Digital Communications and Networks 10 (2024) 158–167

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

NFA: A neural factorization autoencoder based online telephony fraud detection

Abdul Wahid [a,*], Mounira Msahli [a], Albert Bifet [a,b], Gerard Memmi [a]

[a] Department of Computer Sciences and Networks (INFRES), LTCI, Telecom Paris, Institut Polytechnique de Paris, Palaiseau, 91120, France
[b] Artificial Intelligence Institute, The University of Waikato, Private Bag 3105, 3240, Hamilton, New Zealand

ARTICLE INFO

KEYWORDS:
Telecom industry
Streaming anomaly detection
Fraud analysis
Factorization machine
Real-time system
Security

ABSTRACT

The proliferation of internet communication channels has increased telecom fraud, causing billions of euros in losses for customers and the industry each year. Fraudsters constantly find new ways to engage in illegal activity on the network. To reduce these losses, a new fraud detection approach is required. Telecom fraud detection involves identifying a small number of fraudulent calls from a vast amount of call traffic. Developing an effective strategy to combat fraud has become challenging. Although much effort has been made to detect fraud, most existing methods are designed for batch processing, not real-time detection. To solve this problem, we propose an online fraud detection model using a Neural Factorization Autoencoder (NFA), which analyzes customer calling patterns to detect fraudulent calls. The model employs Neural Factorization Machines (NFM) and an Autoencoder (AE) to model calling patterns and a memory module to adapt to changing customer behaviour. We evaluate our approach on a large dataset of real-world call detail records and compare it with several state-of-the-art methods. Our results show that our approach outperforms the baselines, with an AUC of 91.06%, a TPR of 91.89%, an FPR of 14.76%, and an F1-score of 95.45%. These results demonstrate the effectiveness of our approach in detecting fraud in real-time and suggest that it can be a valuable tool for preventing fraud in telecommunications networks.

1. Introduction

Telecom fraud poses a major challenge for Mobile Network Operators (MNOs) and their customers globally, as the phone has become a vital mode of communication. Fraud is reported as the main cause of revenue loss in the telecom industry [1]. A recent study by the Communications Fraud Control Association (CFCA-2021) [2] found that fraud in the telecom sector costs billions of dollars annually, with a 28% increase, or approximately USD 11.6 billion, compared to 2019, as shown in Fig. 1. To prevent such substantial losses, a real-time fraud detection system is necessary. Due to the large volume of normal call records and the ever-evolving cheating tactics of fraudsters, detecting telecom fraud has become a highly challenging task [3].

Telecom networks generate vast amounts of interconnected data streams, requiring a robust offline storage and management system to collect and analyze these records. A large-scale distributed computing system is also necessary for efficient execution. To meet the low latency requirements for online services, efficient data access for online prediction is crucial. Equally important is the development of approaches for feature detection and extraction.

In recent years, there have been numerous efforts to detect various types of fraud in the telecom sector, with most research focused on rule-based, profile-based, and signature-based approaches. A few studies have also proposed anomaly detection-based methods to detect fraud in the telecom industry [4,5]. However, constantly changing fraud patterns over time greatly reduce the effectiveness of expert-summarized rules. Many existing methods (supervised and unsupervised) fail to capture the complex patterns of fraudsters, as they are designed for batch processing and are not suitable for real-time fraud detection.

This paper addresses these challenges and presents a method for online fraud detection based on fraud patterns. Call patterns, which capture typical caller behaviour such as the number of incoming and outgoing calls, call duration, domestic and international calls, etc., are used to represent personalized data that evolves slowly over time. To prevent losses from fraudsters, modelling these fraud patterns is necessary. However, this modelling is challenging due to the difficulty of extracting and evaluating large volumes of call traffic in real-time to detect potentially fraudulent calls.

* Corresponding author.
E-mail addresses: abdul.wahid@universityofgalway.ie (A. Wahid), mounira.msahli@telecom-paris.fr (M. Msahli), albert.bifet@telecom-paris.fr (A. Bifet), gerard.memmi@telecom-paris.fr (G. Memmi).

https://doi.org/10.1016/j.dcan.2023.03.002
Received 1 September 2022; Received in revised form 6 February 2023; Accepted 1 March 2023
Available online 8 March 2023

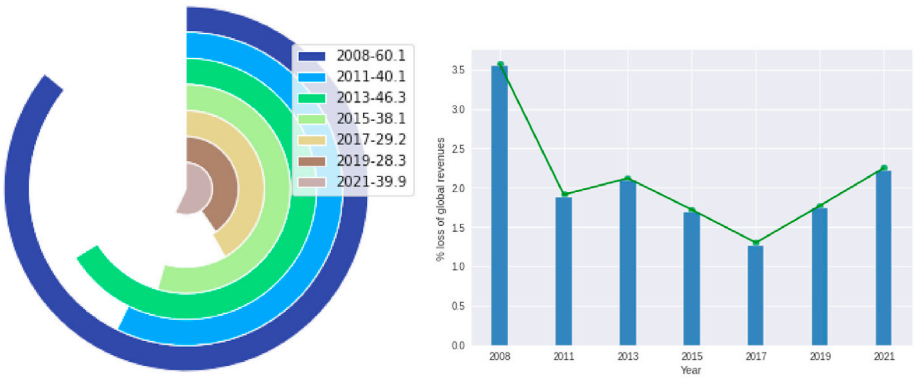

Fig. 1. (a) Global telecom fraud loss in billion USD and (b) % loss of global telecom revenues. Data source: [2].

1.1. Motivations

In this study, real-time telecom fraud detection is motivated by the following observations:

- Inadequacy of Existing Methods: Traditional fraud detection methods may not be able to keep up with the evolving tactics of fraudsters, leading to a higher rate of false negatives and missed fraud incidents.
- Increased Fraud Incidence: With the growth of the telecom industry and the increasing use of mobile and internet services, the incidence of telecom fraud has also increased. There is a need for a more effective and efficient method for real-time fraud detection.
- Cost and Reputation impact: Telecom fraud can result in significant financial losses and harm to a company's reputation. A more effective real-time fraud detection system can reduce these losses and improve the overall security of the telecom network.
- Customer Experience: A timely and accurate response to fraud incidents can reduce the impact on customers and improve their overall experience.

In summary, the motivation for proposing a new approach for real-time telecom fraud detection is due to the inadequacy of existing methods, the increasing incidence of fraud, the need to reduce costs and protect reputation, and the importance of improving the customer experience.

