POSTER: Practical Fraud Transaction Prediction

Longfei Li, Jun Zhou, Xiaolong Li, Tao Chen
Ant Financial Services Group, China
{longyao.llf, jun.zhoujun, xl.li, boshan.ct}@antfin.com

ABSTRACT

Nowadays, online payment systems play more and more important roles in people's daily lives. A key component of these systems is to detect and prevent fraud transactions. In industrial practice, such a task is separated into two phases: 1) mining evidential features to describe users, 2) building an effective model based on these features. Generally speaking, the most popular fraud transaction detection systems use elaborately designed features to build tree based models, sometimes a subsequent linear model is added to improve the behaviour. However, the designed features usually contains only static features, while dynamic features are not considered. In addition, the subsequent model can only learn a linear combination, which may always be unsatisfactory. To address these issues, we present a systematic method, which extracts not only users' static features but also dynamic features based on their recent behaviors. Moreover, N-GRAM model is employed to handle the dynamic features so that time series information is addressed. Based on the extracted features, a tree based model is applied and the outputs of it are regarded as new generated feature representations, which will be further inputted into a Deep Neural Network (DNN) to learn the complex relationships and form the final classification model. Extensive experiments show that our proposed model (with both static and dynamic features) significantly outperforms the existing methods.

KEYWORDS

Fraud Transaction Detection; GBDT; DNN

1 INTRODUCTION

Online payment is becoming more and more important in people's daily lives. A giant online payment system, such as PayPal, WeChat pay, and Alipay, can have hundreds of millions of users and generates millions of online transactions per day. Although online payment brings more convenience to people, it also faces more and more challenges as the network environment becoming more and more complex. One of the critical challenges is fraud transactions which will harm both the account holders and platform. The ability to detect and prevent fraud transactions is thus very crucial for online payment systems.

Fraud transaction detection is very challenging. First, it is not a simple binary classification problem, as we need to build a regression or scoring model and choose a threshold to differentiate fraud and normal transactions. Furthermore, the suspicious transaction will be suspended with a notification sending to the involved users.

A false alarm of fraud transaction will lower down transaction success rate and even harm the user experience. Last but not least, in an online payment platform, in order to protect the user accounts, usually we use multiple authentication methods such as face recognition, SMS etc, which will cost a lot of money. An accurate data-driven method can help to reduce such cost. In this paper, we will present a practical data-driven method for fraud transaction detection in our production system, for which we shall share our experience in feature engineering and model design.

Feature Engineering. Traditionally, we build a model based on users' static features, such as trade amounts and trade frequencies, for fraud detection. However, static features only contain the historical profiles of a user, which do not include dynamic user behaviors. In this work, we propose to not only extract static user features from their profiles, but also dynamic features from their recent behaviors. Specifically, we find that the dynamic features from user behaviors help to significantly improve model performance compared with only using the static features in our task.

Model Design. Usually we can build a linear model such as Logistic Regression(LR), or a nonlinear model such as Deep Neural Network(DNN), or tree based model, such as RF and GBDT for the task. For linear or nonlinear model, usually some feature transformation such as feature discretization need to be conducted before feeding the raw features into the model. For tree-based model, the raw features can be directly fed in, as the model is able to split the numerical values accordingly. This property and the strong representation power of tree based models make them widely adopted in the industries. Despite this, an RF or GBDT model is a linear combination of separate trees, which can be observed from Eq. (1),

$$F(x) = \sum_{i=1}^{n} \gamma_i h_i(x) + const, \tag{1}$$

where γ_i is the weight of the i-th trees, and $h_i(x)$ is the output of i-th tree. In practice, these linear weights may not be optimal, since the old weights $\gamma_1, \dots, \gamma_i$ are never updated when a new tree h_{i+1} is introduced. To solve this issue, some researchers use LR to re-learn the weights and treat the tree-based model as feature transform method [3], which archives better results. However, LR cannot catch the connections among the features, which may limit the performance of the transaction risk estimation. So, we propose to use DNN for learning the new weights of different trees, denoted as GBDT-DNN for convenience. Extensive experiments show that our GBDT-DNN hybrid model with dynamic feature significantly outperforms all the other baselines.

In summary, our main contributions are:

- We propose to extract dynamic user behaviors for the task of fraud transaction detection;
- We propose a GBDT-DNN model for the task;

CCS '17, October 30-November 3, 2017, Dallas, TX, USA
© 2017 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4946-8/17/10.
https://doi.org/10.1145/3133956.3138826

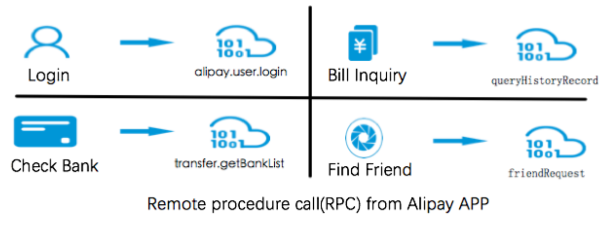

Figure 1: Remote Procedure Call Protocol

- Extensive experiments show our methods significantly outperform other baselines. We deployed our method in a real application and shared our experience.

