IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 7, 2021, accepted July 23, 2021, date of publication July 30, 2021, date of current version August 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3101738

Temporal Transaction Scraping Assisted Point of Compromise Detection With Autoencoder Based Feature Engineering

FUAT OGME, A. GOKHAN YAVUZ, M. AMAC GUVENSAN, (Member, IEEE), AND M. ELIF KARSLIGIL

Intelligent System Laboratory, Department of Computer Engineering, Yildiz Technical University, 34220 Istanbul, Turkey

Corresponding author: Fuat Ogme (fuato@yildiz.edu.tr)

ABSTRACT Credit card fraudsters exploit various methods to capture card information. One of the common methods is to duplicate the credit cards by skimming. In this study, we introduce a new point of compromise detection method in order to trace and identify merchants where the skimming operation took place and card information has been captured by criminals. The proposed method first extracts discriminative features by using principle component analysis(PCA) and Autoencoder extractors and then it clusters similar fraudulent transactions with K-Means algorithm, afterwards it highlights possible merchants that are involved in this scheme by finding matching merchants in the produced clusters with a retrospective analysis of all transactions. Our experiments showed that the proposed method could achieve promising results with zero-knowledge on the existing skimming points. The application of our proposed method on real-life card transactions enabled us to pinpoint 7 out of 9 point of compromise previously identified by the reporting bank.

INDEX TERMS Financial fraud, point of compromise detection, credit card skimming, clustering, autoencoder, retrospective analysis.

I. INTRODUCTION

Fraudulent transactions arise from various means, such as lost, stolen or skimmed credit cards, credit cards that are not received by the users, or fake credit card applications [1]. Credit card skimming is one of the most exploited methods. Skimming may be done physically or online. In the physical method, the credit card is run through the skimmer and the skimmer device acquires the credit card number, expiration date, and the user's full name. In the online method, credit card information is obtained from the user or from the servers of an e-commerce system through cyber-attacks such as phishing, SQL injection, or keylogging [2]–[4]. The location, where the credit card information is stolen, is called point of compromise (POC) or common point of purchase (CPP). Although locating a POC would facilitate implementing precautions to prevent skimming at that location, currently, a POC is only detected when the corresponding bank personnel conduct an investigation upon being notified by card users about fraudulent use. However, such notifications run in the tens of thousands. Therefore, detecting a POC manually is difficult due to the large volume of data and time consuming nature of required checks. Thus, there is a strong need for an autonomous system that detects POCs to minimize financial loss arising from such credit card fraud. Scientific studies focus on present-day analysis for POC detection, whereas patents highlight retrospective analysis, but they are limited by their proposed statistical methods, and are not tested experimentally in real life.

In this study, a system has been designed and put into action to detect POCs by using the fraudulent transactions reported to the banks. The proposed system traces POCs by grouping the fraudulent transactions with similar characteristics via deep learning approach and running a retrospective temporal analysis on these groupings. This analysis leveraged not only fraudulent transactions, but also regular ones. The success of the introduced system was tested with real POC data used as groundtruth provided by 35 members of the Interbank Card Center (BKM).

To the best of our knowledge, our study is the first-ever in the current literature to employ deep learning approach,

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Piccialli.

presenting the impact of the retrospective analysis empirically through various temporal and spatial scenarios, while using actual card transactions. The contributions of this paper are given as follows:

- In our proposed method regardless of the issuer reports, we continuously cluster fraud transactions based on their similarity. To this end, features extracted via Autoencoder were used as input to the K-Means clustering algorithm.
- We provide a concrete implementation of retrospective analysis on the clusters and evaluate its effectiveness on the tracing and detection of POCs. In this regard, to the best of our knowledge, our proposed method is the first attempt to examine the impact of retrospective analysis which was only suggested in a theoretical manner to improve performance in patents.
- Our introduced model requires zero-knowledge on the existing POCs. Therefore, it utilizes an unsupervised approach to cluster fraud transactions with the aim of targeting undetected POCs.
- To the best of our knowledge, it is the first-ever study which utilizes a real transaction dataset to detect POCs.

Section II presents a literature review on detecting card frauds and POCs. Section III sets out the main steps of the proposed system to solve the issue and probes the possible methods to be used in these steps. Section IV explains the components of the proposed system for POC detection and evaluates the results that are obtained by measuring system performance with an actual dataset. Finally, Section V provides a general assessment of the proposed system.

II. RELATED WORK

POC detection is directly related to credit card fraud issues. Thus, we first looked into the previous studies on credit card fraud detection, then moved on to the studies that focus directly on detection of POCs.

A. CREDIT CARD FRAUD DETECTION

Credit card fraud is a major problem that takes a financial toll on banks, companies and individuals. There are a number of studies on the issue that are based on various methods, such as artificial intelligence and machine learning. Each of these studies presents both advantages and disadvantages [5].

As part of the literature review, we also examined the datasets that are used in credit card fraud studies [6], [7]. Researchers that investigate credit card fraud have a hard time accessing actual credit card transactions due to the risks involved around confidentiality of personal data and availability of sensitive data such as credit card transactions. That is why some researchers, for example Aleskerov [8] and Behera [9], conducted their research on synthetic data. In these studies, the Gaussian method was used to synthetically create the product category, and the amount and time information for the spending.

The approaches that are based on machine learning to detect card fraud involve supervised, unsupervised and hybrid methods.

In the studies using supervised methods, fraudulent transactions are detected based on patterns, which are established by analyzing labeled previous card transactions. Strong models have been developed by utilizing the profiles that are based on the card type and spending amount, in addition to the card transaction information [10]. Supervised approaches in credit card fraud include the Bayesian Network, the Artificial Neural Networks model [11], the Support Vector Machine (SVM), the Decision Tree [12], the Logistic Regression, and the Random Forest [13]–[15] methods. An analysis of these studies shows the Bayesian Network, the Random Forest, and the Decision Tree methods to be more successful. The highly unbalanced nature of credit card data impairs the success of the supervised classifiers [16]. A majority of the studies employ undersampling and oversampling methods to overcome this issue [17]–[19].

Unsupervised methods handle fraud detection and card fraud issues as an issue of outlier detection. The Gaussian Mixture Model, K-Nearest Neighbors, Isolation Forest, Self Organizing Map and other similar methods are employed for outlier detection [20]. Malini and Pushpa [21] used the KNN model to examine the card fraud problem. Hybrid models [22], [23] that involve the combined use of unsupervised and supervised methods stand out as successful in card fraud detection. Additionally, deep learning methods that can be trained with large scale data and produce complex models are used for card fraud detection. Roy *et al.* [24] used the Artificial Neural Network (ANN), Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM), and Gated Recurrent Unit (GRU) in their study. The study concludes that the GRU method provides better results in comparison to others. The Autoencoder and the Restricted Boltzmann Machine (RBM) deep learning methods were also utilized in card fraud detection [25], [26]. These methods allow unsupervised learning for training data distribution. Once the model is trained, it becomes possible to detect the samples that do not come from the same distribution. Kazemi and Zarrabi [25] showed that the Autoencoder produced better results than other methods.

