Credit Card Fraud Detection: A Novel Approach Using Aggregation Strategy and Feedback Mechanism

Changjun Jiang, Jiahui Song, Guanjun Liu, *Member, IEEE*, Lutao Zheng, and Wenjing Luan, *Student Member, IEEE*

***Abstract*—With the rapid development of electronic commerce, the number of transactions by credit cards are increasing rapidly. As online shopping becomes the most popular transaction mode, cases of transaction fraud are also increasing. In this paper, we propose a novel fraud detection method that composes of four stages. To enrich a cardholder's behavioral patterns, we first utilize the cardholders' historical transaction data to divide all cardholders into different groups such that the transaction behaviors of the members in the same group are similar. We thus propose a window-sliding strategy to aggregate the transactions in each group. Next, we extract a collection of specific behavioral patterns for each cardholder based on the aggregated transactions and the cardholder's historical transactions. Then we train a set of classifiers for each group on the base of all behavioral patterns. Finally, we use the classifier set to detect fraud online and if a new transaction is fraudulent, a feedback mechanism is taken in the detection process in order to solve the problem of concept drift. The results of our experiments show that our approach is better than others.**

***Index Terms*—Behavioral patterns, concept drift, credit card fraud, machine learning, sliding window.**

I. Introduction

WITH the popularization of mobile devices, online shopping becomes a popular mode of daily purchases. However, the Internet environment is open, online shopping systems have bugs, and criminals can use some bad techniques, such as Trojan and pseudo base-station. All these result in a serious increasing of credit card fraud events. When a criminal steals or cheats the information of the credit card of a cardholder, the criminal can use the credit card to consume. According to the Nilson report in October 2016, more than \$31 trillion were generated worldwide by online payment systems in 2015, increasing 7.3% than 2014. Worldwide losses from credit card fraud rose to \$21 billion in 2015, and will possibly reach \$31 billion by 2020 [1].

Manuscript received January 11, 2018; revised February 20, 2018; accepted March 7, 2018. Date of publication March 15, 2018; date of current version November 14, 2018. This work was supported in part by the National Key Research Development Program of China under Grant 2017YFB1001804 and in part by the Shanghai Science and Technology Innovation Action Plan Project under Grant 16511100900. *(Corresponding author: Guanjun Liu.)*

The authors are with the Department of Computer Science, Key Laboratory of Embedded System and Service Computing, Shanghai Electronic Transactions and Information Service Collaborative Innovation Center, Tongji University, Shanghai 201804, China (e-mail: cjjiang@tongji.edu.cn; demon.song@tongji.edu.cn; liuguanjun@tongji.edu.cn).

Digital Object Identifier 10.1109/JIOT.2018.2816007

Credit card fraud detection is an important way to prevent fraud events, which is usually categorized into two techniques: 1) anomaly detection and 2) classifier-based detection. Anomaly detection focuses on calculating the distance among the data points in space. By calculating the distance between the incoming transaction and the cardholder's profile, an anomaly detection method can filter any incoming transaction which is inconsistent with the cardholder's profile. The second technique utilizes some supervised learning methods to train a classifier on the basis of the given normal transactions and fraud ones. The supervised learning focuses on extracting fraud features from fraud transactions. However, both of them have limitations. For the anomaly detection, it has no ability to portray fraud features although it can portray cardholders' transaction behaviors. For the classifier-based detection, it fails to distinguish different normal behaviors from different cardholders although it can catch fraudsters' behaviors. As revealed in [2], transaction habits of a person vary often since they are easily influenced by their incomes, resources, ages and characters. Thus, their distribution evolves over time because of seasonality and new attack strategies [3]. This is called the problem of concept drift [4] that is difficult to be resolved by the above detection methods.

On the other hand, both of them are not aware of the adaptive capacity of the model. For example, a person may involve some new transaction behaviors in a specific period which has never happened in his/her history. Most of the proposed methods just keep the recent instances for model training, but do not consider the adaptiveness of the model.

Facing the above challenges, we extract the transaction behaviors of a cardholder using both his/her historical transaction data and the data of some similar cardholders. Furthermore, we propose a feedback mechanism which can adapt to the cardholder's transaction behaviors seasonally. This paper can be summarized as the following four aspects.

1) By using clustering method, all cardholders are divided into three groups based on transaction amount, i.e., high, medium, and low.
2) We propose a sliding-window-based method to aggregate the transactions in every group, i.e., derive a set of additional features from windows to characterize a cardholder's behavioral patterns.

3) After preprocessing features, we train a set of classifiers for each group using the data consisting of each specific behavioral pattern and extracted fraud features.
4) Finally, the classifier set trained for a group is assigned to each cardholder in the group as his/her own behavioral patterns, and the classifier with the highest rating score is viewed as his/her recent behavioral pattern. Based on the three classifier sets, we propose a fraud detection method in which a feedback mechanism is taken in order to solve the concept drift problem.

The rest of this paper is organized as follows. Section II reviews the related work on fraud detection. The proposed approach is introduced in Section III. Section IV demonstrates the experimental results. Finally, we conclude this paper in Section V.

II. Related Work

In the earlier study, researchers focused on identifying unusual behaviors from historical transactions and classifying them as fraudulent or genuine. Chen *et al.* [5] proposed a novel personalized approach based on SVM and ANN for detecting fraud. Shen *et al.* [6] tested some classification methods to detect fraud and found that the neural-network-based detection and the logistic-regression-based one both outperform the decision-tree-based one. Quah and Sriganesh [7] used the self-organization map to decipher, filter and analyze the cardholders' behaviors for the detection of fraud.

Srivastava *et al.* [8] proposed an hidden Markov model (HMM) based credit card fraud detection. It can detect fraud by analyzing the transaction patterns on each card and figure out any inconsistency with respect to the usual transaction patterns. Although this model has considered time features, it does not address the concept drift problem. Because HMM model remembers all historical behavioral patterns of a cardholder, it cannot forget the behaviors which may not happen in recent time.

Dempster–Shafer's theory is used to combine multiple evidences and an initial belief is computed by Panigrahi *et al.* [9]. The proposed fraud detection method is composed of four major components. A cardholder profile is used to characterize transactions behaviors. A rule-based component measures the degree of the incoming transaction based on cardholder profiles. Next, the Dempster–Shafer adder combines these evidences to derive an overall belief. Finally, a Bayesian learner can weaken or strengthen this belief using history database.

Seyedhossein and Hashemi [10] constructed a series of time windows to extract the behavioral patterns of an individual credit card. Based on some observation on real data, transactions have some periodic structures. In the first stage, they use k-means to extract a seven-day period pattern on preprocessed data. At the next stage, once the incoming transaction arrives, the model can measure the distance between this new window and the cardholder's profile by using the accumulated transactions. The proposed approach has improved timeliness and detection rate while decreasing the cost in some circumstances.

In the recent years, the machine-learning-based fraud detection has been an important topic, and the supervised methods are widely used. Most of the recent studies do not only consider the raw transaction data for training, but also address serval factors that have an important impact during the training phase, such as aggregation strategy from raw data, cost-sensitivity of the application, skewness of the data, and short-time response of the system. Bahnsen *et al.* [11] addressed the cost-sensitivity problem, and expanded the transaction aggregation strategy to create a new set of features using the von Mises distribution. Vlasselaer *et al.* [12] proposed a novel, automated and real-time approach to detect credit card fraud by mapping the past behavioral patterns into some meaningful features.

