filtering on the input data. The principle is to train a regression algorithm based on the training set so as to assign weights to each of the input features. We trained a random forest regression algorithm [7] to sort the features according to their weights. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. In our task, we used the following steps to execute the random forest regression algorithm: we first randomly selected K data samples from the training data to build their corresponding decision trees. Notice that a single decision tree algorithm takes all sample points in the data to build a tree, while in our random forest algorithm, we select only the K most representative points for the construction. Then, we set the hyperparameter N as the number of decision trees in the random forest and iterate the above steps N times to construct the random forest. For each feature, we put it into each decision tree and predict it separately, and finally we take the average of these predicted values as the score of our random forest, and these scores are importance weights that will be further used in the following neural network.

3.3 Feature extraction layer

In this layer, we construct a deep learning based neural network model to train the features selected in 3.2. Specifically, our neural network is divided into four layers. Assuming that the filtered feature x_f in 3.1 above is our model input, we first perform the initial feature extraction using a fully connected neural network.

$$x_{ful} = \sigma(W_{ful} \cdot x_f + b_{ful}) \quad (1)$$

Where W_{ful} is the weight matrix, b_{ful} is the biased matrix. σ can be any activation function, we use \tanh in our task. After that, in order to extract the sequential features from the model, we introduced a bidirectional LSTM network [8] to map the features. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. It is well-suited to classifying, processing and making predictions based on timeseries data, since there can be lags of unknown duration between important events in a time

series.

$$x_{hid}, x_{out} = \text{BiLSTM}(x_{ful}) \quad (2)$$

Where x_{hid} is the hidden state of the last time step, and x_{out} denotes the output for the entire sequential features. In our case, we only leverage the x_{out} as an input for the further Transformer layer [9].

Then, we use Transformer model to map x_{out} to higher dimension. Transformer model is a feature extractor based on multi-headed attention mechanism, proposed by Google [9]. This allows Transformer to compute the weighted representation of all input features per unit of time. Compared with traditional feature extractors such as RNN, CNN, etc., this feature that Transformer can compute in parallel greatly reduces the computation cost and time cost, thus improving the speed of model training while improving the model capture of sequential features.

It is worth noting that Transformer model is an end2end encoder-decoder structure, in our task, we only use the encoder part of the Transformer to extract the features. Specifically, given the feature matrix x_{out} , we first compute its weighted representation through the self-attention mechanism as follows:

$$a_{out}^i = \sigma(W_{out}^i \cdot x_{out}^i + b_{out}^i) \quad (3)$$

$$w_{out}^i = \frac{e^{a_{out}^i}}{\sum_{j=1}^T e^{a_{out}^j}}$$

$$h_{out} = \sum_i^T a_{out}^i \cdot w_{out}^i$$

Where a_{out}^i denotes the attention score of i -th token, while w_{out}^i represents the normalized weight of i -th token. H_{out} is the summed representation for feature matrix x_{out} .

Since each Transformer block is composed of several identical weighted representations H_{out} gained from the self-attention mechanism aforementioned, we therefore aggregate each independent weighted representation as H'_{out} as our output of each transformer block:

$$H'_{out} = \text{Normalization} \left(\text{addition} \left(H_{out}^1, H_{out}^2, \dots, H_{out}^i \right) \right) \quad (4)$$

Where i is the number of heads in the transformer block, addition is an element-wise operation that adds each weighted representation alongside the first dimension. Normalization is a scaled layer normalization operation after the addition operation, we would not illustrate the detailed process here due to the limited space.

Then, the H'_{out} is fed into a position-wise fully-connected feed forward network for a higher dimensional feature mapping:

$$H_{out}^f = \text{FeedForward}(H'_{out}) \quad (5)$$

The output of the above layer will be transferred to the same layer as it did in Equation 6 to generate the output of the entire transformer block. We cast the operations from Equation 3-7 as Transformer block. Finally, a certain figure of Transformer blocks will be connected to produce the final output $H_{transformer}$:

$$H_{transformer} = \text{Transformer block}_{1,2,\dots,N}(X_{out}) \quad (6)$$

Where N is the number of the transformer blocks used in our task.

