Complex Systems Informatics and Modeling Quarterly (CSIMQ)
eISSN: 2255-9922
Published online by RTU Press, https://csimq-journals.rtu.lv
Article 118, Issue 20, September/October 2019, Pages 72–88
https://doi.org/10.7250/csimq.2019-20.04

Fraud Detection in Payments Transactions: Overview of Existing Approaches and Usage for Instant Payments

Alexander Diadiushkin[1,2*], Kurt Sandkuhl[1,2] and Alexander Maiatin[2]

[1] University of Rostock, Albert-Einstein-Str. 22, 18059 Rostock, Germany
[2] ITMO University, St. Petersburg, Russia

dyadyushkin.a@yandex.ru, kurt.sandkuhl@uni-rostock.de, mayatin@mail.ifmo.ru

Abstract. Financial industries are undergoing a digital transformation of their products, services, overall business models. Part of this digitalization in banking aims at automating most of the manual work in payment handling and integrating the workflows of involved service providers. The focus of the work presented in this paper is on fraud discovery and steps to fully automate it. Fraud discovery in financial transactions has become an important priority for banks. Fraud is increasing significantly with the expansion of modern technology and global communication, which results in substantial damages for the banks. Instant payment (IP) transactions cause new challenges for fraud detection due to the requirement of short processing time. The paper investigates the possibility to use artificial intelligence in IP fraud detection. The main contributions of our work are (a) an analysis of problem relevance from business and literature perspective, (b) a proposal for technological support for using AI in fraud detection of instant payment transactions, and (c) a feasibility study of selected fraud detection approaches.

Keywords: Artificial Intelligence, Enterprise Modeling, Digital Transformation, Instant Payment.

1 Introduction

Financial industries are currently undergoing a change process that many researchers consider as digital transformation. From a business-centric perspective, digital transformation focuses in general on the transformation of products, processes, and organizational aspects triggered by new technologies [1]. This opens up a variety of opportunities for changing business models and value chains in order to meet constantly increasing customer requirements and offer services faster, more intelligently and more efficiently. For the financial sector, examples for new products and services are robot advisory and auto-trading, value-added services based on account information and transaction history, or ad-hoc loans in online-banking. Many of these services

[*] Corresponding author

© 2019 Alexander Diadiushkin et al. This is an open access article licensed under the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0).

Reference: A. Diadiushkin, Kurt Sandkuhl and A. Maiatin, "Fraud Detection in Payments Transactions: Overview of Existing Approaches and Usage for Instant Payments," Complex Systems Informatics and Modeling Quarterly, CSIMQ, no. 20, pp. 72–88, 2019. Available: https://doi.org/10.7250/csimq.2019-20.04
Additional information. Author's ORCID iD: K. Sandkuhl – https://orcid.org/0000-0002-7431-8412. PII S225599221900118X. Received: 25September 2019. Accepted: 28 October 2019. Available online: 31 October 2019.

are facilitated by applications of Artificial intelligence (AI) approaches, which provide the necessary functionality for automating certain steps of work processes or the overall end-to-end process. However, among the prospective users of AI and the decision-makers in organizations, there is often no clear picture of how AI should be put into operation and where the limits are [2].

The focus of this work is a specific aspect of digital transformation, which concerns both, new kinds of services and the application of AI in these services. More concrete, the focus is on the newly established instant payment (IP) service and ways of fully automating fraud discovery in IP transactions. Fraud discovery in financial transactions has become an important priority for banks. Fraud is increasing significantly with the expansion of modern technology and global communication, which results in substantial damages for the banks and new regulations. Instant payments are expected to bring a new complexity to fraud detection; the European Central Bank and Central Bank of the Russian Federation have already proposed the introduction of IP systems. Compared to conventional Single Europe Payment Area (SEPA) transactions, in instant payments fraud detection has to be completed within a few seconds instead of a day or more. New technological approaches are required to achieve this goal.

In the above context, the article is an extended version of a paper presented at the ILOG 2019 workshop in the context of BIR 2019 conference in Katowice, Poland: Alexander Diadiushkin, Kurt Sandkuhl and Alexander Maiatin: Fraud Detection in Instant Payments as Contribution to Digitalization in Banks. Joint Proceedings of the BIR 2019 Workshops and Doctoral Consortium.pp. 107-117 (http://ceur-ws.org/Vol-2443/) and aims at a contribution to quick fraud discovery by investigating, which approaches can be utilized in the real-world fraud detection task. For this purpose, publications about existing approaches were analyzed to explore their utilization in the area of instant payments. Two approaches were selected for implementation with an explicit focus on efficiency. To evaluate performance in terms of speed and precision, a benchmarking of the approaches was performed.

The main contributions of our work are (a) an analysis of problem relevance from business and literature perspective, (b) a proposal for technological support for using AI in fraud detection of instant payment transactions, and (c) a feasibility study of selected fraud detection approaches. The remainder of this article structured as follows: Section 2 summarizes the foundation for our work from fraud detection in payment transactions including important terms. Section 3 introduces the research approach taken. Section 4 investigates the problem’s relevance. Section 5 is dedicated to fraud detection and the feasibility study. Section 6 summarizes the results and gives an outlook on future work.

2 Theoretical Foundations

2.1 Instant Payments

Originally, banks could take their time to process a payment transaction order. The procedure might take hours and even days. Formally, it consists of clearing and settlement of order. The clearing is a process of transmitting, reconciling and, in some cases, confirming transfer orders prior to settlement. The settlement is the completion of a transaction or processing with the aim of discharging participants’ obligations through the transfer of funds [3].

To reduce the amount of time it takes to proceed with an order, the European Central Bank and Central Bank of Russia developed the proposal of instant payment systems [4], [5]. Instant payments will dramatically increase the speed at which payments are made and received in Euro in the European Union. Today it normally takes one business day for a payment to reach the beneficiary. With instant payments, this will happen in real-time, 24 hours a day, 365 days a year. The funds will be available immediately for use by the recipient.

The Euro Retail Payments Board (ERPB) [6] has defined instant payments as "electronic retail payment solutions available 24/7/365 and resulting in the immediate or close-to-immediate interbank clearing of the transaction and crediting of the payee’s account with confirmation to the payer (within seconds of payment initiation)". This is irrespective of the underlying payment instrument used (credit transfer, direct debit or payment card) and of the underlying arrangements for clearing (whether bilateral interbank clearing or clearing via infrastructures) and settlement (e.g. with guarantees or in real-time) that make this possible [4].

As described by the Committee on Payments and Market Infrastructures [7], the idea of “instant” or sometimes called “fast” is not new. Technically, speed comes from instant clearing of the transaction, and only the settlement process is being delayed. According to Mastercard [8], such an approach is the default for many countries, but not for Europe, and called Single-Message clearing, during which authorization and clearing in payment network is done in one dispatch. On the contrary, Dual-Message clearing separates authorization and clearing processes in time [9].

2.2 Bank Fraud

Fraud is wrongful or criminal deception intended to result in financial or personal gain [10]. Thus, bank fraud is commonly described as a criminal act that occurs when a person uses illegal means to receive money or assets from a bank or other financial institution. Bank fraud is distinguished from bank robbery by the fact that the perpetrator keeps the crime secret, in the hope that no one notices until he has gotten away. The term bank fraud also refers to attempts by a person to obtain money from a bank’s depositors by falsely pretending to be a bank or financial institution [11].

In the work, we focus on bank fraud cases, related to instant payment systems. Mainly, on identity thieves, stealing, duplication or skimming of card information, which may often be the result of phishing and Internet fraud. In other words, our main attention is on fraud approaches that utilize genuine payment card credentials.

In 2016, total fraud involving Single European Payment Area (also known as SEPA: the EU Member States plus Switzerland, Iceland, Lichtenstein, Norway) cards decreased to 1.8 billion euros, which is 0.8% less than in 2015. Card fraud at ATMs dropped by 12.4% and online fraud rose significantly, accounting for 73% of the total value of card fraud in 2016. One Euro for every 2,428 Euros spent on payment cards was lost to fraud. In relative terms, i.e. as a share of the total value of card transactions of 4.38 trillion euros, fraud dropped by 0.001 percentage point to 0.041% in 2016, down from 0.042% in 2015. This is the first decrease since 2011 [12].

