BankSealer: An Online Banking Fraud Analysis and Decision Support System

Michele Carminati[1], Roberto Caron[1], Federico Maggi[1], Ilenia Epifani[2], and Stefano Zanero[1]

[1] Politecnico di Milano, Italy
Dipartimento di Elettronica, Informazione e Bioingegneria
{michele.carminati,roberto.caron,federico.maggi,stefano.zanero}@polimi.it

[2] Politecnico di Milano, Italy
Dipartimento di Matematica
ilenia.epifani@polimi.it

Abstract We propose a semi-supervised online banking fraud analysis and decision support approach. During a training phase, it builds a profile for each customer based on past transactions. At runtime, it supports the analyst by ranking unforeseen transactions that deviate from the learned profiles. It uses methods whose output has a immediate statistical meaning that provide the analyst with an easy-to-understand model of each customer's spending habits. First, we quantify the anomaly of each transaction with respect to the customer historical profile. Second, we find global clusters of customers with similar spending habits. Third, we use a temporal threshold system that measures the anomaly of the current spending pattern of each customer, with respect to his or her past spending behavior. As a result, we mitigate the undertraining due to the lack of historical data for building of well-trained profiles (of fresh users), and the users that change their (spending) habits over time. Our evaluation on real-world data shows that our approach correctly ranks complex frauds as "top priority".

Keywords: fraud detection, bank fraud, anomaly detection

1 Introduction

The popularity of Internet banking has led to an increase of frauds, resulting in substantial financial losses [15,4]. Banking frauds increased 93% in 2009–2010 [6], and 30% in 2012–2013 [8].

Internet banking frauds are difficult to analyze and detect because the fraudulent behavior is dynamic, spread across different customer's profiles, and dispersed in large and highly imbalanced datasets (e.g., web logs, transaction logs, spending profiles). Moreover, customers rarely check their online banking history such regularly that they are able to discover fraud transactions timely [15].

We notice that most of the existing approaches build black box models that are not very useful in manual investigation, making the process slower. In addition, those based on baseline profiling are not adaptive, also due to cultural and behavioral differences that vary from country to country. Instead of focusing on *pure detection* approaches, we believe that more research efforts are needed toward systems that *support* the investigation, and we had the unique opportunity to work on a real-world, anonymized dataset.

In this paper we propose BankSealer, an effective online banking semi-supervised decision support and fraud analysis system. BankSealer automatically ranks frauds and anomalies in bank transfer transactions and prepaid phone and debit cards transactions. During a training phase, it builds a local, global, and temporal profile for each user. The local profile models past user behavior to evaluate the anomaly of new transactions by means of a novel algorithm that uses the Histogram Based Outlier Score (HBOS). The global profiling clusters users according to their transactions features via an iterative version of Density-Based Spatial Clustering of Applications with Noise (DBSCAN). For this, we use Cluster-Based Local Outlier Factor (CBLOF). This approach allows to handle undertraining, which is particularly relevant for new users, which lack of training data. The temporal profile aims to model transactions in terms of time-dependent attributes. For this, we design a series of thresholds and measure the anomaly in terms of the percentage gap from the thresholds once they are exceeded. We handle the concept drift of the scores with an exponential decay function that assigns lower weights to older profiles.

We tested the BankSealer on real-world data with a realistic ground truth (e.g., credential stealing, banking trojan activity, and frauds repeated over time) in collaboration with domain experts. Our system ranked fraud and anomalies with up to 98% detection rate. Given the good results, a leading Italian bank is deploying a version of BankSealer in their environment to analyze frauds.

In summary, our main contributions are:

- a general framework for online semi-supervised outlier-detection based on the marginal distribution of the attributes of the user's transactions.
- a combination of different models to discover different types of frauds. The scores calculated by BankSealer have a clear statistical meaning, aiding the analyst's activity. Our approach is adaptive to non-stationary sources and can deal with concept drift and data scarcity.
- We developed an automatic decision support system for banking frauds, evaluated it in real-world setting, and deployed it to a large national bank.

2 Online Banking Fraud Detection: Goals and Challenges

Our goal is to support the analysis of (novel) frauds and anomalies by analyzing bank transfer logs, prepaid cards and phone recharges. From an analysis of the literature (summarized in §6) and a real-world dataset obtained from a large national bank (described in §4.1), we found peculiar characteristics that make the analysis of these datasets particularly challenging: skewed and unbalanced distribution of the attribute values, high number of nominal attributes, high cardinality associated with some of attributes (e.g., IP and IBAN take several thousands of distinct values). We also noticed the prevalence of users who perform a low number of transactions—an issue not considered in the literature.