Unsupervised methods require little or no prior classifications to anomalies. Hence, they are suitable for the transactions with no label. Outlier detection is becoming a growingly useful tool applied to the credit card fraud detection. Gurjar *et al.* [13] proposed an outlier detection method based on the mutuality of the relationship in terms of nearness of a data point and its neighbors, and improved the effectiveness and performance of the dataset consisting of clusters with special shapes, such as lines or circles. SODRNN [14] is another unsupervised approach. It uses the reversed k nearest neighbor algorithm to detect the outliers for fraud detection and uses a data stream technique to scan the data only once instead of serval times, which is more common.

Most techniques mentioned above require some form of supervised training before the system can be applied. However, the transaction behaviors of a cardholder often change over time. To address this problem, the supervised model must be retrained to address the legitimately changing context of the cardholder which can result in the delay of fraud detection system.

To deal with the concept drift problem, Malekian and Hashemi [3] proposed a new concept drift management framework. In their work, a temporary profile is used to retain new concepts, and an initial profile is used to remember all historical behaviors of a cardholder. By using the suitable profile, the result is predicted more accurately when the concept drift happens. Robinson and Aria [15] proposed an HMM to detect fraud in real-time for merchants. By using the fixed window size, it can calculate KL divergence [16] between the recent sequence and the latest sequence. Once the KL divergence is greater than the specific threshold, the model raises a fraud alert, otherwise updates the current HMM to be the most recent HMM.

To the best of our knowledge, though some classification methods and anomaly detection are used to detect frauds, they still have no ability to forget the outdated behavioral patterns due to the concept drift problem. This is the main reason that the false alarm rate is very high in their work. In other literatures, such as [8], [10], and [11], the aggregation models are used to catch the fraud features, but they fail to solve timeliness of the fraud detection. In this paper, our on-line method can adjust each cardholder's profile timely and adapt to the cardholder's recent transaction behaviors. Namely, it can perform different results over time.

TABLE I
RAW FEATURES OF CREDIT CARD TRANSACTION

Attribute name	Description
Transaction ID	Transaction identification number
Cardholder ID	Identification of the cardholder
Amount	Amount of the transaction
Time	Date and time of the transaction
Label	The genuine/fraudulent transaction

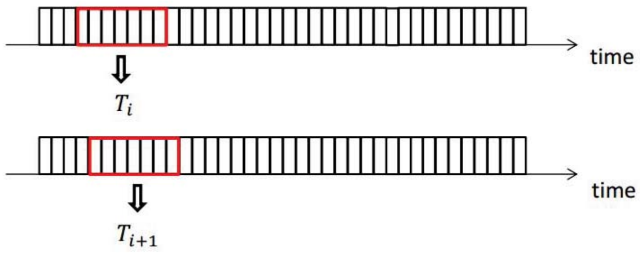


Fig. 1. Sliding window algorithm for aggregating transactions and deriving features.

III. PROPOSED METHOD

There exist serval problems when we use all transactions to build one classifier. For example, in a real world, a cardholder has his/her own behavioral patterns but one classifier trained via all transactions ignores the personalized behaviors of the cardholder. More importantly, binary classification has no ability to label each behavioral pattern of a cardholder. Hence, traditional method with one classifier cannot solve the concept drift problem due to the lack of classified label information.

Instead, our proposed method extracts the behavioral patterns precisely from the aggregated data and labels each behavioral pattern by using a clustering method. Thus, we can build each cardholder's behavioral profile via these behavioral patterns and our on-line model can adapt to a cardholder's transaction behaviors timely. Our method contains four steps.

1) Preprocessing data.
2) Clustering behavioral patterns.
3) Classifying behavioral patterns and assignment.
4) Updating cardholders' behaviorial profile by a feedback mechanism.

A. Preprocessing Data

In our proposed method, the initial set of features (raw features) are summarized in Table I.

1) Cardholder Clustering: We first use the clustering method k-means [18] to divide all cardholders into three similar groups which are, respectively, labeled as high (h), medium (m), and low (l) based on transaction amount. Therefore, we assume that $V = \{l, m, h\}$ and $|V| = 3$. Note that l, m, and h can be viewed as three sets of id's of all cardholders in the corresponding group and id is the identification of the cardholder. Transaction data of all cardholders in a group are more helpful to solve the sparse problem of data compared with the transaction data of a single cardholder. More importantly, in this way each cardholder's behavioral patterns can be composed of two parts: his/her own behaviors reflected by his/her historical transactions, and other behaviors recommended by other members in the same group which may happen in the future but are not reflected by his/her historical transactions. The latter can enrich a cardholder's behaviors and improve the adaptiveness of individual model.

2) Sliding Window: After dividing all users into three similar groups, we propose a sliding-window-based algorithm [19] to aggregate the transactions and then derive some new amount-related/time-related features from the aggregated data in order to characterize the behavioral patterns of a cardholder more precisely. The sliding-window-based algorithm is an incremental mining technique and often used to detect the image objects. Furthermore, due to the fixed window size, this algorithm can quickly drop the first element and append the next new element to do data statistics by using the partial information from the previous window.

Fig. 1 illustrates the method to aggregate transactions and derive new features. Each block represents one transaction of a cardholder. Let $t^{\text{id}} = \{t_1^{\text{id}}, \ldots, t_n^{\text{id}}\}$ be a transaction sequence of the cardholder, where n is the number of the cardholder's transactions in history. The process of aggregating transaction is to select those transactions in a fixed window size p

$$T_i^{\text{id}} = \text{WINDOW}(id, i, p) = \left\{t_j^{\text{id}} | j \in \left[i, i+p-1\right]\right\} \qquad i < n - p + 1 \qquad (1)$$

where WINDOW is a function that creates a new window of a transaction sequence which is denoted as T_i^{id}, i is the new window index, and p is the size of the new window.

As introduced in [10], individual behavioral patterns have strong weekly and monthly periodic structure and it is important to decide the size of a sliding window. We can analyze the frequency of transactions in a fixed time to choose a period such as a week or a month and decide p by human choice. For example, we can choose 50 transactions as a window size if the average number of each cardholder's transactions in a month is 50 (note that this can be obtained from the statistics of data).

3) Feature Extraction: After the aggregation process, we use these new windows to derive some new features. It is important to assume that some fraudsters are not familiar with the transaction behaviors of a cardholder. They are trying to get the most profits by performing high value transactions. Hence, four features we need to extract is the maximum, minimum, average amount of T_i^{id} and the amount of the last transaction in a window:

1) maximum function: $x_{i1}^{\text{id}} = \text{MAX\_AMOUNT}(T_i^{\text{id}})$;
2) minimum function: $x_{i2}^{\text{id}} = \text{MIN\_AMOUNT}(T_i^{\text{id}})$;
3) average function: $x_{i3}^{\text{id}} = \text{AVG\_AMOUNT}(T_i^{\text{id}})$;
4) last transaction amount: $x_{i4}^{\text{id}} = \text{AMOUNT}(T_i^{\text{id}})$.