Online card fraud is naturally increasing as digital services develop further and are becoming more and more sophisticated. The most common types of online fraud reported by the industry are “clean fraud” – where criminals obtain genuine cardholder details including 3D Secure and Address Verification credentials – and “identity theft” – where the fraudster steals the cardholder’s personal data in order to make unauthorized online transactions. However, in recent years there has been an increase in “friendly fraud”, where the payer first makes a genuine transaction then claims that their card has been used fraudulently and asks for money back [13].

2.3 Fraud Discovery Approaches

In this section, an overview of related works found in public access is presented. In summary, more than 40 papers on fraud detection were analyzed in the process of collecting related researches [14]. Quality highly varies between them; some even do not present any implementation or lack well-defined example of evaluation. Correlation to the banking fraud also divides into fraud in the area of loan approvals and area of transactions, sometimes even datasets from one area applied for evaluation of an approach for another, which seems to be not appropriate. A short overview of selected papers is presented below.

Vishwakarma et al. [15] propose an approach for fraud analytics for the NFC-enabled mobile payment system. A multi-layer solution is presented where each subsequent layer is responsible for separated parts of fraud analysis. However, the article presents only a generic view on the problem and its solution, avoiding implementation at all.

Kultur et al. [16] propose a novel cardholder behaviour model for detecting credit card fraud. They propose building a model by clustering transaction amounts of a user, with respect to merchant category code (MCC) of the transaction, using the Expectation Maximization algorithm. The evaluation was done on a real-world dataset provided by a leading bank in Turkey, which is not available in public. The proposed approach showed the detection of 43% of fraud transactions, presented in the dataset. However, no information about the application of this approach in the real-world was provided.

Carminati et al. [17] propose a supervised auto-tuning approach for a banking fraud detection system, called Banksealer [18]. They describe the application of the Multi-Objective Genetic Algorithm (MOGA) for the task of feature weighting task, this way freeing end-users from the need in the manual configuration of this unsupervised system. This gain up to 35% of performance in detecting some sophisticated fraud cases.

Patil et al. [19] implemented a supervised artificial neural network with back propagation algorithm for the purpose of classifying transactions for fraud detection. Experimental evaluation was made on an old dataset of applications for credit loans, which seems to be unrelated to the task of fraud detection in bank transactions. However, accuracy up to 98% was shown during evaluation.

Hatamikhal et al. [20] present a concept drift detection solution based on streaming ensemble algorithm with deep belief network utilized in it. Concept drift problem highly affects fraud detection due to variable user's behaviour. The evaluation was done on MNIST and SEA datasets, comparing the proposed solution with Morelli's method. The F1-score of the proposed method for the evaluation is 50.41%.

Nami et al. [21] developed a two-stage approach for fraud detection. In the first stage, the kNN algorithm was utilized to rate similarity between past user's transactions and incoming ones. In the second stage, dynamic random forest algorithm was applied for initial detection along with the minimum-risk model for cost-sensitive fraud detection. The evaluation was made on data from a private bank and future deployment in the real-world is only proposed for research.

Panigrahi et al. [22] built fraud detection systems that combine several approaches. Initially, the proposed approach checks for address mismatch and which outliers using the DBSCAN algorithm. Afterwards, the results of previous checks are combined using Dempster-Shafer adder. If the result falls into a certain threshold, additionally Bayesian learner was applied to make the optimal decisions. The evaluation was done on a synthetic dataset with up to 98% of true positive cases and less than 10% of false positive ones. No information about future applications in the real world is presented.

Wang et al. [23] propose the application of K-means clustering algorithm and Hidden Markov Model (HMM) for the purpose of fraud detection. K-means are utilized for translating incoming transactions into a set of symbols (states) for further application in HMM, which used to model user's behaviour. If incoming transaction highly deviates from expected user's behaviour, it is considered as fraudulent. Real-world bank data is utilized for evaluation; however, no further real-world application is mentioned.

Behera et al. [24] implemented a hybrid approach using fuzzy clustering and neural network. After initial user authentication and card details verification, a fuzzy c-means clustering algorithm is applied to perform initial transaction scoring. If the score falls into a certain threshold, the transaction is considered either legitimate, suspicious or fraudulent. In case of suspicious transactions, a neural network algorithm is applied to make the final decision. Evaluation of synthetic dataset showed up to 93.90% of true positive cases and less than 6.10% of false positive ones.

Wei et al. [25] proposed the detection of sophisticated online banking fraud on extremely imbalanced data. This solution is highly limited on online banking due to the utilization of ContrastMiner algorithm to examine deviations in user's behaviour on a bank's website (for instance, if after login user instantly traverses to a page different than the home page). Cost-sensitive neural network and random forest are utilized for the purpose of transaction scoring. The dataset for evaluation was provided by a major Australian bank, however, no sign of further application is provided.

Li et al. [26] proposed a solution for the selection of globally optimal (business) rules for detecting fraud. The proposed MCGminer algorithm is based on the Max Coverage Gain metric, which scores how good a rule performs globally. Datasets for evaluation were borrowed in public sources along with one provided by a major Australian bank. Algorithms were successfully deployed in the aforementioned bank.

Jarovsky et al. [27] focused on the problem of business rule sharing between financial institutions for the purpose of improving fraud detection efficiency. Their work allows rules defined in the context of one financial institution (say, a bank in the U.S.A.) being translated into the context of another financial institution (say, a bank in Europe) using GOLDRUSH algorithm. This allows better collaboration between these institutions for the purpose of fraud detection and prevention. For evaluation, real-world transactions are used from a private dataset. However, no sign of further application in the real world is presented.

Hormozi et al. [28] implemented a fraud detection system based on the Artificial Immune System algorithm called Negative Selection Algorithm. Parallel implementation of this algorithm based on Hadoop allowed to slightly improve training time and cut detection time almost in half. Evaluations were made on a private dataset from a large Brazilian bank, however, no further application of this solution is mentioned.

Dhankhad et al. [29] made a comparative study of supervised machine learning algorithms of their application for the fraud detection task. Ten algorithms were compared, among which Random Forest proved to be one of the best. Evaluations were done on the dataset from Kaggle.

Zheng et al. [30] demonstrate how user's behaviour profile can be represented in a form of a graph, where vertexes represent different values of certain transaction feature and weighted edges represent the correlation between them, i.e. how likely value of one attribute would be present if the value of another is. Incoming transactions are evaluated against this profile to measure how unlikely the user's current behavior is. Once again, the dataset from Kaggle is utilized in evaluation and no information about the further real-world applications is present.

3 Research Approach

The overall research paradigm we follow in our work is design science research (DSR) as proposed by [31]. The research goal is to investigate the potential of using AI as an element in the digitization of fraud detection in instant payments (IP) with a focus on confirming problem relevance and feasibility study. The artefact envisioned as a long-term result and thus in focus of our DSR project is method support for introducing AI in IP fraud discovery in combination with technological components implementing AI approaches.

Within the DSR frame, we use different research methods in different phases of the research work. Problem relevance is investigated by an interview study in different banks and financial service providers (see Section 4). This business-oriented aspect of the problem relevance is accompanied by a literature analysis to discover relevant existing work in the scientific body of knowledge (see Section 2). The main research question of the problem relevance investigation is "What challenges do organizations in the financial industry experience in implementing fraud detection in instant payments?".

As the problem relevance investigation confirms the need for changes in IP fraud detection, we propose an initial design of the envisioned technological support, i.e. the AI component. This initial version serves as a feasibility study for fraud detection in instant payment transactions applying AI. Lessons learned from the feasibility study and requirements derived from the interviews form input for the next design-evaluate cycle of the artefact. The initial version of the method is not discussed in this article but presented in related work [32].