Given the scarcity of labeled datasets, such a system must be able to work in an unsupervised or semi-supervised fashion. This conflicts with the requirement of the system being able to provide "readable" evidence to corroborate each alert. These peculiarities have remarkable implications for the typical statistical and data mining methods used in the outlier detection field.

3 Approach and System Description

BankSealer characterizes the users of the online banking web application by means of a local, a global and a temporal *profile*, which are built during a training phase. As depicted in Fig. 1, the training phase takes as input a list of *transactions*. As summarized in Tab. 1 we take into account three types of transactions. Each type of profile (i.e., local, global, temporal) extracts different statistical *features* from the transaction attributes (e.g., average, minimum, maximum, actual value), according to the type of model built.

BankSealer works both under semi-supervised and unsupervised assumptions by using a sample of the unlabeled dataset as training data. In the first case, the assumption is that the training data contains only legitimate transactions. In the second case, which is more realistic, the assumption is that the training dataset contains frauds, although these are a minority. As shown by [5,13], we can safely assume that the data in input contains frauds but, due to the fact that they are rare, the learned model is unbiased.

Once the profiles are built, BankSealer processes new transactions and ranks them according to their anomaly score and the predicted risk of fraud. The *anomaly score* quantifies the statistical likelihood of a transaction being a fraud w.r.t. the learned profiles. The *risk of fraud* prioritizes the transactions by means of anomaly score and amount.

BankSealer is not a classifier: It provides the analysts with a ranked list of fraudulent transactions, along with the anomaly score of each user. Top-ranked transactions have higher priority. As described in §1, the rationale behind this design decision is that analysts must investigate reported alerts in any case: Therefore, the focus is on collecting and correctly ranking evidence that support the analysis of fraudulent behavior, rather than just flagging transactions.

3.1 Local Profiling

The goal of this profiling is to characterizes each user's individual spending patterns. During training, we aggregate the transactions by user and approximate each feature distribution by a histogram. More precisely, we calculate the empirical marginal distribution of the features of each user's transactions. This representation is simple, readable and effective.

Table 1: List of attributes for each type of transaction. "CC_ASN" is the country code of the autonomous system of the client's IP. "Card type" takes values such as "Mastercard" or "VISA". Attributes in **bold** are hashed for anonymity needs.

Dataset	Attributes
Bank Transfers	Amount, CC_ASN, IP, **IBAN**, IBAN_CC, Timestamp, **Recipient**
Phone recharges	Amount, CC_ASN, IP, Phone operator, **Phone number**, Timestamp
Prepaid Cards	Amount, Card type, Card number, CC_ASN, IP, Timestamp

At runtime, we calculate the anomaly score of each new transaction using the HBOS [7] method. The HBOS computes the probability of a transaction according to the marginal distribution learned. As described in §3.1, we improved the HBOS to account for the variance of each feature and to allow the analyst to weight the features differently according to the institution's priorities.

Training and Feature Extraction. The features are the actual values of all the attributes listed in Tab. 1. During training we estimate the marginal distribution of each feature using uni-variate histograms in a way similar to what is done in HBOS [7,17,13]. However, we do not consider correlation between features in order to gain lower spatial complexity and better readability of the histograms. Uni-variate histograms are indeed directly readable by analysts who get a clear idea of the typical behavior by simply looking at the profile. In addition, they easily allow to compute the anomaly score as the sum of the contributions of each feature, giving an intuitive justification of the resulting anomaly score. For *categorical attributes* (e.g., IP, CC), we count the occurrences of each category. For *numerical attributes* (e.g., Amount, timestamp) we adopt a static binning and count how many values falls inside each bin. After this, we estimate the marginal frequency of the features, computing the relative frequency.

