IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 October 2023, accepted 21 November 2023, date of publication 28 November 2023,
date of current version 11 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3337635

RESEARCH ARTICLE

An Adversary Model of Fraudsters' Behavior to Improve Oversampling in Credit Card Fraud Detection

DANIELE LUNGHI[1,2,3,4], GIAN MARCO PALDINO[1], OLIVIER CAELEN[5], AND GIANLUCA BONTEMPI[1], (Senior Member, IEEE)

[1]Machine Learning Group, Computer Science Department, Université Libre de Bruxelles, 1050 Brussels, Belgium
[2]Department of Computer and Decision Engineering, Université Libre de Bruxelles, 1050 Brussels, Belgium
[3]Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 15784 Athens, Greece
[4]Athena Research Centre, 15125 Athens, Greece
[5]Worldline S.A., 1130 Brussels, Belgium

Corresponding author: Daniele Lunghi (daniele.lunghi@ulb.be)

The work of Daniele Lunghi was supported by the HORIZON EUROPE Marie Sklodowska-Curie Actions under Grant 955895. The work of Gian Marco Paldino and Gianluca Bontempi was supported by Service Public de Wallonie under Grant 2010235-ARIAC.

ABSTRACT Imbalanced learning jeopardizes the accuracy of traditional classification models, particularly for what concerns the minority class, which is often the class of interest. This paper addresses the issue of imbalanced learning in credit card fraud detection by introducing a novel approach that models fraudulent behavior as a time-dependent process. The main contribution is the design and assessment of an oversampling strategy, called ''Adversary-based Oversampling'' (ADVO), which relies on modeling the temporal relationship among frauds. The strategy is implemented by two learning approaches: first, an innovative regression-based oversampling model that predicts subsequent fraudulent activities based on previous fraud features. Second, the adaptation of the state-of-the-art TimeGAN oversampling algorithm to the context of credit card fraud detection. This adaptation involves treating a sequence of frauds from the same card as a time series, from which artificial frauds' time series are generated. Experiments have been conducted using real credit card transaction data from our industrial partner, Worldline S.A, and a synthetic dataset generated by a transaction simulator for reproducibility purposes. Our findings show that an oversampling approach incorporating time-dependent modeling of frauds provides competitive results, measured against common fraud detection metrics, compared to traditional oversampling algorithms.

INDEX TERMS Fraud detection, imbalance learning, machine learning, oversampling, synthetic data, time series, threat model.

I. INTRODUCTION

Learning from imbalanced data is a classical challenge in machine learning. Unfortunately, most machine learning algorithms assume data to be balanced and suffer from severe accuracy degradation when this assumption does not hold [1]. In a binary classification problem, this happens when a class (the *minority class*) is under-represented with respect to the other one (the *majority class*). Moreover, the minority class is often the class of interest: in fields like medicine [2], [3], disaster predictions [4], and fault prediction [5], we are especially interested in detecting what differs from the norm. A common problem with imbalanced data is that algorithms trained to maximize accuracy tend to ignore the observations of the minority class altogether. State-of-the-art algorithms are often designed to deal with balanced data sets and cannot correctly learn from heavily imbalanced data.

A significant example of imbalanced learning problems is credit card fraud detection, i.e., the task of automatically analyzing transactions to detect fraudulent ones [6], [7]. Credit card fraud detection is a very challenging data-driven

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

task since it presents many learning challenges, notably concept drift [8], verification latency [9], and a complex feature engineering process [10].

Moreover, fraudulent transactions are heavily outnumbered by genuine ones [11]. Specifically, the number of frauds can be less than two transactions per 1000 [12], making the distribution highly skewed towards the majority class.

In this paper, we focus on the problem of imbalanced learning in credit card fraud detection. In particular, we consider data-level, or sampling techniques, like *undersampling* [13] and *oversampling* [14] which reduce the majority class or size up the minority class, respectively. The currently existing methodologies fail to acknowledge that fraudsters may employ various strategies to circumvent card blocking and enhance the success rate of their fraudulent activities. Such strategies could involve complex decision-making patterns involving multiple fraudulent transactions masked as genuine transactions. Since, to the best of the authors' knowledge, the behavior of fraudsters is not yet seriously taken into consideration in the literature, we propose a strategy for augmenting the observed dataset with synthetic frauds that emulate the fraudster attitude.

First, we assessed whether the actions of fraudsters are somewhat related to the transaction history of the card they attack. We modeled the dependency between the last genuine and the first fraudulent card transaction, but we did not find sufficiently accurate results. Instead, we realized that the behavior of fraudsters does indeed exhibit time-dependent patterns. To leverage this, we propose a new framework, named *Adversary-based Oversampling* (ADV-O) for oversampling credit card fraudulent transactions, which relies on two different learning algorithms.

The first algorithm, named *MIMO ADV-O*, models the dependency between consecutive fraudulent transactions executed with the same card, emulating the behavioral patterns exhibited by fraudsters. Specifically, we train a multi-target regression model to forecast some characteristics of a fraudulent transaction based on the preceding one, and we use such a model to generate a set of artificial fraudulent transactions.

The second algorithm, called *TimeGAN ADV-O*, is based on TimeGAN [15], a popular adaptation of Generative Networks for time series. This algorithm is used to model each chain of fraudulent transactions as a time series.

Both MIMO ADV-O and TimeGAN ADV-O differ from traditional oversampling strategies used in fraud detection. Conventional techniques like SMOTE and GAN treat each fraudulent transaction as an independent sample from the same distribution and do not consider any dependency on the card used for the transaction or its transaction history. ADV-O algorithms, instead, model transactions as complex time-dependent problems.

The main contribution of this paper is a new framework to manage the imbalance in fraud detection data by explicitly modeling the behavioral patterns of fraudsters and their time-dependent dynamics. In particular

- We propose the first quantitative model of fraudsters' behavior, where the fraudsters' actions depend on the card they have access to and their previous actions.
- We uncover the existence of a substantial dependency between consecutive fraudulent transactions through a comprehensive analysis of transactions history.
- We design a novel framework composed of two oversampling algorithms, both based on fraudsters' behavior study, called MIMO ADV-O and TimeGAN ADV-O, respectively.
- We conduct a comprehensive experimental evaluation of the proposed methodology on over 20 million real credit card transactions, comparing the proposed approach with state-of-the-art oversampling techniques.
- To ensure the reproducibility of our results, we design a transactions simulator inspired by existing literature and replicate our experiments on synthetic, publicly available data.

The organization of the paper is the following: Section II reviews the current literature, Section III provides a formal description of the problem, Section IV describes the main contributions, while Section V assesses the effectiveness of our algorithm on both a large real dataset (20M+ transactions over two months) provided by our industrial partner and a synthetic dataset. We discuss the results obtained and analyze the plausible next research steps in Section VI.

II. STATE OF THE ART

Imbalanced learning plays an important role in multiple domains, such as rare events prediction [16], intrusion detection [17], churn prediction [18] and fraud detection [7]. We can categorize the approaches used to address this problem into two main families, *algorithm-level* and *data-level* techniques [19]. Since we consider imbalanced learning in fraud detection, modeled as a binary classification task, by the algorithm we mean the binary classifier used to discriminate fraudulent from genuine transactions.

Algorithm-level techniques address the problem by modifying the classifiers to guarantee robust accuracy concerning data imbalance. This includes adjusting the cost matrix to increase the importance of misclassified samples from the minority class [20] and using ensemble learning techniques, training multiple learners on different parts of the training set to artificially increase the importance of observations from the minority class [21].

Data-level approaches modify the training set before classification, allowing the use of standard classifiers. The two most common approaches in the literature are *undersampling*, which drops a certain number of majority class observations from the training set, and *oversampling*, which adds some artificial minority class samples to the dataset. Undersampling tends to work better for large datasets [11] where the likelihood that the maintained subsample carries enough signal is higher. The main issues with this approach concern the potential loss of useful information given by the removal of observations from the training set and the

potentially damaging change in the prior of the majority class [22]. An effective solution to the mentioned problem is EasyEnsemble [21]: this method exploits an ensemble of classifiers that are trained on different balanced subsamples of the original data. Each subsample is obtained through a random undersampling of the majority class, reducing the risk of losing information. The ensemble nature of a Random Forest algorithm makes it particularly suitable for the implementation of an EasyEnsemble technique: the model obtained is called Balanced Random Forest, and each of its Decision Trees is trained on a different data subsample [23]. The role of undersampling has been largely studied in our previous works [11], [24], showing good results in the fraud detection scenario.