B. POINT OF COMPROMISE DETECTION

There are a limited number of studies on detecting POCs. One of the main reasons could be lack of available datasets. The paragraphs below assess the relevant patents and a conference paper.

The patent studies attempt to associate POCs to the number of pre-fraud accounts that are used in the businesses [27]–[29]. Pre-fraud accounts or pre-fraud transactions comprise all retrospective transactions starting with the time of the fraudulent transaction up to the time when skimming had taken place. Therefore, pre-fraud transactions can be detected by analyzing all retrospective transactions within that time period.

Klebanoff [27] determined a pre-fraud rate by dividing the number of pre-fraud accounts at a business by the total number of payment accounts. This study postulates that when the pre-fraud rate exceeds a threshold, that business could be defined as a PPOC. They also proposed that the daily pre-fraud rate at a business could be used for the same purpose. Moreover, they grouped fraudulent transactions according to the Merchant Category Code (MCC) and conducted weekly retrospective analyses. On the other hand, Yan [28] asserted that fraud transactions could be grouped by taking into consideration the accounts from which large amounts of money were withdrawn at various locations on the same day. They then attempted to detect one or multiple POCs by analyzing the pre-fraud account groups for a certain period of time prior to the day the frauds had taken place. The study calls this period of time an exposure window, which may last for a day, a week or a month. However, there are other studies suggesting that using an exposure window of more than a month may produce better results [29]. Nevertheless, these studies have not been conducted on an experimental level, nor do they provide a concrete and quantitative success assessment, or present a success rate increase. Some of the studies employ profiles such as the card user or the merchant, in addition to the pre-fraud rate, in the retrospective analysis. Information such as the daily transaction volume at a business, location and MCC are used to create a merchant profile, while card information, payment method, spending habits, etc. are applied to form a card user profile. Moreover, other studies attempt to increase the success rate of POC detection by associating these profiles with each other [30], [31]. Another study proposes an approach that can detect card POCs for a certain exposure window by using the information of a credit card that is known to have been skimmed [32]. One thing these studies have in common is that they conduct retrospective analysis based on the statistics that are obtained from fraudulent transactions and that they are based on statistical calculations. The results of these studies do not provide a quantitative success rate.

Araujo *et al.* [33] likens the POC detection problem to that of detecting malware in a computer file system and uses a bipartite graph to solve this problem. However; for data privacy concerns, the study does not provide details about the use of datasets. Additionally, it does not disclose how many POCs were detected for which time window, nor does it state the number of POCs that would be used as reference for a success assessment. Another drawback of the method proposed in the study is that detecting a POC requires other POCs that have been detected before.

The literature review shows that the proposed methods for POC detection, with the exception of scientific publications in patent format, rely on the current state and do not conduct retrospective analyses. Meanwhile, the studies involving patents propose that POC detection accuracy and success rate could be improved by starting off with the pre-current state transactions by employing a retrospective approach. Nevertheless, these approaches have been proposed only conceptually; they have not been tested on any datasets, nor have their impact on success been studied experimentally. Furthermore, the retrospective analysis period in the patents were set completely with a hypothetical approach. Also, a review of these studies reveal that they adopted a general approach to conduct retrospective analysis through fundamental statistical methods. POC is a point which is used at a time before detection of the fraud and at which the fraudulent cards were used. Therefore, a retrospective analysis is a factor in determining success. In our study, we combined machine translation techniques with a retrospective analysis while employing the actual card transactions in a six-month span, and put forth the factors that affect the success rate both temporally and spatially.

III. SYSTEM DESIGN

This study aims at detecting POCs by identifying and clustering similar fraudulent card transactions, followed by a retrospective analysis of respective transactions in each cluster to establish common spending points. To this end, consequent to the necessary feature extraction an unsupervised learning was employed. The transactions in the obtained clusters were then subjected to the aforementioned retrospective analysis with regard to the spending location and amount to trace POCs. Figure 1 provides the architecture of the proposed system. The dataset and system details are explained in the respective sub-sections.

A. DATASET

As mentioned earlier, one of the distinguishing characteristics of our proposed system is the fact that it was built and evaluated using real card transactions. On that account, we collaborated with the BKM in our research. BKM facilitates and supports the systems, platforms and infrastructure for every money transfer and card payment between 35 banks in Turkey. This collaboration also facilitated a thorough retrospective analysis as a result of having all past transactions, fraudulent or not, at our disposal. It is important to note that actual skimming could have taken place during customer's legal transactions. Therefore, the temporal analysis must include all the transactions. In order to evaluate the performance of our proposed system we compared our results regarding detected POCs to the groundtruth data provided by one of the largest and well-established member banks of BKM.

The dataset for tracing the POCs was constructed using fraudulent transactions reported to the BKM between 2017 and 2018, and consists of 681.862 instances. The groundtruth data consists of 9 points of purchase, which have been established to be actual POCs by a BKM member bank in December 2018. Table 1 presents the selected 21 features, as well as the groups for these features related to the transactions in the dataset. Feature groups are explained in detail in the experiments section.

Banks consider different scenarios as a means to conduct a fraudulent card transaction. These scenarios include on one