1.2. Contributions

The key contributions of our paper are summarized as follows:

- **Real-time Telephony Fraud Detection**: We design and develop a new fraud detection system based on the customers' calling patterns.
- **Identifying Frauds Pattern**: We leverage the power of the Neural Factorization Machine (NFM) and the Autoencoder (AE) to model customers' calling patterns.
- **Memory Module**: In order to detect fraudulent calls, we develop a memory module that tracks fraud patterns and updates it regularly using a First In First Out (FIFO) memory update policy.
- **Effectiveness**: As part of our validation of our proposed system, we performed extensive experiments on real CDR datasets (of 670,211 call records) provided by our industrial partners[1]. In addition, we demonstrate that the proposed system is more effective than baseline approaches in combating telephony fraud.

[1] https://www.araxxe.com/

1.3. Roadmap

This paper is structured as follows: Section 2 explores existing work on online and offline fraud detection systems. Section 3 defines the problem statement. Section 4 describes the methodology and functional architecture of the system proposed in this paper. Section 5 discusses the dataset in detail and reviews the proposed system's implementation, evaluation, and comparison with the baseline approaches. Finally, Section 6 presents the conclusion and possible directions for future research.

2. Related work

The detection of real-time fraud in the telecom industry is a challenging task. Identifying fraudulent schemes is crucial to minimize revenue losses and enhancing customer satisfaction. In recent years, various solutions have been proposed to address this issue. Early fraud detection systems relied on rule-based methods, using correlation, statistics, or logical comparisons to determine the authenticity of a record. Currently, many commercial fraud detection programs still employ rule-based algorithms to detect fraudulent behaviour on a network. It's worth mentioning that these rules are made up of multiple conditions, typically created by experienced fraud analysts."

In [6], the authors proposed a method for detecting credit card fraud based on association rules. In another study, Rosset et al. [7] presented a two-stage system incorporating an additional rule selection mechanism with the C4.5 rule generator, which can detect telecom fraud. Then, Zhao et al. [8] designed a telecom fraud detection system by understanding the call's contents. They formed rules to detect fraudulent activities by identifying similar contents within the same call. A new study published in Ref. [9], examined pattern-based Wangiri fraud detection. Their approach focused on defining three Wangiri fraud patterns and then identifying fraud using machine learning (ML) and outlier detection techniques. Each method employs a set of fraud detection rules summarized by experts. In real-world applications, fraud techniques constantly evolve, making some patterns obscure. As a result, these methods either fail to detect all instances of fraud or perform poorly. Therefore, searching for fraud patterns in streaming datasets is challenging in order to mobilize expert knowledge effectively.

Recently, there has been a surge in the number of researchers who have presented sequence-based approaches to the problem of fraud detection. When a sequence, sub-sequence, or sequential pattern deviates significantly from normality, it is classified as an abnormal [10]. In Ref. [11], the authors proposed a highly intuitive method for computing pairwise similarities between sequences. The study [12] focused on window-based mining and detecting partial anomalies in entire sequences. In some works, such as [13,14], Recurrent Neural Networks (RNN) was used to detect fraud based on sequences. In Ref. [15],

Bernardo et al. proposed an RNN architecture for real-time fraud detection and a machine learning (ML) pipeline.

These articles focus on sequential information rather than effectively using the intrinsic information associated with each calling behavior. In order to address the aforementioned problem, Zhu et al. [16] suggested using a Hierarchical Explainable Network (HEN) for modelling user behavior sequences, improving fraud detection, and explaining the inference process. In the HEN model, Factorization Machines (FMs) approximate feature interaction by calculating the inner product between embedding vectors. Our proposed work also utilizes a sequence-based fraud detection method to detect fraudulent activities in complex non-fraudulent call patterns, similar to HEN [16]. The proposed method overcomes HEN limitations by employing an improved memory-enhanced FM adaptation.

In recent years, few articles have discussed the latest approaches for evolving data streams to detect outliers and thus be placed in fraud scenarios. In a recent paper [17], Zhang et al. proposed a privacy-preserving based real-time system for identifying unfamiliar callers and classifying them based on four distinct categories: taxi drivers, delivery and takeout staff, telemarketers and fraudsters, and normal users. Another researcher, Togbe et al. [18], found that combining a sliding window with a randomly constructed half-space tree can detect anomalies in streaming data. In addition, several anomaly-detection-based methods have also been proposed, such as ECOD [19], COPOD [20], SUOD [21], DeepSVDD [22], LODA [23], iForest [24], and Memstream [25]. Despite being related to our work, these methods did not emphasize the specific sequential patterns associated with telecom fraud.

3. Problem statement

The problem of real-time fraud detection in telecommunications involves detecting fraudulent activities in near real-time as they occur in telecommunication networks. This is a crucial challenge as fraudsters continually find new ways to exploit weaknesses in telecommunication systems, leading to financial losses for both service providers and their customers. Real-time fraud detection aims to identify and prevent fraudulent activities as soon as they happen before they cause significant damage. This requires the development of advanced machine learning and statistical models that can effectively analyze large amounts of data from various sources, such as call detail records, network logs, and subscriber information.

A real-time fraud detection system must handle high volumes of data and perform complex analysis in real time while being accurate and minimizing false positives. The system must also be scalable and adaptable to changing fraud patterns and emerging threats. To summarize, the problem of real-time fraud detection in telecommunications is to develop an effective and efficient system that can identify fraudulent activities as they happen, reducing financial losses and protecting both service providers and their customers. Fig. 2 shows a stream of call events that we are trying to classify as normal or fraudulent by looking at the call pattern of the customer. The problem is defined as follows:

Definition 1. The goal is to determine whether or not a call is fraudulent in real time using call detail records (CDR).

Suppose that a customer's call event sequence is $E = [e_1, e_2, \ldots, e_T]$, where T is the length of the sequence. Each behavior event $e \in E$ has n features, such as *call_origin*, *call_destination*, *caller_initial_location*, etc. The customer's call event is denoted as $e_t = [f_1^t, f_2^t, \ldots, f_n^t]$, $(1 \leq t \leq T)$, where f_i^t denotes the value of the i^{th} feature. The objective is to determine whether the target of the customer's calling event e_T is fraudulent or not using a set of normal patterns $P = [p_1, p_2, \ldots, p_{T'}]$, where $P \subseteq E$, and $T' \leq T$, and available information about the target event e_T. This task can be formulated as a binary classification task in such a setting. A description of the symbols used in the paper is given in Table 1.