A naive oversampling is achieved by randomly sampling and duplicating the observations of the minority class. This method increases the proportion of minority samples but is prone to overfitting. More complex oversampling methods, such as SMOTE [25], use data locality to generate new minority samples through interpolation. For instance, SMOTE randomly selects close samples in the feature space and interpolates between them to generate a new artificial sample. SMOTE has encountered vast success in the oversampling literature, and many variants have been proposed over the years. Some of the most cited and renowned techniques try to limit the application of SMOTE to some defined subspaces. For example, ADASYN [26] uses a weighted distribution for different minority class examples, where more samples are generated for classes harder to learn, while Borderline-SMOTE [27] samples from all classes (and thus not only from the minority class) only next to the border between different classes, as samples in those areas are more likely to be informative for the classifier.

A more recent version is K-Means SMOTE [28], which employs the k-means clustering algorithm together with SMOTE to re-balance skewed datasets. Such a combination aims to reduce the impact of noise and addresses both between-class and within-class imbalances. The method consists of three steps: clustering, filtering, and oversampling. The first step consists in clustering the input space into k groups. The second step selects the clusters with a high proportion of minority class samples and assigns more synthetic samples to generate clusters where minority samples are sparsely distributed. The third step applies SMOTE in each selected cluster. Unfortunately, most SMOTE implementations do not allow maintaining the structure in the data, as the interpolation between admissible observations may not be an admissible observation (for instance, you cannot have a non-integer cardholder age). More generally, such methods strongly rely on the data locality assumption and may be detrimental in the case of complex distributions of the minority class.

Generative methods for oversampling are based on the assumption that samples from the minority class follow a specific distribution [29]. The idea, common for Generative Adversarial Networks (GAN) [30] and Variational Auto Encoders (VAE) [31], is to model the distribution from historical data and use the estimated distribution to sample as many artificial minority observations as required. The adoption of GAN is considered a promising approach for fraud detection problems [32]: its main idea is to train two networks, a generator and a discriminator, that compete against each other. The generator takes a white noise signal as input and aims to generate new samples that resemble the original ones as much as possible. The discriminator is trained instead to discriminate real from artificially created instances. Their joint competitive learning increases the capability of the generator to produce representative artificial observations. An interesting example is CT-GAN [33], a GAN-based method to model tabular data distribution and sample rows from the distribution. CT-GAN authors propose a model-specific normalization to handle complex distributions, like non-Gaussian and multimodal. Furthermore, a conditional generator is proposed to deal with the imbalanced discrete columns not adequately handled by traditional GANs.

The existence of multiple techniques to deal with imbalanced learning in fraud detection led to the necessity of analyzing and comparing them. A recent survey [34] shows that SMOTE is the most used resampling technique in fraud detection but suggests that no single algorithm consistently outperforms all the alternatives. An empirical comparison of three resampling techniques in the context of fraud detection is performed in [35]. The authors compare Random Undersampling, SMOTE, and a variant of SMOTE called SMOTE Tomek [36], together with multiple features engineering techniques. Interestingly, this work results in Random Undersampling, obtaining the best performance among all algorithms, with SMOTE being the second-best performer. However, all performances are similar and significantly improved compared to the baseline, further showcasing the importance of resampling techniques for fraud detection.

Finally, when we consider the time dynamics of the frauds, we move into the domain of time series. Cardholders have specific spending patterns, creating time dependences among their transactions. In fraud detection literature, multiple works consider genuine transactions as time series to detect anomalies [37], [38], [39]. Instead, [40] uses the time series nature of the transactions to train classifiers explicitly designed for time series. In line with other works that suggest the possible utility of deep learning in fraud detection [41], they use an ensemble of Long Short Term Memory recurrent networks [42], which work on data rebalanced with a new hybrid resampling method called SMOTE-ENN, which combines oversampling and undersampling algorithms to best resample the data.

Closer to our work, the authors of [43] model both genuine and fraudulent transactions as time-dependent and use this to craft new features. Interestingly, explicitly considering

frauds as time series is much less common. The most related work [44] starts from the assumption that frauds are a time series and designs a variant of a GAN to exploit this fact in generating synthetic frauds, that are then used to perform oversampling. The resulting synthetic frauds are then analyzed to verify whether they are similar to the original frauds in the dataset. Their influence on a classifier is then compared to that of a VAE and a generic GAN using accuracy as the primary metric. The main limitations of the work lie in the lack of comparison with other oversampling algorithms, the lack of metrics more suited for imbalanced classification tasks, and the absence of an explicit model of fraudsters' behavior.

Time series can also be employed for oversampling, where synthetic data are generated in series instead of single data points. This is particularly relevant for our work, as considering the frauds as a time series allows us to compare such methods to classical approaches. Defining a good generative model for time series data is not trivial, as the generated data should resemble the original points in terms of *point-to-point* similarity (i.e., the statistical similarity between all points in a time series) and conditional dependence between sequential values. For our work, we opted to use TimeGAN [15], a GAN specifically designed to maximize both criteria jointly. This method aims to generate artificial time series data that are indistinguishable from real fraud sequences, thereby helping the model learn more complex patterns in fraud data and, ultimately, improving the fraud detection system. For a complete description of TimeGAN, we refer to Section IV.

Finally, it should be noted that binary classification with imbalanced data sets can also be addressed by anomaly detection techniques, where minority class instances may be detected as anomalous with respect to the ''normal'' genuine ones. The theoretical advantage of such an approach is that it requires virtually no sample of the minority class, as both unsupervised and semi-supervised techniques can be employed. In credit card fraud detection, anomaly detection can detect fraudulent credit card usage, working both ''by-user'' and ''by-operation,'' identifying anomalous transactions or user's behaviors [45]. However, as discussed in our previous work [24], the main limitation of unsupervised approaches relates to the high number of False Positives due to the difficulty of covering all possible scenarios of legitimate transaction activities [46]. As a result, fraud detection is mainly performed through supervised methods [6]. Here, we focus on addressing the problem of imbalance learning for classification, since this can be beneficial both in the fully supervised approach and/or when anomaly-detection and supervised techniques are combined [24].

III. PROBLEM FORMULATION AND NOTATION

We assume to have access to a training set D_{train} and a test set D_{test} of transactions related to a set $C_1, C_2 \dots C_c$ of c cards. Each transaction is denoted by $z_{i,j}$[1] where j stands for the transaction number and i for the associated card. A transaction $z_{i,j}$ is described by a feature vector $x_{i,j}$ of size N and a label $y_{i,j}$ indicating whether the transaction is fraudulent ($y_{i,j} = 1$) or genuine ($y_{i,j} = 0$). For the sake of conciseness, we will also use the notation $z^+_{i,j}$ to indicate that the transaction $z_{i,j}$ is a fraud ($y_{i,j} = 1$) and $z^-_{i,j}$ if $z_{i,j}$ is genuine ($y_{i,j} = 0$). The transaction features belong to three main categories: *i) the card's features*, e.g. the transaction history and the cardholder information, *ii) the terminal's features* e.g., its location, activity time, and transactions history, and *iii) the transaction features*, e.g., the amount and time of the transaction. We will refer to generic terminal features with TRM and to transaction features with TRX. The fraud detection problem is formulated as a binary classification problem [7] where a classifier associates to each feature vector $x_{i,j}$ an estimated probability $\hat{p}_{i,j}$ of fraud. The assessment of the detection procedure consists of training the classifier on D_{train} and testing its accuracy on D_{test} according to a set of conventional metrics described in the experimental section.

IV. CONTRIBUTIONS

The main assumption of this paper is that fraudsters exhibit a behavioral pattern that can be partially inferred from the transactions they perform. This pattern can be conceptually decomposed into two components:

- fraudsters adapt to the characteristics of cards they have access to;
- they perform various transactions according to a certain plan, meaning we can see each chain of frauds as a series of highly correlated and time-dependent elements.