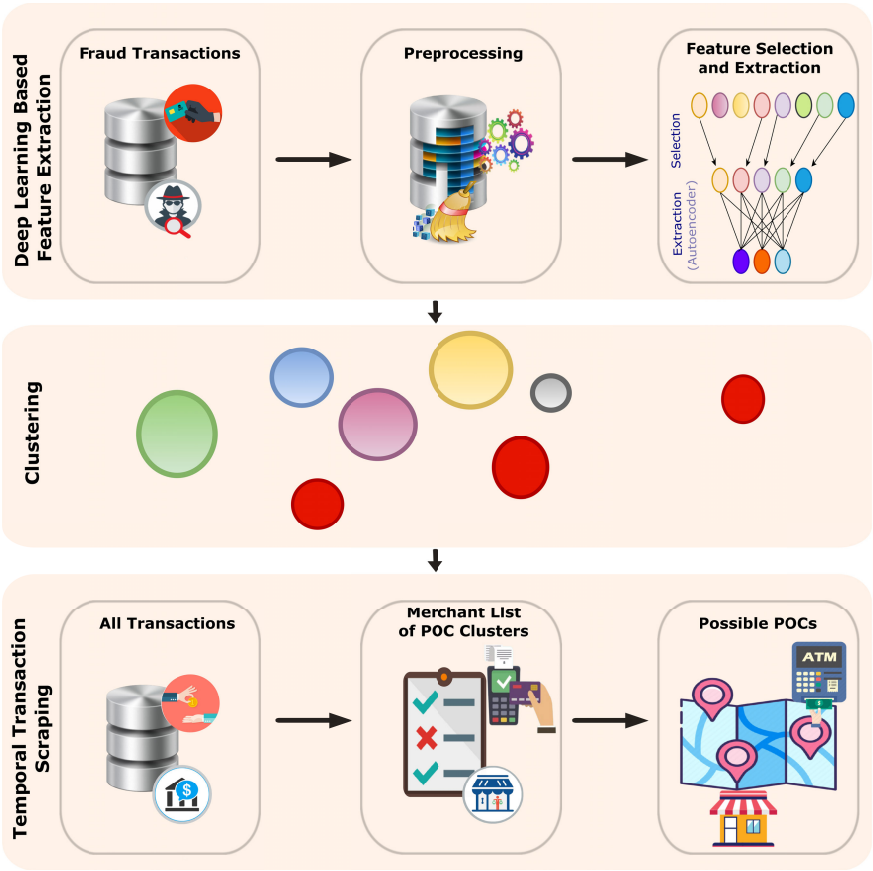


FIGURE 1. High-level data flow diagram of the proposed system, Red clusters indicates PPOC clusters.

TABLE 1. Feature types and selected features of fraud transactions.

Feature Type	Selected Features
Bank Features	Source Member Name
	Microfilm Reference Number
	Destination Member Name
	Pos Terminal Capability
Transaction Features	ATM Flag
	Contactless Flag
	Chip Flag
	Ecommerce Flag
	Transaction Channel
	Purchase Date
	Amount
	Card Brand
	Cardholder Id Method
	Pan Entry
	Payment Schema
Merchant Features	Merchant No
	Merchant City
	Merchant Country
	Merchant Category Code(MCC)
Fraud Features	Fraud Date
	Fraud Type

hand cards reported as lost, stolen or not received by the users, and on the other hand transactions stated as counterfeit by the card users or the issuers. This information is provided by the banks in the "Fraud Type Code" field of the TC40 [34] message. The code corresponding to counterfeit transactions is defined as "4". Unfortunately, at the time of reporting a fraudulent transaction by the user, this transaction could not be categorized as skimming immediately without further investigation. Therefore, the code "4" is ambigious in its definition and in real-life both banks and BKM assume that a code of "4" almost always indicates a counterfeit transaction. Even if further investigation reveals that the reported transaction took place in a POC, the reason code still remains "4". Therefore, all TC40 records with a reason code of "4" are generally treated as counterfeit transactions.

Investigation of the TC40 data revealed that skimming cases took place either at ATMs or at various businesses, particularly in restaurants and clothing stores, as well as gas stations. One skimming case showed that after 20 credit cards were skimmed at a restaurant, 181 fraud transactions took place in a span of three months. Ninety percent of these fraudulent transactions took place in the form of ATM withdrawals in similar amounts. In addition to ATM withdrawals, purchases were made in the close-by districts of the same city. Investigation of another case revealed a total of 41 fraudulent transactions, 40 of which were in the form of ATM

withdrawals, after seven credit cards were skimmed at a sports center. ATM withdrawal amounts were again similar to each other. Further investigation yielded similar results. Therefore, we could assume that cards skimmed at a given POC were subjected to similar spending characteristics.

B. PREPROCESSING

Credit card transaction data consist of the transaction details that banks report to the BKM. Since banks gather information from various sources, they may provide the data in different forms. Therefore, the need for preprocessing becomes ever more important. Thus, we applied the following operations. *data cleansing, aggregation, normalization,* and *data transformation*.

Inaccurate, incomplete or missing values reduce the success of the models, and thus, they should be fixed. Therefore, the following actions were performed on the dataset with regard to data cleansing.

- Redundant fields within the instances were removed.
- Format differences for the same field resulting from inconsistencies among the banks were fixed and unified.
- Incomplete/missing fields were remedied through imputation by filling out the empty values either with the values of similar instances or the average value of the respective field.

Data aggregation is a preprocessing action that is required to identify the data patterns and trends. Spending dates were aggregated using the week that the spending took place in order to express them holistically and to reduce the number of features required to represent the spending dates. The same approach was also used for special occasions such as the New Year, National/Religious holidays, and Black Friday. Min-Max normalization was then applied to the transaction amount and to the spending dates after they have been aggregated as outlined above. Afterwards, all the categorical features within the dataset were converted into corresponding numerical features. This was achieved by applying One-Hot Encoding [35] which converts categorical variables in numeric form during the preliminary data preparation. It takes a column with categorical data and creates new columns with as many different categorical values as in this column.

C. FEATURE SELECTION AND EXTRACTION

Various analyses were conducted for a better assessment of the dataset, and correlation matrices were created to examine the association of pair of features. When creating the correlation matrices, different metrics were used based on the type of features. If both features were categorical, Cramer's V [36]; if one was categorical and the other was numerical, Correlation Ratio [37]; and if both were numerical, Pearson correlation metrics [38] were used to determine the correlation values.

- *Cramer's V metric:* If both features are categorical, then the Cramer's V metric is used to determine the correlation value. In the equation 1; χ^2 is the chi-squared statistic, N is the sample number, and k represents the number of categories of the binary feature with the fewest number of categories.

$$\text{Cramer's V} = \sqrt{\frac{\chi^2}{N(k-1)}} \tag{1}$$

- *Pearson correlation metric:* If both features are numeric, then the Pearson correlation metric is used to determine the correlation value. In the equation 2; X and Y represent feature variables of two instances, $\overline{X}$ and $\overline{Y}$ represent the average value of these variables, and n is the number of total samples.

$$Pearson_{XY} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \tag{2}$$

- *Correlation Ratio metric:* If one feature is categorical and the other is numerical, then the correlation ratio is used to determine the correlation values. In the equation 3; X and Y are random variables, $Var(Y)$ is the variance of Y, $Var(Y|X)$ is the conditional variance, E is the expected value, or in other words, its average.

$$\eta^2_{Y|X} = 1 - \mathsf{E}\left[\frac{\mathsf{V}ar(Y \mid X)}{\mathsf{V}ar(Y)}\right] \tag{3}$$

Some metrics also provide the direction of the correlation by producing results in the range of $[-1, 1]$. However, as the correlation itself was the determining factor, the absolute value function was utilized in order to equalize the outcomes of the above mentioned metrics into the [0, 1] range.

Feature selection is done by analysing correlation matrix to eliminate highly correlated feature pairs. In order to determine high correlation between features, we used a threshold with value 0.9. After elimination process, we had 21 selected features.