Runtime and Anomaly Score Calculation. We score each new transaction using HBOS [7]. It considers the relative frequency of each bin to quantify the log-likelihood of the transaction to be drawn from the marginal distribution. In other words, for each feature t_i of the transaction t we calculate $\log \frac{1}{\mathrm{hist}_i(t_i)}$, where hist_i indicates the frequency of the i-th feature. The resulting values are summed to form the anomaly score $\mathrm{HBOS}_i(t)$. The logarithm makes the score less sensitive to errors caused by floating point precision. To account for the higher

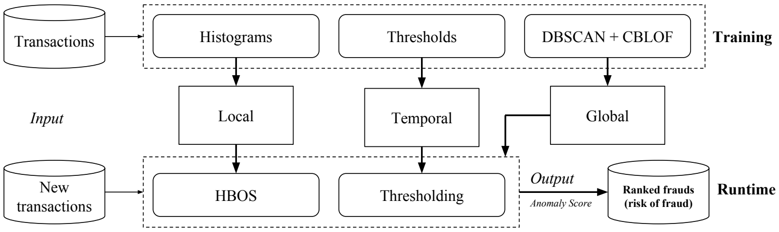

Figure 1: BankSealer architecture.

relevance of frauds in high-amount transactions, we multiply the anomaly score by the transaction amount.

Feature Normalization, Weighting and Rare Values. One of the main drawbacks of the original HBOS is that it does not take into account the variance of the features. For example, if Alice typically uses the banking web application from 5 distinct IPs and Bob from 2, the HBOS for Alice would be much lower than the HBOS for Bob. However, their activity is equally legitimate. To avoid this problem we apply a min-max normalization [9, pp. 71–72] to the histogram, where the minimum is zero, and the maximum is the highest bin.

In addition to the normalization, we added a weighting coefficient w_i to each feature to allow the analyst to tune the system according to the institution's priorities. In our experiments, however, we fixed all the weights at 1, except for IP and IBAN, which were fixed at 0.5 because of their high variance.

When a feature value has never occurred during training for a user (i.e., zero frequency within the local profile), the respective transaction may be assigned a high anomaly score. However, a user may have just changed his spending habits legitimately, thus causing false positives. To mitigate this, we calculate the frequency of unseen values as $k/1 - f$, where f is the frequency of that value calculated within a particular cluster, if the global profiling is able to find a cluster for that user. Otherwise, f is calculated on the entire dataset. This method quantifies the "rarity" of a feature value with respect to the global knowledge. The parameter k is an arbitrarily small, non-zero number. In our experiments we set it to 0.01.

Profile Updating. We update the histograms using an exponential discount factor, expressed in terms of the time window W and its respective sampling frequency. Every month we recursively count the values of the features in the previous months discounted by a factor $\lambda = e^{-\tau/W}$, where $W = 365$ days (up to 1 year). The rationale is that business activities are typically carried out, throughout an entire year, with a monthly basis. The parameter τ/W influences the speed with which the exponential decay forgets past data. In our case we empirically we set $\tau = 5$, because it seemed to best discount past data with respect to time and sampling windows.

Undertrained and New Users. An undertrained user is a user that performed a low number of transactions. In BankSealer this value is a parameter, which empirically we set at $T = 3$ as this is enough to get rid of most of the false positives due to undertraining.

For undertrained users, we consider their global profile (see §3.2) and select a cluster of similar users. For each incoming transaction, our system calculates the anomaly score using the local profile of both the undertrained user and the k nearest neighbor users (according to the Mahalanobis distance as detailed in §3.2). For new users, we adopt the same strategy. However, given the absence of a global profile, we consider all the users as neighbors.

3.2 Global Profiling

The goal of this profiling is to characterize "classes" of spending patterns. During training, we first create a global profile for each user. Then, we cluster the resulting profiles using an iterative version of the DBSCAN. Finally, for each global profile we calculate the CBLOF score, which tells the analyst to what extent a user profile is anomalous with respect to its closest cluster. The global profile is also leveraged to find local profiles for undertrained or new users. The rationale is that users belonging to the same cluster exhibit spending patterns with similar local profiles.

Training and Feature Extraction. Each user is represented as a feature vector of six components: total number of transactions, average transaction amount, total amount, average time span between subsequent transactions, number of transactions executed from foreign countries, number of transactions to foreign recipients (bank transfers dataset only). To find classes of users with similar spending patterns, we apply an iterative version of the DBSCAN, using the Mahalanobis distance [11] between the aforementioned vectors.