4 Problem Relevance

The investigation of problem relevance was performed in two steps: first, we performed interviews with three different payment service providers about their way of performing fraud detection in conventional SEPA payments. The interviews were conducted on the basis of a structured questionnaire. The objective of the interviews was to understand which steps in conventional SEPA fraud detection could no longer be performed in instant payment fraud detection because of the short time frame. In SEPA payments, banks usually have one bank day for fraud detection, in instant payments – max. 10 seconds. Thus, the interviews aimed at gaining a better understanding of the process of processing suspected cases. As a result of the interviews, we discovered similar processes at all three organizations that simplified consists the following steps:

- A specific back-office software monitors all transactions and identifies "suspicious cases„ based on rule sets tailored for the payment service provider
- The transactions identified as "suspicious cases„ are assigned a rating that indicates the severity of the case
- The fraud officers at the payment service provider work on most severe cases first in an IT-supported but mostly manual process
- The investigation of suspicious cases includes checking payment history (amounts, recipients, geographic distribution, etc.) of the customer
- If the suspicion is confirmed, either the customer making the payment and/or the recipient's bank are contacted by phone call
- Based on the manual check, payment is blocked or released
- If a fraud case can be confirmed, all relevant information is documented and a police investigation is initiated.

For investigating the potential use of AI, it was also interesting to understand what information is available about suspicious cases. The fraud officer commonly receives or fetches the following information:

- Reason for displaying the suspicious case, e.g. known suspicion/fraud pattern, rule(s)
- Assessment result of the criticality of the suspected case, e.g. using multi-level scale
- Information about the triggers of the transaction, for instance, name, age, address data; transaction/sales history
- Information about the content and recipient of the transaction, such as account information of the trigger, amount, intended use, name and bank details of the recipient
- Any further information about the trigger of the transaction, such as the service agent in the bank assigned to the customer.

The basic process flow of fraud detection takes an average time for the manual parts between 5–10 minutes and up to 30 minutes for difficult cases. All interviewees confirmed that the above process is not applicable for IP due to the drastically reduced time frame of 10 seconds for the whole process. With millions of transactions performed every day, the payment service providers participating in the interview estimated the time frame available for checking a single transaction to a few milliseconds. This is confirmed by earlier discussions in the financial sector, for instance in a discussion paper by Mastercard [8].

Feasibility Study: Fraud Detection in IP Based on AI

Section 2.3 shows that there are many fraud detection approaches but that publications describing these approaches do not provide sufficient information for using or implementing them. The only exception detected in the analysis was the Banksealer approach. Thus, we decided to apply the Banksealer for the feasibility study. Furthermore, we aimed to understand performance issues in implementing AI-based fraud detection. In order to have a way of interpreting the performance of Banksealer, we decided to implement a second approach to compare with. Here, we selected a general approach, the random forest approach, as an element of the feasibility study.

As there are other general approaches that are potentially applicable in fraud detection (for instance Dempster-Shafer or K-means clustering), we followed a component-oriented software design offering the possibility to easily integrate implementations of additional approaches. More concrete, we designed generic interfaces to classifiers and score detectors that have to be specialized by the actual approach used – in our case Banksealer and random forest. The class diagrams provided in Section 5.3 provide more information and illustration of the design.

This section will first cover the random forest approach (Section 5.1), followed by more details on Banksealer (Section 5.2). Section 5.3 and 5.4 focus on implementation issues for both approaches and the evaluation, which includes a comparison.

4.1 Random Forest

Random Forest is an ensemble classifying algorithm that represents ensembles a collection of Decision Trees, each of which is built on a randomly selected set of features see an example from famous people domain in Figure 1. A decision tree is a tree where each node represents an attribute and edges following from it represent a condition, under which the edge can be traversed. On leaves of the tree target classes are located. The final decision is represented by the majority of results. It is easy to see that the model behind Random Forest can be easily visualized and analyzed for investigation, thus, results of classification can be explained in a reasonable amount of time.

Methods for the creation of Random Forest mainly consist of three approaches: bagging, random split and a random set of weights. Bagging is made by sapling original training data set randomly until a certain size is reached. This way, training data sets made by bagging may contain duplicates. For random splitting, a tree is built using K attributes from the training data set, selected at random. The last approach is similar to bagging but duplicates are represented by a weighting of instances – the more instance's weight, the more copies of it were sampled.

A decision is made by traversing the tree from root to leaf by a path that meets conditions associated with it. At the end, arriving at a leaf presents the result of the classification process. Random Forest runs classification on each tree it consists of during the runtime.

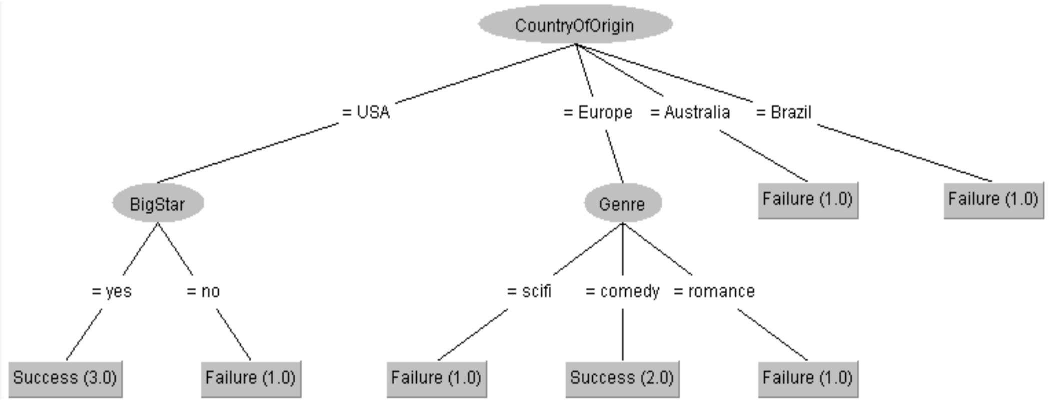


Figure 1. Example of Decision Tree visualization

4.2 Banksealer

On contrast with Random Forest, Banksealer [17] consists of a few major algorithms, helped by some tiny ones, altogether presenting an approach that is more complex (see Figure 2).

User’s behavior is represented by three types of profiles:

- Local profile – represents past user’s activity in the form of a histogram
- Temporal profile – represents statistic values of past user’s activity
- Global profile – aggregates by the similarity of their spending patterns

The local pattern is built by evaluating HBOS algorithms on past user’s transactions. The transaction has the following features: amount, country code of autonomous system of the client’s IP, client’s IP, **IBAN**, IBAN country code, timestamp, and **recipient**. Features highlighted boldly are hashed for privacy preservation.

Initially, a histogram is built for each feature separately, representing its distribution. For evaluating of new transaction HBOS [33] is calculated by Formula 1.

$$HBOS(t) = \sum_{i=0}^{d} \log\left(\frac{1}{hist_i(t)}\right) \tag{1}$$

Where $hist_i(p)$ represents score for i-th feature of transaction t, calculated by a histogram, associated with the feature.

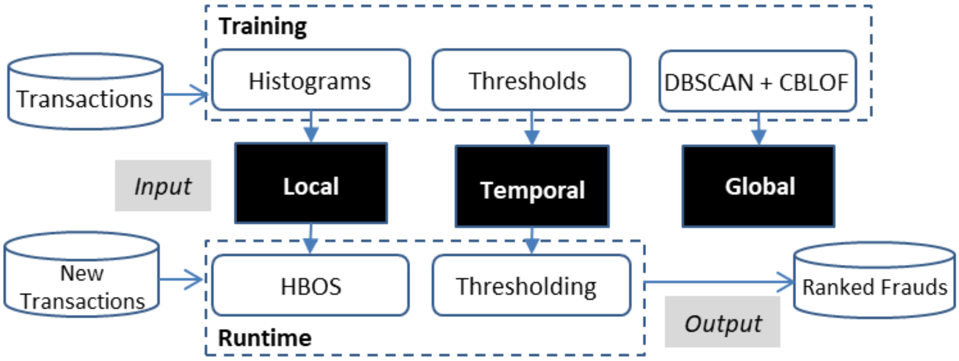


Figure 2. Banksealer architecture (adopted from [17])

The temporal profile is built by calculating statistic values, mean and standard deviation. The following features are extracted during training: total amount, total and maximum daily number of transactions. For each numerical feature, statistic values are calculated. A threshold is set to the sum of mean and standard deviation. During the evaluation of the new transaction, for each of the mentioned above features, cumulative value is calculated with a certain frequency. The positive delta between each cumulative value and the threshold sum up into the anomaly score.