In addition, more cautious fraudsters try to imitate a cardholder's transaction behaviors and perform low value transactions in a short time. Therefore, we are interested in analyzing some time-related features. We set $p-1$ time intervals which are, respectively, calculated from a transaction and its previous

TABLE II
EXAMPLE OF DERIVED FEATURES

Raw features					Derived features						
TranID	CardID	Time	Amount	Label	x_{i1}^{id}	x_{i2}^{id}	x_{i3}^{id}	x_{i4}^{id}	x_{i5}^{id}	x_{i6}^{id}	y_{i}^{id}
1	1	2017/01/07	350	0	/	/	/	/	/	/	/
2	1	2017/01/08	400	0	/	/	/	/	/	/	/
3	1	2017/01/10	250	0	400	250	333	250	1	2	0
4	1	2017/01/13	500	0	500	250	383	500	2	3	0
5	1	2017/01/14	150	1	500	150	300	150	3	1	1
6	1	2017/01/14	100	1	500	100	250	100	1	0	1
7	1	2017/01/14	100	1	150	100	117	100	0	0	1

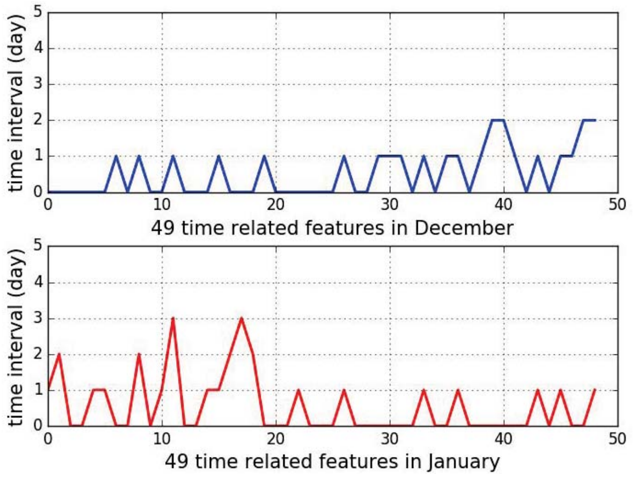


Fig. 2. One cardholder with two different time-related curves in two different months.

transaction in a window. For example, with the window size $p = 50$, amount-related features are $\{x_{i1}^{\text{id}}, x_{i2}^{\text{id}}, x_{i3}^{\text{id}}, x_{i4}^{\text{id}}\}$, and time-related features are $\{x_{i5}^{\text{id}}, x_{i6}^{\text{id}}, \ldots, x_{i53}^{\text{id}}\}$. We denote $X_i^{\text{id}} = (x_{i1}^{\text{id}}, x_{i2}^{\text{id}}, \ldots, x_{i53}^{\text{id}})$ as a vector constructed by these derived features. Finally, we label each window as normal or abnormal by using the label of the last transaction and the label function is $y_i^{\text{id}} = \text{LABEL}(T_i^{\text{id}})$.

Additionally, Table II shows an example of derived features, where the length of transaction sequence is 7 and the sliding window size p is 3. Besides, $X_i^{\text{id}} = (x_{i1}^{\text{id}}, x_{i2}^{\text{id}}, \ldots, x_{i6}^{\text{id}})$ is a vector constructed by the aggregated transactions, we thus start to derive six new features from the third transaction and "/" is denoted as empty. For the first window constructed by transactions 1-3, $(400, 250, 333, 250, 1, 2, 0)$ is a vector and 400, 250, and 333 are the maximum, minimum, and average amount in this window, respectively. Because transaction 3 is the last one in this window, we have that $x_{i4}^{\text{id}} = 250$ and $y_i^{\text{id}} = 0$. In addition, $x_{i5}^{\text{id}} = 1$ is the time interval between transactions 2 and 1. $x_{i6}^{\text{id}} = 2$ is the time interval between transactions 3 and 2.

Finally, for each cardholder we can obtain two datasets

$$G^{\text{id}} = \left\{X_i^{\text{id}} | y_i^{\text{id}} = 0, i \in \left[0, n-p+1\right]\right\} \quad (2)$$

$$F^{\text{id}} = \left\{X_i^{\text{id}} | y_i^{\text{id}} = 1, i \in \left[0, n-p+1\right]\right\} \quad (3)$$

Algorithm 1: Sliding-Window-Based Algorithm for Aggregating Transactions and Deriving Features

```
Input: The unique $id$ of a cardholder, a transaction sequence
       $t^{id}$ and the size of sliding window $p$;
Output: The feature sets of the cardholder $G^{id}$ and $F^{id}$
$n :=$ the length of $t^{id}$ ;
$G^{id} := \emptyset$ ;
$F^{id} := \emptyset$ ;
for ($i$:=0; $i < n - p + 1$; $i$++) do
    $T_i^{id} := \emptyset$ ;
    //WINDOW Function
    for ($j := i$; $j \le i + p - 1$; $j$++) do
        $T_i^{id} = T_i^{id} \cup t_j^{id}$ ;
    end
    $x_{i1}^{id} = MAX\_AMOUNT(T_i^{id})$ ;
    $x_{i2}^{id} = MIN\_AMOUNT(T_i^{id})$ ;
    $x_{i3}^{id} = AVG\_AMOUNT(T_i^{id})$ ;
    $x_{i4}^{id} = AMOUNT(T_i^{id})$ ;
    $f_i := 4$ ; //$f_i$ is a counter
    for ($j := i + 1$; $j \le i + p - 1$; $j$++) do
        $f_i := f_i + 1$ ;
        $x_{if_i}^{id} = TIME(t_j^{id}) - TIME(t_{j-1}^{id})$ ;
        //$TIME(t_j^{id})$ denotes the transaction time of $t_j^{id}$
    end
    $X_i^{id} = (x_{i1}^{id}, ..., x_{if_i}^{id})$ ;
    $y_i^{id} = LABEL(T_i^{id})$ ;
    if $y_i^{id} = 0$ then
        $G^{id} = G^{id} \cup X_i^{id}$;
    else
        $F^{id} = F^{id} \cup X_i^{id}$;
    end
end
```

where G^{id} is the normal feature set of the cardholder and F^{id} is the abnormal feature set. Algorithm 1 shows the process of aggregating transactions and deriving features.

B. Clustering Behavioral Patterns

After extracting new features from each window, X_i^{id} can be regarded as a single behavior pattern of a cardholder and G^{id} is the set of all behavioral patterns of the cardholder with id. Fig. 2 illustrates an example of the visualization of 49 time-related features obtained by Algorithm 1 that uses a cardholder's transactions in December and January and the window size $p = 50$. We can see that time-related curves are often different in different months which means that a cardholder's transaction behaviors are variable with seasons. Based on every cardholder's normal feature set, we can obtain the set of all normal features for each group

$$G_j = \cup_{id \in j} G^{\text{id}} \quad \forall j \in V. \tag{4}$$

In real world, it is a difficult job to classify these normal feature sets to specific behavioral patterns. The reason is that the definition of behavioral patterns may be obscure when they are concluded by using human domain knowledge. However, it is more convenient to use a clustering method to solve this unsupervised learning problem which can automatically organize high-level abstract knowledge. For example, G_l represents all normal feature sets of the low consumption group. Then, for each group $j \in V$, we carry out the cluster method k-means [18] over G_j and thus obtain a set of clusters $B_j = \{b_1, b_2, \ldots, b_k\}$ where k is the number of different behavioral patterns and preset up by hand. For example, B_l is the behavioral pattern set of the low consumption group. Each cluster can be thought of as the specific behavioral patterns of group j. In other words, those aggregated transactions in a fixed cluster are of similar features.

C. Classifying Behavioral Patterns and Assignment

In the previous step, we have obtained serval specific behavioral patterns from those aggregated transactions for each group. In each group, we have serval normal behavioral patterns but we still have no ability to predict the incoming transaction due to the lack of abnormal features. Hence, in this step, we first collect all abnormal features from the three groups and form an abnormal feature set

$$F = \cup_{id \in V} F^{\text{id}}. \tag{5}$$

For each b_i, F and their labels, we utilize random forest to training a classifier c_i. Random forest is one of the state-of-the-art ensemble methods. This method can produce a classifier that is constructed by combing serval different independent-base classifiers. This technique is known as bagging, or bootstrap aggregation which has a significantly lower risk of overfitting. By using multiple trees based on a majority voting on the individual predictions, we reduce the error rate and variance of a classifier which is more accurate than a single-base classifier [26]. After training, we obtain a set of classifiers for each group

$$C_j = \{c_1, \ldots, c_k\}, \quad j \in V. \tag{6}$$

Here, each classifier can be viewed as a profile of single behavioral pattern.