In order to refine the characteristic features obtained at the end of the feature selection process and to achieve dimensionality reduction, feature extraction was applied. This process also helped to transform possibly correlated features into a smaller number of uncorrelated features. Both Principal Component Analysis (PCA) and Autoencoder methods were examined for feature extraction.

1) PRINCIPLE COMPONENT ANALYSIS

PCA [39] is a linear conversion method that minimizes the dataspace by promoting the features with high variance. PCA is not a suitable method for categorical feature data because it is dependent on variance values. Since the nominal categorical features may have different values in the dataset used, building the PCA model would be almost impossible due to the data sparsity and the large dataspace following the One-Hot Encoding.

Incremental PCA (IPCA) [40] was preferred because this PCA model is a more suitable variant for memory use in large datasets. In PCA methods, examining the effect of the principal components on cumulative variance is one of the approaches that is applied to establish the size of the new feature space. The size of the new feature space was established

by examining the cumulative variance values for the trained IPCA model.

2) AUTOENCODER

Autoencoder [41] is a type of artificial neural network with a multi-layered structure made up of perceptrons that are developed by modeling the neurons of the human brain. Autoencoders represent unsupervised learning because the class information is the data itself in them. Autoencoders may also be employed for purposes such as reducing data noise or identifying anomalies. They were used for feature extraction in this study. Different feature spaces were experimented by examining the reconstruction error to establish the size of the new feature space that would be attained with the Autoencoder.

Parameter and model selection for the PCA and Autoencoder is explained in the Experiments section.

D. CLUSTERING SIMILAR FRAUD TRANSACTIONS

Since we observed that the skimmed credit cards were usually used in batches, we examined whether the POCs could be detected by grouping collective fraudulent transactions. The K-means algorithm was applied to cluster the fraudulent transactions. However, since we cannot foresee the number of clusters in a given fraudulent transaction dataset, the value of k, which represents the number of clusters, must be determined dynamically. In order to establish an effective k value, we first utilized the Elbow method [42] by generating Elbow curves corresponding to the instances under consideration. In doing so, for different k values within-cluster sum of squares (WCSS) values were obtained. Equation 4 presents the within-cluster sum of squares (WCSS) where x gives the instance in the ith cluster, and c_i the cluster centers. There are a number of proposed approaches for detecting the optimal knee in such discrete data. According to [43], we chose to apply the Kneedle algorithm to determine the optimal k value.

$$\text{WCSS} = \sum_{i=1}^{k} \sum_{x \epsilon C_i} \text{Distance}(c_i, x)^2 \tag{4}$$

For each transaction within the fraudulent transaction dataset, a fraud type code is given as depicted in Table 2. As mentioned earlier in Section III-A upon feedback from card users, banks especially examine transactions with a fraud type code of "4". Consequently, such transactions initially reported as counterfeit would turn up as skimming frauds, i.e. as pointers to eventual POCs. Unfortunately, some skimming frauds could not be identified as a result of the lack of user feedback. On the other hand, fraudulent transactions reported as "0" through "3" could also have been used in a skimming operation indirectly. Therefore, it is imperative to cluster all the fraudulent transactions not just only transactions with a type code of "4". This allows us to examine the counterfeit transactions that were not identified as such in the dataset.

TABLE 2. Different types of frauds in dataset.

Fraud Type Code	Fraud Type
0	Lost card
1	Stolen card
2	Card never arrived fraud
3	False application fraud
4	Counterfeit and skimming fraud
5	Other frauds

After clustering, the counterfeit transactions would be scattered to some clusters more intensely than the others due to their similarity. As per our observation the credit cards involved in those counterfeit transactions would have been skimmed at some common business during casual spending transactions.

E. TEMPORAL TRANSACTION SCRAPING

In temporal transaction scraping all the candidate POC clusters, where each cluster consists of similar counterfeit transactions, were put through a retrospective analysis. In order to methodically determine candidate POC clusters, we formulated the counterfeit ratio value. This value represents the ratio of transactions with a fraud type code of "4" to all the transactions in a given cluster. We call clusters with a ratio exceeding a given threshold d as candidate POC clusters or POC clusters in short. Specific details on the selection of this value and its effect on the performance are elaborated in Experiments section. For the purpose of carrying out the aforementioned retrospective analysis the transactions in each POC cluster should be related to the corresponding card. However, this would yield duplicate cards. Therefore, for each POC cluster duplicate cards were eliminated and the rest of the analysis was carried out using distinct cards. The next step was merely a database query to establish the point of purchases where these cards had been used within a given backward time period. The obtained merchants were then expressed as time-ordered intersection sets. This operation enables us to obtain the discredited shopping points. After sorting this merchant list based on their appearances and scores, the first N entries of each set, i.e. highly suspicious shopping centers, were used to form the candidate POC list. Similarly, this list contained duplicate merchant entries which were eliminated too. The details of the temporal transaction scraping is given in Algorithm 1.

IV. EXPERIMENTS

In the experiments we employed card transactions reported to the BKM between 2017 and 2018. There are over 2.44 billions of total transactions for this time period. Of those transactions 681.862 were fraudulent and furthermore 21.298 of those belonged to December 2018. The 9 POCs, that were provided, were detected as follow up investigations of the reporting bank team triggered by customer complaints regarding their expenditure in December 2018. Therefore, the 21.298 fraudulent transactions belonging to this time period, were used as the initial seed for clustering. On the

Algorithm 1 Temporal Transaction Scraping Algorithm

Inputs:

Dataset: $X = \{x_1, x_2, \ldots, x_n\}$, Number of Clusters: k, Cluster Labels: $L = \{l(x) \mid x = 1, 2, 3, \ldots, n\}$, Fraud Transaction Clusters: $C = \{c_1, c_2, \ldots, c_k\}$

Outputs:

Set of Possible POC(s): $PPOC\{\}$

Parameters:

$t(integer)$: exposure window size as months, $d(float)$: counterfeit ratio threshold for selecting PPOC cluster, $N(integer)$: maximum number of candidates for a cluster

Algorithm:

$candidateList[\,] \leftarrow list()$
for $c_i \in C\{\}$ **do**
  $counterfeitRatio_i \leftarrow \frac{\sum counterfeits \in c_i}{\sum allfrauds \in c_i}$
  if $counterfeitRatio_i \geq d$ **then**
    $cards_i\{\} \leftarrow$ get distinct cards in *fraud transactions* $\in c_i$
    $prefraudCounts_i\{\} \leftarrow dict()$
    for $card_{ij} \in cards_i\{\}$ **do**
      $prefrauds_{ij}\{\} \leftarrow db.getAllTransactions(card_{ij})$ in exposure window(t)
      for $prefraud_{ijt} \in prefrauds_{ij}\{\}$ **do**
        $merchant \leftarrow$ get merchant from $prefraud_{ijt}$
        if $merchant \in prefraudCounts_i.keys$ **then**
          $prefraudCounts_i[merchant] \leftarrow prefraudCounts_i[merchant] + 1$
        else
          $prefraudCounts_i[merchant] \leftarrow 1$
        end if
      end for
    end for
    $Merchants_i\{\} \leftarrow prefraudCounts_i.keys$
    $merchantScores_i\{\} \leftarrow db.getMerchantScores(Merchants_i)$
    $sortedMerchants_i\{\} \leftarrow$ sort $Merchants_i\{\}$ by $prefraudCounts_i\{\}$ and $merchantScores_i\{\}$
    $selectedMerchants_i\{\} \leftarrow$ select first N merchants from $sortedMerchants_i\{\}$
    $candidateList[\,].append(selectedMerchants_i)$
  end if
end for
$PPOC\{\} \leftarrow$ get distinct candidates in $candidateList$