3.4 Classification layer

In this layer, we classify each given samples by using a binary classifier. Concretely, we feed the output of transformer encoder into a feed forward network for a dimensional transformation [10].

$$f = \text{Feed Forward}(H_{transformer}) \quad (7)$$

Then, each logit of the training sample will be calculated via a sigmoid classifier alongside their first dimensions, i.e., their batch size. In the inference stage, the sample with a logits higher than and equals to 0.5 will be classified as 1, i.e., fraud. On the other hand, the sample with a logits lower than 0.5 will be classified as 0, i.e., non-fraud.

$$p = \text{sigmoid}(f) \quad (8)$$

3.5 Training

In our task, we utilize a binary cross entropy loss as our loss function to optimize the model mentioned above:

$$\mathcal{L} = \sum_n \sum_i^n p_i' \cdot \log(p_i) \quad (9)$$

4 EXPERIMENT

4.1 Dataset

We adopt a public research dataset to conduct our experiments where the dataset contains transactions made by credit cards in September 2013 by European cardholders [11]. This dataset presents transactions that occurred in two days, which contains 492 fraud samples out of 284,807 transactions. In this sense, the dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. All the sensitive information has been desensitized due to confidentiality issues. The features $V1, V2, \dots, V28$ are the principal components obtained with PCA, the only features which have not been transformed with PCA are Time and Amount. Feature Time contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature Amount

is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature Class is the response variable and it takes value 1 in case of fraud and 0 otherwise [12].

The figure 2 reflects the cosine similarity between different dimensions of the data, and the brighter color indicates the stronger correlation. From the figure 2, we can see that the correlation between features $V1$ and $V2$, and between features $V7$ and Amount is very strong, so we visualize them separately, as shown in the later figure 3, and can analyze the similarity and overlap of their feature spaces.

As recommended by the creator, we evaluate the experiment using accuracy metric.

4.2 Baseline models

We compared our model with several baselines. Autoencoder [6] proposed a denoising autoencoder neural network (DAE) algorithm which can not only oversample minority class sample through misclassification cost, but it can denoise and classify the sampled dataset. Convolutional neural network [13] that have used in this project are 2 layer MLP, and two 1D Conv layer with kernel size equals to 29. We implemented a bi-directional LSTM model as they did in [14, 15] to conduct our experiments. Similar to the conventional MLP models applied in various of deep learning area, we also implemented a MLP with hidden size equaling to 718 in our task for a comparison. The last baseline model is Bayesian-based neural network [3] which updates its weights via Bayes' rule. It is a potential choice for our choice.

4.3 Experimental results

We used Random Forest regressor to filter the feature with high importance. The result is shown in figure 4:

In the experiment, we tried several hyperparameters to find the best setting. We used multi-head attention because it allows the model to jointly attend to information from different representation space, which help the network extracting more features. But redundant heads can be negative to the model, therefore we want to find the number of heads that performs better. We tested heads number 1, 2, 4, 8, 16 and the result is as follow in Table 1:

The result suggests that 4 heads can yield highest accuracy. We set the number of heads to 4 and tested our approach with other baselines. The result is shown in Table 2:

To minimize error, the result is the average of 10 rounds. Our model has the highest Precision, Recall rate and F1 score on test set. It is worth noticing that because of the unbalanced dataset as shown in figure 5, all methods have similar Precision(99+), but how the fraud cases are classified is more important.

Recall rate can represent how many fraud cases are classified correctly. The recall rate of our model is 43.75, while the recall rate of AutoEncoder is 10.81. The ROC curve is as figure 6:

The ROC area under the curve(AUC) is used to quantify the classification accuracy of the model; the higher the area, the greater the difference between true positives and false positives, and the better the model is at classifying members of the training data set. An area of 0.5 corresponds to a model that does not perform better than random classification, and a good classifier will move as far away from it as possible. An area of 1 is ideal, and the closer the AUC is to 1, the better.