To mitigate the drawbacks of the classic DBSCAN when applied to skewed and unbalanced datasets such as ours (i.e., one large cluster and many small clusters), we run 10 iterations for decreasing values of ε from 10 to 0.2, which is the maximum distance to consider two users as connected (i.e., density similar). High values of this parameters yield a few large clusters, whereas low values yield many small clusters. At each iteration, we select the largest cluster and apply DBSCAN to its points with the next value of ε. The smaller clusters at each iterations are preserved. We stop the iterations whenever the number of clusters exhibits an abrupt increase (i.e., a knee). In all of our experiments, we empirically observed that this happens at 0.2. As a result, we obtain a set of clusters, which contain similar user profiles.

Anomaly Score Calculation and Updating. The global profile is used to assign each user profile a global anomaly score, which tells the analyst how "uncommon" their spending pattern is. For this, we compute the unweighted-CBLOF [2] score, which considers small clusters as outliers with respect to large clusters. More precisely, the more a user profile deviates from the dense cluster of "normal" users, the higher his or her anomaly score will be. The CBLOF anomaly score is the minimum distance of a user profile from the centroid of the nearest largest cluster. CBLOF takes only two parameters (α and β), which we evaluated empirically by considering as "normal" the 90%-percentile of the user profiles. The clustering is re-run according to the sampling frequency (i.e., 1 month). Moreover, we update the CBLOF anomaly score using an exponential discount, as described in §3.1.

3.3 Temporal Profiling

The goal of this profiling is to deal with frauds that exploit the repetition of legitimate-looking transactions over time (e.g., frequent wire transfers of

Table 2: Amount transferred for each dataset and scenario.

Fraud scenario	**Amount transferred (€)**		
	Bank transfers	Phone recharges	Prepaid cards
1: Information Stealing	10,000–50,000	250–255	750–1,000
2: Transaction Hijacking	10,000–50,000	250–255	750–1,000
3: Stealthy Fraud very low amount	50–100	5–10	50–100
low amount	100–500	10–25	100–250
medium amount	500–1,000	25–50	250–500

amounts that not violating the local or global profile). We construct a temporal profile for each user having a sufficient amount of past transactions. During training, we aggregate the transactions of each user over time and calculate the sample mean and variance of the numerical features. These are used as thresholds during runtime to calculate the anomaly score as the delta.

After a first training, updating profiles and anomaly scores is necessary because users may change their spending habits. We use a time window, which size can be easily chosen given the hardware resources available, as show by our experiments. Within such time window, the features of the transactions are aggregated with a daily sampling frequency.

Training and Feature Extraction. For each user, we extract the following aggregated features: total amount, total and maximum daily number of transactions. During training, we compute the mean and standard deviation for each feature, and set a threshold at mean plus standard deviation to classify transactions as anomalous. Undertrained users are excluded from temporal profiling because occasional transactions have a high variance, unsuitable for this kind of analysis. The anomaly scores are updated as in the global profile (i.e., exponential discount to account for evolving spending habits).

Runtime and Anomaly Score Calculation. xAt runtime, according to the sampling frequency, we calculate the cumulative value for each of the aforementioned features for each user. Then, we calculate the delta between each cumulative value and the respective threshold. Positive deltas are summed up to form the anomaly score.

4 Experimental Evaluation

The goals of our evaluation is to measure (1) the effectiveness and (2) the computational resource requirements of BankSealer in correctly ranking fraudulent transactions never seen before among the top.

4.1 Dataset Description and Fraud Scenarios

We obtained 3 months of anonymized data collected from a large national consumer bank between April and June 2013: we used 2 months for training and 1 month for testing, as suggested in [9, pp.359–364].

The dataset consists of 371,137 *bank transfers* (47,650 users), the money transfers made from any account of the bank to any other account, 54,141 *prepaid cards* transactions (16,093 users), the transactions to top up credit on prepaid cards, and 34,986 *phone recharge* transactions (8,415 users), the transactions to refill prepaid cellphone accounts. The dataset is unlabeled, but it contains no known frauds as confirmed by the bank we collaborate with.

The evaluation of BankSealer is particularly difficult because, like any unsupervised analysis tool, it produces novel knowledge. In addition, no frauds were known or reported at the bank in the 3 months of observation. Therefore, we relied on domain experts (bank operators) to enrich our testing dataset with generated frauds based on three fraud scenarios that, based on their experience, well replicate the typical real attacks performed against online banking users. We focus on the most important and challenging fraud schemes nowadays, those driven by banking trojans (e.g., ZeuS, Citadel) or phishing.