Global profile builds by applying an iterative modification of the DBSCAN algorithm for the purpose of clustering users. Afterwards, CBLOF is applied to measure how a user’s behaviour deviates from the most common one.

DBSCAN [34] algorithm takes into account three parameters: distance metric, epsilon – radius in which algorithm searches point’s neighbours and the minimal number of neighbours in this radius, required it not to be noise. The modified version, used in Banksealer, iteratively applied DBSCAN to the larger cluster, each time lowering the search radius. This is done for the purpose of splitting larger clusters into smaller ones, to further present a more detailed view on how user’s behaviour patterns aggregate.

After building clusters, information about them passed into CBLOF [35] algorithm, along with three parameters: distance metric, alpha – the percentage of points that at least larger clusters must accumulate in summary and beta – for clusters, ordered by size, rate of sizes between larger and smaller subsequent clusters. Initially, all clusters are separated into large and small ones, according to incoming parameters. The process terminates as soon as one of the conditions is met. Afterwards, for each point score is calculated by Formula 2.

$$CBLOF(p) = \begin{cases} |C_i| \cdot \min\left(d(p, C_j)\right)\ if\ C_i\ \in SC\ where\ p\ \in\ C_i\ and\ C_j \in LC \\ |C_i| \cdot d(p, C_j)\ if\ C_i\ \in LC\ where\ p\ \in\ C_i \end{cases} \tag{2}$$

Where SC represents a set of small clusters and LC represents a set of large clusters. In other words, if a point belongs to a large cluster, its score will be the distance to the centre of the cluster. Otherwise, its score is the distance to the center of the closest large cluster. Mahalanobis distance [36] is utilized in the building of the global profile.

Resulting anomaly score is the sum of scores gained by the aforementioned approaches, multiplied by the transaction amount. Finally, the transactions are ordered by scores assigned to them for further investigation by analytics.

4.3 Implementation

For evaluation purpose, only the data set from Kaggle [37] was utilized in this work. The reason for this decision is that it consists of real-world transactions, ready to be utilized in the classification algorithm. The disadvantage of this data set is high anonymity, thus, it can’t be utilized fully for certain algorithms as they require some knowledge of users. Features, presented in this data set include the following information:

- Time – between the current transaction and first transaction in the data set,
- V1-V28 – anonymized features of the transaction,
- Amount – transaction amount,
- Class – a nominal attribute that classifies the transaction as fraudulent or not.

Since information about users is vital for correct evaluation of algorithms, it is necessary to look for synthetic data. Only one suitable simulator of bank transactions was found: PaySim [38].

For Random Forest, the implementation from WEKA [39] was chosen, as will be demonstrated further. WEKA is probably the most popular, open-source, production-ready library, that provide support of many algorithms. It supports many data formats and even connection to SQL databases via JDBC. Official GUI allows experimenting and the result visualization without the need for a single line of code, just like similar commercial products, for instance, RapidMiner. There are three main approaches to build Random Forest classifier: bagging, random split and random weighting. WEKA implementation of the algorithm supports combining of first and last approach with a random split.

Since parametrization of this algorithm may vary depending on incoming data, it is necessary to create a generic classification detector for high customization and de-duplication purpose. Generic interface for classifiers in WEKA is called Classifier. Source data in WEKA presented as a collection, named Instances, each of which is presented by interface Instance. Data attributes are represented by the attribute class. Figure 2 shows the class diagram of generic classifier detector implementation, which was part of the implementation.

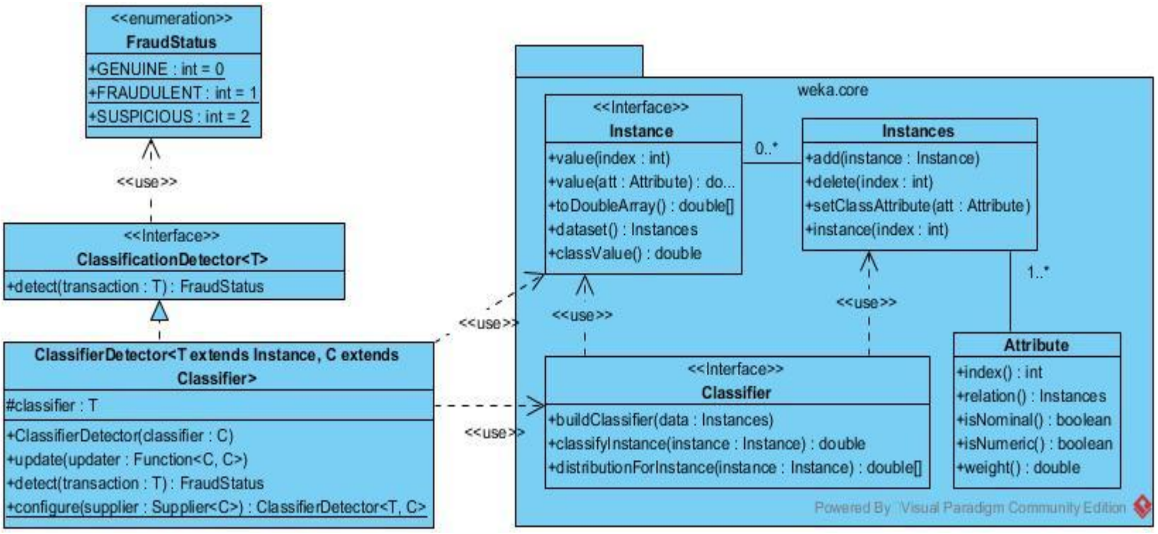

<<enumeration>>
FraudStatus
+GENUINE : int = 0
+FRAUDULENT : int = 1
+SUSPICIOUS : int = 2
<<use>>
<<Interface>>
ClassificationDetector<T>
+detect(transaction : T) : FraudStatus
ClassifierDetector<T extends Instance, C extends Classifier>
#classifier : T
+ClassifierDetector(classifier : C)
+update(updater : Function<C, C>)
+detect(transaction : T) : FraudStatus
+configure(supplier : Supplier<C>) : ClassifierDetector<T, C>
<<use>>
<<use>>
weka.core
<<Interface>>
Instance
+value(index : int)
+value(att : Attribute) : do...
+toDoubleArray() : double[]
+dataset() : Instances
+classValue() : double
0..*
Instances
+add(instance : Instance)
+delete(index : int)
+setClassAttribute(att : Attribute)
+instance(index : int)
1..*
<<use>>
<<use>>
<<Interface>>
Classifier
+buildClassifier(data : Instances)
+classifyInstance(instance : Instance) : double
+distributionForInstance(instance : Instance) : double[]
Attribute
+index() : int
+relation() : Instances
+isNominal() : boolean
+isNumeric() : boolean
+weight() : double
Powered By Visual Paradigm Community Edition

Figure 3. Class diagram of generic classifier detector implementation

Further in implementation, Instances, Instance and Attribute were used for storing and handling of data. Rich set of operations can be performed with Filter class, which makes it easy to manipulate large amounts of data without the need in manual implementation with aforementioned classes and interfaces.

To implement score detector of Banksealer approach, algorithms, utilized by it require efficient implementation first. First of all, the HBOS algorithm implementation is required. The original paper by Goldstein et al. [33] that presents this algorithm referenced implementation for RapidMiner platform [35]. Since original implementation was also written in Java, it served as a basis for porting it under WEKA data structures. However, due to the significant number of incoming parameters, a separate class diagram was designed in order to separate algorithm parametrization from the creation of a model (see Figure 4).

