2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE) | 978-1-6654-1540-8/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICBAIE52039.2021.9389940

Credit Card Fraud Detection Using Autoencoder-Based Deep Neural Networks

Jiatong Shen*
Shanghai High School
Shanghai, China
dorashenshanghai@163.com

***Abstract*—It is now customary to pay by credit cards due to the ease of purchase and the ability to keep customers' credit history. At the same time, credit card fraud emerges as a widely-concerned issue in the electronic payment sector, urging the development of fraud detection techniques. In this paper, I develop a credit card fraud detection model using an autoencoder-based deep neural network. To ensure accuracy and robustness, I train two models on normal transactions and fraud transactions respectively. After that, I combine these two individual models into a hybrid model, which is the major innovation of this creation. ROC score along with the AUC value is the evaluation method here. It turns out that the mean of the AUC of the hybrid model is 0.9577, which is between that of the two individual models and shows the ensured accuracy and increased stability.**

Keywords- Credit card fraud, Neural networks, Autoencoder, L1 regularization

I. INTRODUCTION

Using a credit card is a considerably significant way of payment. The credit card transaction amount of China has risen from 19.7 trillion CNY, 2014, to 38.2 trillion CNY, 2018, and the overall trend is increasing [10]. Besides, it is statistically analyzed that credit card usage in the US has steadily increased from 2016 to 2018, rising from 18% to 23% [11].

However, transactions can be bothersome when fraud occurs. By means of Genetic Algorithm, SVM, HMM, and many other detection methods, lots of researchers dedicated themselves to deal with credit card fraud, in which many of them applied hybrid models. A hybrid model is the good complementary of each model included, where the disadvantages of a model alone can be diminished with the help of another model combined. Lei et al [1] proposed a supervised (SICLN) combining unsupervised (ICLN) learning network for credit card fraud detection, which increases stability and reduces training time. Patidar et al trained a three-layer backpropagation neural network combining genetic algorithms [2]. These contributions of the predecessors become my inspirations to create a hybrid model.

In my study, two unsupervised models are combined after being separately trained, which is different from those methods which combine solely supervised or supervised plus unsupervised models.

To establish the hybrid model, the reconstruction error, and the corresponding class types are extracted. After that, the values of the 'Reconstruction error' column are added and the sum is defined as the reconstruction error of the hybrid model. In the experiment phase, parameters such as the epochs are adjusted, aiming at yielding more desirable testing results. Model evaluations show that this hybrid model improves stability and maintains a similar accuracy to individual models.

The main contributions are as follows: First, it reduces the frequency of people bumping into fraud behaviors. Second, the result is meant to lead the progress of security and make people feel cared for, enhancing a sense of peacefulness and ensuring long-term development.

II. DATA DESCRIPTION

The data set is downloaded from www. Kaggle. com [3], which contains data of credit card transactions in September 2013 by European cardholders. It presents transactions that occurred in two days, with 492 frauds out of 284315 transactions, which is highly imbalanced. This fact will influence our choice of models for training.

The 'V1, V2…' columns are features of transactions, hiding users' identity or sensitive information. The 'Time' column shows seconds elapsed between a transaction and the first transaction. The 'Amount' column suggests the transaction amount. The 'Class' column represents the transaction type. 0 refers to normal transactions while 1 refers to fraud transactions.

TABLE I. MEAN AND STANDARD DEVIATION OF THE AMOUNT FOR THE TRANSACTIONS OF EACH CLASS

	Mean of amount	STD of amount
Normal	88.35	250.12
Fraud	122.21	256.68

Table 1 shows the mean of the 'Amount' and the standard deviation of the 'Amount' in terms of two different classes. The standard deviation of fraud transactions 'Amount' is slightly larger. The mean of the 'Amount' of fraud transactions is larger, from which we can generalize that the transaction amount of fraud transactions is larger. The average 'Amount' of fraud transactions is about 38.3% larger than that of normal transactions, indicating that the 'Amount' of transactions can be a helpful feature for classification.

978-1-6654-1540-8/21/$31.00 ©2021 IEEE

III. DATA EXPLORATION

The following figure shows the relationship between time of transaction and amount by class.