4. Methodology

This section discusses the methodology proposed for real-time fraud detection in the telecommunications industry. The focus of this section is to address the challenge of real-time fraud detection in the telecom industry. We propose a new method that combines Neural Factorization Machines (NFM) and Autoencoders (AE) to detect fraudulent calls that deviate from normal calling patterns in real time. The following sections detail our Neural Factorization Autoencoder (NFA) based detection model, the algorithms utilized, and the functional architecture proposed.

Traditional machine learning models tend to suffer from overfitting due to the multidimensional nature of CDR data and many missing feature values. Although manual feature engineering has been used to enhance the performance of machine learning models, this approach becomes impractical for interactions between higher-order features. To overcome this challenge, embedding features such as factorization and neural network models can be used, which learn feature interactions

Table 1
Description of symbols.

Symbol	Description
e_t	Call event at current time t
E	Set of call events
T	Length of call sequences
n	Number of calling features
N	Size of memory
β	A threshold value to classify records
f_i^t	The i^{th} feature value call at t
P	Set of normal calling patterns
$Z(e_t)$	An intermediate representation of call at t
ψ_i	Embedding matrix or Look-up table of i^{th}*field*
$\mathbb{R}$	Set of real numbers
ϑ_i	The i^{th} embedding vector
d	Dimensions of embedding vector
l	Number of distinct values in a categorical field
$\odot$	The Hadamard product
w_i^t	Attention weight for embedding vector v_i^t
L_{AE}	Autoencoder reconstruction error
S	Set of fraud patterns
η	Number of epochs

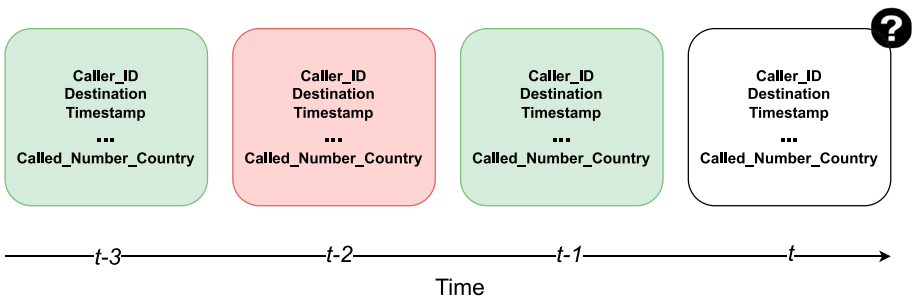

Fig. 2. Call traffic as interconnected streams.

directly from raw data [26–28]. A factorization machine (FM) [27] models the interactions between features as the dot product between two embedding vectors.

To the best of our knowledge, this concept has not been applied to unsupervised learning, specifically in autoencoders that learn a low-dimensional representation from a high-dimensional sparse input. In the context of sparse data, the output of an autoencoder reconstruction quickly loses its sparsity and collapses to the average value of the input features [29]. Wider and deeper autoencoders are not sufficient to capture latent, low-dimensional information because more capacity can lead to autoencoders copying input [28]. In high-dimension and sparse settings, NFM and autoencoders are used to find an effective latent representation of the data that is regularized by adversarial learning.

Fig. 3 illustrates the flow diagram of NFA that uses NFM, AE, and a memory module that is initially trained on a subset of a normal dataset. The memory modules store the patterns associated with normal phone calls. A feature extractor is used to highlight each call event's structure. The current call is analyzed and scored by computing the fraudulent score based on the similarity between its encoding and that stored in memory. Depending on this score, the calling event is used to update the memory if it is deemed normal; otherwise, it is discarded. Memory is required to continuously maintain the update process in order to keep track of the changing patterns of the current data distribution. As data continuously arrives in a streaming system, older patterns stored in memory may no longer be relevant to the current patterns. Therefore, a FIFO replacement policy is used to update the memory module.

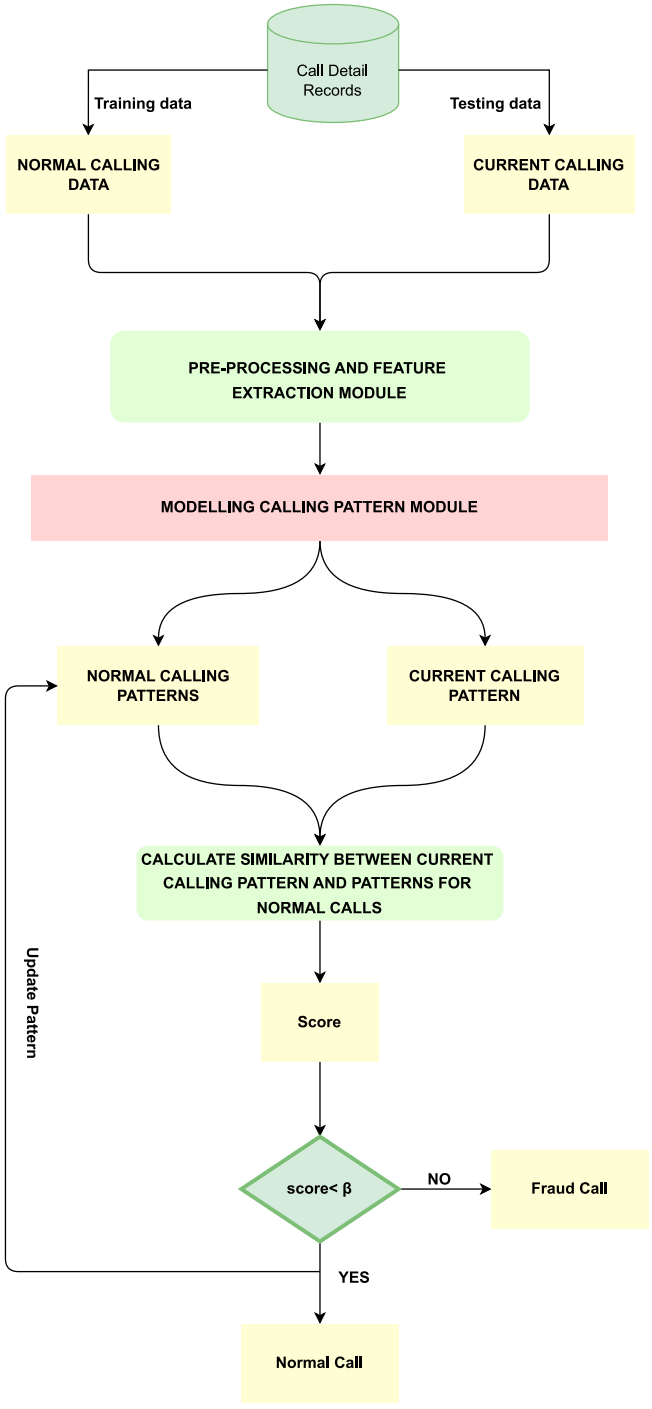

Fig. 3. Flow diagram for NFA.