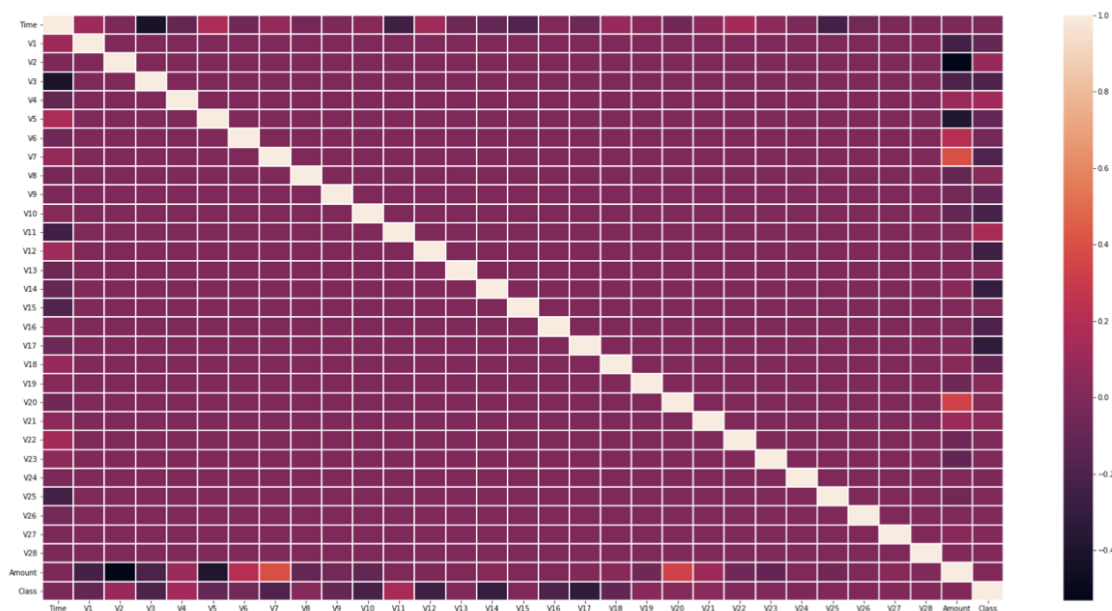


Figure 2: Correlation feature matrix heat map

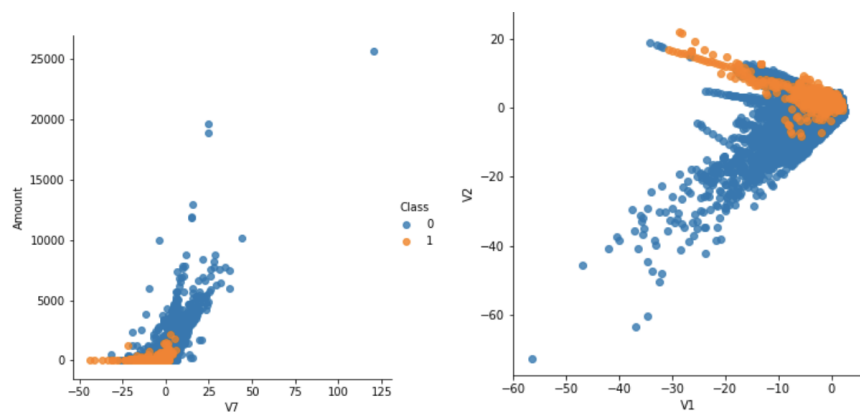


Figure 3: Feature correlation map of V1 and V2, Amount and V7

Table 1: Performance with different number of heads

	Accuracy on test set	Accuracy on training set
Head = 1	99.51	99.48
Head = 2	99.736	99.84
Head = 4	99.85	99.85
Head = 8	99.83	99.85
Head = 16	99.56	99.85