[1]Object db stands for an interface class that allows us to execute database queries.
[2]Subscripts i, j, t denotes indices for respectively clusters, cards, and transactions.

other hand, all the fraudulent transactions, i.e. 681.862, were exploited in the feature extraction phase. However, as a preliminary step, 45 raw features were analyzed using the correlation based approach described in Section III-A and were eliminated to 21 features (Table 1). 18 out of 21 features were converted into their one hot encoded representation since they were categorical features. Consequently, the initial feature space was expanded to 14.802 features. Then, we opted to utilize and compare the performances of two different feature extractors, namely PCA and Autoencoder. The aforementioned clustering took place via the extracted features.

The distribution of the binary combination of the principle components of the PCA extractor and the deep features of the Autoencoder extractor are given in Figure 2. The plots d-e clearly demonstrate that Autoencoder extracted features produce better clustering than PCA.

Before performing the retrospective transaction analysis for the cards existing in the clusters, clusters having a d value below the threshold were eliminated. Empirical results showed that a threshold value greater than 0.7 resulted in excessive elimination of clusters whereas less than 0.4 did not contribute to the elimination process substantially. Therefore, we concluded to select the value d as $0.4 \leq d < 0.7$. The remaining clusters were then subjected to the retrospective transaction analysis as outlined in Section III-E.

For all the experiments, we utilized the IBM PowerAI system, which is designed for use in deep learning studies.

A. CLUSTERING FRAUD TRANSACTIONS VIA PCA EXTRACTED FEATURES

Considering the huge number of instances to be included in the feature extraction process, we utilized the IPCA

FIGURE 2. Clusters projected onto a reduced feature space containing binary combinations of the extracted features. PCA (top), Autoencoder (bottom).

model [40], a PCA method variant, in order to improve the memory usage. The IPCA model was realized incrementally in batches of 500 fraudulent transactions. Changes in the number of eigenvalues were examined for dimension estimation of the feature set. The resulting scree plot is given in Figure 3. Kneedle algorithm was applied to the y-axis values to determine the optimum knee point which turned out to be "3" for our dataset. Thus, in the rest of experiments we utilized those extracted four PCA features.

December 2018 fraudulent transactions were then subjected to feature extraction using the PCA model obtained and K-means clustering was performed. The resulting Elbow graph is given in Figure 6. Although the Kneedle algorithm pointed to 25 clusters, further analysis showed that better results could be achieved with 30 clusters. Therefore, we moved on with 30 clusters.

B. CLUSTERING FRAUD TRANSACTIONS VIA AUTOENCODER EXTRACTED FEATURES

As previously mentioned all the fraudulent transactions were input to the Autoencoder model to obtain extracted features. The Autoencoder model architecture consisted of a simple model architecture made up of a single hidden layer shown in Figure 4. We employed Rectified Linear Unit activation [44] with L1 regularizer [45] for encoding phase and sigmoid activation for decoding phase. We also employed the AdaDelta Optimization [46] both for speedy

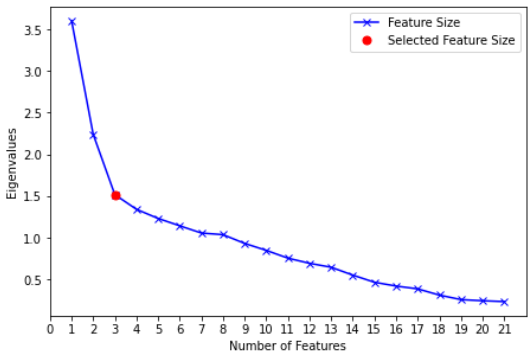


FIGURE 3. Scree plot of PCA showing eigenvalues of each components.

binary cross entropy and achieving convergence with fewer iterations.

In order to determine the optimum number of features the output of the Autoencoder was varied from 1 to 20 in steps of 2 and for each output size the Autoencoder was run 10 times. The mean square error was used as the reconstruction error criterion. The resulting graph, given in Figure 5, shows that no particular change was observed in the reconstruction error for output sizes greater than five. Also, application of the Kneedle algorithm yielded the value of three. Therefore, we decided to use an Autoencoder extractor with an output size of three.

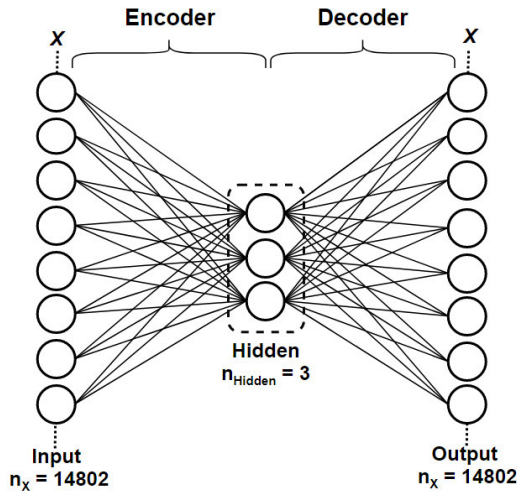


FIGURE 4. Single hidden layered autoencoder model architecture.

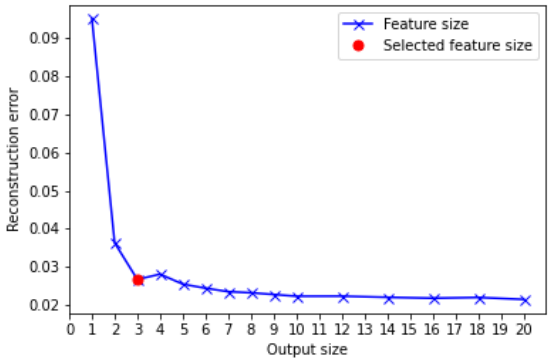


FIGURE 5. Scree plot of Autoencoder showing reconstruction error for different latent feature sizes.