During the HBOS model creation, the value of each selected feature is observed from minimum to maximum value, resulting in the creation of histograms on the way. These histograms are represented by a set of bins, which account for values falling into a certain range. Depending on input parameters, the size of each bin may be dynamic or static. The score for each bin represents how often values from the training data set fall into it. Finally, each bin is normalized to maximal feature value, represented in training data set. The resulting score for a feature value is determined by the score of a bin, into which this value falls.

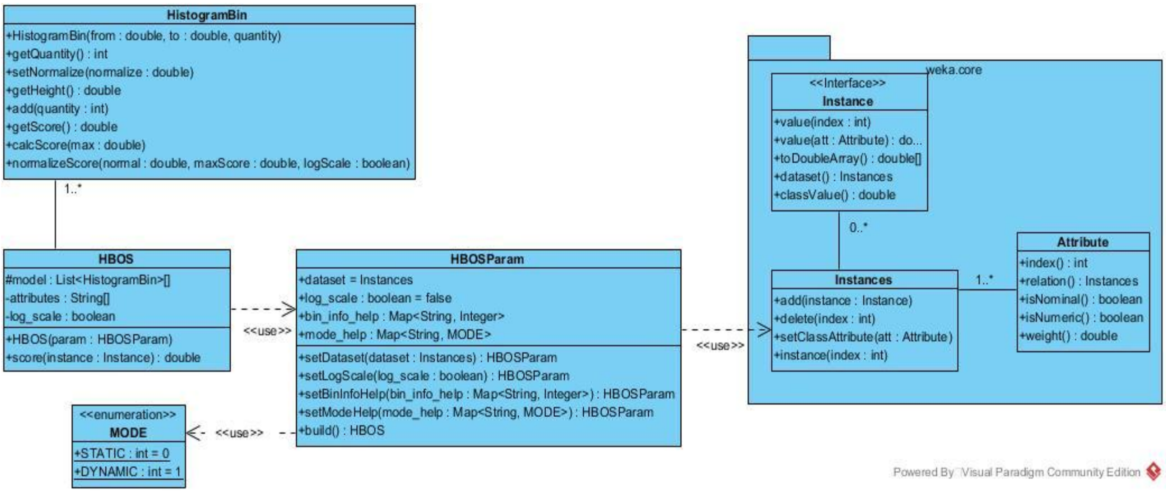

HistogramBin
+HistogramBin(from : double, to : double, quantity)
+getQuantity() : int
+setNormalize(normalize : double)
+getHeight() : double
+add(quantity : int)
+getScore() : double
+calcScore(max : double)
+normalizeScore(normal : double, maxScore : double, logScale : boolean)
1..*
HBOS
#model : List<HistogramBin>[]
-attributes : String[]
-log_scale : boolean
+HBOS(param : HBOSParam)
+score(instance : Instance) : double
<<use>>
HBOSParam
+dataset = Instances
+log_scale : boolean = false
+bin_info_help : Map<String, Integer>
+mode_help : Map<String, MODE>
+setDataset(dataset : Instances) : HBOSParam
+setLogScale(log_scale : boolean) : HBOSParam
+setBinInfoHelp(bin_info_help : Map<String, Integer>) : HBOSParam
+setModeHelp(mode_help : Map<String, MODE>) : HBOSParam
+build() : HBOS
<<enumeration>>
MODE
+STATIC : int = 0
+DYNAMIC : int = 1
<<use>>
<<use>>
weka.core
<<Interface>>
Instance
+value(index : int)
+value(att : Attribute) : do...
+toDoubleArray() : double[]
+dataset() : Instances
+classValue() : double
0..*
Instances
+add(instance : Instance)
+delete(index : int)
+setClassAttribute(att : Attribute)
+instance(index : int)
1..*
Attribute
+index() : int
+relation() : Instances
+isNominal() : boolean
+isNumeric() : boolean
+weight() : double
Powered By Visual Paradigm Community Edition

Figure 4. Class diagram of the implementation of the HBOS algorithm

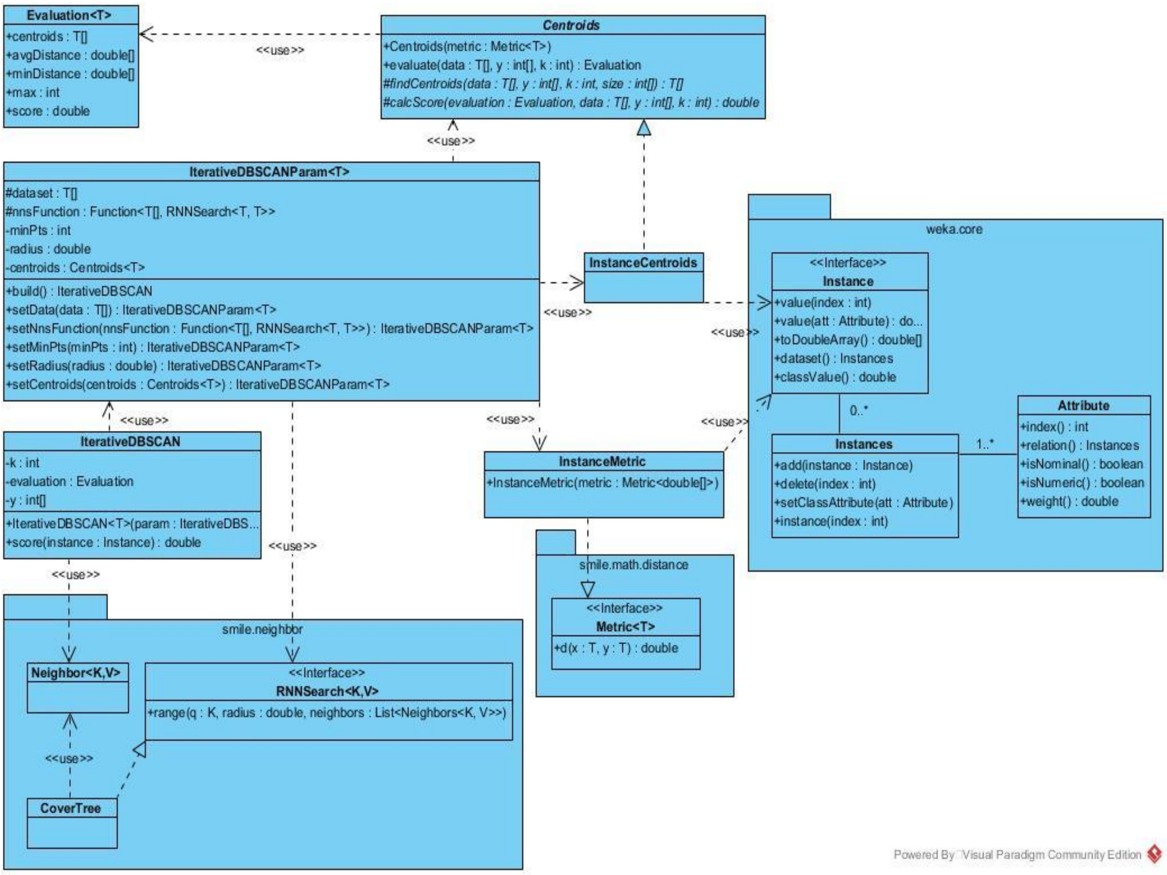
HistogramBin class represents a bin, created by HBOS algorithm. Scoring is performed as a sum of scores for each Instance feature, specified during training. In addition, it is possible to implement a weighted score, where each for feature is assigned a certain weight, which is multiplied by the determined score for a value to form the resulting score.

For the DBSCAN algorithm [34], the situation is similar (see Figure 5). However, the speed of the algorithm highly depends on the index acceleration method. According to Kriegel et al. [40], one of the authors of this algorithm, in our situation the best approach would be to choose Cover Tree for this purpose, provided by Smile library.

However, we still need to modify the algorithm into the iterative approach. Also, the distance metric suggested by Banksealer serves Mahalanobis distance which requires computation of the covariation matrix. This metric is already presented in Smile library.

Information about the distance between classified points is encapsulated into RNNSearch and Centroids instances. The first interface is utilized for searching of nearest neighbours and the second – for calculating information about cluster which can be used later on: its quality score, centroids and some additional data.