Our work aims to prove these assumptions, design a quantitative model of fraudsters' behavior from real data, and show that this model is predictive of fraudsters' actions. We use such model in a new framework for oversampling in fraud detection, named ''Adversary Based Oversampling'' (ADV-O).

To model the fraudsters' behavior, two scenarios may be considered:

- *genuine-to-fraud* scenario: the hypothesis is that if a fraudster wants to cloak their frauds as genuine transactions, he will imitate the cardholder's behavior the first time he uses the compromised card.
- *fraud-to-fraud* scenario: the hypothesis is that the fraudster uses a specific logic to generate a series of fraudulent transactions. Hence, we can model his behavior as a stochastic process like a Markov Chain, where each fraud is a state and the next state can be seen as a stochastic function of the previous one.

We mainly focus on the fraud-to-fraud scenario since the experiments reported in Section V-E1 suggest that the genuine-to-fraud scenario is too hard to tackle from a data-driven perspective. In what follows, we consider two implementations of the ADV-O strategy.

[1] Bold letters, such as $\mathbf{z}$, indicate vectors.

A. MIMO ADV-O

The first ADV-O algorithm, denoted *MIMO ADV-O* (pseudo-code in Algorithm 1), models the fraudster behavior with a Multi-Input Multi-Output (MIMO) regression model. Specifically, MIMO ADV-O applies machine learning regression to learn the set of multivariate dependencies g existing between the N features of $z^+_{i,j}$ and the ones of $z^+_{i,j+1}$. We decompose the learning problem into a set of N Multi-Input Single-Output (MISO) problems. This means that for each fraud $z^+_{i,j+1}$ we create N regression tasks where the output is the n^{th} transaction feature $x^n_{i,j+1}$, $n = 1 \dots N$ and the input is the feature set $[x^1_{i,j}, x^2_{i,j}, \dots, x^N_{i,j}]$ of $z^+_{i,j}$. For each of the N MISO tasks, we train a regressor $\mathcal{R}_n$, $n = 1, \dots, N$, which takes as input all the features of the previous fraud and outputs only the nth feature of the following one. The composition $\mathcal{E}$ of the N single-output regressors $\mathcal{E} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_N\}$ estimates the MIMO mapping $g : \mathbb{R}^N \rightarrow \mathbb{R}^N$ between the features of two consecutive frauds whose aim is to model the fraudsters' behavior in the fraud-to-fraud scenario. Since regression returns numerical variables, categorical features are converted into numerical ones before training (Section V-A). In summary, for each fraud $z^+_{i,j} \in D_{train}$, MIMO ADV-O adds a new artificial fraud by applying all the regressors in $\mathcal{E}$, and adds the predicted fraud to the training set.

Algorithm 1 MIMO ADV-O Algorithm

Require: D_{train} ▷ Training dataset
Require: r ▷ Desired oversampling ratio
Ensure: D_{train} augmented with artificial frauds to achieve ratio r

```
Initialize E = {R_1, R_2, ..., R_N}  ▷ Initialize the set of regressors
for n = 1, ..., N do  ▷ For each transaction feature
    Train R_n using D_train and the n-th feature as the target
end for
while the ratio of frauds in D_train is less than r do  ▷ While desired ratio is not reached
    for each fraud z^+_{i,j} ∈ D_train do  ▷ For each fraud in the training set
        for n = 1, ..., N do  ▷ For each transaction feature
            Predict the n-th feature of the following fraud using R_n
            Add the predicted feature to z^+_art
        end for
        Add z^+_art to D_train
    end for
end while
```

B. TimeGAN ADV-O

MIMO ADV-O is built on an implicit Markov Chain representation of fraudsters' behavior, which considers each fraudster's action as based solely on the previous fraud, ignoring longer time patterns.

In the second ADV-O approach, we consider each fraud chain as a time series. This has two advantages: *i)* it takes into consideration possible dependencies among non-consecutive frauds *ii)* it allows us to re-use oversampling algorithms from the time series literature. The second ADV-O algorithm is denoted *TimeGAN ADV-O* (pseudo-code in Algorithm 2), and employs TimeGAN [15], a commonly used algorithm for time series oversampling, and applies it to the fraud chains.

Algorithm 2 TimeGAN ADV-O Algorithm

Require: D_{train} ▷ Training dataset
Require: r ▷ Desired oversampling ratio
Ensure: D_{train} augmented with artificial fraud chains to achieve ratio r

```
Initialize TimeGAN with Embedder E, Recovery function R, Generator G, Discriminator D, and Supervisor S
Train the TimeGAN model on D_train
while the ratio of fraud chains in D_train is less than r do  ▷ While desired ratio is not reached
    for each fraud chain X_i = x_{i,1}, x_{i,2}, ..., x_{i,T} in D_train do
        Generate a random noise series ε = ε_1, ε_2, ..., ε_T
        Create synthetic time-series in the latent space: Z_i = G(ε)
        Transform synthetic latent series into the original feature space: z^+_art
        Add z^+_art to D_train
    end for
end while
```

First, we express the fraudulent transactions performed on the i_{th} card with $X_i = x_{i,1}, x_{i,2}, \dots, x_{i,T}$, i.e. a time series of length T. We then employ TimeGAN to consider longer dependencies among the frauds. The TimeGAN framework consists of four main components: an autoencoder composed of an Embedder E and a Recovery function R, a Generator G, a Discriminator D, and a Supervisor S. The framework aims to learn the underlying feature distributions $P(x_{i,t})$ and temporal dynamics $P(x_{i,t+1}|x_{i,t})$. We train our model in four steps: (a) the Embedder E maps the original data X_i to a latent space to generate $E(X_i)$, (b) the Recovery function R tries to reconstruct the original data from the embedded data, producing $R(E(X_i))$, which should ideally be as close as possible to X_i, (c) the Generator G creates synthetic data $Z_i = z_{i,1}, z_{i,2}, \dots, z_{i,T}$ in the latent space, (d) the Supervisor S is trained to predict the next timestep in the latent space, i.e., $S(E(X_i)_{1:T-1}) \approx E(X_i)_{2:T}$.

The Discriminator D is trained to distinguish between the real embedded data $E(X_i)$ and the synthetic data Z_i. To synthesize artificial fraud data, TimeGAN starts with a random noise series $\epsilon = \epsilon_1, \epsilon_2, \dots, \epsilon_T$, generates a synthetic time-series in the latent space using $Z_i = G(\epsilon)$, and then transforms this synthetic latent series into the original

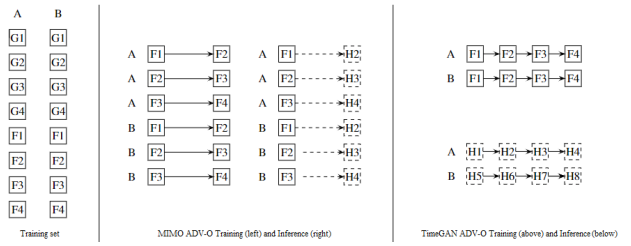

FIGURE 1. Schematic representation of ADV-O and TimeGAN training and inference phases.

feature space using the Recovery function R, i.e., $R(Z_i)$. Interestingly, TimeGAN's loss is the composition of two losses: *i)* a supervised loss, aimed at learning the latent dynamics of both real and synthetic data *ii)* an unsupervised loss, aimed at maximizing the similarity between the original observations and the generated data. Notably, this approach differs from classic generative models, as it learns the distribution of the fraud chains. Classical oversampling algorithms, instead, learn the distribution of single frauds, ignoring the dependency among the frauds within each chain.

C. MIMO VS TimeGAN ADV-O

MIMO ADV-O models the process leading to the generation of a fraud $z^+_{i,j+1}$ given the previous fraud $z^+_{i,j}$ through a discriminative model. The resulting function is then applied to all frauds in the dataset, emulating how a generic fraudster would perform the next fraud given the previous one. TimeGAN ADV-O instead models each chain of frauds as a time series, meaning it does not explicitly estimate the fraudsters' behavior. Moreover, TimeGAN ADV-O is a generative model, meaning that the artificial frauds are not explicitly built on the ones in the original training data. The two approaches are different, yet they present strong similarities, as both approaches base their ability to emulate fraudsters' behavior on modeling the dependency among frauds performed with the same card, even if the approaches are different.