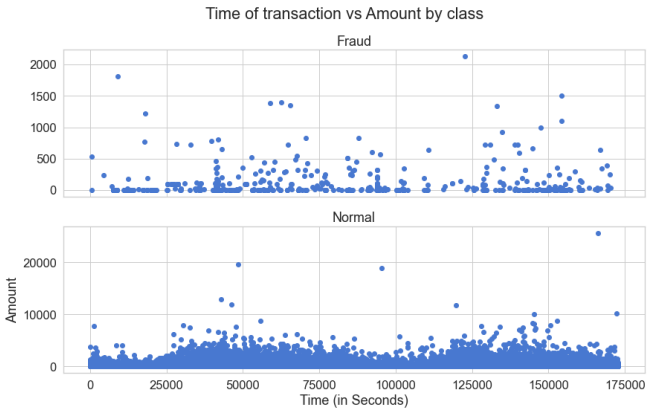

Figure 1. Time of transaction vs. Amount by class.

Figure 1 illustrates that when the normal transaction amount is low, the fraud transaction amount is relatively higher. And normal transactions are more frequent than fraud ones, which means in certain periods, the number of normal transactions is larger than fraud ones.

Distribution plots can be useful to see the relationship between values of a feature of different class types. These features all form a sharp contrast and are conducive for the model to recognize the transaction type. Take V1 as an example, Figure 2 shows that the values of the feature V1 for normal transactions have low ebb, where those of fraud transactions are relatively high. This phenomenon forms a recognizable discrepancy between normal and fraud transactions.

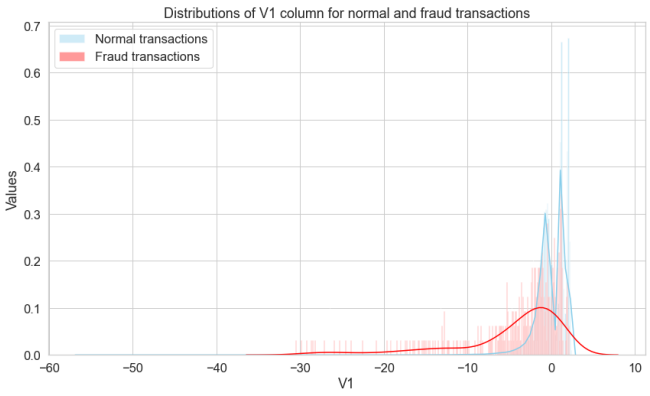

Figure 2. Distributions of V1 column for normal and fraud transactions.

IV. METHODS

I develop the hybrid model for credit card fraud detection by combining two individual unsupervised models and compare this method with logistic regression as a baseline. The AUC value of logistic regression is only 0.8087, while that of the hybrid autoencoder model is 0.9577. The autoencoder is an unsupervised model that handles imbalanced data very well and can be trained using one class of data without the data labels. In contrast, logistic regression is a supervised learning method, which requires labeled data of different classes. Therefore, the applicability of logistic regression to unbalanced data is not ideal.

I first split the data into a training set and a test set. To ensure that the split is repeatable, I fix the random seed.

Next, I apply L1 regularization through the regularizer to create sparsity in my autoencoder model based on normal transactions because of huge numbers of data. The intensity of regularization is 10e-5. The model based on fraud transactions does not necessarily need sparsity because its number of data is relatively smaller.

Then, I used a function in StandardScaler to normalize the data. It changes the values of inputs into (-1, 1). Normalization is of great help to increase the efficiency of model training, accelerating the gradient descent process.

After that, I use the deep learning tool TensorFlow to build variational as well as sparse autoencoder networks for model training. The input dimension is set as 29, and the code dimension is 14.

Finally, I save the model into two files. When using the model to predict, I load the model from the files, access the two models, and tables showing their corresponding reconstruction errors are generated. Next, I access the 'reconstruction error' column of the two tables and add them to combine the two autoencoder models.

A. Training methods

1) Implements for model training.

I apply several techniques in the training process, including TensorFlow and Keras, as well as L1 Regularization, which will be introduced in the following sections.

a) TensorFlow and Keras.

Tensorflow is a symbolic maths library created by Google. [4]. TensorFlow and Keras are both open-source deep learning libraries. Keras is an open-source artificial neural network library written in Python that can be used as a high-level application interface for Tensorflow for the design, debugging, evaluation, application, and visualization of deep learning models[5]. We use Keras to define the structure, interface, and hyper-parameters of the autoencoder network. Tensorflow is used as the back end of Keras.

b) L1 Regularization.