4.1. Modelling calling patterns

Fraud detection uses two types of patterns: profile-based and anomaly detection [30–33]. Profile-based methods have a database of aberrant attacks, while anomaly detection uses each call's pattern as the baseline to detect fraud by comparing new traffic against individual patterns. If an activity deviates significantly from normal, it may indicate compromise. This article focuses on the profile-based method for fraud detection.

Fig. 4 demonstrates NFM and AE learning the current event's calling pattern (e_t) represented as $Z(e_t^{'})$. NFM captures higher-order feature interactions by feeding dense embeddings into FM. An autoencoder (encoder and decoder, as in Ref. [29]) is used to learn feature representations, with a pattern being the intermediate representation of input data encoded by the encoder. The decoder reconstructs input samples using these intermediate representations.

4.1.1. Look-up embedding

Look-up embedding is a technique in deep learning used for

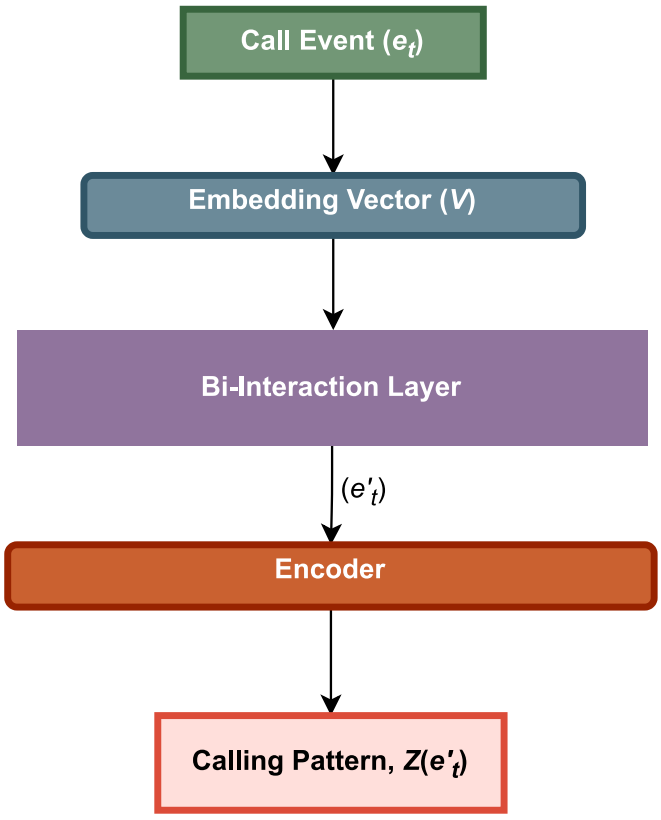

Fig. 4. Modelling calling pattern.

converting discrete variables, such as words or categorical features, into continuous vectors or embeddings. A look-up embedding layer is a dictionary that maps each discrete value to a vector in a high-dimensional space. These vectors are then used as input to the rest of the model. This allows the model to handle categorical variables more effectively than traditional methods like one-hot encoding. Look-up embeddings capture the semantic relationships between the discrete values and enable the model to learn the representations of the variables in a continuous space, leading to improved performance in various tasks such as natural language processing, recommendation systems, and classification.

We employed the look-up embedding method to handle the dense representation of our dataset. This approach has proven effective in modelling user behavior and making predictions, as shown in previous studies like [16,26,34]. In practice, there are two types of fields: categorical fields with a limited number of distinct values (e.g. Network Service Provider or Calling Country) and numerical fields with continuous data. These two types of fields require different look-up embedding techniques. The embedding matrix for the i^{th} field can be expressed as follows:

$$\psi_i \in \begin{cases} \mathbb{R}^{d\times l}, & \text{if } i^{th} \text{ field is a categorical field} \\ \mathbb{R}^{d}, & \text{if } i^{th} \text{ field is a numerical field} \end{cases} \tag{1}$$

Here d is the embedding vector dimension, and l is how many distinct values are in a categorical field {1, 2, …, m}. Next, we compute the embedding vector for the i^{th} feature (f_i^t) as

$$\vartheta_i^t = \begin{cases} \psi_i[f_i^t], & \text{if } i^{th} \text{ field is a categorical field} \\ f_i^t \times \psi_i, & \text{if } i^{th} \text{ field is a numerical field} \end{cases} \tag{2}$$

where $\psi_i[f_i^t]$ is the f_i^t row of ψ_i.

4.1.2. Bi-interaction layer

The Bi-Interaction layer is a common type of layer in recommendation systems and other deep-learning models that deal with sparse data. The Bi-Interaction layer's purpose is to capture the pairwise interactions between the input features. The dot product of each pair of input features is calculated in the Bi-Interaction layer, and all dot products are concatenated to form a high-dimensional dense vector. This dense vector captures the second-order feature interactions between the input features. This helps to capture complex relationships between features that are not captured by simple linear or single-layer models by considering pairwise interactions between features.

The purpose of the look-up embedding matrices in this layer is to extract the feature-level information needed for analysis. This feature extractor also assigns a score to each feature's importance as part of the extraction process. Research has shown that high-order feature interactions are effective [26,27,35,36]. Building upon the success of Factorization Machines [27], we use a novel feature extractor that captures both first- and second-order feature interactions, feeding that information into the autoencoder to capture higher-order interactions. The Bi-interaction layer is defined as follows:

$$e_t^{'} = \sum_{i=1}^{n} w_i^t \vartheta_i^t + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \vartheta_i^t \odot \vartheta_j^t \tag{3}$$

or it can be reformulated in linear runtime [27] as

$$e_t^{'} = \sum_{i=1}^{n} w_i^t \vartheta_i^t + \frac{1}{2}\left[\left(\sum_{i=1}^{n}(\vartheta_i^t)\right)^2 - \sum_{i=1}^{n}(\vartheta_i^t)^2\right] \tag{4}$$

where t^{th} event's call event embedding is represented by $e_t^{'}$, and its i^{th} feature embedding is represented by ϑ_i^t. A hadamard product is represented by the symbol $\odot$, and the attention weight w_i^t for ϑ_i^t is calculated as follows:

$$w_i^t = \frac{exp(a_{f_i^t}^i)}{\sum_{j=1}^{n} exp(a_{f_i^t}^i)} \tag{5}$$

where $a_{f_i^t}^i$ is the learnable parameter for the embedding of f_i^t of the i^{th} field. It is important to note that the learnable parameter $(a_{f_i^t}^i)$ is the same for all values of f_i^t of the numerical field irrespective of the value of f_i^t. This *Bi-interaction* level extractor is able to choose informative features in each call event record.