Next the classifier set C_j will be assigned to each cardholder in group j. For example, the classifier set C_l will be assigned to cardholder u where $u \in l$ and C_l^u is denoted as the classifier set of u. Thus, each group member has many specific profiles from the similar group. By using group's profiles instead of using individual profiles, we enrich a cardholder's behavioral patterns, some of which may not occur in his/her historical transactions but may happen in the future. Finally, for each cardholder u in group j, our method will choose the most suitable classifier from set C_j^u as the cardholder's recent profile. This can always keep the trends of the cardholder's transaction behaviors and the outdated behaviors can be forgotten.

Algorithm 2: Updating the Rating Score of a Set of Classifiers of a Cardholder

```
Input: The unique $id$ of a cardholder, a set of 2-tuples
       $\{< r_i, c_i > | i = 1, ..., k\}$ and an incoming transaction
       with $p - 1$ previous transactions
Output: a new set of 2-tuples $\{< r_i', c_i > | i = 1, ..., k\}$
1  $X^{id}$ := a vector of the incoming transaction with $p - 1$ previous
   transactions obtained by Algorithm 1 ;
2  $c^*$ := a classifier with the highest rating score obtained by a
   Priority Queue ;
3  $pred$ := the predicted label of $X^{id}$ by using $c^*$ ;
4  $label$ := the True label of the incoming transaction ;
5  if $pred \neq label$ and $label = 0$ then
6      for $(i := 1;\ i \leq k;\ i{+}{+})$ do
7          $pred_i$ = the predicted label of $X^{id}$ by using $c_i$;
8          if $pred_i \neq label$ then
9              $r_i' := r_i - 1$ ;
10         else
11             $r_i' := r_i + 1$ ;
12         end
13     end
14 end
```

D. Updating Cardholders' Behavioral Profile by Feedback Mechanism

To the best of our knowledge, the True label of a transaction in test dataset has never been used to update the profile of the cardholder. We note that the True label information is useful because it can reflect the changes of a cardholder's transaction behaviors indirectly. Hence, our proposed method uses feedback mechanism to update the profile of each cardholder when a new transaction comes. Each cardholder u in group j has a set $C_j^u = \{c_1, c_2, \ldots, c_k\}$. A rating score will be assigned to each classifier, and the cardholder's profile is represented as a 2-tuple $< r_i, c_i >$, where:

1) c_i is one of the behavioral pattern in set C_j^u;
2) r_i is one of the rating score of the classifier c_i.

Fig. 3 illustrates the framework of the proposed fraud detection method. In the last stage, we can see that a priority queue is used to choose a classifier with the highest rating score, which is highlighted as the gray block. Once the method produces a wrong prediction, it considers that the recent transactions cannot conform to the newest profile of the cardholder. The True label of the incoming transaction will be used to change the rating score of the classifier and our method tries to find out the most suitable classifier as the newest profile of the cardholder. Hence, we propose a feedback mechanism for updating the rating score. The incoming transaction inconsistent with the newest profile will be input to each classifier, and the classifier c_i will be rewarded (i.e., $r_i = r_i + 1$) if it predicts correctly, else it will be punished (i.e., $r_j = r_j - 1$). By using this feedback mechanism, the next transaction can

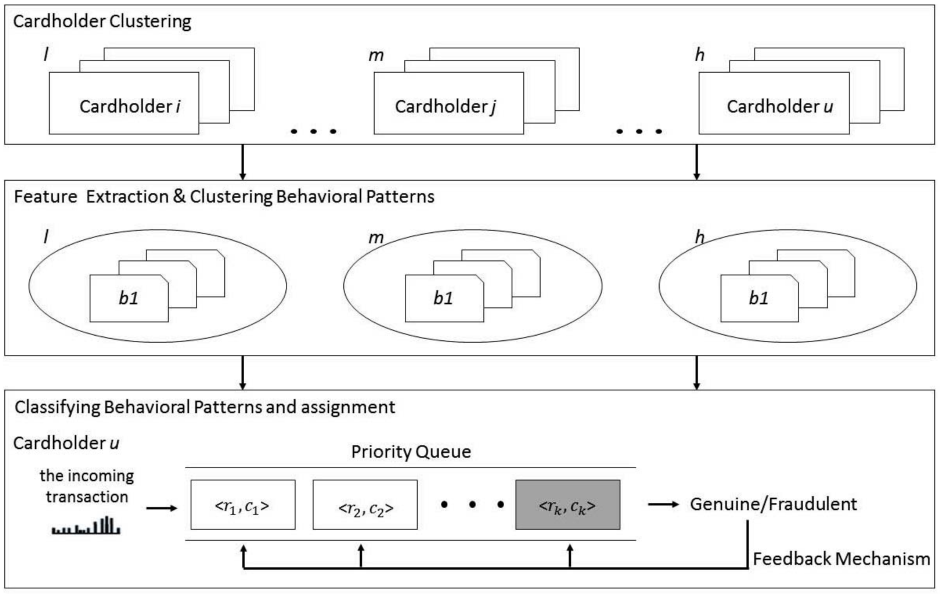


Fig. 3. Framework of the proposed fraud detection method.

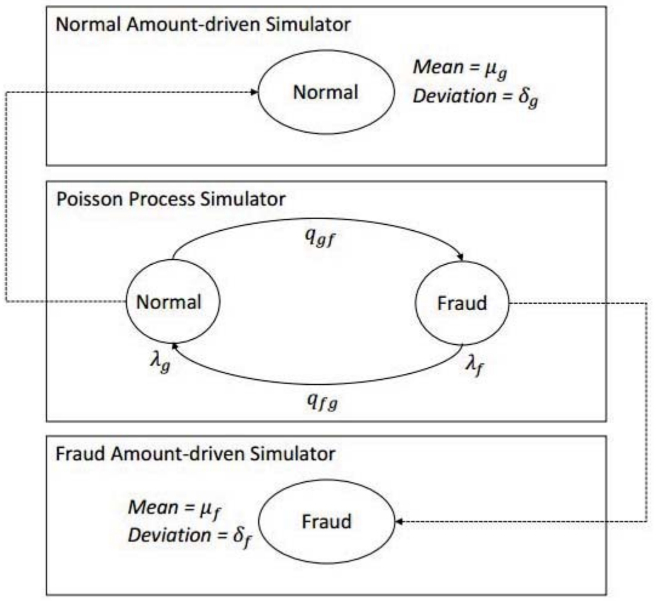

Normal Amount-driven Simulator
Normal
Mean = μg
Deviation = δg
Poisson Process Simulator
qgf
Normal
Fraud
λg
λf
qfg
Fraud Amount-driven Simulator
Mean = μf
Deviation = δf
Fraud

Fig. 4. Transaction simulator.

be predicted by a classifier c^* such that r^* is the highest rating score in $\{r_1, r_2, \ldots, r_k\}$. The more details can be seen in Algorithm 2.

Clearly, for each cardholder, our method derives serval behavioral patterns for each cardholder from the group including him/her and dynamically chooses the most suitable classifier as his/her recent profile. In this way, the feedback mechanism makes our on-line method have the ability to adapt to the cardholder's transaction behaviors.