Similar to the PCA feature extractor the obtained Autoencoder model was applied to the fraudulent transactions of December 2018. The resulting Elbow graph of the K-means clustering applied to the extracted features is given in Figure 6. Again, the Kneedle algorithm indicated an optimum cluster size of 25, but further analysis showed that, as in the PCA case, 30 clusters would yield better results.

C. TEMPORAL ANALYSIS FOR POINT OF COMPROMISE

As outlined in the Experiments Section, the d value is crucial for a correct elimination of spurious clusters to accelerate and to improve the POC tracing process. However, in order to determine the optimum d value temporal analysis should also be run and the number of suggested and correctly detected POCs must be evaluated against groundtruth POC data. For a thorough analysis, while varying the d value from 0.4 to 0.7 in 0.1 steps, as given in Figure 7, we carried out a retrospective temporal analysis starting with one month and ranging up to six months in one-month intervals. The whole process was executed for both PCA and Autoencoder feature extractors, thus, giving a total of 120 different scenarios. The summary of these scenarios are given in Figure 7 and Figure 9. These figures also include the results for d values below 0.4 and above 0.7 just to illustrate the validity of the aforementioned range of 0.4-0.7 for d. As can be seen below 0.4 no feasible cluster elimination would be attained and above 0.7 some of the viable clusters would be lost. Consequently, we opted to choose a d value of 0.4.

TABLE 3. Effect of different N values on both suggested and detected POCs.

N	Suggested POCs	Detected POCs
10	23	2
20	45	3
30	67	4
40	89	5
50	105	7
60	145	7
70	193	7
80	236	7
90	274	7

The remaining clusters were subjected to the temporal scraping algorithm given in Algorithm 1. In order to be on the safe side, we extended the backward-looking analysis up to 12 months (Figure 8) which in turn revealed the fact that the number of fraudulent transactions decrease gradually both when getting closer to time period when the actual skimming occurred and the time period when the POC was identified. Thus, the fraudsters put some time frame after the skimming before actually exploiting the cards. Then, the number of exploits grow to a peak point and decrease eventually as the risk of the identification of the POC increases.

Similar to the analysis of the d value, the backward-looking analysis showed that the optimal backward range lies in two or three months as given in Figure 9. After that the success rate falls considerably. Therefore, a three-month time period was chosen as the exposure window. Although we are fully aware of the fact that the exposure window size is tightly correlated to the dataset contrary to the current literature [27]–[29] which suggests a window size between one week to one month our research clearly showed that given the characteristics of the fraudulent transactions an exposure window size of three months would yield considerably better results.

Although each cluster contains unique instances applying, a temporal analysis on this instances would produce duplicate merchants which is actually a desirable outcome. Because recurring merchants as a result of this backward-looking analysis indicates an intersection of these pseudo unrelated transactions at some time in the past at a specific location. Naturally, duplicate merchant ids should be unified. The last varying parameter of N is used to somehow control the number of obtained matching merchant ids. Up to some point increasing the number N also increases the number of matched merchants but also produces a larger list of suggested POCs. As can be seen in Table 3, our analysis showed that a value greater than 50 did not change the number of detected POCs but considerably increased the number of suggested POCs. Therefore, for our dataset, we chose the value of N as 50.

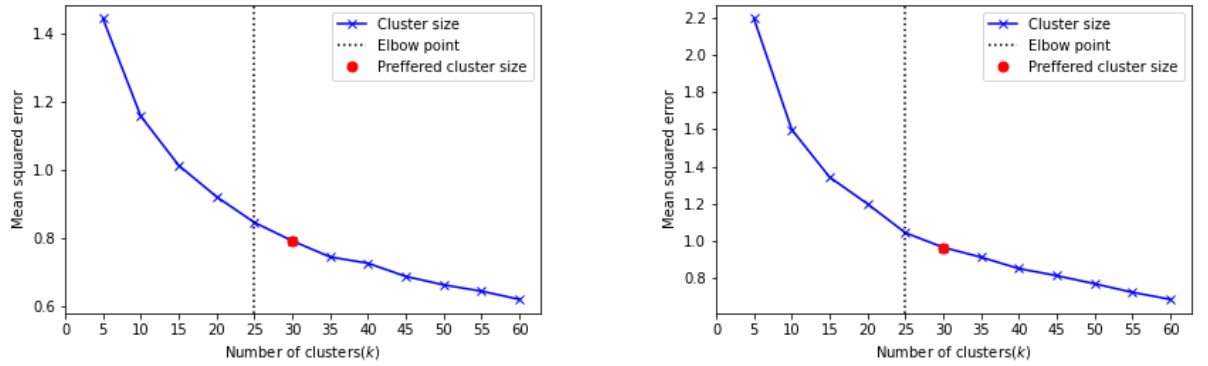


FIGURE 6. Elbow plots of clustering on feature extraction methods: PCA and Autoencoder.

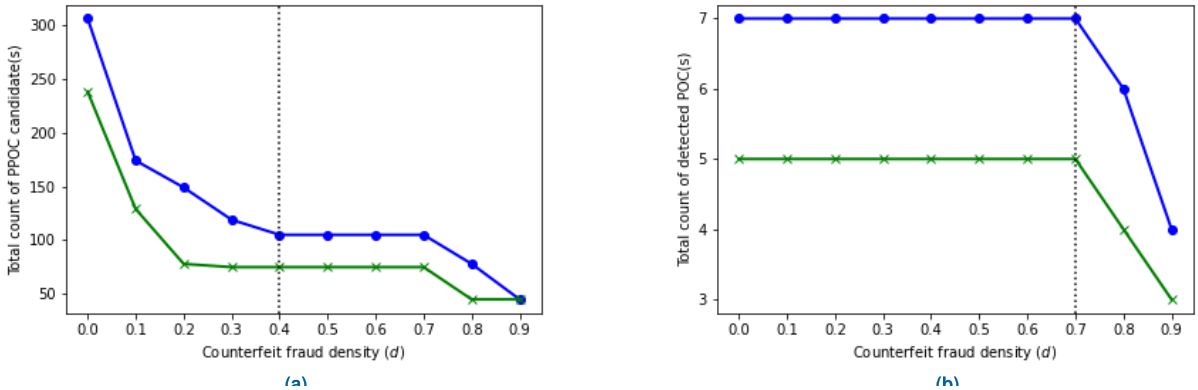


FIGURE 7. (a) Total count of Possible POC candidates predictions for each different counterfeit density threshold(d) selection (b) Total count of detected POCs for each different counterfeit density threshold(d) selection.