Scenario 1: Info stealing. The trojan modifies the login form to deceive the victim into entering an one time password (OTP) along with the login credentials. This information is used by the fraudster to execute a transaction (with a high amount) towards a his account, where the victim never sent money to. We test both the case of the connection coming from a national and foreign IP address. To inject the fraud, we randomly choose a victim from the testing dataset and used a random timestamp for the transaction.

Scenario 2: Transaction Hijacking. The trojan, not the fraudster, hijacks a legitimate bank transfer by manipulating the victim's browser. The challenge is that the connection comes from the victim's computer and IP address. Moreover, we execute the fraudulent transaction within ten minutes from a real one, to emulate a session hijacking.

Scenario 3: Stealthy Fraud. The strategy of the fraudster is to execute a series of low–medium amount transactions, repeated daily for one month during working hours, to better blend in. We analyze three cases (very low, low and medium daily amounts). We use the same number of users of the previous scenarios, each performing 30 fraudulent transaction.

For the bank transfers dataset, the money can be transferred to a national or foreign account, whereas for the phone recharges and prepaid debit cards the money is charged on an unknown card. Tab. 2 shows the amounts for each dataset and scenario.

4.2 Evaluation Approach and Metrics

After training, we inject n fraudulent transactions (or users) in the testing dataset. Then, we use the local profiles to rank transactions, and the temporal profiles to rank users, according to the respective anomaly scores. The global profiles are used to mitigate undertraining.

We analyze the top n transactions or users in the ranking, where n is the number of injected transactions (or users). In our case, n accounts for 1% of the testing dataset. Depending on the specific scenario, a fraud may trigger either the local or temporal profile, or both. Either way, thanks to the presence of

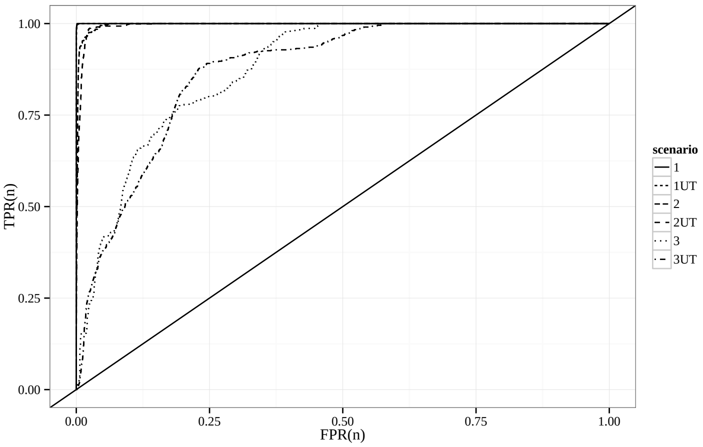

Figure 2: True positive rate (TPR) and false positive rate (FPR) for $n \in [1, N]$, where N is the size of the testing dataset. The label "UT" stands for "undertraining". In **Scenario 1** and **2** BankSealer detects about 98.26% of the frauds with 99.98% of precision (0.19% FPR). **Scenario 3** (users) is the most challenging because of stealthy, small-amount frauds: BankSealer still detect 69.73% fraudulent users with 83.10% of precision (14.03% FPR).

both profiling, the analyst is able to notice the fraud. We count as true positives the number of fraudulent transactions (or users) in the top n positions, and the remainder ones (to the whole n) are either false positives or negatives.

The **overall results** are summarized in Fig. 2. BankSealer outperforms the state of the art. For instance, [15] detects up to 60–70% of the frauds with an unreported precision. Remarkably, the effect of undertraining is negligible.

Experiment 1: Well-trained Users. We first tested BankSealer without the noise due to non well-trained users (i.e., we removed users with less than 3 transactions from the dataset). All test are repeated 10 times and the results are averaged to avoid biases.

As Tab. 3 shows, the combination of local and temporal profiles guarantees that frauds are ranked high at either transaction level, thanks to the local profiles, or user level, thanks to the temporal profile.

The results on the information stealing frauds (**Scenario 1**) are very promising. Transaction hijacking frauds (**Scenario 2**) are particularly challenging, because the malware does not alter the overall amount of transactions performed: It leverages existing transactions by diverting them to a different recipient. The IP address is one of those usually used by the user and, in the case where the recipient fraudulent account is national, these transactions blend in quite easily. It is likely that with more training data these features will become more significant. However, even for this last case, thanks to the temporal profile anomaly score BankSealer correctly ranked 58% of the frauds, up to 89% in the case of the phone recharges dataset. Indeed, if we consider the action of refilling phones or prepaid cards, we expect users to do this only towards a few fixed numbers or cards. This means that a recharge towards an unknown phone number or card

Table 3: Experiment 1 results on transactions and users. Blank cells indicate inapplicable dataset-scenario combinations (e.g., phone recharge transactions have no IBAN, phone recharge or prepaid card transactions are only nation-wise).