As a stop condition, we utilized the Davies-Bouldin index [41], which evaluates clustering quality. The lower index value – the better clustering was performed. Since on each iteration, DBSCAN is applied to largest cluster and smaller clusters are preserved, a new index acceleration structure needs to be generated each time so that already points from other clusters would not affect the results of the algorithm. To evaluate the score, it is required to calculate cluster centroids. Since centroids also required further in CBLOF algorithm, they are also stored as a result of DBSCAN.



Evaluation<T>
+centroids : T[]
+avgDistance : double[]
+minDistance : double[]
+max : int
+score : double
<<use>>
Centroids
+Centroids(metric : Metric<T>)
+evaluate(data : T[], y : int[], k : int) : Evaluation
#findCentroids(data : T[], y : int[], k : int, size : int[]) : T[]
#calcScore(evaluation : Evaluation, data : T[], y : int[], k : int) : double
<<use>>
IterativeDBSCANParam<T>
#dataset : T[]
#nnsFunction : Function<T[], RNNSearch<T, T>>
-minPts : int
-radius : double
-centroids : Centroids<T>
+build() : IterativeDBSCAN
+setData(data : T[]) : IterativeDBSCANParam<T>
+setNnsFunction(nnsFunction : Function<T[], RNNSearch<T, T>>) : IterativeDBSCANParam<T>
+setMinPts(minPts : int) : IterativeDBSCANParam<T>
+setRadius(radius : double) : IterativeDBSCANParam<T>
+setCentroids(centroids : Centroids<T>) : IterativeDBSCANParam<T>
InstanceCentroids
<<use>>
<<use>>
weka.core
<<Interface>>
Instance
+value(index : int)
+value(att : Attribute) : do...
+toDoubleArray() : double[]
+dataset() : Instances
+classValue() : double
0..*
Instances
+add(instance : Instance)
+delete(index : int)
+setClassAttribute(att : Attribute)
+instance(index : int)
1..*
Attribute
+index() : int
+relation() : Instances
+isNominal() : boolean
+isNumeric() : boolean
+weight() : double
<<use>>
<<use>>
<<use>>
IterativeDBSCAN
-k : int
-evaluation : Evaluation
-y : int[]
+IterativeDBSCAN<T>(param : IterativeDBS...
+score(instance : Instance) : double
InstanceMetric
+InstanceMetric(metric : Metric<double[]>)
<<use>>
<<use>>
smile.math.distance
<<Interface>>
Metric<T>
+d(x : T, y : T) : double
smile.neighbor
Neighbor<K,V>
<<Interface>>
RNNSearch<K,V>
+range(q : K, radius : double, neighbors : List<Neighbors<K, V>>)
<<use>>
CoverTree
Powered By Visual Paradigm Community Edition

Figure 5. Class diagram of the iterative DBSCAN algorithm implementation

The class diagram shown in Figure 6 and the sequence diagram in Figure 7 provide more information on the CBLOF implementation. The CBLOF algorithm was proposed by one of the authors of the HBOS algorithm in another research [42]. However, it is also implemented in

RapidMiner plug-in mentioned earlier. Another well-documented version is implemented in ELKI library. However, everything necessary for implementation was calculated before during evaluation of the DBSCAN algorithm, and actual implementation is as trivial as the calculating distance to cluster centroids.

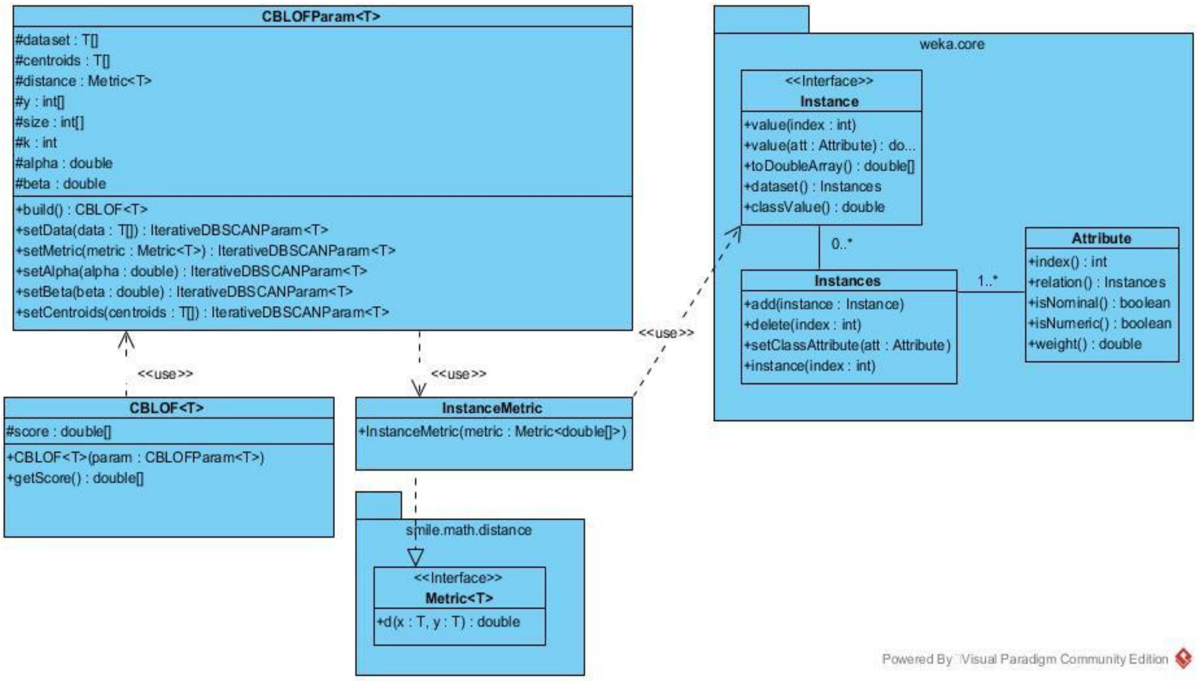


Figure 6. Class diagram of CBLOF algorithm implementation

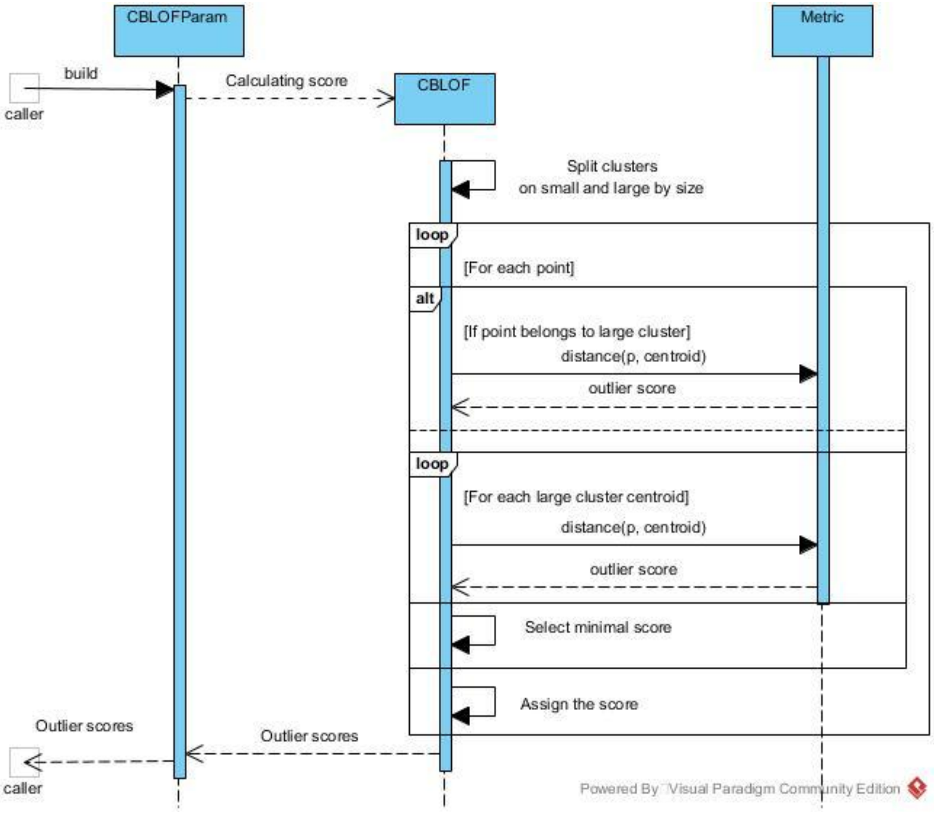


Figure 7. Sequence diagram showing a simplified algorithm of calculating CBLOF

Finally, combining results of previous algorithms it is possible to implement Banksealer score detector (see Figure 8).