L1 regularization forces the weights of uninformative features to be zero by subtracting a small amount from the weight at each iteration and thus making the weight zero. The intensity of the regularizer can be adjusted. The higher the intensity is, the more sparsity it brings[6].

2) Autoencoder

Autoencoder is an unsupervised learning method. As a specific type of feed-forward neural network, the autoencoder has its input identical to the output. Essentially, it learns a mapping between the input and a lower-dimensional vector called code and another mapping between the code and the output [7].

The most common type of autoencoder is called the variational autoencoder [8], which is also what I am using when training the model for fraud transactions. However, when training a model for normal transactions, I apply the sparse autoencoder. In a variational autoencoder model, all neurons between layers are fully connected. However, in a sparse autoencoder, not all neurons between layers are fully connected in case of overfitting.

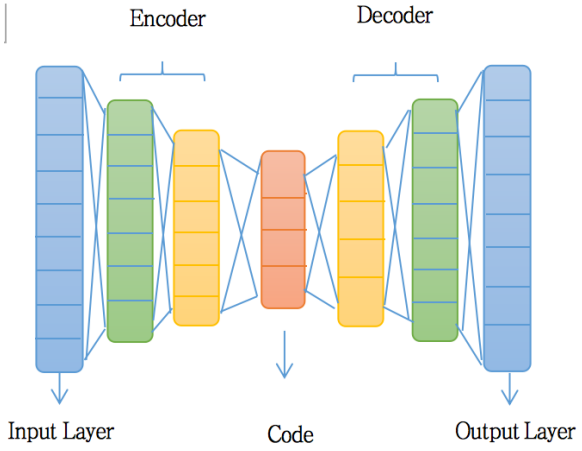

Figure 3. Autoencoder model.

Figure 3 describes the autoencoder model structures. It illustrates that an autoencoder model has more than one hidden layer, and the dimensions for inputs and outputs are the same!

However, the inputs and outputs are not equal. There exists the reconstruction error and we need to minimize it.

$$\Phi: x \to C \tag{1}$$

$$\Psi: C \to x \tag{2}$$

$$\Phi, \Psi = \arg\min ||X - (\Phi \circ \Psi)X||^2 \tag{3}$$

The function above well-illustrates the minimizing process. It finds the value of parameters in Φ, Ψ by stochastic gradient descent to make the L2 norm of $X - (\Phi \circ \Psi)X^2$ the smallest.

I use stochastic gradient descent to minimize the cost function [9].

Here, Φ is the encoding process while Ψ is the decoding process.

Φ can be further specified:

$$z = \sigma(Wx + b) \tag{4}$$

Here, x refers to an input of the encoder while z refers to an output of the encoder. w and b are vectors. We apply a linear transfer on x to reduce its dimension, and then we apply a nonlinear transfer σ to get the output.

Ψ can also be specified:

$$\hat{x} = \hat{\sigma}(\hat{W}z + \hat{b}) \tag{5}$$

Here, z refers to an input of the decoder while $\hat{x}$ refers to an output of the decoder. $\hat{W}$ and $\hat{b}$ are vectors, and this time we apply a linear transfer on Z to increase its dimension, then we apply a nonlinear transfer $\hat{\sigma}$ to get the output. These two functions σ and $\hat{\sigma}$ are also called activation functions. In my study, the activation function including Sigmoid, ReLu, and Tanh is applied to make the inputs become nonlinear outputs.

The loss function can be written as the following, where values of four parameters $W, \hat{W}, b, \hat{b}$ are to be found. $\sigma, \hat{\sigma}$ are hyper-parameters, whose values must be defined before training the model.

$$\mathcal{L}(x, \hat{x}) = ||x - \hat{x}||^2 = \left||x - \hat{\sigma}(\hat{W}(\sigma(Wx + b)) + \hat{b})\right||^2 \tag{6}$$

3) Sparse autoencoder.

We have known that variational autoencoder connects each neuron well but this tends to cause overfitting. Therefore, there appear sparsity constraints in sparse autoencoder which can cut some connections and avoid overfitting to some extent. A sparse autoencoder is an autoencoder that has a sparsity penalty $\delta(h)$ on the code layer h.

The sparsity can be achieved in two ways. One is exploiting the KL divergence and the other is minimizing the sum of the absolute values of outputs of all hidden layers by applying L1 or L2 regularization terms.

4) Model combination.