Fraud scenario		**Correctly ranked frauds (%)**					
		Bank transfers		Phone recharges		Prepaid cards	
		Transactions	Users	Transactions	Users	Transactions	Users
1: Information Stealing	foreign IP and IBAN	**98**	57	**96**	87	**95**	68
	foreign IP, national IBAN	**91**	57				
	national IP, foreign IBAN	**98**	57	87	**95**	67	**96**
	national IP and IBAN	**91**	57				
2: Transaction Hijacking	foreign	**75**	58				
	national	22	**58**	83	**89**	71	**77**
3: Stealthy Fraud	foreign, very low amount	**73**	64				
	foreign, low amount	**68**	67				
	foreign, medium amount	69	**73**				
	national, very low amount	42	**64**	64	97	**99**	93
	national, low amount	37	**67**	94	**99**	92	**97**
	national, medium amount	42	**72**	95	**99**	94	**98**

is always anomalous, even if the transaction amounts is low and the IP address of the connection is one of those commonly used by the user.

Stealthy frauds (**Scenario 3**) are also challenging: the local profile performs well when the recipient account is foreign, or with phone recharge and prepaid card frauds. Interestingly, stealthy frauds involving very low amounts (50–100€) are correctly ranked better than transactions involving low amounts (100–500€). The reason is because the very-low amounts are rarer in the dataset, and thus obtain higher anomaly scores.

Experiment 2: Undertrained and New Users. We evaluated the capabilities of the global profile to lookup a good replacement local profile for undertrained and new users. We proceeded similarly to what we did in the previous experiment, injecting 1% of fraudulent transactions, but we spread the injections evenly across well trained, undertrained, and new users.

Tab. 4 summarizes the percentage of correctly ranked transactions overall, for well-trained users only, for undertrained uses only, and finally for new users only. Performance is similar to the previous experiment, even if the percentage of correctly ranked frauds are obviously a little lower due to the additional noise.

The fact that undertrained sometimes obtain better ranking than well-trained users, especially when in the attack scenario the frauds are masked to be similar to common transactions, is an artifact due to the fact that in undertrained users' profiles even frauds designed to appear as legitimate transactions can receive a high score if the (few) transactions already observed for them are very different from the injected ones. Frauds injected in new users, instead, are ranked incorrectly when are designed to be similar to legitimate transactions. This is due to the fact that, for new users, transactions are tested against the average

Table 4: Experiment 2 results on well-trained, undertrained, new users only, and overall. As in Tab. 3, blank cells indicate inapplicable dataset-scenario combinations.

Fraud scenario		**Correctly ranked frauds (%)**											
		Bank transfers				Phone recharges				Prepaid cards			
		Overall	Well trained	Undertrained	New	Overall	Well trained	Undertrained	New	Overall	Well trained	Undertrained	New
1: Information Stealing	foreign IP and IBAN	96	98	99	92	80	99	99	43	80	85	94	59
	foreign IP, national IBAN	75	81	95	52								
	national IP, foreign IBAN	95	97	93	88	76	97	100	29	73	81	96	42
	national IP and IBAN	73	84	93	41								
2: Transaction Hijacking	foreign	63	43	91	51								
	national	24	13	57	3	40	29	83	6				
3: Stealthy Fraud	foreign, very low amount	52	40	91	22								
	foreign, low amount	56	38	92	39								
	foreign, medium amount	61	42	93	49					32	18	73	6
	national, very low amount	27	13	68	1	16	3	42	4	56	45	94	28
	national, low amount	29	14	71	2	19	5	51	0	59	40	91	47
	national, medium amount	34	19	78	3	19	2	56	0	72	64	94	59

profile of all transactions in the dataset, and thus transaction with common attributes will receive low scores.

In the experiments on the phone recharges dataset, we obtain a slightly lower percentage of correctly ranked frauds than those in Tab. 3. On the other hand, for the stealthy fraud (**Scenario 3**) the percentages are considerably lower. A factor is the huge number of undertrained users in the phone recharges dataset (2,932 transactions for well trained users vs. 11,505 transactions in total). Similar considerations hold for the prepaid cards dataset, in particular for the higher percentage of correctly ranked frauds with undertrained users.