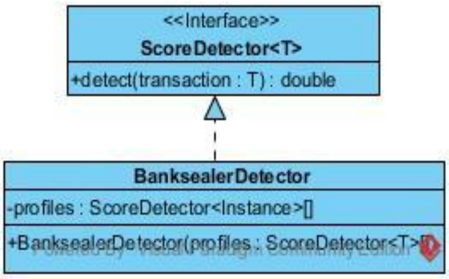

<<Interface>>
ScoreDetector<T>
+detect(transaction : T) : double
BanksealerDetector
-profiles : ScoreDetector<Instance>[]
+BanksealerDetector(profiles : ScoreDetector<T>[

Figure 8. Class diagram of Banksealer score detector implementation

While it is not mentioned directly, it seems that all scores derived from profiles are multiplied together. Utilizing this knowledge, it is possible to create a simple detector that multiplies the results of others. Using FunctionalInterface annotation provided by Java 1.8, it is possible to simplify the creation of detectors, allowing their declaration via lambda expressions. Since information from the global profile and transaction amount is supplied along with incoming transaction, it helps to simplify the process of detector creation. Local profile detectors are expected to be created per each user of a bank. Resulting score is computed by multiplying HBOS score, transaction amount and global score.

4.4 Evaluation

For initial unit testing of different algorithms' implementation and for testing assumptions about their performance the Junit [43] library was utilized. For benchmarking purpose, JMH [18] from OpenJDK was applied. A plugin for Gradle [44] allowed running the benchmarking process as simple as possible. Different metrics can be extracted during evaluation but since detection must fit into a certain amount of time, operation per second is the one that was used.

Since it is unknown whether the Random Forest model must be built for each user separately or can be global, the first approach was assumed. Loading of data is separated from benchmark evaluation into the setup, since implementation mainly requires only a vector of float point numbers, so transformation into this representation would depend on actual data source. In addition, building a detector and actual detection always split into different benchmarks. Typically, 10 iterations of the building process and 50 iterations of detection were benchmarked.

Evaluation also has to include the effectivness of the approaches under comparison, which basically means that actual fraud cases included in the data sets should be classified as fraudulent (true positives - TP) and not as correct (false positives - FP) whereas correct cases should not be classified as fraudulent (false negatives - FN) but as correct (true negatives - TN). For the purpose of evaluating effectiveness, we used Fβ-scores [45] which are based on precision[†] and recall[‡]. Fβ-scores were calculated with β = 1 and β = 2. The F1-score is the harmonic mean of precision and recall. The F2-score weighs recall is higher than precision by putting more emphasis on FN. Payment providers usually would like to detect all TP and avoid FN.

Results of the evaluation

The evaluation was performed on the transaction data set, generated by the modified version of PaySim. This data set contained around 15 thousand transactions, more than 600 of which were fraudulent and 450 clients were involved in the simulation.

[†] Precision = TP / (TP+FP)

[‡] Recall = TP / (TP + FN)

Performance measurement was performed by 5 benchmarks, each of which were executed for 101 iterations with 5 iterations of warm-up. The evaluation was performed on typical PC-class machine with 16 GB of RAM and Intel® Core™ i5-3450. The results are reflected in Table 1.

Since the global profile of the user is derived from payee of a transaction and represented by a single score, it can be derived outside the detector. Since the resulting score is a multiplication of all scores and amount value, it is correct to assume that detection time for the global profile is constant, as it only depends on how results of the profile creation are stored.

The Fβ-score for the Banksealer approach is equal for both cases due to utilization of the evaluation method used in original paper: the precision value equals the recall value for fraud, as only the number of inserted fraud transactions is observed.

While the Banksealer approach takes more time for profile building, detection time is slightly faster than it is for Random Forest. However, Random Forest seems to provide more precise results. Generally, both approaches perform well and can be utilized together.

Table 1. The result of the evaluation

	Random Forest	Banksealer		Computing Platform
		HBOS	DBSCAN & CBLOF	
Building (sec)	0.003170	0.009644	0.115 ± 0.001	PC-class machine with 16 GB of RAM and Intel® Core™ i5-3450
		0.000023 (per user)		
Detection (sec)	0.000003	0.0000004	Depends on storage	
Fβ-score(β=1)	0.950	0.919		
Fβ-score(β=2)	0.926	0.919		
Fβ-score(β=1) weighted average	0.987	0.979		

5 Summary

Digital transformation in financial industries and the introduction of new payment methods, such as IP, cause technological challenges and lead to discussions about the potential of AI for financial industries. Starting from an investigation into the state-of-the-art of fraud detection for payment processes and an interview study with payment providers, this work aimed at providing insights into the feasibility of AI use for fraud detection in IP.

Based on the observations and analysis results of the problem analysis presented in Section 4, we argue that there is a need for additional technological support for fraud discovery in instant payments and propose an AI implementation using the random forest or Banksealer approach. To overcome the unavailability of proper test data, a modification of existing payment simulator was proposed and implemented. This allowed the successful evaluation of the explored approaches. The selected approaches were efficiently implemented and tested for the applicability in the instant payments area. The limitation currently is the lack of real-world test data, which can be used for developing and evaluating fraud detection approaches.

Further research can be done by improving the implementation with other production-ready approaches. In addition, the area of payment simulators can be improved to generate more suitable transactions for different kinds of payment methods. This would allow gathering more useful context information that can slightly improve the quality of the detection process.

Acknowledgements

The research presented has been supported by the Government of Russian Federation, Grant 08-08 and by German International Exchanges Agency (DAAD), Grant 57464138.

References

[1] C. Matt, T. Hess, and A. Benlian, “Digital transformation strategies,” *Business & Information Systems Engineering*, vol. 57, no. 5, pp. 339–343, 2015. Available: https://doi.org/10.1007/s12599-015-0401-5

[2] J. Rifkin, *The Third Industrial Revolution: How Lateral Power Is Transforming Energy, the Economy, and the World*. St. Martin’s Griffin, 2013.

[3] European Central Bank, “Glossary of terms related to payment, clearing and settlement systems,” 2009. [Online]. Available: https://www.ecb.europa.eu/pub/pdf/other/glossaryrelatedtopaymentclearingandsettlementsystemsen.pdf.

[4] European Central Bank, “Instant Payments,” 2019. [Online]. Available: https://www.ecb.europa.eu/paym/retpaym/instant/html/index.en.html.

[5] The Central Bank of the Russian Federation, “Instant payment system,” 2019 (in Russian). [Online]. Available: https://www.cbr.ru/psystem/sistema-bystrykh-platezhey/.