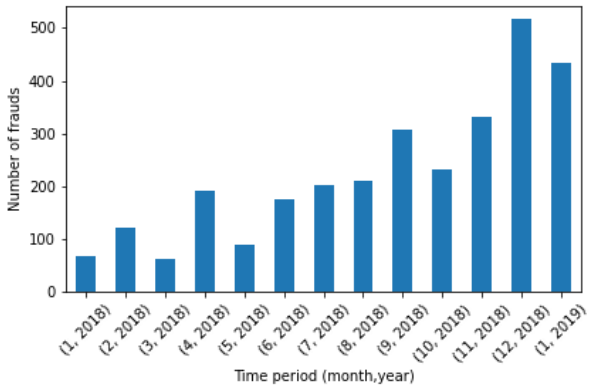


FIGURE 8. Past fraudulent transaction counts of compromised accounts related with reference POCs.

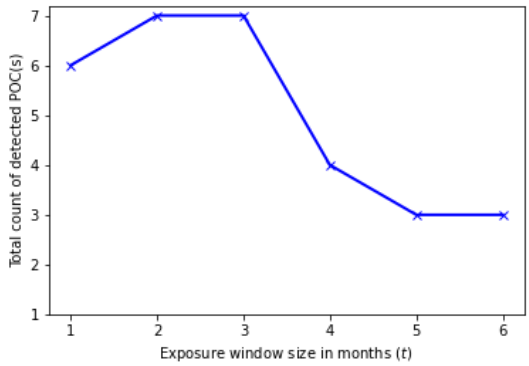


FIGURE 9. Effects of using different exposure windows on detecting POCs.

TABLE 4. Results of different methods.

Methods	Candidate POC Clusters	Suggested POCs	Detected POCs
PCA & K-means	2	75	5
AE & K-means	3	105	7

Table 4 presents the number of candidate clusters after elimination, the number of suggested and identified POCs as produced by our approach. It is evident that Autoencoder extracted features clustered with K-means outperforms the K-means clustering with PCA extracted features. Therefore, for the dataset under consideration the best approach is K-means clustering precedeed by Autoencoder extraction and followed by a retrospective analysis within an exposure window of three months.

V. CONCLUSION

In this study, we introduced a new POC detection mechanism using the similarity information of fraud transactions in order to pinpoint the skimming points which requires an exhaustive manpower to be identified. Our novel approach enabled us to trace POCs with zero-knowledge on the existing POCs and

to prevent possible credit card fraudulent transactions which might occur in the future.

The test results obtained from real-life transaction dataset showed us utilizing Autoencoder extractor in our method outperforms the PCA extractor. Thus, we were able to identify 7 POCs out of 9 by giving Autoencoder features into K-means algorithm and then analyzing the transactions in the obtained clusters retrospectively. The whole process was completed within a period of 5 minutes involving the aforementioned dataset with 2.44 billions of transactions. It is also important to note that our system only suggested 105 possible target POCs among thousand merchants. On the other hand, we presented the effect of retrospective analysis and determined that looking three months backward would be sufficient enough to identify the maximum number of skimming points. This evaluation could avert unnecessary labor for analyzing millions of transactions older than three months. Thus, the manpower could be benefited at minimum during POC tracing.

ACKNOWLEDGMENT

The authors would like to sincerely thank Bankalararası Kart Merkezi (BKM) who prepared the data set and helped the authors to understand the fraudulent behaviors and data set features as well as provided the necessary infrastructure for GPU-based machine learning on big data.

REFERENCES

[1] W. Jolly. (2021).*5 Most Common Types of Credit Card Fraud Explained*. [Online]. Available: https://www.savings.com.au/credit-cards/credit-card-fraud

[2] P. Muncaster. (2021). *Web Skimmers Use Phishing Tactics to Steal Data*. [Online]. Available: https://www.infosecurity-magazine.com/news/web-skimming-attacks-phishing/

[3] W. Ashford. (2021). *Magento E-Commerce Sites Urged to Apply Security Update*. [Online]. Available: https://www.computerweekly.com/news/252460661/Magento-e-commerce-sites-urged-to-apply-security-update

[4] Sansec. (2021). *Cardbleed: A Massive Magento1 Hack*. [Online]. Available: https://sansec.io/research/cardbleed

[5] S. B. E. Raj and A. A. Portia, "Analysis on credit card fraud detection methods," in *Proc. Int. Conf. Comput., Commun. Electr. Technol. (ICCCET)*, Mar. 2011, pp. 152–156.

[6] H. Hofmann. (2021). *UCI Machine Learning Repository*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

[7] (2021). *UCI Machine Learning Repository*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)

[8] E. Aleskerov, B. Freisleben, and B. Rao, "CARDWATCH: A neural network based database mining system for credit card fraud detection," in *Proc. IEEE/IAFE Comput. Intell. Financial Eng.*, Mar. 1997, pp. 220–226.

[9] T. K. Behera and S. Panigrahi, "Credit card fraud detection: A hybrid approach using fuzzy clustering & neural network," in *Proc. 2nd Int. Conf. Adv. Comput. Commun. Eng.*, May 2015, pp. 494–499.

[10] B. Can, A. G. Yavuz, E. M. Karsligil, and M. A. Guvensan, "A closer look into the characteristics of fraudulent card transactions," *IEEE Access*, vol. 8, pp. 166095–166109, 2020.

[11] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick, "Credit card fraud detection using Bayesian and neural networks," in *Proc. 1st Int. Naiso Congr. Neuro Fuzzy Technol.*, 2002, pp. 261–270.

[12] Y. G. Şahin and E. Duman, "Detecting credit card fraud by decision trees and support vector machines," in *Proc. Int. Multi-Conf. Eng. Comput. Scientists (IMECS)*, Hong Kong, vol. 1, Mar. 2011, pp. 1–6.

[13] A. C. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, "Detecting credit card fraud using periodic features," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 208–213.

[14] A. Mishra and C. Ghorpade, "Credit card fraud detection on the skewed data using various classification and ensemble techniques," in *Proc. IEEE Int. Students' Conf. Electr., Electron. Comput. Sci. (SCEECS)*, Feb. 2018, pp. 1–5.

[15] K. Randhawa, C. K. Loo, M. Seera, C. P. Lim, and A. K. Nandi, "Credit card fraud detection using AdaBoost and majority voting," *IEEE Access*, vol. 6, pp. 14277–14284, 2018.

[16] A. D. Pozzolo, O. Caelen, Y.-A. L. Borgne, S. Waterschoot, and G. Bontempi, "Learned lessons in credit card fraud detection from a practitioner perspective," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4915–4928, 2014.

[17] T. M. Padmaja, N. Dhulipalla, R. S. Bapi, and P. R. Krishna, "Unbalanced data classification using extreme outlier elimination and sampling techniques for fraud detection," in *Proc. 15th Int. Conf. Adv. Comput. Commun. (ADCOM)*, Dec. 2007, pp. 511–516.

[18] U. Fiore, A. De Santis, F. Perla, P. Zanetti, and F. Palmieri, "Using generative adversarial networks for improving classification effectiveness in credit card fraud detection," *Inf. Sci.*, vol. 479, pp. 448–455, Apr. 2019.