IV. Experimental Results

It is difficult to test the proposed method using real data. Banks do not, in general, agree to share their data with researchers. There is neither benchmark data set available for experiments. Therefore, we use a simulator to generate the transaction data that is also used in most of the related literatures [8], [9], [17]. We first use the amount-driven simulator presented in [8]. The number of normal transactions in a given length of mixed transactions is normally distributed with a cardholder specified μ (mean) and δ (standard deviation). Fig. 4 illustrates our transaction simulator. The first and the last block are the amount-drive simulator which can generate the amount of a transaction for each cardholder. We also assume that a fraudster is familiar with a cardholder's transaction amount since the fraudster can obtain the history information of credit card. Hence, the fraudulent transaction amount is similar to normal transactions, but the only difference among them is the frequency of transactions in a period of time since the fraudster always wants to consume as soon as possible. Table III shows the simulation parameter settings where each consumption group has different amount mean (μ_g) and deviation (δ_g) and finally these groups with these parameters make up one of the whole dataset.

However, the above simulator only constructs transaction amount. We also need other transaction features, such as weekly and seasonal patterns. Since the average frequency of transactions in a week/month keeps a fixed value as introduced in [11]. We use a poisson process [27] to capture these practical situations. For example, Table III shows one of the λ_g which is a normal weekly pattern. We assume that a cardholder has 50 transactions in a month and each month has four weeks. Hence, the number of transaction is different in each week, i.e., $\lambda_g = [0.6, 0.2, 0.15, 0.05]$ is, respectively, the ratios, i.e., the number of transactions in a week accounts for the whole month. Then we will get [30, 10, 7, 3] transactions by λ_g for each week in a month. Finally, as Fig. 4 shows, the second block can use poisson distribution to simulate the transaction time for each week based on λ_g. Furthermore, three main categories of cardholders have their own frequency pattern of

TABLE III
SIMULATION PARAMETER SETTINGS

Simulator settings	Parameter values							
	μ_g	μ_f	δ_g	δ_f	q_{gf}	q_{fg}	λ_g	λ_f
low consumption group	1170	1110	234	220	0.1	0.9	$[0.6, 0.2, 0.15, 0.05]$	$[0.25, 0.25, 0.25, 0.25]$
medium consumption group	5000	6000	1200	1000			$[0.05, 0.4, 0.5, 0.05]$	
high consumption group	10000	11200	2000	2240			$[0.05, 0.15, 0.2, 0.6]$	

TABLE IV
CLASSIFICATION CONFUSION MATRIX

	Actual positive y = 1	Actual negative y = 0
Predict positive $pred = 1$	True positive(TP)	False positive(FP)
Predict negative $pred = 0$	False negative(FN)	True negative(TN)

transactions which are low, medium, and high, respectively. To simulate concept drift, we introduce two main types of concept drifts as mentioned in [28], namely, incremental and sudden. Incremental drift involves changes happened slowly over time. Sudden drift represents cases that the variable class assignment is changed instantly and abruptly. In this paper, we use incremental drift cases as normal transactions and sudden drift cases as fraud transactions, because a sudden change generally means that something unexpected have happened in recent. Thus, we inject two kinds of transactions to some cardholders as fraud: 1) fraud transactions from fraudsters and 2) transactions that happened in history but inconsistent with the recent transaction behaviors. Finally, the parameters q_{gf} and q_{fg} can control the ratio between normal transactions and fraudulent ones.

A confusion matrix is shown in Table IV, where *y* is the True label of the transaction, and *pred* is the predicted label. We use the standard binary classification metrics, such as true positive (TP), false positive (FP), false negative (FN), true negative (TN) [20], receiver operating characteristic (ROC) [21] and area under the ROC curve statistics [22], to measure the effectiveness of the proposed method. TP represents the number of fraudulent transactions correctly predicted as fraudulent, while FP is the number of genuine transactions predicted as fraudulent. For most of the methods, when they result in a high value of TP, they can also cause the increase of FP. By using the metrics in Table IV, *Accuracy* and *Recall* can be calculated as follows:

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{7}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{8}$$

A. Experiment Design

Since there are two parameters in our fraud detection method which are the size *p* of the sliding window and the number *k* of behavioral patterns, we need to change one at a time while keeping another one unchanged. In Section III, we have discussed the size of the sliding window is 50 due to strong weekly and monthly periodic structure from the observation on the simulation data. In experiments, *p* is varied from 25 to 100 in steps of 4. The threshold values is set by 10%, 30%, 50%, 70%, and 90%. *k* is varied from 1 to 11 in steps of 6. Thus, there are $4 \times 6 = 24$ possible combinations.

We only show the summarized results since it is not convenient to present each result of the 24 combinations. In Table V we present the results for each value of sliding window size *p* averaged over all the six kinds of *k*. Similarly, in Table VI we show the results for each value of *k* averaged over all the four sliding window sizes. As mentioned above, *Accuracy* presents the whole efficiency of the model, *Recall* presents the fraction that the fraudulent transactions can be detected as fraudulent by the proposed method.

In Tables V and VI, the highest value of *Accuracy* and *Recall* have been highlighted for each row. From Table V it can be seen that *Accuracy* shows a clear increasing trend with the increasing of the threshold. The sliding window size has no influence on *Acurracy*, but the highest value of *Recall* always takes place when the sliding window size is 50 or 100. Therefore, we can conclude that *Recall* can be periodically influenced by the sliding window size that is important to decide each cardholder's behavioral period. From the Table VI we can conclude that a smaller *k* and a higher threshold can lead to a higher *Accuracy*. However, a smaller *k* and a higher threshold can result in a lower *Recall*. Therefore, we should choose an appropriate threshold and *k* in order to keep two indicators as high as possible. From Table V it is seen that *Recall* is high for window size 50% in 60% of the cases and *Accuracy* is high for the same window size in 50% of the cases. Hence, we choose 50 as the size of sliding window for optimum performance. From Table V the threshold could be set to either 50% or 70%. Although *Accuracy* is higher for threshold = 70%, *Recall* is decreased more than 50%. To maximize *Accuracy* and *Recall* as high as possible, we choose threshold = 50%. Due to the unclear indication from the above summarized information presented in Tables V and VI, we use $G{-}mean$ [23] to take a detailed look at data when threshold = 50%, as shown in Table VII.

It is seen that when $p = 50$, the highest value of $G - mean$ occurs for $k = 7$. Thus, our parameters' setting is given as follows.

1) The size of sliding window $p = 50$.
2) The number of behavioral patterns $k = 7$.
3) Threshold value = 50%.