[6] European Central Bank, “Euro Retail Payments Board,” 2019. [Online]. Available: https://www.ecb.europa.eu/ecb/access_to_documents/document/dialogue/euro_retail_payments_board_(erpb)/html/index.en.html

[7] Committee on Payments and Market Infrastructures, “Fast payments – Enhancing the speed and availability of retail payments,” 2016. [Online]. Available: https://www.bis.org/cpmi/publ/d154.pdf

[8] S. Herbst-Murphy, “Clearing and Settlement of Interbank Card Transactions: A MasterCard Tutorial for Federal Reserve Payments Analysts,” FRB of Philadelphia – Payment Cards Center Discussion, paper No. 13–01, 2013. Available: https://doi.org/10.2139/ssrn.2348276

[9] Committee on Payment and Settlement Systems, Bank for International Settlements, “Clearing and Settlement Arrangements for Retail Payments in Selected Countries,” 2000. [Online]. Available: https://www.bis.org/cpmi/publ/d40.pdf

[10] Oxford English Dictionary, “Fraud,” 2019. [Online]. Available: https://en.oxforddictionaries.com/definition/fraud

[11] Legal Dictionary, ‘Bank Fraud’, 2015. [Online]. Available: https://legaldictionary.net/bank-fraud/

[12] European Central Bank, “ECB report shows a fall in card fraud in 2016,” 2018. [Online]. Available: https://www.ecb.europa.eu/press/pr/date/2018/html/ecb.pr180926.en.html

[13] US Securities and Exchange Commission, “Mastercard Incorporated,” *Fiscal Year 2017 Form 10-K Annual Report*, 2017. [Online]. Available: https://www.sec.gov/Archives/edgar/data/1141391/000114139118000009/ma12312017-10xk.htm

[14] A. Diadiushkin, “Automation of transaction analysis for fraud detection in instant banking payments,” M.S. thesis, Institute of Computer Science, University of Rostock, Germany, 2019.

[15] P. Vishwakarma, A. K. Tripathy, and S. Vemuru, “A Layered Approach to Fraud Analytics for NFC-Enabled Mobile Payment System,” *Distributed Computing and Internet Technology, ICDCIT 2018*, Springer, LNCS, vol. 10722, pp. 127–131, 2018. Available: https://doi.org/10.1007/978-3-319-72344-0_9

[16] Y. Kültür and M. Çağlayan, “A Novel Cardholder Behavior Model for Detecting Credit Card Fraud,” *Intell. Autom. & Soft Comput.*, vol. 24, no. 4, pp. 808–817, 2018. Available: https://doi.org/10.1080/10798587.2017.1342415

[17] M. Carminati, L. Valentini, and S. Zanero, “A Supervised Auto-Tuning Approach for a Banking Fraud Detection System,” *Cyber Security Cryptography and Machine Learning*, Springer, LNCS, vol. 10332, pp. 215–233, 2017. https://doi.org/10.1007/978-3-319-60080-2_17

[18] OpenJDK, “Code Tools: jmh,” 2019. [Online]. Available: https://openjdk.java.net/projects/code-tools/jmh/

[19] P. S. Patil and N. V. Dharwadkar, “Analysis of banking data using machine learning,” *Proc. of the International Conference on IoT in Social, Mobile, Analytics and Cloud, I-SMAC 2017,* pp. 876–881, 2017. Available: https://doi.org/10.1109/I-SMAC.2017.8058305

[20] N. Hatamikhah, M. Barari, M. R. Kangavari, and M. A. Keyvanrad, "Concept Drift Detection via Improved Deep Belief Network," *Proc. of the 26th Iranian Conference on Electrical Engineering, ICEE 2018*, pp. 1703–1707, 2018. Available: https://doi.org/10.1109/ICEE.2018.8472481

[21] S. Nami and M. Shajari, "Cost-sensitive payment card fraud detection based on dynamic random forest and k-nearest neighbors," *Expert Syst. Appl.*, vol. 110, pp. 381–392, 2018. Available: https://doi.org/10.1016/j.eswa.2018.06.011

[22] 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. Available: https://doi.org/10.1016/j.inffus.2008.04.001

[23] X. Wang, H. Wu, and Z. Yi, "Research on Bank Anti-Fraud Model Based on K-Means and Hidden Markov Model," *2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC)*, pp. 780–784, 2018. Available: https://doi.org/10.1109/ICIVC.2018.8492795

[24] T. K. Behera and S. Panigrahi, "Credit Card Fraud Detection: A Hybrid Approach Using Fuzzy Clustering & Neural Network," *Proc. of the 2nd IEEE International Conference on Advances in Computing and Communication Engineering, ICACCE 2015*, pp. 494–499, 2015. Available: https://doi.org/10.1109/ICACCE.2015.33

[25] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, Springer, vol. 16, no. 4, pp. 449–475, 2013. Available: https://doi.org/10.1007/s11280-012-0178-0

[26] J. Li, C. Wang, L. Cao, and P. S. Yu, "Efficient Selection of Globally Optimal Rules on Large Imbalanced Data Based on Rule Coverage Relationship Analysis," *Proc. of the 2013 SIAM International Conference on Data Mining*, pp. 216–224, 2013. Available: https://doi.org/10.1137/1.9781611972832.24

[27] A. Jarovsky, T. Milo, S. Novgorodov, and W. C. Tan, "Rule sharing for fraud detection via adaptation," *Proc. of the IEEE 34th International Conference on Data Engineering, ICDE 2018*, pp. 125–136, 2018. Available: https://doi.org/10.1109/ICDE.2018.00021

[28] E. Hormozi, M. K. Akbari, M. S. Javan, and H. Hormozi, "Performance evaluation of a fraud detection system based artificial immune system on the cloud," *Proc. of the 8th International Conference on Computer Science and Education, ICCSE 2013*, pp. 819–823, 2013. Available: https://doi.org/10.1109/ICCSE.2013.6554022

[29] S. Dhankhad, E. A. Mohammed, and B. Far, "Supervised machine learning algorithms for credit card fraudulent transaction detection: A comparative study," *Proc. of the 2018 IEEE 19th International Conference on Information Reuse and Integration for Data Science, IRI 2018*, pp. 122–125, 2018. Available: https://doi.org/10.1109/IRI.2018.00025

[30] L. Zheng, G. Liu, C. Yan, and C. Jiang, "Transaction fraud detection based on total order relation and behavior diversity," *IEEE Trans. Comput. Soc. Syst.*, vol. 5, no. 3, pp. 796–806, 2018. Available: https://doi.org/10.1109/TCSS.2018.2856910

[31] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007. Available: https://doi.org/10.2753/MIS0742-1222240302

[32] K. Sandkuhl, "Putting AI into Context – Method Support for the Introduction of Artificial Intelligence into Organizations," 2019 *IEEE 21st Conference on Business Informatics* (CBI), pp. 157–164, 2019. Available: https://doi.org/10.1109/CBI.2019.00025

[33] M. Goldstein and A. Dengel, "Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm," *Proc. Poster and Demo Track of the 35th German Conference on Artificial Intelligence (KI-2012)*, pp. 59–63, 2012.

[34] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proc. of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[35] M. Goldstein, "RapidMiner Extension for Anomaly Detection," 2014. [Online]. Available: https://github.com/Markus-Go/rapidminer-anomalydetection.

[36] P. C. Mahalanobis, "On the generalized distance in statistics," *Proc. of the National Institute of Science of India*, pp. 49–55, 1936.

[37] Machine Learning Group – ULB, "Credit card fraud detection," 2016. [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud

[38] E. Lopez-Rojas, A. Elmir, and S. Axelsson, “3 Paysim : A financial mobile money simulator for fraud detection,” *28th Eur. Model. Simul. Symp. EMSS 2016*, 2016.

[39] WEKA: Class RandomForest, 2019. [Online]. Available: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/RandomForest.html

[40] H.-P. Kriegel, E. Schubert, and A. Zimek, “The (black) art of runtime evaluation: Are we comparing algorithms or implementations?” *Knowl. Inf. Syst.*, vol. 52, no. 2, pp. 341–378, 2017. Available: https://doi.org/10.1007/s10115-016-1004-2

[41] D. L. Davies and D. W. Bouldin, “A Cluster Separation Measure,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 1979. Available: https://doi.org/10.1109/TPAMI.1979.4766909

[42] M. Amer and M. Goldstein “Nearest-Neighbor and Clustering based Anomaly Detection Algorithms for RapidMiner,” *3rd RapidMiner Community Meeting and Conferernce*, 2012.

[43] JUnit, “JUnit 4,” 2019. [Online]. Available: https://junit.org/junit4/

[44] C. Champeau, “JMH Gradle plugin,” 2019. [Online]. Available: https://github.com/melix/jmh-gradle-plugin.

[45] C. J. Van Rijsbergen, *Information Retrieval* (2nd ed.). Butterworth-Heinemann, 1979.