4.1.3. Autoencoder

In nonlinear data embedding and representation, the use of autoencoders is becoming increasingly popular. Autoencoders often experience difficulty learning a representation when the input is high-dimensional and sparse, and changing a useful representation is difficult [29]. Encoders and decoders with wider and deeper capacities can easily learn to copy input without extracting any useful information about the latent distribution [29]. By utilizing non-linear, higher-order interactions between input features, autoencoders have difficulty learning a satisfactory hash function of the input. Therefore, an explicit layer of feature interaction is proposed as a modelling technique for autoencoders.

The autoencoder is designed to map each data point e_t in the given calling event sequences $E = [e_1, e_2, \ldots, e_T]$ into a low-dimensional, discrete representation in the coding space $e_t^{'} \in R^B$ through an encoding function $f : e_t \rightarrow e_t^{'}$. A decoding function $g : e_t^{'} \rightarrow \hat{e}_t$ is then used to recover e_t as $\hat{e}_t$ from $e_t^{'}$. It is crucial to leverage higher-order interactions when dealing with high-dimensional data to improve the model performance. With its ability to model feature interaction, this encoder finds meaningful representations of the input. The autoencoder is structured as follows:

- $f : e_t \rightarrow e_t^{'}$ is an encoder function that maps e_t into a vector $e_t^{'}$. AE is modelled as a multi-layer perceptron (MLP) with a ReLU activation function at the end.
- $g : e_t^{'} \rightarrow \hat{e}_t$ is a decoder function that reconstructs the input e_t as $\hat{e}_t$. Similarly to the encoder, the decoder is modelled with an MLP.
- The AE determines its parameters by minimizing the squared error reconstruction cost as follows:

$$L_{AE} = \|\hat{e}_t - e_t\|_2^2 = \|g(f(e_t)) - e_t\|_2^2 \tag{6}$$

4.2. Algorithm

The NFM and AE utilize a limited set of CDR datasets to recognize typical call patterns, and the memory component is initialized. When a new call event (e_t) takes place, the NFM and AE model the call pattern and produce a compressed representation (line 3). Using this compact output, $z(e_t)$, they compute the l_2 distance (line 4). Then, they calculate the final score (line 5). This score is compared to the user-defined threshold (β) (line 6). If the score is below the threshold, the memory is updated with a new pattern based on the FIFO update policy. If the fraudulent score surpasses the threshold, the alarm is triggered, and the call is blocked (lines 9 and 10). Finally, the score for the current call e_t is returned (line 11). Based on the patterns of the fraudulent calls, they are classified into various categories, such as Smishing, Wangiri, MTRA, etc. If there is no existing fraud category, it is declared as a new fraud.

Algorithm 1: Neural Factorization Autoencoder (NFA)

Input: Call event sequences $E = \{e_1, e_2, ..., e_t\}$
Output: Fraudulent score for each call event
Initialize memory module M with a normal patterns, as $\{z_1, z_2, ...z_M\}$; // Here, M is the size of memory
for *each call event* e_t **do**
  Model calling pattern, $z(e'_t)$ // Calculate the difference between the current calling pattern and patterns stored in a memory
  $\text{diff}(z_t, z_i) = \|z_t - z_i\|_2$, for all $i \epsilon M$
  // Assign a score to each call event
  $Score(e_t) = \sum_{i=1}^{M} \frac{diff(z_t, z_i)}{|M|}$
  // Memory updation
if $Score(e_t) < \beta$ **then**
  Update the pattern as *FIFO* policy.
else
  Trigger Alarm();
  Block the event, e_t;
  $e_t \rightarrow S$ // Assign fraud calls to different categories of fraud based on their patterns
Output fraudulent score, $Score(e_t)$;

5. Experimental study

Here, we discuss the experimental setup, evaluation metrics, comparative analysis, and baseline approaches. To validate the effectiveness of our model, a series of experiments were conducted on 670,211 call records. To assess the effectiveness of our proposed approach in detecting fraudulent calls, we compared it with one online and six offline anomaly detection methods. In this experimental study, parameters and hyper-parameters were fine-tuned, tested, and retained based on their performance. The parameter values used in the experiment are given in Table 2.

5.1. Dataset description

For our experimental study, we focus on Wangiri fraud. This type of fraud, also known as "one ring and cut" or "ping-calls" or "call-back scams," involves fraudsters calling thousands of random numbers and disconnecting after a few rings in hopes that the recipients will call back. If they do, they are then exposed to a pre-recorded message and charged an exorbitant fee. Detecting Wangiri fraud is challenging due to the lack of a chargeable duration and visibility, as the terminating carrier cannot determine the origin country's risk.

To collect a real-world Call Detail Record (CDR) dataset, we utilized a telecom honeypot from our French industrial partners. The dataset includes millions of CDRs, each with information such as the caller and recipient's phone numbers, country of origin, target country, start time, etc. Throughout our analysis, we do not use any personal information, and all identifiable information (such as phone numbers and geographical locations) is encrypted to protect privacy. The telecom operator labeled each incoming call as either "fraudulent" or "normal," providing us with 28,172 fraudulent calls[2] to analyze.

Table 2
Parameter settings for our experimental study.

Parameter	Values
Learning rate	$1e-2$
Batch size	2000
Epochs	7500
Activation function	ReLU
Optimization method	Adam

5.2. Experimental setup

This experiment was conducted on a CDR dataset using Google Colab with a GPU. The NVIDIA Tesla T4 GPU was utilized, along with the necessary libraries and software being installed. The algorithms used in the experiment generated an outlier-ness score, with a higher score indicating a more anomalous sample. Four performance metrics were evaluated, including the True Positive Rate (TPR), False Positive Rate (FPR), Area Under the Curve of Receiver Operating Characteristics (AUC-ROC), and F1-score. Detailed descriptions of these metrics can be found in Section 5.3.