Figure 1 illustrates the MIMO ADV-O and TimeGAN ADV-O training and inference phases with a toy example. The figure delineates the transactional activities of two distinct cards, labeled as 'A' and 'B'. The genuine transactions conducted with these cards are denoted by 'G1', 'G2', 'G3', and 'G4', while fraudulent activities are represented as 'F1', 'F2', 'F3', and 'F4'. The inference phase leads to generating artificial transactions, which are indicated as 'H1', 'H2', and so forth.

V. EXPERIMENTS

This section describes the experimental assessment of the ADV-O strategy and is organized as follows. We describe in Sections V-A, V-B and V-C the real dataset, the metrics, and the statistical tests used for the experiments, respectively. Section V-D introduces the synthetic data generator. Finally, Section V-E shows and discusses the results of the experiments on the industrial dataset, and Section V-F reproduces the results on the synthetic dataset.

A. THE REAL DATASET

Real-world fraud detection systems are typically composed of five modules (detailed in [6]), where transactions are first checked (e.g., PIN) at the terminal level, then filtered by simple blocking rules. If the transaction is not discarded, a series of additional checks are performed using scoring expert-based rules and data-driven models. Finally, transactions raising a fraud alert are controlled by human investigators, who assess whether the transaction is effectively fraudulent. In this perspective, it is important to control the false alarm rate since human investigators are a critical and limited resource.

The industrial partner Worldline S.A provided the transactional data used for the experimental assessment. The dataset refers to a period from 01/05/2018 to 30/09/2018 (DD / MM / YYYY) and includes over 60 Million online, international transactions (each associated with a card and a terminal). Our experiments are performed using a sliding window approach, where the data are divided into windows of two weeks. Each window is split into two 7-day smaller windows: the first is the training set, and the second is the test set. We report the average and standard deviation of the results over the various windows, and we use the results obtained over the various windows to assess the statistical relevance of the performance differences among the various classifiers.

The ratio between frauds and total transactions is significantly less than 1% (though its exact value cannot be disclosed), yielding a strongly unbalanced learning problem. The raw dataset contains 46 features, some categorical and some numerical. Though the exact nature of the original features is not disclosable for confidentiality reasons, it is important to remark that they contain information about the transaction, the card-holder, the terminal and that some of them result from a feature engineering process extracting meaningful aggregates (e.g., average expenditure in the last month). Given the presence of categorical variables in the raw dataset and the adoption of regression techniques in our strategies, special attention has to be devoted to categorical encoding. Several techniques for the encoding of categorical variables exist in literature [47] like *Integer Encoding* and *One Hot Encoding*. In our experiments, we adopt a technique for encoding categorical variables called "target encoding," already used in our previous works on fraud detection [48]. We replace each category with the empirical frequency with which a transaction belonging to such a category is fraudulent. This encoding maps each category to an informative numerical value representing the conditional a-priori probability that a transaction belonging to such a category is fraudulent. For this work, we decided to work on a significantly simplified version of the problem, where we selected only a small subset of features. We decided to focus on the amount (AMOUNT) and two other features,

both categorical, that we deemed the most important for classification. For simplicity and to conceal their true names, we call them X_TERMINAL and Y_TERMINAL, as both features are linked to characteristics of the terminals used in the transactions. The goal is to show the results in a simplified yet realistic context to extend the work to the full dataset in future works. This also allows us to reproduce the results on simulated data without dealing with the complexities of emulating a high-dimensional dataset.

B. METRICS

The assessment of our proposed algorithm can be done at two different levels: the accuracy of the fraudster model (described in Section V-B1) and the quality of the oversampling algorithm (discussed in Section V-B2).

1) ACCURACY METRICS OF THE FRAUDSTER MODEL

The fraudster behaviour is modeled with a set of N independent MISO regression tasks (Section IV) whose accuracy may be measured with conventional regression metrics, like the coefficient of determination. The coefficient of determination

$$R_n^2 = 1 - \frac{\sum_{i,j}\left(x_{i,j}^n - \hat{x}_{i,j}^n\right)^2}{\sum_{i,j}\left(x_{i,j} - \bar{x}^n\right)^2}, \quad n = 1, \ldots, N \tag{1}$$

measures the proportion of the variation of the feature x^n that can be explained by the nth regression model, where $x_{i,j}^n$ is the true value of the nth feature of the transaction $z_{i,j}^+$, $\hat{x}_{i,j}^n$ is the predicted value and $\bar{x}^n$ the average of $x_{i,j}^n$ over all cards and transactions. In order to assess the overall accuracy of the MIMO task, we compute the average $R^2 = \frac{\sum_{n=1}^N R_n^2}{N}$ over all the N features.

Note that a R^2 score significantly larger than 0 suggests good predicting abilities, with 1 being the score of a perfect regressor. Any negative or null value implies no predictive power.

2) OVERSAMPLING QUALITY METRICS

We measure here the quality of oversampling through its impact on the performance of a classifier trained on the real data enhanced with the oversampling. The most common metrics to assess fraud detection are detailed in the chapter "Performance Metrics" of [49]. For binary classifications, they are the following:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{f1} = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{3}$$

where TP, or *True Positive*, indicates the number of correctly classified frauds; FP, or *False Positive*, indicates the number of wrongly classified frauds and FN, or *False Negative*, indicates the wrongly classified genuine transactions. Though widely adopted, those metrics rely on setting a specific threshold for the classification, which can strongly influence the results, especially in unbalanced settings. This is the reason why we prefer a metric like the Area Under the Precision-Recall Curve (*prauc*), which is a well-reputed way to assess classification accuracy in strongly unbalanced settings [48], [50]. Another common metric in the fraud detection literature is the *Precision Top k*

$$\text{P}_k = \frac{TP_k}{k} \tag{4}$$

where TP_k indicates the number of True Positive in the k top returned alerts [6], [48].

Once a fraud is discovered, the corresponding credit card is blocked. Hence, considering multiple transactions from the same card in the metrics measurement can lead to overestimating the accuracy. Therefore, *Grouping by Card*, considering only the highest fraud score between different transactions from the same card, is a good practice.

C. STATISTICAL TESTS

Statistical tests play a crucial role in evaluating and comparing different algorithms, enabling researchers to determine the statistical significance of their results and draw meaningful conclusions. In the context of imbalanced learning, where the performance of various oversampling techniques is assessed, it is essential to employ rigorous statistical tests to ensure that observed differences in performance are not due to chance or random variations in the data. In this study, we employ the Friedman test, followed by the Nemenyi post-hoc test, to conduct pairwise comparisons of the proposed Adversary-based oversampling technique with other state-of-the-art algorithms, as suggested by [51]. The Friedman test is a non-parametric test designed for comparing multiple treatments, while the Nemenyi test facilitates multiple comparison procedures and helps to identify significant differences between specific pairs of algorithms.

The results of these tests are presented using a Critical Distance (CD) plot, which provides a visually intuitive way to represent the relative performance of the algorithms and their statistical significance. The CD plot consists of horizontal lines with markers representing the average ranks of each algorithm, while a vertical line indicates the critical distance. If the markers for two algorithms are farther apart than the critical distance, it can be concluded that their performance differs significantly. This approach enables researchers to quickly and effectively assess the comparative performance of various oversampling techniques and identify the most effective ones in tackling the challenges of imbalanced learning in fraud detection. When presenting the performance of various oversampling algorithms in Tables 3 and 5, we apply report in bold the methods that are not significantly worse than the best one according to the Friedman-Nemenyi test. For the results of each individual test, we refer to the Appendices.

D. A SYNTHETIC GENERATOR FOR REPRODUCIBLE RESULTS

Results on real data are an important part of our analysis, yet they are unfortunately not reproducible because of the evident confidentiality issues in financial-related domains [52]. Though some of the authors of this paper contributed to releasing an encoded fraud detection dataset (the Kaggle credit card dataset in [22]), such a dataset is not suitable here (missing card identifiers) and no other examples of public datasets exist to validate the proposed approach. To increase the reproducibility of the present work, this section presents then a simplistic simulator of transactional data, which may be used to assess our approach with non-confidential data. The generator, in spite of its simplistic nature, has some interesting characteristics:

- it models the terminal and the cardholder with a small number of features: the terminal is described by two geographical coordinates while the cardholder is described by two geographical coordinates, and the spending behaviour

TABLE 1. Accuracy of ADV-O in predicting initial fraud features following genuine transactions on real data.