[19] S. Dhankhad, E. Mohammed, and B. Far, "Supervised machine learning algorithms for credit card fraudulent transaction detection: A comparative study," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, Jul. 2018, pp. 122–125.

[20] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognit.*, vol. 74, pp. 406–421, Feb. 2018.

[21] N. Malini and M. Pushpa, "Analysis on credit card fraud identification techniques based on KNN and outlier detection," in *Proc. 3rd Int. Conf. Adv. Electr., Electron., Inf., Commun. Bio-Inf. (AEEICB)*, Feb. 2017, pp. 255–258.

[22] F. Carcillo, Y.-A. L. Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Inf. Sci.*, vol. 557, pp. 317–331, May 2021.

[23] A. Srivastava, A. Kundu, S. Sural, and A. K. Majumdar, "Credit card fraud detection using hidden Markov model," *IEEE Trans. Depend. Sec. Comput.*, vol. 5, no. 1, pp. 37–48, Jan./Mar. 2008.

[24] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, and P. Beling, "Deep learning detecting fraud in credit card transactions," in *Proc. Syst. Inf. Eng. Design Symp. (SIEDS)*, Apr. 2018, pp. 129–134.

[25] Z. Kazemi and H. Zarrabi, "Using deep networks for fraud detection in the credit card transactions," in *Proc. IEEE 4th Int. Conf. Knowl.-Based Eng. Innov. (KBEI)*, Dec. 2017, pp. 630–633.

[26] A. Pumsirirat and L. Yan, "Credit card fraud detection using deep learning based on auto-encoder and restricted Boltzmann machine," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 1, pp. 18–25, 2018.

[27] V. F. Klebanoff, "Method and system for assisting in the identification of merchants at which payment accounts have been compromised," U.S. Patent 7 580 891, Aug. 25, 2009.

[28] S. Yan, "System and method for detecting account compromises," U.S. Patent 8 600 872, Dec. 3, 2013.

[29] K. P. Siegel, R. A. Paynter, R. L. Grossman, C. Brown, C. R. Byce, T. Dwyer, and A. Chen, "System and method for identifying a point of compromise in a payment transaction processing system," U.S. Patent 8 473 415, Jun. 25, 2013.

[30] S. M. Zoldi, L. Wang, L. Sun, and S. G. Wu, "Mass compromise/point of compromise analytic detection and compromised card portfolio management system," U.S. Patent 7 761 379, Jul. 20, 2010.

[31] S. Zoldi and M. Urban, "Detection of compromise of merchants, ATMS, and networks," U.S. Patent 10 115 153, Oct. 30, 2018.

[32] G. Forman, "Determining point-of-compromise," U.S. Patent 10 654 821, Mar. 10, 2005.

[33] M. Araujo, M. Almeida, J. Ferreira, L. Silva, and P. Bizarro, "Breachradar: Automatic detection of points-of-compromise," in *Proc. SIAM Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, 2017, pp. 561–569.

[34] VISA. (2021). *Visa Global Acquirer Risk Standards*. [Online]. Available: https://usa.visa.com/dam/VCOM/download/merchants/visa-global-acquirer-risk-standards.pdf

[35] D. Harris and S. L. Harris, *Digital Design and Computer Architecture*. San Mateo, CA, USA: Morgan Kaufmann, 2010.

[36] H. Cramér, *Mathematical Methods of Statistics*, vol. 43. Princeton, NJ, USA: Princeton Univ. Press, 1999.

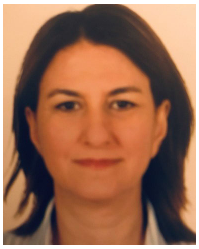
[37] R. A. Fisher, "Statistical methods for research workers," in *Breakthroughs in Statistics*. New York, NY, USA: Springer, 1992, pp. 66–70.

[38] K. Pearson, "VII. Note on regression and inheritance in the case of two parents," *Proc. Roy. Soc. London*, vol. 58, nos. 347–352, pp. 240–242, 1895.
[39] F. R. S. K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *London, Edinburgh, Dublin Philosoph. Mag. J. Sci.*, vol. 2, no. 11, pp. 559–572, 1901.
[40] M. Artac, M. Jogan, and A. Leonardis, "Incremental PCA for on-line visual learning and recognition," in *Proc. Int. Conf. Pattern Recognit.*, vol. 3, Aug. 2002, pp. 781–784.
[41] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length, and Helmholtz free energy," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, pp. 3–10.
[42] R. L. Thorndike, "Who belongs in the family?" *Psychometrika*, vol. 18, no. 4, pp. 267–276, Dec. 1953.
[43] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a 'Kneedle' in a haystack: Detecting knee points in system behavior," in *Proc. 31st Int. Conf. Distrib. Comput. Syst. Workshops*, 2011, pp. 166–171.
[44] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 1–8.
[45] J. Lokhorst, "The lasso and generalised linear models," Honors Project, Dept. Statist., Univ. Adelaide, Adelaide, SA, Australia, 1999.
[46] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*. [Online]. Available: http://arxiv.org/abs/1212.5701

FUAT OGME received the B.Sc. and M.Sc. degrees in computer engineering from Yildiz Technical University, Istanbul, Turkey, where he is currently pursuing the Ph.D. degree. He is also a Research and Teaching Assistant in computer engineering with Yildiz Technical University. He is also a member of the Intelligent Systems Laboratory. His current research interests include information processing, cloud systems, artificial intelligence, machine learning, and deep learning.

A. GOKHAN YAVUZ received the Ph.D. degree in computer engineering from Yildiz Technical University, Istanbul, Turkey. He is currently an Associate Professor with the Department of Computer Engineering, Yildiz Technical University and the Co-Director of the Intelligent Systems Laboratory. His current research interests include systems and network security, cloud computing, and big data.

M. AMAC GUVENSAN (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in computer engineering from Yildiz Technical University, Istanbul, Turkey, in 2002, 2006, and 2011, respectively. From 2009 to 2010, he visited the Wireless Networks and Embedded Systems Laboratory, University at Buffalo, The State University of New York, for a period of six months. He is currently an Associate Professor with the Department of Computer Engineering, Yildiz Technical University. His current research interests include pervasive and ubiquitous computing, mobile technologies and applications, intelligent transportation systems, machine learning, and the Internet of Things.



M. ELIF KARSLIGIL received the Ph.D. degree in computer engineering from Yildiz Technical University, Istanbul, Turkey. From October 2001 to October 2002, she was a Postdoctoral Associate with NTT-Keihanna, Japan. She is currently an Associate Professor with the Department of Computer Engineering, Yildiz Technical University and the Co-Director of the Intelligent Systems Laboratory. Her research interests include machine learning and deep learning with applications in signal, image, and video processing.