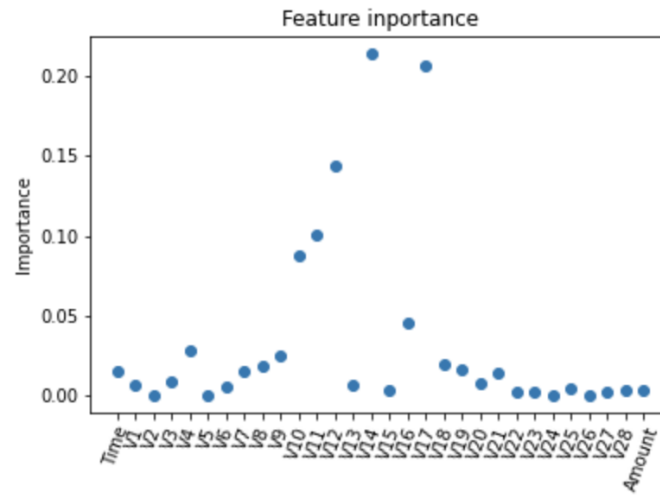


Figure 4: feature importance

Table 2: Comparison with baselines

	Precision	Recall	F1 score
Multi-layer Perceptron (MLP)	99.85	20.10	33.46
Bayesian neural network (BNN)	99.84	37.51	46.5
Convolutional neural network (CNN)	99.84	40.10	57.22
Long-short term memory (LSTM)	98.20	40.86	57.71
AutoEncoder	99.48	10.81	19.50
Our model	99.85	43.75	60.85

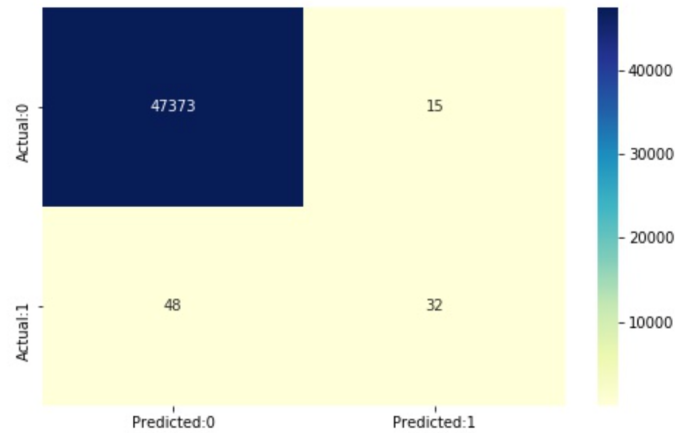


Figure 5: confusion matrix

5 CONCLUSION

In this paper, we proposed a multi-stage method to tackle the credit card fraud detection task in data science domain. We firstly extracted the top-k important features from the original dataset, where they are further used to train the self-designed deep learning

model. Experimental results shown that our model can achieve better results compared to the baseline models. Since we also incorporated transformer layer into our deep learning model, we also tested its performance and the results prove the effectiveness.