We have also analyzed the time taken by the fraud detection phase, which is performed online for each cardholder's incoming transaction. Fig. 5 shows the plot of model detection time against the size of sliding window. As the size of

TABLE V
VARIATION OF ACCURACY AND RECALL FOR DIFFERENT SLIDING WINDOW SIZES

threshold(%)	Accuracy average over all the 6 kinds of k for different sliding window sizes				Recall average over all the 6 kinds of k for different sliding window sizes			
	25	**50**	**75**	**100**	**25**	**50**	**75**	**100**
10	**0.6852**	0.5889	0.6183	0.5757	0.6131	0.7418	0.7390	**0.7556**
30	**0.7319**	0.6954	0.7021	0.6571	0.4546	0.6055	0.5839	**0.6123**
50	0.7859	**0.8181**	0.7844	0.7891	0.3297	**0.4174**	0.3746	0.3874
70	0.8506	**0.8782**	0.8539	0.8732	0.1472	**0.1531**	0.1395	0.1093
90	0.8951	0.8997	0.8969	**0.9000**	0.0441	**0.0444**	0.0346	0.0262

TABLE VI
VARIATION OF ACCURACY AND RECALL FOR DIFFERENT k (THE NUMBER OF BEHAVIORAL PATTERNS)

threshold(%)	Accuracy average over all the 4 sliding window sizes for different k						Recall average over all the 4 sliding window sizes for different k					
	1	**3**	**5**	**7**	**9**	**11**	**1**	**3**	**5**	**7**	**9**	**11**
10	**0.8973**	0.8542	0.7094	0.5260	0.4012	0.3033	0.1229	0.6931	0.8114	0.8601	0.8923	**0.9002**
30	**0.9032**	0.8666	0.7700	0.6540	0.5240	0.4483	0.0651	0.4670	0.6291	0.6933	0.7661	**0.7835**
50	**0.9032**	0.8815	0.8420	0.7779	0.7185	0.6524	0.0396	0.2748	0.4000	0.4647	0.5342	**0.5682**
70	**0.9018**	0.8962	0.8782	0.8568	0.8455	0.8163	0.0164	0.1014	0.1395	0.1556	0.1541	**0.2340**
90	**0.9006**	0.9021	0.8999	0.8975	0.8973	0.8921	0.0035	0.0330	0.0403	0.0468	0.0432	**0.0535**

TABLE VII
DETAILED RESULT OF $G-$MEAN FOR THRESHOLD = 50%

k	Sliding Window Size p			
	25	**50**	**75**	**100**
1	0.2376	0.2146	0.1873	0.1435
3	0.4699	0.5632	0.5394	0.4625
5	0.5438	0.6244	0.6045	0.6135
7	0.5610	**0.6748**	0.5934	0.6268
9	0.5710	0.6600	0.5841	0.6396
11	0.5841	0.6471	0.6013	0.6161

sliding window increases, the sliding-window-based algorithm performs more stably and quickly than the one without sliding window.

After setting the parameters, we study the performance of our method compared with two state-of-the-art ones under various combinations of input data.

B. Performance Evaluation

Shen *et al.* [6] provided a useful framework to choose the best model for fraud detection and their result shows that logistic regression [25] performs well. Logistic regression is similar to a linear regression but is suitable for the case that the dependent variable is dichotomous. Therefore, logistic regression is tested over the raw data (i.e., disaggregated data).

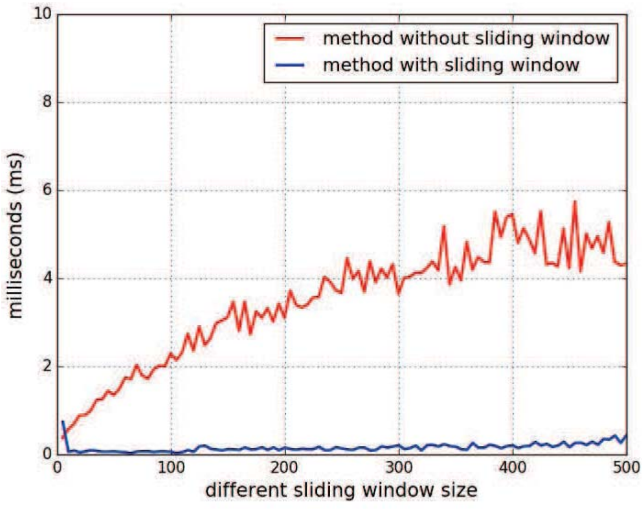


Fig. 5. Average time (ms) to model detection and different sliding window size.

Whitrow *et al.* [24] indicated that transactions aggregation is particularly effective when random forest [26] is used for classification. Therefore, random forest is tested just like ours, over the aggregated data. For convenience, the two methods and ours are, respectively, represented as follows.

1) Logistic regression with raw data (*RawLR*).
2) Random forest with aggregated data (*AggRF*).
3) Random forest and feedback technique with aggregated data (*AggRF+FB*) that is our method.

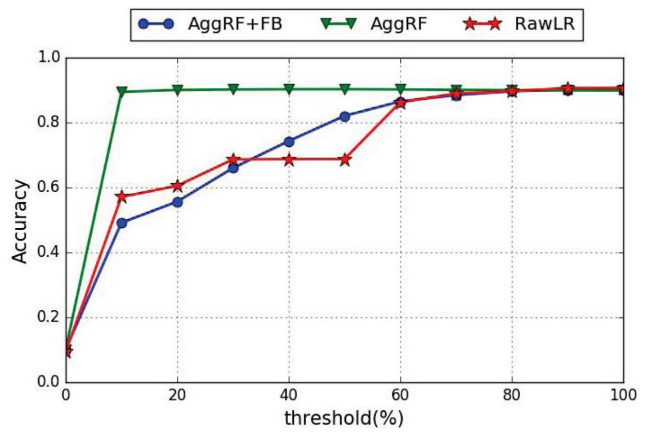


Fig. 6. Accuracy comparison of three different methods, trained by *AggRF+FB, AggRF,* and *RawLR*.

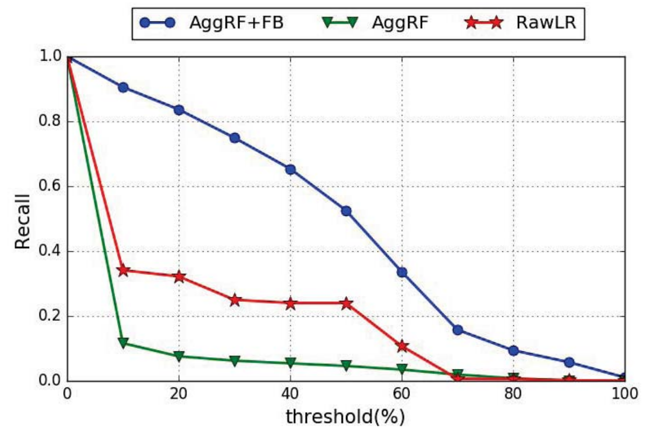


Fig. 7. Recall comparison of three different methods, trained by *AggRF+FB, AggRF,* and *RawLR*.

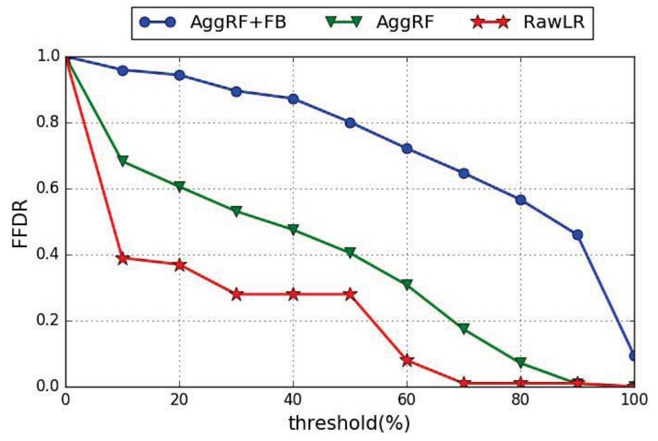


Fig. 8. FFDR comparison of three different methods, trained by *AggRF+FB, AggRF,* and *RawLR*.