Experiment 3: Performance and Resource Requirements. To test the performance of BankSealer, we measured both the computational requirements at runtime (as this is a constraint for the practical use of the system in production), and peak memory requirements at training time (as this is a constraint on the dimension of the dataset that can be handled).

Table 5: Computation time required at runtime under various conditions. In the typical use case, the system works on a daily basis, thus requiring 6 minutes (worst case).

Testing interval	**Elapsed time**		
	Bank transfers	Phone recharge	Prepaid cards
1 day, no undertrained/new users	$1'00''$	$0'18''$	$0'07''$
1 day, undertrained/new users	$4'00''$	$0'24''$	$0'10''$
1 month, no undertrained/new users	$6'00''$	$0'30''$	$0'12''$
1 month, undertrained/new users	$93'00''$	$2'30''$	$1'00''$

For computational power requirements, we tested the time to analyze one day and one month of data, both with and without the handling of undertrained and new users explained in §3.1. Our experiments have been executed on a desktop-class machine with a quad-core, 3.40Ghz Intel i5-3570 CPU, 8GB of RAM, running Linux 3.7.10 x86_64. Processing times are taken using the *time* library. The results are listed in Tab. 5. As we can see, the processing time varies on the basis of the context being tested, and there is a significant difference induced by the handling of undertrained/new users. In production BankSealer will analyze transactions day by day. Therefore, the maximum time required would be 4 minutes per day for the bank transfers context. In conclusion, we believe that BankSealer could be suitable for online fraud monitoring.

We tested the scalability of the system by measuring the RAM consumption at training time, which is the most memory-intensive phase. We used the bank transfers dataset, the largest one. We relied on memory-profiler and psutil . As Fig. 3 shows, the peak RAM consumption increases linearly with the number of days and quadratically with the number of users. This is expected, as the most memory-intensive data structure is the distance matrix, a square matrix of the size of the number of users.

5 Discussion

The main barrier in this research field is the lack of publicly available, real-world frauds and a ground truth for validation. Indeed, we had to resort to synthetically generated frauds. The absence of non-anonymized text fields does not allow us to analyze, for instance, their semantics. In future extensions, BankSealer will compute the models on the bank side and export privacy-preserving statistics for evaluation.

The prototype is also constrained by the RAM consumption of the clustering phase. This technical limitation can be mitigated in two possible ways. First, a triangular data structure to store the distance matrix. Second, a parallel version of DBSCAN [16], which splits the dataset on multiple machine.

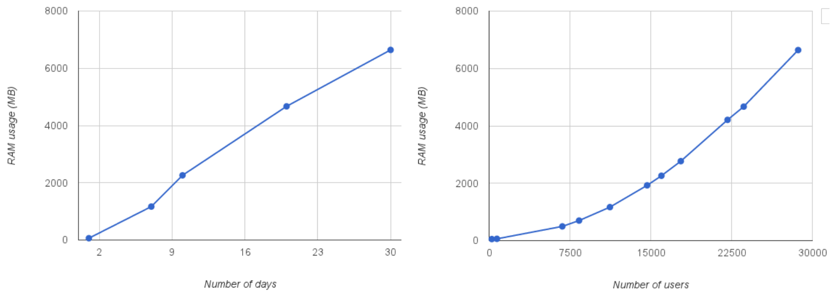

Figure 3: RAM requirements for increasing values of W (left) and users profiled (right).

6 Related Work

Fraud detection, mainly focused on credit card fraud, is a wide research topic, for which we refer the reader to [5,14,4].

Limiting our review to the field to banking fraud detection, supervised approaches based on contrast patterns and contrast sets (e.g., [3]) have been applied. Along a similar line [1] proposed a rule-based Internet banking fraud detection system. The proposed technique does not work in real time and thus is profoundly different from ours. Also, supervised techniques require labeled samples, differently from BankSealer.