For training the model, we use 80% mixed data, while in testing, we use 20% fraud and legitimate call data. With NFA, we analyze customer usage patterns with the open-source PyTorch implementation of Neural Factorization Machines (NFMs) [26] and extract features using an autoencoder (AE). Both encoders and decoders use Neural Nets and Adam optimizers with $1e-2$ learning rates. In addition, we use a memory module that keeps track of current call patterns and updates them based on the normal calling behavior using a First In First Out (FIFO) memory update policy. In this case, the size N was searched in the following memory sizes: {64, 128, 256, 512, 1024, 2048, 4096, and 8192}. We searched for the threshold (β) in {0.0001, 0.001, 0.01, 0.1, and 1.0} to determine whether or not to update the memory. The results are based on the best results obtained from 15 experiments unless explicitly specified for each parameter setting.

The previous system for real-time fraud detection in telecommunications used a rule-based approach, which relied on a set of predefined rules and patterns to identify fraud. However, these approaches had several limitations, including the inability to detect new and evolving fraud patterns and a high false positive rate. This paper proposes a novel approach that uses NFM and Autoencoder to detect fraud in real-time. The key innovations of our approach include using NFM to learn the underlying fraud patterns and using an autoencoder to reduce the dimensionality of the data and improve performance.

5.3. Performance metrics

In this section, we present evaluation metrics such as TPR, FPR, F1-score, and AUC for assessing the accuracy of the fraud detection algorithm. Based on the experimental results, which are presented in the form of a confusion matrix, the following terms are defined:

- True Positive (TP): The number of normal calls predicted as normal by the algorithm.
- False Positive (FP): The number of fraudulent calls incorrectly predicted by the algorithm as normal calls.
- True Negative (TN): The number of fraudulent calls correctly predicted by the algorithm as fraud.
- False Negative (FN): The number of normal calls predicted as fraud by the algorithm.

Due to the imbalance of our CDR dataset, we consider both TPR and FPR in our analysis. TP, FP, TN, and FN can calculate these performance metrics. In this study, we used the following metrics to measure performance.

True Positive Rate (TPR): In machine learning, this is also called

[2] Our industry partners provide the data for the experimental works.

sensitivity or recall and refers to a method's ability to estimate how many calls were correctly identified in a dataset. Calculating it mathematically, it can be written as follows:

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

False Positive Rate (FPR): False Positive Rate (FPR) is a measure of the likelihood that an actual value will be reported as positive when it actually is negative. This can be expressed mathematically as follows:

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

In some cases, we may prefer high precision and recall at the expense of another metric. When precision is maximized, the number of false positives will be minimized, whereas when the recall is maximized, the number of false negatives will be minimized. The F1 score can be employed to find an optimal balance between precision and recall in cases where we wish to find an optimal blend of the two metrics.

F1-score: In terms of precision and recall, the F1-score is calculated as follows:

$$F1 - score = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{9}$$

A classification model with the highest F1 scores best balances precision and recall.

AUC-ROC: The performance of a fraud detection system can also be measured using this metric. In ROC analysis, the y-axis represents the true positive rate (TPR) of the test. At the same time, the x-axis indicates the false positive rate (FPR), which represents the probability that the test will produce a false alert. This AUC-ROC value is simply a measure of how much of an area is under the curve, and a perfect value for the AUC-ROC is 1.

Execution Time: In our study, we evaluated the execution time of different algorithms on a complete dataset and presented the results in Table 3. It is evident from the table that our proposed approach, NFA, has a higher execution time compared to other baseline methods. However, this increase in execution time is a trade-off for the improved performance of the algorithm in real-time fraud detection. NFA is designed to handle real-time fraud detection in a dynamic and complex environment and requires a more sophisticated and computationally intensive approach, aiming to improve the algorithm's overall detection performance. This is why the NFA uses advanced algorithms and techniques to analyze and process large amounts of data effectively.

To demonstrate the feasibility of our approach in real-time environments, where the detection speed is crucial, we measured the execution time of our proposed approach on a single call event and found that it takes only 937.22 ms to classify the call event as fraud or normal. This low execution time demonstrates the feasibility of our approach in real-time environments. The fast-processing speed of our approach ensures that call events can be quickly and accurately classified, reducing the risk of fraud and minimizing the impact on the customer experience. This high performance is achieved through a combination of NFM and Autoencoder and efficient implementation, allowing our approach to be applied in real-world scenarios where the speed of detection is crucial.

5.4. Results and discussion

This section discusses the performance of different algorithms in detecting fraud. The experimental results of NFA with all baseline algorithms are shown in Table 3. It reports the detection performance of the NFA and other baseline methods in terms of the TPR, FPR, AUC, and F1-score, along with their corresponding execution times.

We conducted an experiment on NFA without the factorization machine (FM) to evaluate the impact of the FM on detection performance. Table 3 compares the results of NFA and NFA (without FM), demonstrating that NFA outperforms the NFA (without FM). This highlights the significance of incorporating the FM into our proposed approach.

Next, we compared NFA with other baseline methods, and Table 3 shows that NFA offers superior detection performance compared to these methods based on selected performance metrics. Our results reveal the following insights:

- NFA outperforms baseline methods, primarily due to its implementation of the NFM approach that captures higher-order feature interactions using the FM and considers event sequences in recognizing calling patterns. This allows NFA to capture hierarchical structures effectively.
- The NFA improves performance by adding the bi-interaction layer and bi-interaction embedding reconstruction to the autoencoder, and we found that the factorization component of the NFA greatly contributes to the improvement of the model. This shows that the presence of a factorization component enhances detection performance significantly.
- FM-based methods perform better than non-FM methods in sequence-based fraud detection. The NFA effectively learns higher-order feature interactions from original data, which is an advantage over non-FM approaches. NFA's ability to adapt to different distributions in calling pattern analysis is also due to its complex parametric layers.
- Compared to MEMSTREAM and other offline methods, NFA performs well and achieves the best detection performance. While offline methods may be effective in batch-wise analyses where data is stored for some time before processing, the performance of LODA, an offline detection method, is inferior to all other methods. This is because LODA's goal of preserving pair-wise distances of the original space makes it unable to estimate outliers accurately.

Finally, we conduct a security analysis of the NFA. The telecommunications industry is prone to false positives (FP), which can cause customer dissatisfaction. A high FP rate results in normal calls being misidentified as fraudulent calls, damaging telecommunications companies' reputations and financial health. On the other hand, a high rate of false negatives (FN) leads to fraudulent calls being misidentified as legitimate calls. To effectively detect telecom fraud, a detection system must have low rates of both FP and FN. Compared to the baseline

Table 3
Performance comparison of NFA with baseline methods.