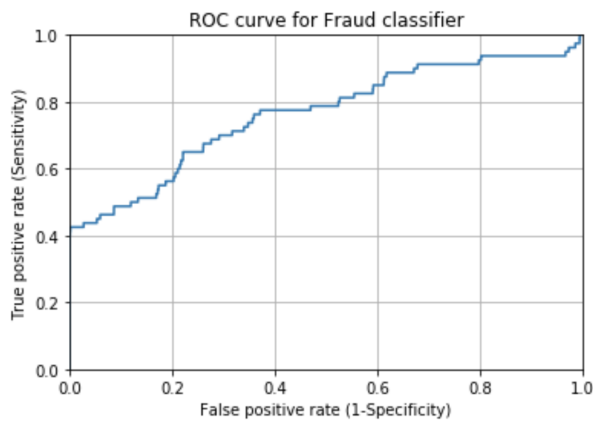


Figure 6: ROC

REFERENCES

- [1] Maja Puh, Ljiljana Brkić, "Detecting Credit Card Fraud Using Selected Machine Learning Algorithms", Information and Communication Technology Electronics and Microelectronics (MIPRO) 2019 42nd International Convention on, pp. 1250-1255, 2019.
- [2] S. Dhankhad, E. Mohammed and B. Far, "Supervised Machine Learning Algorithms for Credit Card Fraudulent Transaction Detection: A Comparative Study," 2018 IEEE International Conference on Information Reuse and Integration (IRI), 2018, pp. 122-125, doi: 10.1109/IRI.2018.00025.
- [3] Sun S, Zhang Z, Huang B C, *et al.* Sparse-softmax: A Simpler and Faster Alternative Softmax Transformation[J]. arXiv preprint arXiv:2112.12433, 2021.
- [4] Roy, A. , Sun, J. , Mahoney, R. , Alonzi, L. , Adams, S. , & Beling, P. . (2018). Deep learning detecting fraud in credit card transactions. 2018 Systems and Information Engineering Design Symposium (SIEDS). IEEE.
- [5] Cao, Jiarun, *et al.* "CONNER: A Cascade Count and Measurement Extraction Tool for Scientific Discourse." Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021). 2021.Lin, Y., & Wu, J. (2021). Detection of gravitational waves using Bayesian neural networks. Physical Review D, 103(6).
- [6] Zou J, Zhang J, Jiang P. Credit card fraud detection using autoencoder neural network[J]. arXiv preprint arXiv:1908.11553, 2019.
- [7] Shantanu Rajora, Dong-Lin Li, Chandan Jha, Neha Bharill, Om Prakash Patel, Sudhanshu Joshi, Deepak Puthal, Mukesh Prasad, "A Comparative Study of Machine Learning Techniques for Credit Card Fraud Detection Based on Time Variance", Computational Intelligence (SSCI) 2018 IEEE Symposium Series on, pp. 1958-1963, 2018.
- [8] Save, P., Tiwarekar, P., N., K., & Mahyavanshi, N. (2017). A Novel Idea for Credit Card Fraud Detection using Decision Tree. International Journal Of Computer Applications, 161(13), 6-9. <https://doi.org/10.5120/ijca2017913413>
- [9] Sylvester, EVA, Bentzen, P, Bradbury, IR, *et al.* Applications of random forest feature selection for fine-scale genetic population assignment. Evol Appl. 2018; 11: 153– 165. <https://doi.org/10.1111/eva.12524>.
- [10] Cao J, Ananiadou S. GenerativeRE: Incorporating a Novel Copy Mechanism and Pretrained Model for Joint Entity and Relation Extraction[C]//Findings of the Association for Computational Linguistics: EMNLP 2021. 2021: 2119-2126.
- [11] Graves A., Fernández S., Schmidhuber J. (2005) Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. In: Duch W., Kacprzyk J., Oja E., Zadrozny S. (eds) Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005. ICANN 2005. Lecture Notes in Computer Science, vol 3697. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11550907_126
- [12] Vaswani A, Shazeer N, Parmar N, *et al.* Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.
- [13] Sun S, Zhang Z, Huang B C, *et al.* Sparse-softmax: A Simpler and Faster Alternative Softmax Transformation[J]. arXiv preprint arXiv:2112.12433, 2021.
- [14] Shantanu Rajora, Dong-Lin Li, Chandan Jha, Neha Bharill, Om Prakash Patel, Sudhanshu Joshi, Deepak Puthal, Mukesh Prasad, "A Comparative Study of Machine Learning Techniques for Credit Card Fraud Detection Based on Time Variance", Computational Intelligence (SSCI) 2018 IEEE Symposium Series on, pp. 1958-1963, 2018.
- [15] Cao J, Wang C, Gao L. A joint model for text and image semantic feature extraction[C]//Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence. 2018: 1-8.