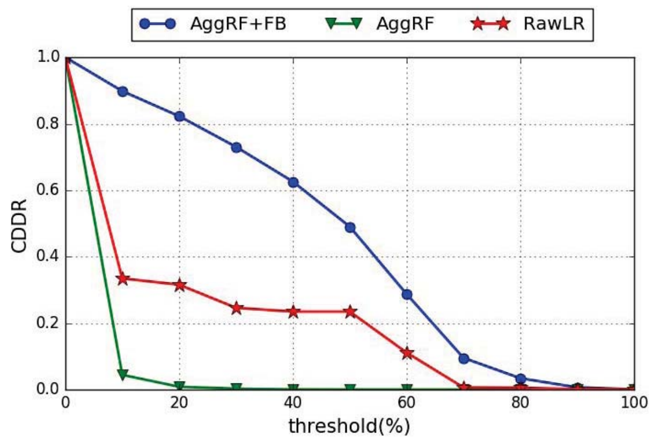


Fig. 9. CDDR comparison of three different methods, trained by *AggRF+FB, AggRF,* and *RawLR*.

As mentioned before, the fraudulent transactions compose of two different parts: 1) label 1 and 2) label 2. Label 1 represents fraudulent transactions happened on the cardholder due to fraud features. Label 2 represents fraudulent transactions due to the sudden concept drift. We define two indications: 1) fraud features detection rate (FFDR) and 2) concept drift detection rate (CDDR). FFDR is the fraction of the transactions ($label = 1$) identified as fraudulent, and CDDR is the fraction of the transactions ($label = 2$) identified as fraudulent. Then we evaluate the three methods in the same data set with different thresholds. Evaluation indicators include *Accuracy*, *Recall*, FFDR, and CDDR.

There are 8.33% transactions with $label = 2$ and 1.14% transactions with $label = 1$ in the data set trained by three methods. Our main purpose is to make both FFDR and CDDR as high as possible. The results are shown in Figs. 6–9. It is observed by Fig. 6 that *Accuracy* of the model trained by *AggRF* are better than other two methods. However, *Recall* are worst. As shown in Fig. 7, *AggRF+FB* performs best. It is interesting that *AggRF* performs better than *RawLR* on FFDR but worse on CDDR. One of the reason is that the fraud features can be portrayed more precisely with aggregated data. Hence, transactions which occur in history are identified as genuine due to the sudden concept drift. In Figs. 8 and 9, it is seen that *AggRF+FB* performs both best on both FFDR and CDDR. The value of FFDR increases 97.8% in comparison with *AggRF* and the value of CDDR increases 109% in comparison with *RawLR* when threshold $= 50\%$.

However, it is seen that the trends of each curve in Figs. 7 and 9 are similar due to the large ratio of the concept drift data. In real data set, the transactions due to the concept drift may take up a small amount of data. Hence, we simulate four different datasets where the ratio of the concept

TABLE VIII
FOUR DIFFERENT TRANSACTIONS DATA SETS

	All transactions	The concept drift transactions $label = 2$	Fraud features transactions $label = 1$
dataset 1	100%	0%	1.14%
dataset 2	100%	8.33%	1.14%
dataset 3	100%	16.67%	1.14%
dataset 4	100%	25%	1.14%

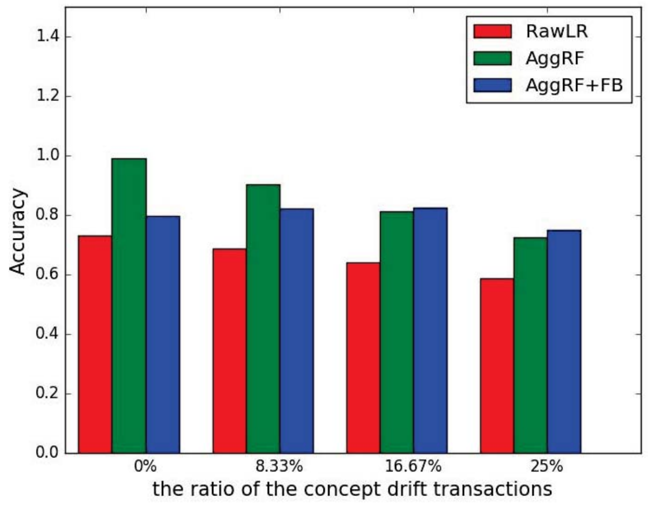


Fig. 10. Accuracy comparison of four different datasets, trained by *AggRF+FB, AggRF,* and *RawLR* when threshold = 50%.

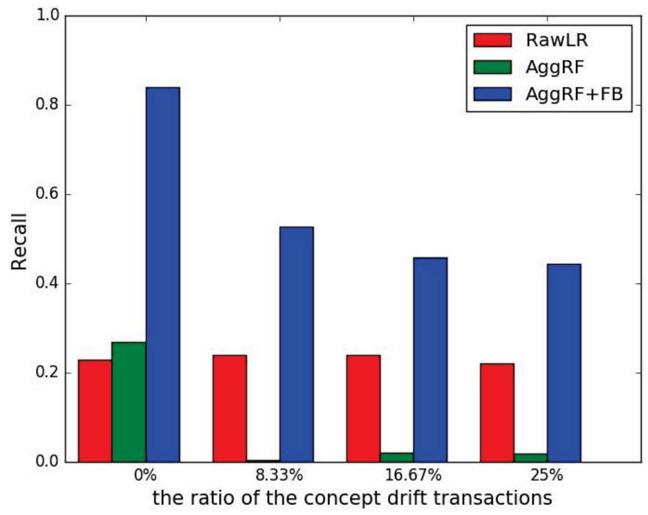


Fig. 11. Recall comparison of four different data sets, trained by *AggRF+FB, AggRF,* and *RawLR* when threshold = 50%.

drift transactions are, respectively, 0%, 8.33%, 16.67%, and 25% and it can be seen in Table VIII.

The results are shown in Fig. 10. It is observed that *AggRF* performs well on *Accuracy* and *AggRF+FB* is better than it when the ratio of the concept drift transactions is higher. However, in Fig. 11 *AggRF* performs better than *RawLR* only when ratio = 0%. *AggRF+FB* is the best compared with them on *Recall*.

V. Conclusion

In this paper, we propose a novel fraud detection method. We utilize the behavioral patterns from the similar cardholders to build a recent behavioral profile of a cardholder. In this way, we propose a method to solve the adaptive capacity of the model. A feedback mechanism can make full use of the True label information from transactions in order to solve the concept drift problem. The classifier will adjust its own rating score according to a series of incoming transactions. This on-line fraud detection method can dynamically change its parameters to adapt to a cardholder's transactions behaviors timely. Experimental results show the performance and effectiveness of our method. Compared with other two methods, all of them can achieve 80% accuracy at the detection of transactions. However, *AggRF* only performs well on FFDR, and (RawLR) only performs well on CDDR. Our proposed method can enhance the performance on both FFDR and CDDR, and increase the average recall and accuracy.

The future work aims to design an individual periodic time window p in order to continue to improve the performance of fraud detection.

Acknowledgment

The authors would like to thank the four reviewers for their constructive comments.