The flow chart above suggests the combination process, where RC1 stands for the reconstruction error of the model based on normal transactions, RC2 for that of fraud ones, and RC3 stands for that of the hybrid model. The addition makes the reconstruction error of the fraud transactions larger and more remarkable when staying together with normal transactions so that it will be easier for the autoencoder to learn.

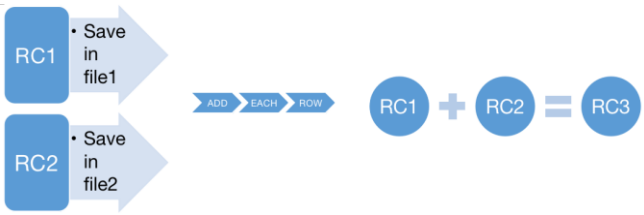

Figure 4. Flow chart of model combination.

B. *Evaluation methods*

The following are some implements that I use, including the Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC). I repeated the experiment many times to acquire the mean and standard deviation of the AUC value.

V. RESULTS AND DISCUSSIONS

A. *Hyper-parameter Study*

To find out the most suitable and practical model, I changed the values of batch size, the number of epochs, the encoding dimension, and the number of layers to figure out a relatively better model. It shows that batch size equals 64 is a better choice.

TABLE II. AUC VALUES OBTAINED WITH DIFFERENT BATCH SIZES

Batch size	AUC1	AUC2	AUC3	Mean of AUC
32	0.9522	0.9556	0.9532	0.9537
64	0.9532	0.9540	0.9549	0.9540
128	0.9534	0.9512	0.9543	0.9530

TABLE III. AUC VALUES OBTAINED WITH DIFFERENT EPOCHS

Number of epochs	AUC1	AUC2	AUC3	Mean of AUC
10	0.9535	0.9545	0.9530	0.9537
15	0.9472	0.9481	0.9490	0.9481
20	0.9437	0.9478	0.9487	0.9467

It shows that as the number of epochs increases, the AUC declines. So, the number of epochs equals 10 is a desirable choice.

TABLE IV. AUC VALUES OBTAINED WITH DIFFERENT ENCODING DIMENSION

Encoding dimension	AUC1	AUC2	AUC3	Mean of AUC
10	0.9537	0.9530	0.9537	0.9535
14	0.9528	0.9540	0.9540	0.9536
18	0.9501	0.9526	0.9530	0.9519

There appears to be several AUC of the same value, which implies that the effects brought about from the encoding dimension are not so significant. When the encoding dimension is equal to 14, the value of the AUC of this model is larger.

This is a table showing different numbers of encoding layers and decoding layers and the corresponding AUC.

TABLE V. AUC VALUES OBTAINED WITH DIFFERENT NUMBERS OF ENCODING LAYERS

Number of layers	AUC1		AUC2	AUC3	Mean of AUC
5	0.9537		0.9542	0.9545	0.9541
7	0.9537		0.9550	0.9545	0.9544
9	0.9547		0.9554	0.9550	0.9550

Figure 5 is presented as follows to take a closer look at the results.

It shows that as the number of layers increases, the AUC of the model also increases. But when the number of layers is set to be 7 and 9, the reconstruction error is undesirable because it does not converge well. Therefore, the number of layers equals 5 is a better choice.

It turns out that changing epochs has the most significant influence, while other hyper-parameters have impacts to certain extents.

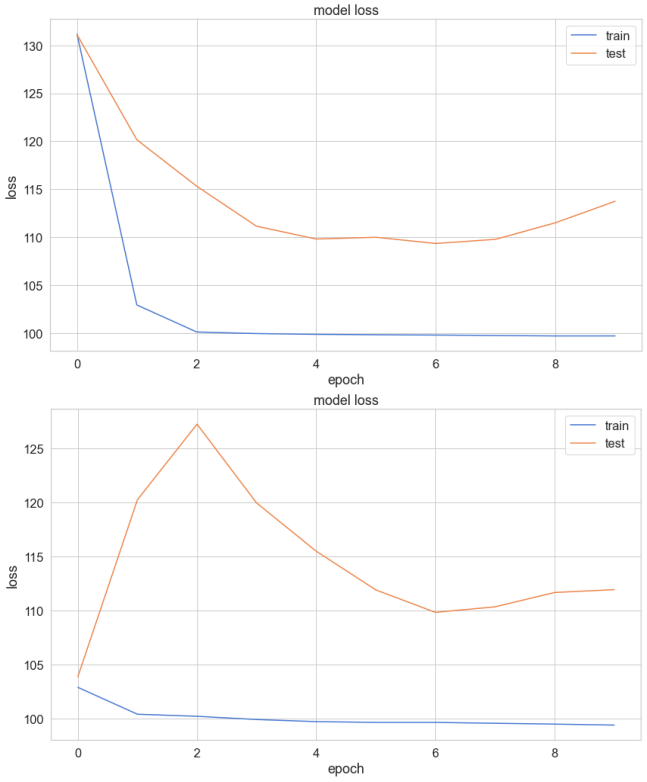

Figure 5. The model loss for 'number of layers=7' (upper) and 'number of layers=9' (lower).