	MLP		Ridge		RF		Naive	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*	*Mean*	*Variance*
X_TERMINAL	-0.90	0.50	-0.03	0.0011	-0.05	0.0018	-9.71e+07	6.41e+16
Y_TERMINAL	-0.02	0.00074	-0.01	0.00029	0.0084	0.00051	-7.64e+05	3.13e+12
TX_AMOUNT	-0.03	0.0017	-0.0082	0.00032	-0.0034	0.00066	-0.31	0.19

TABLE 2. Accuracy of ADV-O in predicting subsequent fraud features on real data.

	MLP		Ridge		RF		Naive	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*	*Mean*	*Variance*
X_TERMINAL	-4.40	78.59	0.19	0.017	0.21	0.008	-6.55e+08	8.92e+18
Y_TERMINAL	0.035	6.01	0.93	0.001	0.93	0.001	-3.28e+07	4.57e+16
TX_AMOUNT	0.12	0.025	0.13	0.023	0.085	0.076	-0.16	0.048

- it preserves the customer-terminal-transaction nature of the original dataset (like the ULB-MLG generator in [49]) which allows identifying pairs of consecutive frauds,
- it creates a statistical association between consecutive frauds, in concordance with the assumption of this paper that has been corroborated by the real data experiment.

The proposed generator's structure is the following: first, two populations (customer and terminal profiles) are created by sampling their low dimensional distributions. As far as a cardholder is not hacked, he is repeatedly associated with a random terminal based on the geographic location, and her transactions are generated in an i.i.d. manner. Then, as simulation time goes by, a portion of cardholders switches from the genuine to the fraudster category. The behavior of fraudulent cardholders differs in terms of the association strategy to a terminal and the correlation existing between consecutive amounts. Though it is evident that such a simulator simplifies the real process for the sake of reproducibility, it preserves three important properties of the transaction process: the large imbalance ratio, a difference between fraudulent and genuine users in choosing the terminal, and above all, the existence of a temporal association (corresponding to the mapping g estimated in Section V-E1) in fraudulent sequences only. More details on the generator and related synthetic datasets are available at https://github.com/FaramirHurin/ADV-O.git

E. RESULTS ON REAL DATA

On the basis of the metrics discussed before, we first assess the accuracy of the models predicting the fraudsters' behavior (Section V-E1) and then the quality of the related oversampling algorithm (Section V-E2).

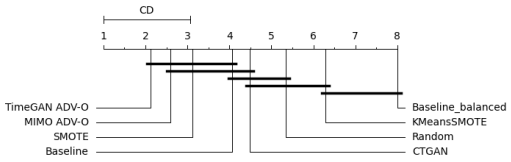

FIGURE 2. The critical difference (CD) plot shows the results of the Friedman-Nemenyi test for model comparison using the metric Pk1000 over real data.

1) ASSESSMENT OF THE FRAUDSTER MODEL: RESULTS AND DISCUSSION

This section considers four different regressors to predict the fraudster behavior in both genuine-to-fraud and fraud-to-fraud scenarios (Section IV): Random Forests, Feed Forward Neural Networks, Ridge Regressors, and a simple persistent model (called "Naive") which returns the latest observed value as a prediction. All algorithms (except Naive) are trained with a standard random grid cross-validation strategy for hyperparameters tuning[2] [53].

The preliminary experiment concerns the genuine-to-fraud scenario in its simplest setting by considering only continuous inputs. Table 1 shows the accuracy of ADV-O in predicting the features of the first fraud perpetrated on a hacked card: the table displays the regression results for the variables X_TERMINAL, Y_TERMINAL, and TX_AMOUNT, as predicted by MLP, RF, and Naive models on real data. Both the mean and standard deviation of the R2 metric for each variable predicted by the models are presented. The results suggest that it is particularly hard to have accurate predictions in this setting, though the Random Forest can predict some features with an R^2 somehow larger than 0 (and than the naive R^2). A possible explanation might be that many characteristics of a card, including its transaction history, may be impossible to retrieve for someone who has stolen or cloned the card. However, it must be noted this does not prove that fraudsters do not adapt their behavior to the card they have access to, but that we do not have any hint in this direction from our data.

We then consider the fraud-to-fraud scenario. We compare the accuracy of different regressors to a third approach, called naive, which simply reproduces the previous fraud. The results, shown in Table 2, show a significant difference

[2]Note that this may penalize algorithms like the Neural Network, which sometimes requires a more ad hoc training procedure than the other learners.

TABLE 3. Comparison of accuracy metrics for different models on real data.

	Baseline		Baseline_bal		CTGAN		KMSMOTE		Random		SMOTE		TimeGAN ADV-O		MIMO ADV-O	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
Precision	**0.18**	0.05	0.01	0.0	0.06	0.01	0.01	0.0	0.01	0.0	0.08	0.02	**0.1**	0.02	**0.09**	0.02
Recall	0.07	0.02	**0.76**	0.03	0.26	0.05	**0.48**	0.04	**0.47**	0.04	0.25	0.05	0.24	0.05	0.25	0.05
F1	0.1	0.03	0.02	0.0	0.1	0.02	0.02	0.0	0.03	0.01	**0.12**	0.02	**0.14**	0.02	**0.13**	0.02
PRAUC	0.05	0.02	**0.11**	0.02	0.05	0.01	0.06	0.01	0.06	0.02	0.06	0.02	**0.07**	0.02	**0.07**	0.02
PRAUC_C	0.08	0.02	**0.21**	0.03	0.08	0.02	0.1	0.02	0.1	0.02	0.09	0.02	**0.11**	0.02	0.1	0.02
Pk50	**0.29**	0.13	0.08	0.04	0.18	0.08	0.15	0.07	0.15	0.08	**0.26**	0.1	**0.25**	0.09	**0.25**	0.12
Pk100	**0.29**	0.13	0.08	0.03	0.18	0.06	0.15	0.06	0.16	0.06	**0.23**	0.09	**0.25**	0.08	**0.25**	0.1
Pk200	**0.26**	0.1	0.08	0.03	0.19	0.06	0.15	0.05	0.17	0.06	**0.24**	0.09	**0.27**	0.1	**0.25**	0.1
Pk500	**0.24**	0.1	0.08	0.02	0.2	0.08	0.15	0.05	0.17	0.07	**0.23**	0.09	**0.27**	0.1	**0.26**	0.11
Pk1000	**0.22**	0.08	0.08	0.02	0.21	0.07	0.15	0.05	0.17	0.06	**0.23**	0.09	**0.25**	0.07	**0.24**	0.08
Pk2000	0.18	0.06	0.08	0.02	0.19	0.06	0.16	0.05	0.2	0.06	0.2	0.07	**0.23**	0.07	**0.22**	0.06

TABLE 4. Accuracy of ADV-O in predicting subsequent fraud features on synthetic data.

	MLP		Ridge		RF		Naive	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*	*Mean*	*Variance*
X_TERMINAL	0.938	0.00023	0.914	0.00023	0.948	0.00020	-150.87	46111.70
Y_TERMINAL	0.930	0.00022	0.899	0.00027	0.950	0.00017	-105.09	22282.78
TX_AMOUNT	0.858	0.00023	0.855	0.00023	0.825	0.00025	-0.35	0.71

between the features of the terminal and the amount. The terminal's features present certain regularities, which seem not to be present in the amount. Overall, the results support the assumption that learning a reliable model of the fraudster's behavior from historical data in a fraud-to-fraud scenario is possible and recommended. At the same time, there is still space for improvement. Given the performances shown in Table 2, we adopt Random Forest in the following.

2) ASSESSMENT OF THE OVERSAMPLING APPROACH: RESULTS AND DISCUSSION

This section benchmarks MIMO ADV-O and TimeGAN ADV-O against two Baselines (no oversampling and undersampling through the use of a Balanced Random Forest) and four state-of-the-art oversampling strategies: random oversampling (RANDOM), SMOTE, K-Means SMOTE, and CT-GAN. We implement MIMO ADV-O using the regressor that, on average, performs better in Section V-E1, i.e., the Random Forest Regressor. We then apply the same oversampling factor (10%) to all oversampling algorithms.