Algorithm	TP	FP	TN	FN	AUC	TPR	FPR	F1-score	Time(s)
ECOD [TKDE, 2022]	445843	8468	14017	67840	0.7456	0.8679	0.3766	0.9211	22.8
COPOD [ICDM, 2020]	368111	6740	15745	145572	0.7084	0.7166	0.2997	0.8285	18.43
SUOD [MLSys, 2021]	341382	7720	14765	172301	0.6606	0.6645	0.3433	0.7913	570.52
DeepSVDD [ICML, 2018]	393963	11288	11197	119720	0.6324	0.7669	0.5020	0.8574	184.39
LODA [ML, 2016]	239140	8835	13650	274543	0.5363	0.4655	0.3929	0.6279	56.53
iForest [ICDM, 2008]	380168	11439	11046	133515	0.6156	0.7401	0.5087	0.8398	57.11
Memstream [WWW, 2022]	507184	6169	22003	134855	0.8474	0.7899	0.2189	0.8779	206.75
NFA (without FM)	508278	11621	16551	133761	0.7601	0.7916	0.4125	0.8748	187.56
NFA	590002	4161	24011	52037	**0.9106**	**0.9189**	**0.1476**	**0.9545**	260.81

approaches, the NFA approach has the lowest rates of FP and FN and the highest number of true positives (TP) and true negatives (TN). These results demonstrate that our NFA approach outperforms the baseline approaches.

The advantages of a Neural Factorization Autoencoder (NFA) based real-time telecom fraud detection method over other existing approaches are:

- Improved Fraud Detection Accuracy: NFA uses deep learning algorithms to detect complex and evolving fraud patterns, resulting in higher accuracy than traditional methods.
- Unsupervised Learning: NFA is an unsupervised learning method that does not require labeled data to train the model. This allows for quick adaptation to new fraud patterns, improving the overall accuracy of the fraud detection system.
- Anomaly Detection: NFA uses autoencoder architecture to identify anomalies in the data, making it well-suited for real-time fraud detection.
- Multi-dimensional Data Analysis: NFA can analyze multi-dimensional data such as user behavior, network activity, and call data records to identify fraud, leading to a more comprehensive and effective fraud detection system.
- Scalability: NFA is highly scalable and can process large volumes of data in real-time, making it well suited for use in large telecommunication networks

In summary, the proposed real-time telecom fraud detection method offers improved accuracy, unsupervised learning, anomaly detection, multi-dimensional data analysis, and scalability, making it a promising solution for real-time fraud detection in telecommunication networks.

Further, we analyzed normal and fraudulent phone calls made during a single day and on different days of the week to gain a deeper understanding of their patterns and characteristics. The insights gained from this analysis can be used to improve the security of communication systems, enhance the customer experience, and make data-driven decisions.

The analysis revealed the frequency and timing of normal and fraudulent calls, which can be visualized in Fig. 5(a) and (b). In these figures, we show the number of normal and fraudulent calls made over a specified period and analyze the patterns of fraud calls over a day.

By dividing the data into intervals of time, such as hours or half-hours, and counting the number of normal and fraudulent calls in each interval, we can identify spikes or dips in call volume and determine if fraudulent calls are more likely to occur at specific times of the day. This information is crucial in detecting and preventing fraudulent activity, as it helps to identify trends and anomalies in the data.

Our analysis revealed that the frequency of fraudulent calls increases from 8 in the morning, reaches its peak at noon, and then decreases until 11 at night. The frequency of fraudulent calls during nighttime is lower than in the daytime.

This pattern suggests that fraudulent call activity follows a distinct trend, with higher activity during the day and lower activity at night. The increased frequency of fraud calls during the day can be attributed to various factors such as higher levels of human activity, more potential victims, less awareness, greater opportunities for fraudsters, and weaker security measures.

Similarly, the analysis was conducted for different days of the week to uncover any trends or patterns that are specific to certain days. For instance, fraudulent calls could be more frequent on weekends or weekdays, and the analysis aimed to uncover such trends.

The results depicted in Fig. 6(a) and (b) showed that the total number of calls made on each day of the week, there was little to no difference between the days. However, when we analyzed the percentage of fraudulent calls made each day, we found that approximately 4% of the calls made each day were fraudulent. This indicates that while the total number of calls may not vary much from day to day, the proportion of fraudulent calls is consistent at around 4%.

In conclusion, the analysis of normal and fraudulent phone calls provides valuable insights into the types and patterns of calls being made. It can help organizations detect potentially fraudulent activity. This information can be used to improve the security of communication systems and enhance the overall customer experience.

5.5. Ablation study

We then present an ablation test on the NFA to examine the effect of memory length and threshold (β) on performance. The study is divided into two parts:

(a) Memory Length (N) and Performance: Our study revealed that increasing memory size led to fewer false positives, as shown in Fig. 7. The best performance was achieved at $N = 4096$, and further increases in memory length led to a decline in performance. The results indicate that memory sizes that are either too large (e.g. $N = 8192$) or too small (e.g. $N = 64$) can negatively impact recognition performance as the memory may not correctly store the current pattern. A large memory ensures that only the current trend is learned, but representations of previous trends will also contaminate it. On the other hand, a small memory size does not provide enough representations of the current trend for inclusion in the model, leading to suboptimal performance.

(b) Threshold (β) Impact: To decide whether to update the memory with a record, we use a threshold based on its fraud score. When the threshold is high, memory updates occur frequently; when it's low, memory updates rarely happen. This lets us control the frequency of memory updates based on our preferences. As Table 4 demonstrates, the threshold significantly affects NFA's detection performance. The table shows that NFA performs well when $\beta = 0.001$, but performance deteriorates as the threshold increases.

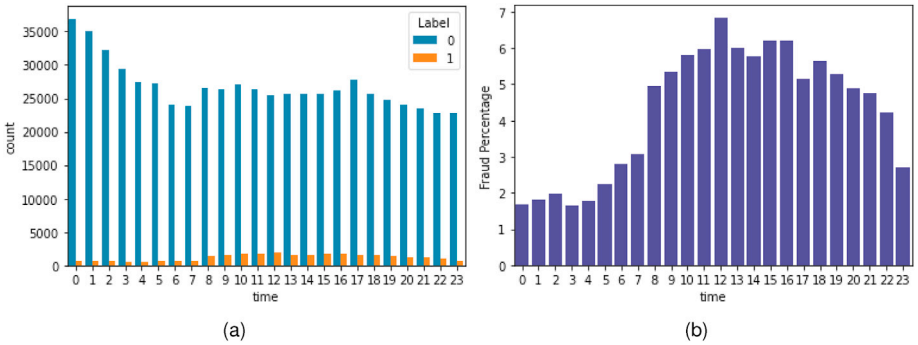

Fig. 5. (a) Hourly Distribution of Normal and Fraud Calls (0: normal, 1: fraud) (b) Hourly Fraud Percentage in Calls.