2 ARCHITECTURE

In this section, we will introduce the details of both feature engineering and model design.

2.1 Feature Engineering

We have mainly used two types of features, one is user profile features which are static, the other is user behavior features which are dynamic.

User profile features: According to a person's transaction and shopping records in the platform, we can get some profiles of the person, such as working place, living place, credit score (similar with FICO score), trading amount, etc, which demonstrates a person's consuming ability and consumption habits. The rationale of using such features is that an unusual transaction amount or location may be with fraud.

Dynamic features: When a user is using the platform's services, we can record which services this user have used. As shown in Fig 1, behaviors like login, check bill, check bank card will be recorded in our database. From these operation behaviors, we can learn the user's common behavior patterns.

The operation behaviors can be treated as a time series data. Common methods for handling such data are discussed as follows.

First, simply using a statistic of the appearance of user behaviors. Specifically, we use a binary vector representation of user behaviors, where the total length of the vector is the dimension of behaviors and the value at each dimension indicate the presence or absence of the corresponding behavior. Second, the behavior sequence matters as the same click behaviors but with difference sequences reveal different transaction status. To make use of such sequential information, we can use N-gram features (capture at most length N sequences). In our study, we found that a binary vector representation of all behavior features has shown to have competitive performance and could be easily deployed in an industrial scale application.

2.2 model design

In this section we present our proposed hybrid model structure: the concatenation of boosted decision trees and deep neural network classifier, as illustrated in Figure 2.

The boosted decision trees are shown to be a powerful model to transform the original features of an instance [3], which can then be utilized by other models to further get even higher accuracy. Specifically, we use each learned individual tree as a categorical feature, where the value is set as the index of the leaf node the instance falls in. As a result, if there are n trees in the GBDT model, the transformed feature of an instance is given in terms of a structured vector $x = (e_{i_1}, ..., e_{i_n})$, where e_{i_k} is the i_k-th unit vector with the dimension of d_k, where d_k is the number of leaf nodes at k-th tree, and i_k is the index of the leaf node where the current instance falls into at k-th tree. Such feature extraction method will not bring too much pressure for online systems. In the training phase, we also assume that a binary label $y \in \{1, 0\}$ is given for each transaction, to indicate whether it is risky or not. Finally, we can use the transformed feature to train a DNN model, which can further outperform GBDT.

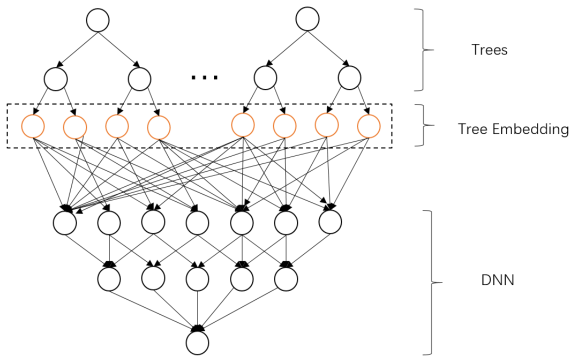

Figure 2: Hybrid model structure. Input features are transformed by boosted decision trees. The output of each individual tree is treated as a categorical input feature feeding to a fully connected DNN.

The DNN model used in our study is a five-layer fully connected network. It has three hidden layers each coupled with a Rectifier function as the activation function. The output layer uses the sigmoid function. Empirical studies show a deeper network structure does not provide significant improvements.

3 EXPERIMENT

In this section, we will represent our experiment.

3.1 Experimental Setup

In our experiments, we used a real historical transaction dataset from a giant online payment platform. The features of each instance consists of transactional information, user information and user behavior records as is mentioned in section 2.1, besides, for dynamic features, user's interactions with the the APP during the recent two days are selected as feature, if the number of interaction is bigger than 200, we will drop the old interactions, at the same time, low-frequency interaction behaviors are filtered out to focus on high-frequency behaviors.

The dimensions of the static features dimension is 273 and dynamic features dimension is 2790.

To perform rigorous and controlled experiments, we prepared a sufficiently large dataset which consists of transactions records from 5 months in 2016. To simulate the online environment, we divide the whole dataset into a training set, a validation set, and a testing set according to the time stamps. Specifically, the data from the first three months is treated as the training set, the next month as the validation set, and the last month as the testing set. As we notice that the risky samples are too rare while non-risk samples are too many, we downsample the non-risky samples to balance the proportion between them. The details of the data sizes are summarized in Table 1.