Concerning SMOTE, we use the implementation provided by the `Imbalanced-learn` [54] library, using the default values for all the parameters and setting the oversampling ratio to the one used in ADV-O. We adopt the default structure for CT-GAN: the generator is a two-layer fully-connected neural network with Batch-normalization and Relu activation functions, followed by a mix of activation functions to generate the synthetic row representation; the discriminator is also a two-layer fully-connected neural network with a LeakyReLU activation function and dropout for each hidden layer. For K-Means Smote, we tune the required proportion of minority class samples to filter a cluster: we set the `cluster balance threshold` to 0.1.

We assess here the impact of the oversampling technique on the final detection accuracy and requires the choice of a specific binary classifier.

We adopt a Random Forest classifier for the following reasons: (i) Random Forests have been shown to outperform other models in our previous studies [7], [23], (ii) they are beneficial to establish feature importance, which is highly valued by investigators to interpret the fraud detection process, and (iii) they provide a natural ensemble that can be easily exploited to address the imbalance, e.g., by feeding each decision tree with a balanced subset of the original data.

The assessment uses the Precision top K and the Area under the precision-recall curve computed for the test set. Overall, Table 3 shows that both ADV-O algorithms outperform all other oversampling algorithms, with timeGAN ADV-O being slightly superior in most metrics. The comparison with the baselines is interesting. First, we can see how the balanced random forest beats all the competitors regarding PRAUC and PRAUC_C. Still, it pays a heavy price regarding precision top K. Conversely, the baseline aligns with most other approaches in all metrics and is the best approach for low values of K. However, for large values of K, MIMO ADV-O and TimeGAN ADV-O are the best methods. Among traditional oversampling algorithms, SMOTE best holds the comparison with the ADV-O algorithms, especially concerning the precision top K. This is partially in line with previous results in fraud detection literature, which highlight how SMOTE has proven to generally be the most competitive oversampling algorithm [36]. Moreover, we performed our experiments on few dimensions. Since SMOTE is known to have issues with high dimensional data [55], this may help explain its good performance. Overall, the choice of approach heavily depends on the metric one wants to maximize. Still, the results show that oversampling can be beneficial when maximizing the precision top K, with time-based approaches being particularly helpful. Concerning the comparison between MIMO ADV-O and TimeGAN ADV-O, both algorithms achieved a remarkably close performance. We expected these results, as both algorithms rely on the same principle of modeling the time-dependent nature of the frauds. Finally, TimeGAN ADV-O performs slightly better in most metrics, although the difference is not statistically significant. More tests on different datasets may help us understand whether a difference exists and if modeling data as time series gives us a significant advantage compared to only considering the direct dependencies between consecutive frauds.

TABLE 5. Comparison of accuracy metrics for different models on synthetic data.

	Baseline		Baseline_bal		CTGAN		KMSMOTE		Random		SMOTE		TimeGAN ADV-O		MIMO ADV-O	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
Precision	**0.26**	0.09	0.05	0.03	0.13	0.08	0.11	0.07	0.09	0.07	0.13	0.08	0.14	0.09	0.14	0.09
Recall	0.18	0.11	**0.9**	0.02	0.51	0.15	**0.59**	0.13	**0.78**	0.05	**0.58**	0.12	0.51	0.15	0.51	0.15
F1	0.2	0.1	0.1	0.05	**0.2**	0.1	0.18	0.1	0.15	0.1	0.21	0.11	**0.21**	0.11	**0.21**	0.11
PRAUC	0.17	0.11	**0.3**	0.11	0.22	0.13	0.2	0.12	**0.23**	0.13	**0.24**	0.13	0.22	0.13	0.22	0.12
PRAUC_C	0.32	0.15	**0.48**	0.08	0.38	0.15	0.39	0.11	**0.45**	0.1	0.41	0.12	0.39	0.15	0.38	0.14
Pk50	**0.53**	0.14	0.08	0.07	**0.43**	0.14	0.18	0.15	0.15	0.13	0.28	0.17	**0.46**	0.13	**0.46**	0.14
Pk100	**0.52**	0.14	0.09	0.08	**0.43**	0.1	0.18	0.12	0.14	0.11	0.28	0.13	**0.46**	0.11	**0.46**	0.12
Pk200	**0.5**	0.14	0.11	0.07	**0.44**	0.1	0.21	0.11	0.14	0.09	0.28	0.12	**0.46**	0.11	**0.45**	0.11
Pk500	0.4	0.16	0.12	0.08	**0.42**	0.11	0.22	0.1	0.13	0.08	0.28	0.11	**0.42**	0.11	**0.42**	0.11
Pk1000	0.31	0.14	0.12	0.07	**0.36**	0.12	0.21	0.09	0.13	0.09	0.27	0.11	**0.36**	0.12	**0.36**	0.11
Pk2000	0.23	0.12	0.13	0.08	**0.28**	0.11	0.2	0.1	0.13	0.09	0.26	0.11	**0.28**	0.11	**0.28**	0.11

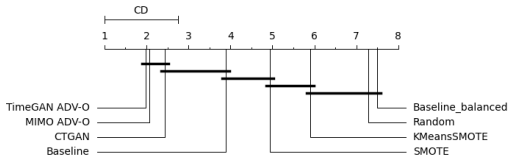

FIGURE 3. The critical difference (CD) plot shows the results of the Friedman-Nemenyi test for model comparison using the metric Pk1000 over synthetic data.

The result of the Friedman-Nemenyi test using the PK1000 over real data can be found in Figure 2. The plot displays the average rank of each model on the x-axis (the lower, the better), with models grouped into sets that are not significantly different from each other (marked by horizontal bars). CD indicates the minimum detectable difference (CD) at a 95% confidence level.

F. RESULTS ON SYNTHETIC DATA

The availability of a simulator allows for the extension of the experimental session to a new synthetic dataset. First, we show that in a simple environment, where the relationship between two consecutive frauds is relatively simple, MIMO ADV-O can predict the features of the following fraud with a significantly high R^2, as shown in Table 4.

Concerning the classic metrics for classification, we repeat here the same analysis performed on real data. We show the results in Table 5. Similarly to what happens in real data, undersampling is the best approach in terms of PRAUC, but it severely damages the precision top K of the algorithm. Again, for low values of k the baseline is the best method, while with larger values, the ADV-O algorithms and CT-GAN are the best approaches. Finally, it should be noted that time-dependent methods perform on pair or better than all other oversampling methods, confirming the results obtained on the real data. The results of the Friedman-Nemenyi test using the metric Pk1000 on synthetic data be found in Figure 3.

VI. CONCLUSION AND FUTURE WORK

Imbalanced learning can severely limit the possibility of learning from the data if not adequately addressed. In this work, we propose a new approach to address this problem that considers the existence of behavioral patterns from fraudsters. We compose these patterns into two components: the adaptation to the card they steal and the correlation and dependency among the frauds they perform with each card. Our experiment shows that the latter is the most promising approach. To improve the quality of oversampling algorithms for fraud detection, we propose a new framework for oversampling in fraud detection that exploits the time-dependent nature of the fraud generation process through two different algorithms.

First, we introduce MIMO ADV-O. This novel oversampling algorithm models the frauds generation process as the dependency of each fraud from the previous one performed with the same card. It uses it to craft plausible frauds for a compromised card. Then, the dependence of each fraud on the previous ones allows us to model each chain of frauds and use oversampling algorithms explicitly designed for time series generation. In particular, we propose TimeGAN ADV-O, an adaptation of the time series oversampling algorithm TimeGAN to the context of fraud detection. Our experiments show that the accuracy of a classifier trained on datasets oversampled with both ADV-O algorithms is comparable to or better than the best competitors.