References

[1] NilsonReport. (Oct. 2016). *The Nilson Report*. [Online]. Available: https://www.nilsonreport.com/upload/content_promo/The_Nilson_Report_10-17-2016.pdf
[2] M. Masud, J. Gao, L. Khan, J. Han, and B. M. Thuraisingham, "Classification and novel class detection in concept-drifting data streams under time constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 6, pp. 859–874, Jun. 2011.
[3] D. Malekian and M. R. Hashemi, "An adaptive profile based fraud detection framework for handling concept drift," in *Proc. Int. Conf. Inf. Security Cryptol.*, Yazd, Iran, 2013, pp. 1–6.
[4] Q. Wei, Z. Yang, Z. Junping, and W. Yong, "Mining multi-label concept-drifting data streams using ensemble classifiers," in *Proc. IEEE Int. Conf. Fuzzy Syst. Knowl. Disc.*, Tianjin, China, 2003, pp. 275–279.
[5] R.-C. Chen, S.-T. Luo, X. Liang, and V. C. S. Lee, "Personalized approach based on SVM and ANN for detecting credit card fraud," in *Proc. IEEE Int. Conf. Neural Netw. Brain*, Beijing, China, 2005, pp. 810–815.
[6] A. Shen, R. Tong, and Y. Deng, "Application of classification models on credit card fraud detection," in *Proc. IEEE Int. Conf. Service Syst. Service Manage.*, Chengdu, China, 2007, pp. 1–4.
[7] J. T. S. Quah and M. Sriganesh, "Real time credit card fraud detection using computational intelligence," in *Proc. IEEE Int. Conf. Neural Netw.*, Orlando, FL, USA, 2007, pp. 863–868.
[8] A. Srivastava, A. Kundu, S. Sural, and A. Majumdar, "Credit card fraud detection using hidden Markov model," *IEEE Trans. Depend. Secure Comput.*, vol. 5, no. 1, pp. 37–48, Jan./Mar. 2008.
[9] S. Panigrahi, A. Kundu, S. Sural, and A. K. Majumdar, "Credit card fraud detection: A fusion approach using Dempster–Shafer theory and Bayesian learning," *Inf. Fusion*, vol. 10, no. 4, pp. 354–363, 2009.
[10] L. Seyedhossein and M. R. Hashemi, "Mining information from credit card time series for timelier fraud detection," in *Proc. IEEE Int. Conf. Telecommun. (IST)*, Tehran, Iran, 2010, pp. 619–624.
[11] A. C. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, "Feature engineering strategies for credit card fraud detection," *Expert Syst. Appl. Int. J.*, vol. 51, pp. 134–142, Jun. 2016.
[12] V. V. Vlasselaer *et al.*, "APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions," *Decis. Support Syst.*, vol. 75, pp. 38–48, 2015.
[13] R. N. Gurjar, N. Sharma, and M. Wadhwa, "Finding outliers using mutual nearness based ranks detection algorithm," in *Proc. IEEE Int. Conf. Optim. Rel. Inf. Technol. (ICROIT)*, Faridabad, India, 2014, pp. 141–144.
[14] V. R. Ganji and S. N. P. Mannem, "Credit card fraud detection using anti-K nearest neighbor algorithm," *Int. J. Comput. Sci. Eng.*, vol. 4, no. 6, 2012, Art. no. 1035.
[15] W. N. Robinson and A. Aria, "Sequential fraud detection for prepaid cards using hidden Markov model divergence," *Expert Syst. Appl.*, vol. 91, pp. 235–251, Jan. 2018.
[16] S. Kullback, *Information Theory and Statistics* (Dover Books on Mathematics). New York, NY, USA: Wiley, 1959.

[17] T. K. Behera and S. Panigrahi, "Credit card fraud detection: A hybrid approach using fuzzy clustering neural & network," in *Proc. IEEE Int. Conf. Adv. Comput. Commun. Eng.*, Dehradun, India, 2015, pp. 494–499.
[18] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means clustering algorithm," *J. Roy. Stat. Soc.*, vol. 28, no. 1, pp. 100–108, 1979.
[19] C.-H. Lee, C.-R. Lin, and M.-S. Chen, "Sliding-window filtering: An efficient algorithm for incremental mining," in *Proc. ACM Int. Inf. Knowl. Manag.*, Atlanta, GA, USA, 2001, pp. 263–270.
[20] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the JAM project," in *Proc. DARPA Inf. Survivability Conf. Expo.*, vol. 2, 2000 pp. 130–144.
[21] R. J. Bolton and D. J. Hand, "Statistical fraud detection: A review," *Stat. Sci.*, vol. 17, no. 3, pp. 235–249, 2002.
[22] A. D. Pozzolo, O. Caelen, Y.-A. Le Borgne, S. Waterschoot, and G. Bontempi, "Learned lessons in credit card fraud detection from a practitioner perspective," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4915–4928, 2014.
[23] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Mach. Learn.*, vol. 30, nos. 2–3, pp. 195–215, 1998.
[24] C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, and N. M. Adams, "Transaction aggregation as a strategy for credit card fraud detection," *Data Min. Knowl. Disc.*, vol. 18, no. 1, pp. 30–35, 2009.
[25] A. M. Flitman, "Towards analysing student failures: Neural networks compared with regression analysis and multiple discriminant analysis," *Comput. Oper. Res.*, vol. 24, no. 4, pp. 367–377, 1997.
[26] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[27] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," *Perform. Eval.*, vol. 18, no. 2, pp. 149–171, 1993.
[28] D. Brzeziński, "Mining data streams with concept drift," M.S. thesis, Inst. Comput. Sci., Poznań Univ. Technol., Poznań, Poland, 2010.

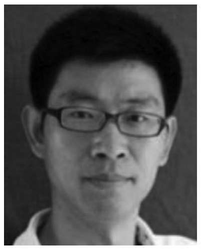
Changjun Jiang received the Ph.D. degree from the Institute of Automation, Chinese Academy of Science, Beijing, China, in 1995.

He is currently the Leader of the Key Laboratory of the Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai, China. He is also a Research Fellow with the City University of Hong Kong, Hong Kong, and a Specialist of the information area of Shanghai. He has led over 30 projects supported by the National Nature Science Foundation, National High Technology Research and Development Program, and National Basic Research Developing Program. He has authored over 300 papers in journals and conference proceedings, including *Chinese Science*, the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, and the IEEE TRANSACTIONS ON FUZZY SYSTEMS. His current research interests include concurrency theory, Petri nets, formal verification of software, cluster, grid technology, intelligent transportation systems, and service-oriented computing.

Dr. Jiang was a recipient of the International Prize and seven national prizes in the field of science and technology.

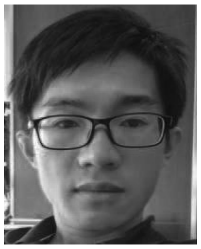
Jiahui Song received the B.S. degree in computer science and technology from Shanghai University, Shanghai, China, in 2016. He is currently pursuing the M.S. degree in computer software and theory at Tongji University, Shanghai, China.

His current research interests include machine learning, deep learning, and natural language processing.



Guanjun Liu (M'16) received the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2011.

He was a Post-Doctoral Research Fellow with the Singapore University of Technology and Design, Singapore, from 2011 to 2013. He was with the Humboldt University of Berlin, Berlin, Germany, from 2013 to 2014 as a Post-Doctoral Research Fellow supported by the Alexander von Humboldt Foundation. He is currently an Associate Professor with the Department of Computer Science, Tongji University. He has authored or co-authored over 60 papers including 13 in IEEE/ACM TRANSACTIONS. His current research interests include Petri net theory, model checking, Web service, workflow, discrete event systems, and information security.



Lutao Zheng received the M.S. degree in computer science and technology from Anhui Normal University, Wuhu, China, in 2015. He is currently pursuing the Ph.D. degree in computer software and theory at Tongji University, Shanghai, China.

His current research interests include credit card fraud detection, machine learning, deep learning, big data, and transfer learning, and natural language processing.

Wenjing Luan (S'16) received the B.S. and M.S. degrees from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively. She is currently pursuing the Ph.D. degree in computer software and theory at Tongji University, Shanghai, China.

Her current research interests include cyberspace security, transportation systems, and smart city.

Ms. Luan was a recipient of the Best Student Paper Award–Finalist of the 13th IEEE International Conference on Networking, Sensing and Control Conference.