The unsupervised approach presented in [15] is interesting as it mitigates the shortcomings of contrast pattern mining by considering the dependence between events at different points in time. However, [15] deals with the logs of the online banking web application, and thus does not detect frauds as much as irregular interactions with the application. Among the unsupervised learning methods, [12] proposed an effective detection mechanism to identify legitimate users and trace their unlawful activities using Hidden Markov Model (HMM)s. [10] is based on an unsupervised modeling of local and global observations of users' behavior, and relies on differential analysis to detect frauds as deviations from normal behavior. This evidence is strengthened or weakened by the users' global behavior. The major drawback of this approach is that the data collection must happen on the client side, which makes it cumbersome to deploy in large, real-world scenarios. In general, a major difference between existing unsupervised and semi-supervised approaches and BankSealer is that they do not give the analyst a motivation for the analysis results, making manual investigation and confirmation more difficult.

7 Conclusions

BankSealer is an effective online banking semi-supervised and unsupervised fraud and anomaly detection approach, with which we implemented a decision support system developed in close collaboration with a large national bank, which deployed it in a pilot project. Even with the strict requirements typical of banking scenarios (e.g., anonymized datasets), we showed that it is possible to create a decision support system that helps the analyst *understanding* the reasons behind frauds.

Most of the future works that aim to overcome BankSealer's limitations require more detailed datasets, which are often difficult to obtain due to privacy restrictions. Some examples are the semantic analysis of the text attributes, improvements of the temporal profile, and the estimation of the number of transactions required to fully train a profile.

A short-term work is to consider the feedback given by the analyst, which however requires careful evaluation because of the possible biases that an analyst may introduce. To this end, we currently deployed BankSealer in a real world environment, to collect the feedback of banking analysts on the detected anomalies.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n 257007, as well as from the TENACE PRIN Project (n. 20103P34XC) funded by the Italian Ministry of Education, University and Research.

References

1. Aggelis, V.: Offline Internet Banking Fraud Detection. In: ARES, IEEE Computer Society (2006) 904–905
2. Amer, M., Goldstein, M.: Nearest-Neighbor and Clustering based Anomaly Detection Algorithms for RapidMiner. (8 2012) 1–12
3. Bay, S.D., Pazzani, M.J.: Detecting Group Differences: Mining Contrast Sets. Data Mining and Knowledge Discovery **5**(3) (July 2001) 213–246
4. Bolton, R.J., David: Statistical fraud detection: A review. Statistical Science **17** (2002)
5. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Comput. Surv. **41** (July 2009) 15:1–15:58
6. Fossi, M., Egan, G., Haley, K., Johnson, E., Mack, T., Adams, T., Blackbird, J., Low, M.K., Mazurek, D., McKinney, D., Wood, P.: Symantec Internet Security Threat Report trends for 2010. Technical report, Symantec (April 2011)
7. Goldstein, M., Dengel, A.: Histogram-Based Outlier Score (HBOS): A Fast Unsupervised Anomaly Detection Algorithm. (2012)
8. Haley, K., Wood, P., et al.: Internet Security Threat Report 2013. **18** (2013)
9. Han, J., Kamber, M.: Data mining: concepts and techniques. 2 edn. Morgan Kaufmann (2006)
10. Kovach, S., Ruggiero, W.: Online banking fraud detection based on local and global behavior. In: ICDS 2011 : The Fifth Intl. Conf. on Digital Society . (2011) 166–171
11. Mahalanobis, P.C.: On the generalized distance in statistics. In: Proc. of the national Institute of Science of India. (1936) 49–55
12. Mhamane, S., Lobo, L.: Internet banking fraud detection using HMM. In: Computing Communication Networking Technologies (ICCCNT), 2012 Third Intl. Conf. on. (2012) 1–4
13. Noto, K., Brodley, C., Slonim, D.: FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection. Data Min. Knowl. Discov. **25**(1) (July 2012) 109–133
14. Phua, C., Lee, V.C.S., Smith-Miles, K., Gayler, R.W.: A Comprehensive Survey of Data Mining-based Fraud Detection Research. CoRR **abs/1009.6119** (2010)
15. Wei, W., Li, J., Cao, L., Ou, Y., Chen, J.: Effective detection of sophisticated online banking fraud on extremely imbalanced data. World Wide Web **16**(4) (July 2013) 449–475
16. Xu, X., Jäger, J., Kriegel, H.P.: A Fast Parallel Clustering Algorithm for Large Spatial Databases. Data Min. Knowl. Discov. **3**(3) (1999) 263–290
17. Yamanishi, K., Takeuchi, J.I., Williams, G., Milne, P.: On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms. Data Min. Knowl. Discov. **8**(3) (May 2004) 275–300