Possible future extensions should concern the impact of feedback delay and concept drift, which have already been investigated in our previous works [6], [8], [23]. Other interesting research lines could concern the impact of feature-engineering and feature interpretability (e.g., by means of Shapley values [56]) on the accuracy of the fraudster model. Moreover, the work done in [40] shows that applying time-series classification techniques to fraud detection data can improve the classification performance of fraud detection engines, even when combined with static oversampling algorithms. Since we generate synthetic time series, feeding the resulting frauds to time series classifiers seems a natural extension of our work. Finally, considering frauds as time series opens the door to totally different approaches to fraud detection, where the compromission of a card can be seen as a *change* instead of an *anomaly*, and time-series classifiers can be used in addition or in assistance to existing methods.

APPENDIX A
STATISTICAL TESTS ON REAL DATA

This appendix section presents the results of the statistical tests performed on real data. The critical difference (CD) plots show the results of the Friedman-Nemenyi test for model comparison using the various metrics over synthetic data. The choice of metrics comes from the fraud detection literature. The plots display the average rank of each model on the x-axis (the lower, the better), with models grouped into sets that are not significantly different from each other (marked by horizontal bars). CD indicates the minimum detectable difference (CD) at a 95% confidence level.

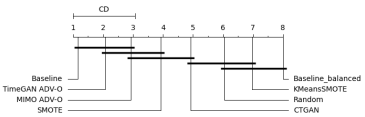

FIGURE 4. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Precision metric.

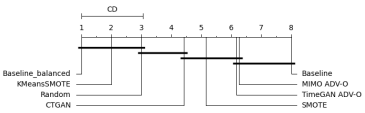

FIGURE 5. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Recall metric.

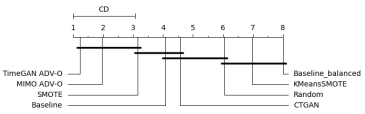

FIGURE 6. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the F1 metric.

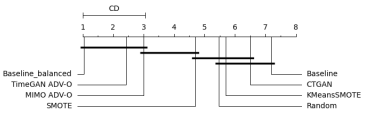

FIGURE 7. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the PRAUC metric.

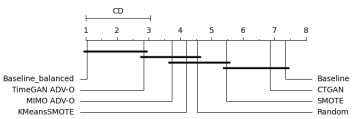

FIGURE 8. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the PRAUC_C metric.

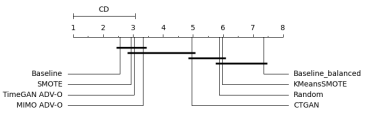

FIGURE 9. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk50 metric.

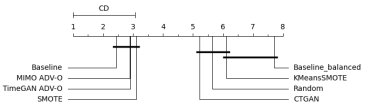

FIGURE 10. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk100 metric.

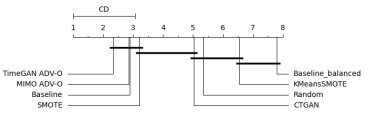

FIGURE 11. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk200 metric.

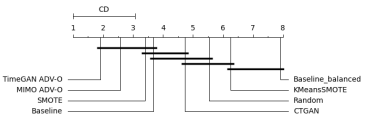

FIGURE 12. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk500 metric.

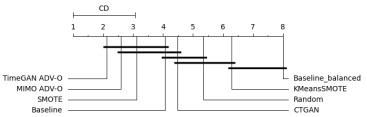

FIGURE 13. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk1000 metric.

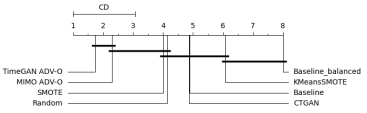

FIGURE 14. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling, using the Pk2000 metric.

APPENDIX B
STATISTICAL TESTS ON SYNTHETIC DATA

This appendix section presents the results of the statistical tests performed on synthetic data. The critical difference (CD) plots show the results of the Friedman-Nemenyi test for model comparison using the various metrics over synthetic data. The choice of metrics comes from the fraud detection

literature. The plots display the average rank of each model on the x-axis (the lower, the better), with models grouped into sets that are not significantly different from each other (marked by horizontal bars). CD indicates the minimum detectable difference (CD) at a 95% confidence level.

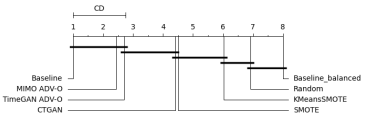

FIGURE 15. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Precision metric.

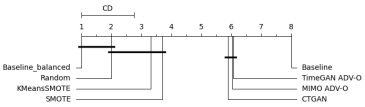

FIGURE 16. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Recall metric.

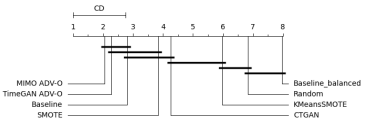

FIGURE 17. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the F1 metric.

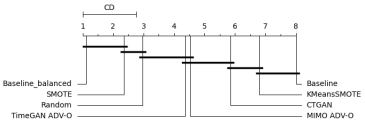

FIGURE 18. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the PRAUC metric.

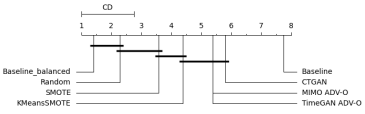

FIGURE 19. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the PRAUC_C metric.

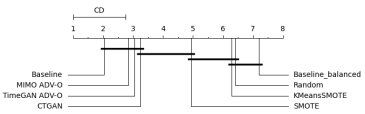

FIGURE 20. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk50 metric.

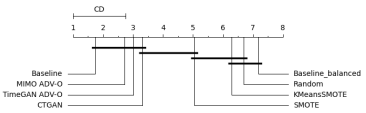

FIGURE 21. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk100 metric.

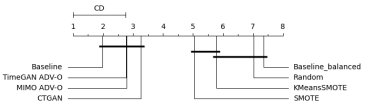

FIGURE 22. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk200 metric.

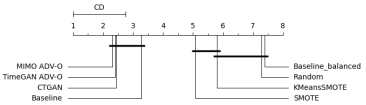

FIGURE 23. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk500 metric.

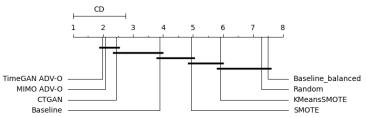

FIGURE 24. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk1000 metric.

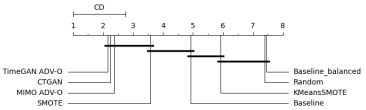

FIGURE 25. Critical difference plot showing the results of Friedman/Nemenyi tests on the results of our classification after oversampling on synthetic data, using the Pk2000 metric.

REFERENCES

[1] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, Oct. 2002.

[2] D.-C. Li, C.-W. Liu, and S. C. Hu, "A learning method for the class imbalance problem with medical data sets," *Comput. Biol. Med.*, vol. 40, no. 5, pp. 509–518, May 2010.

[3] H. Yu, J. Ni, Y. Dan, and S. Xu, "Mining and integrating reliable decision rules for imbalanced cancer gene expression data sets," *Tsinghua Sci. Technol.*, vol. 17, no. 6, pp. 666–673, Dec. 2012.

[4] S. Maldonado and J. López, "Imbalanced data classification using second-order cone programming support vector machines," *Pattern Recognit.*, vol. 47, no. 5, pp. 2070–2079, May 2014.

[5] R. Shatnawi, "Improving software fault-prediction for imbalanced data," in *Proc. Int. Conf. Innov. Inf. Technol. (IIT)*, Mar. 2012, pp. 54–59.

[6] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3784–3797, Aug. 2018.

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

[8] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection and concept-drift adaptation with delayed supervised information," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[9] M. Carminati, R. Caron, F. Maggi, I. Epifani, and S. Zanero, "BankSealer: A decision support system for online banking fraud analysis and investigation," *Comput. Secur.*, vol. 53, pp. 175–186, Sep. 2015.

[10] S. A. Ebiaredoh-Mienye, E. Esenogho, and T. G. Swart, "Artificial neural network technique for improving prediction of credit card default: A stacked sparse autoencoder approach," *Int. J. Electr. Comput. Eng.*, vol. 11, no. 5, p. 4392, Oct. 2021.

[11] A. Dal Pozzolo, O. Caelen, and G. Bontempi, "When is undersampling effective in unbalanced classification tasks?" in *Proc. ECML PKDD, Mach. Learn. Knowl. Discovery Databases*, 2015, pp. 200–215.