B. *Model Performance*

TABLE VI. AUC VALUES OF DIFFERENT MODELS

	Model of normal transactions	Model of fraud transactions	Hybrid Model
AUC value1	0.9603	0.9527	0.9576
AUC value2	0.9637	0.9549	0.9568
AUC value3	0.9591	0.9538	0.9589
AUC value4	0.9613	0.9522	0.9576
AUC value5	0.9621	0.9534	0.9577
AUC value6	0.9577	0.9538	0.9565
AUC value7	0.9579	0.9525	0.9583
AUC value8	0.9631	0.9543	0.9576
AUC value9	0.9602	0.9551	0.9578
AUC value10	0.9637	0.9546	0.9581
STD of AUC value	0.0022	0.0010	0.0007
Mean of AUC value	0.9609	0.9537	0.9577

After training the model, the AUC value can be presented to determine whether the model has been well-trained. The AUC value is between the previous two models, and its standard deviation is the lowest among the various types of models, and its mean is between that of the two individual models. Therefore, it can supplement the weakness of the previous models and increase stability, which also ensures accuracy.

C. Discussions of experiment results

I come up with the idea of combining the two models to balance the strengths and weaknesses of each model and increase the stability as well as ensure the accuracy. Therefore, the total reconstruction error of the two models is extracted and added so that a hybrid model is formed. The reconstruction error in terms of two classes in the hybrid model is closer to that of the model based on fraud transactions. This may be caused by addition. The combination process will make the reconstruction error in terms of fraud transactions be more distinctive. However, I previously expected the combination process to be subtraction or division, but it seems that the addition will turn out a more satisfactory result.

VI. CONCLUSION

The research on credit card fraud detection is practical in today's world. The goal of this research is to make a little difference to the massive via individual efforts to make people feel connected and cared. We are now entering an information-booming era and personal information is likely to be robbed, which will expose us to a greater chance to fall into the dark zone of credit card fraud. Unfortunately, we can never keep our personal information from being robbed. Therefore, measures like credit card fraud detection must be taken to prevent fraud behaviors from going to an extreme and bringing people loss. The research can help design a practical fraud detection system, especially for optimizing the machine learning algorithm of the system.

REFERENCES

[1] Lei, Ali.Ghorbani, "Improved competitive learning neural networks for network intrusion and fraud detection "Neurocomputing 75(2012)135–145.

[2] Patidar, Lokesh Sharma, "Credit Card Fraud Detection Using Neural Network", International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-1, 2011.

[3] Machine Learning Group - ULB. (2018) Credit Card Fraud Detection. https://www.kaggle.com/mlg-ulb/creditcardfraud

[4] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).]

[5] Simple.Flexible.Powerful.Keras. https://keras.io.

[6] L1 and L2 Regularization-Explained. https://mc.ai/l1-and-l2-regularization%e2%80%8a-%e2%80%8aexplained/.2020,March 8

[7] Kramer, Mark A. (1991). "Nonlinear principal component analysis using autoassociative neural networks" (PDF). AIChE Journal. 37 (2): 233–243. doi:10.1002/aic.690370209.

[8] Welling, Max; Kingma, Diederik P. (2019). "An Introduction to Variational Autoencoders". Foundations and Trends in Machine Learning. 12 (4): 307–392. ArXiv: 1906.02691.

[9] Scikit Learn "1.5 Stochastic Gradient Descent." https://scikit-learn.org/stable/modules/sgd.html

[10] Business Daily News(2020.4). " 746 million credit cards represent a staggering trillion-dollar consumer market!" https://zhuanlan.zhihu.com/p/131601080

[11] Erica Stanberg (2020.9). "The Average Number of Credit Card Transactions per Day & Year". https://www.cardrates.com/advice/number-of-credit-card-transactions-per-day-year/