Table 1: Statistics of Datasets in Millions

Data set	train set	validation set	test set
Size(M)	7.3	2.4	2.3

To validate the effectiveness of the proposed GBDT-DNN and dynamic feature, we compare a number of state-of-the-art techniques on our transaction dataset, which includes:

- LR: the Logistic Regression algorithm [1];
- DNN: Deep Neural Networks with plain feature;
- GBDT: Gradient Boosting Decision Tree;
- GBDT-LR: LR with GBDT transformed feature;

Since our dataset is large and security, we use KunPeng algorithm platform which is quite similar with Pettum which contain all the algorithm we used in our experiments except LSTM.

For GBDT, we use 400 trees and restrict the max depth of each tree as 3, to cover more features (and their correlations) and prevent over-fitting. For DNN and GBDT-DNN, they have the same network structure: three fully connected hidden layers with 256,128,128 hidden units, respectively. The Rectifier function is adopted as active function. The output of DNN is compressed by the sigmoid function, and the optimization solver adopted is Adagrad [2].

3.2 Comparison with Baseline Models

Firstly we compare all the methods only based on the static features, where **R**ecall **A**t **T**op **P**ercent (**RATP**) is used to evaluate their performance. RATP@r is the recall of the subset which consists of the instances with the top r percentage of prediction scores. The results are summarized in the Table 2.

Table 2: RATP Result with static feature

Model Structure	RATP@0.02	RATP@0.05	RATP@0.1
LR	0.402	0.534	0.602
DNN	0.425	0.634	0.683
GBDT	0.524	0.718	0.792
GBDT-LR	0.525	0.723	0.796
GBDT-DNN	**0.546**	**0.734**	**0.803**

From the test RATP in Table 2, we observe GBDT-DNN significantly outperforms all the baselines. Since transaction amount is so huge, a little improvement can significantly decrease the number of people to be disturbed. We can find that GBDT-DNN always outperforms all the other baselines when we use different percentages (0.02%, 0.05%, 0.1%). For example, GBDT-DNN is 2.1%, 1.6% better than GBDT under RATP@0.02 and RATP@0.05, respectively. These observations show that the proposed GBDT-DNN can significantly improve RATP.

3.3 Importance of Behavioral Features

In this subsection, we shall examine the importance of behavior features.

Table 3: AUC Results on adding behavior features (Beh) to the original model (Orig).

Model	LR	DNN	GBDT	GBDT-LR	GBDT-DNN
Orig.	0.9546	0.9675	0.9922	0.9923	0.9934
+Beh.	0.9612	0.9721	0.9935	0.9936	**0.9944**

Firstly, we use Area-Under-ROC (AUC) to evaluate the methods, which is shown in Table 3. Since the majority of the transaction is non-fraud, the baseline AUC score is pretty high. Our GBDT-DNN helps increase AUC by more than 0.08% relative to that of GBDT. However, a small improvement on the AUC helps to prevent a large number of fraud transactions. To validate this, again, we evaluate the methods in terms of RATP as shown in Table 4.

Table 4: RATP Results with behavioral features. The relative improvements over static features are shown in brackets.

Model Structure	RATP@0.02	RATP@0.05	RATP@0.1
LR	0.411(+2.2%)	0.551(+3.2%)	0.632(+5.0%)
DNN	0.429(+0.9%)	0.644(+1.6%)	0.714(+4.5%)
GBDT	0.538(+2.7%)	0.732(+1.9%)	0.812(+2.5%)
GBDT-LR	0.541(+3.0%)	0.732(+1.2%)	0.814(+2.3%)
GBDT-DNN	**0.568**(+4.0%)	**0.751**(+2.3%)	**0.832**(+3.6%)

From RATP results in Table 4, we find that after using behavior features, almost every algorithm makes an impressive improvement from at 0.9% to 5.0%. And our proposed GBDT-DNN model still outperforms all the other algorithms.

4 CONCLUSIONS

In this work, we demonstrate a systematic way to extract user features regarding their profiles and recent behaviors for fraud transaction detection task. We further present a hybrid model to tackle this task, which uses DNN to boost GBDT results. Experiments have been conducted on a real large-scale dataset to show the effectiveness of our proposed model. Our model has also been deployed in a giant online payment platform and outperformed the previous production system significantly. As future work, we seek to use a unified model to automatically learn high-level features from raw data for fraud transaction detection.

REFERENCES

[1] Galen Andrew and Jianfeng Gao. 2007. Scalable training of L 1-regularized log-linear models. In *ICML*.
[2] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR* (2011).
[3] Xinran He and et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*.