[12] P. Juszczak, N. M. Adams, D. J. Hand, C. Whitrow, and D. J. Weston, "Off-the-peg and bespoke classifiers for fraud detection," *Comput. Statist. Data Anal.*, vol. 52, no. 9, pp. 4521–4532, May 2008.

[13] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[14] C. Drummond and R. C. Holte, "C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling," in *Proc. ICML Workshop Learn. Imbalanced Datasets*, vol. 11, 2003, pp. 1–8.

[15] J. Yoon, D. Jarrett, and M. van der Schaar, "Time-series generative adversarial networks," in *Advances in Neural Information Processing Systems*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf

[16] G. M. Weiss and H. Hirsh, "Learning to predict extremely rare events," in *Proc. AAAI Workshop Learn. From Imbalanced Data Sets*, 2000, pp. 64–68.

[17] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *J. Big Data*, vol. 8, no. 1, pp. 1–41, Dec. 2021.

[18] J. Burez and D. Van den Poel, "Handling class imbalance in customer churn prediction," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 4626–4636, Apr. 2009.

[19] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Prog. Artif. Intell.*, vol. 5, no. 4, pp. 221–232, Nov. 2016.

[20] N. Thai-Nghe, Z. Gantner, and L. Schmidt-Thieme, "Cost-sensitive learning methods for imbalanced data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2010, pp. 1–8.

[21] T.-Y. Liu, "EasyEnsemble and feature selection for imbalance data sets," in *Proc. Int. Joint Conf. Bioinf., Syst. Biol. Intell. Comput.*, 2009, pp. 517–520.

[22] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *Proc. IEEE Symp. Ser. Comput. Intell.*, Dec. 2015, pp. 159–166.

[23] F. Carcillo, A. Dal Pozzolo, Y.-A. Le Borgne, O. Caelen, Y. Mazzer, and G. Bontempi, "SCARFF: A scalable framework for streaming credit card fraud detection with spark," *Inf. Fusion*, vol. 41, pp. 182–194, May 2018.

[24] F. Carcillo, Y.-A. Le 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.

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[26] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw., IEEE World Congr. Comput. Intell.*, Jun. 2008, pp. 1322–1328.

[27] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning," in *Advances in Intelligent Computing*, 2005.

[28] F. Last, G. Douzas, and F. Bação, "Oversampling for imbalanced learning based on k-means and SMOTE," 2017, *arXiv:1711.00837*.

[29] T. Jebara, *Machine Learning: Discriminative and Generative*, vol. 755. Berlin, Germany: Springer, 2012.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 3, Jun. 2014.

[31] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, pp. 307–392, 2019.

[32] 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.

[33] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional GAN," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[34] M. Alamri and M. Ykhlef, "Survey of credit card anomaly and fraud detection using sampling techniques," *Electronics*, vol. 11, no. 23, p. 4003, Dec. 2022.

[35] Z. Salekshahrezaee, J. L. Leevy, and T. M. Khoshgoftaar, "The effect of feature extraction and data sampling on credit card fraud detection," *J. Big Data*, vol. 10, no. 1, p. 6, Jan. 2023.

[36] B. Jonathan, P. H. Putra, and Y. Ruldeviyani, "Observation imbalanced data text to predict users selling products on female daily with SMOTE, tomek, and SMOTE-tomek," in *Proc. IEEE Int. Conf. Ind., Artif. Intell., Commun. Technol. (IAICT)*, Jul. 2020, pp. 81–85.

[37] R. Devaki, V. Kathiresan, and S. Gunasekaran, "Credit card fraud detection using time series analysis," *Int. J. Comput. Appl.*, vol. 3, pp. 8–10, Feb. 2014.

[38] G. Moschini, R. Houssou, J. Bovay, and S. Robert-Nicoud, "Anomaly and fraud detection in credit card transactions using the arima model," *Eng. Proc.*, vol. 5, no. 1, p. 56, 2021.

[39] P. Rousseeuw, D. Perrotta, M. Riani, and M. Hubert, "Robust monitoring of time series with application to fraud detection," *Econometrics Statist.*, vol. 9, pp. 108–121, Jan. 2019.

[40] E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba, and G. Obaido, "A neural network ensemble with feature engineering for improved credit card fraud detection," *IEEE Access*, vol. 10, pp. 16400–16407, 2022.

[41] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms," *IEEE Access*, vol. 10, pp. 39700–39715, 2022.

[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[43] Y. Lucas, P.-E. Portier, L. Laporte, L. He-Guelton, O. Caelen, M. Granitzer, and S. Calabretto, "Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs," *Future Gener. Comput. Syst.*, vol. 102, pp. 393–402, Jan. 2020.

[44] Z. Zhang, L. Yang, L. Chen, Q. Liu, Y. Meng, P. Wang, and M. Li, "A generative adversarial network–based method for generating negative financial samples," *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 2, 2020.

[45] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009.

[46] M. Krivko, "A hybrid model for plastic card fraud detection systems," *Expert Syst. Appl.*, vol. 37, no. 8, pp. 6070–6076, Aug. 2010.

[47] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *Int. J. Comput. Appl.*, vol. 175, no. 4, pp. 7–9, Oct. 2017.

[48] A. Dal Pozzolo, "Adaptive machine learning for credit card fraud detection," Tech. Rep., 2015.

[49] Y.-A. Le Borgne, W. Siblini, B. Lebichot, and G. Bontempi, *Reproducible Machine Learning for Credit Card Fraud Detection—Practical Handbook*. Belgium: Université Libre de Bruxelles, 2022. [Online]. Available: https://github.com/Fraud-Detection-Handbook/fraud-detection-handbook

[50] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[51] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

[52] S. A. Assefa, D. Dervovic, M. Mahfouz, R. E. Tillman, P. Reddy, and M. Veloso, "Generating synthetic data in finance: Opportunities, challenges and pitfalls," in *Proc. 1st ACM Int. Conf. AI Finance (ICAIF)*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1–8, doi: 10.1145/3383455.3422554.

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[54] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-365
[55] S. Maldonado, C. Vairetti, A. Fernandez, and F. Herrera, "FW-SMOTE: A feature-weighted oversampling approach for imbalanced classification," *Pattern Recognit.*, vol. 124, Apr. 2022, Art. no. 108511.
[56] M. Sundararajan and A. Najmi, "The many Shapley values for model explanation," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9269–9278.

DANIELE LUNGHI received the master's degree in computer science and engineering from Politecnico di Milano. He is currently pursuing the joint Ph.D. degree with Université Libre de Bruxelles (ULB) and the Athena Research Center (ARC), under the auspices of DEDS and the Horizon 2020 Marie Skłodowska-Curie Innovative Training Networks. He actively contributes as a member of the Machine Learning Group, ULB. His current research interests include adversarial machine learning, fraud detection, and imbalanced learning.

GIAN MARCO PALDINO received the joint M.Sc. degree in computer science and engineering from Université Libre de Bruxelles and Politecnico di Milano. He is currently pursuing the Ph.D. degree. His current research interests include credit card fraud detection, automated machine learning (AutoML), time series forecasting, digital twins, and causality.

OLIVIER CAELEN received the two master's degrees in statistics and computer science and the Ph.D. degree in machine learning. He is currently a Machine Learning Researcher with Worldline S.A., a Paytech Pioneer for seamless payment solutions. He also teaches an introductory ML course and an advanced DL course at the university. He is the coauthor of 45 publications in international peer-reviewed scientific journals/conferences, book author, and co-inventor of six patents.

GIANLUCA BONTEMPI (Senior Member, IEEE) is currently a Full Professor with the Computer Science Department, Université Libre de Bruxelles (ULB), Brussels, Belgium, and the Co-Head of the ULB Machine Learning Group. He has been the Director of the ULB/VUB Interuniversity Institute of Bioinformatics (IB^2), Brussels, from 2013 to 2017. He was a Marie Curie Fellow Researcher. He is the author of more than 250 scientific publications. He is the coauthor of several open-source software packages for bioinformatics, data mining, and prediction. His current research interests include big data mining, machine learning, bioinformatics, causal inference, predictive modeling, and their application to complex tasks in engineering (time series forecasting and fraud detection) and life science (network inference and gene signature extraction). He was awarded in two international data analysis competitions and took part in many research projects in collaboration with universities and private companies all over Europe.