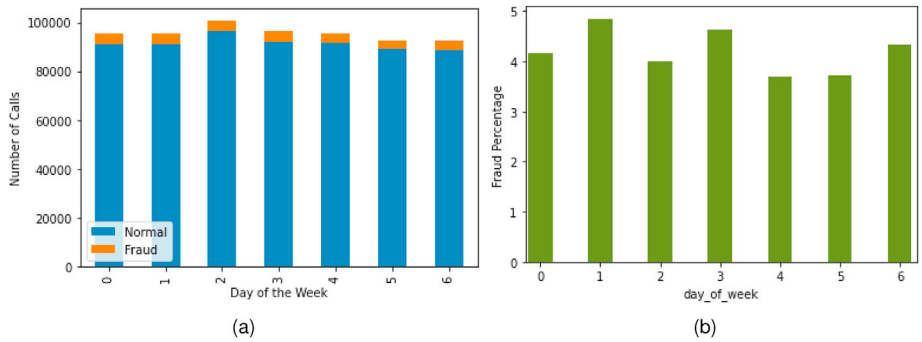

Fig. 6. (a) Distribution of normal and fraud calls by day of the week (b) fraud percentage by day of the week.

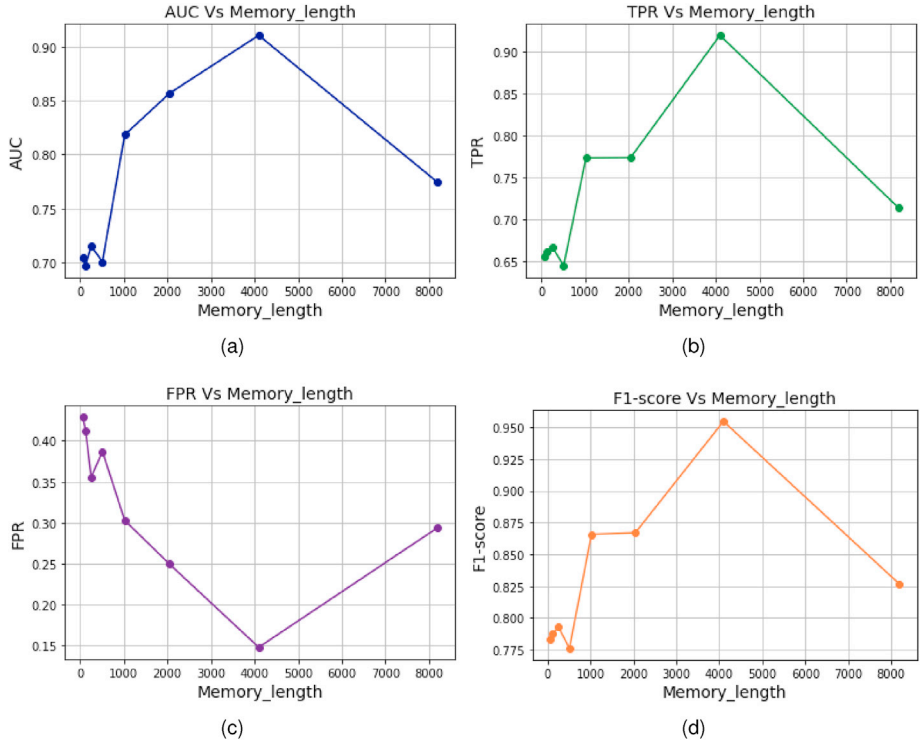

Fig. 7. Effect of memory length on NFA with the following settings: (learning rate = 1e-2, epochs = 750, $\beta = 0.001$).

Table 4
Effect of threshold (β) on NFA with the following settings: (learning rate = 1e-2, epochs = 750, memory length = 4096).

Threshold (β)	AUC	TPR	FPR	F1-score
0.0001	0.8474	0.7899	0.2189	0.8779
0.001	**0.9106**	**0.9189**	**0.1476**	**0.9545**
0.01	0.7602	0.7916	0.4125	0.8748
0.1	0.8568	0.7735	0.2494	0.8669
1.0	0.8184	0.7732	0.3026	0.8655

6. Conclusion and future work

In this paper, we present NFA, a new real-time technique to identify telecom fraud based on customer calling behavior. NFA integrates neural factorization machines (NFM) and autoencoders (AE) to capture customer usage patterns, and its NFM component directly learns feature interactions from raw data, improving fraud detection. Furthermore, NFA employs memory modules to comprehend customer calling patterns. The threshold-based classification algorithm assesses calls and categorizes them as either fraudulent or normal based on their deviation from typical call patterns.

Finally, we demonstrate the effectiveness of our proposed approach through a series of experiments conducted on a real-world CDR dataset comprising 670,211 call records. In order to evaluate the performance of our proposed method, we compared it to several baseline approaches. We considered TPR, FPR, AUC-ROC, and F1 scores to assess the performance. From the experimental results, we can conclude that NFA performed well in detecting fraud in real-time. Our future plans include adapting this approach to other types of telecommunications fraud, such as MTRA and Smishing. Also, we are looking for an optimized solution that can reduce the execution time of the detection system by using the most advanced machine learning methods, such as multi-head attention parallel computing in the Transformer.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that might have appeared to influence the work presented in this article.

Acknowledgements

This research work has been conducted in cooperation with members of DETSI project supported by BPI France and Pays de Loire and Auvergne Rhone Alpes. The authors would like to sincerely thank Lilian Perron from Araxxe and Fabrice Chesneau from Expandium, a VIAVI Solutions company, for supporting our research.

References

[1] K. Wieland, Network & application monitoring-can revenue assurance stop this happening? revenue leakage continues to hamper the telecom industry and operators, understandably, don't want to talk about it in, Telecommun. Int. 38 (8) (2004) 10–11.
[2] CFCA, Fraud Loss Survey. https://cfca.org/wp-content/uploads/2021/12/CFCA-Fraud-Loss-Survey-2021-2.pdf, 2021. (Accessed 7 June 2021).
[3] W. Rodgers, R. Attah-Boakye, K. Adams, Application of algorithmic cognitive decision trust modeling for cyber security within organisations, IEEE Trans. Eng. Manag. 69 (6) (2020) 3792–3801.
[4] Y. Jiang, G. Liu, J. Wu, H. Lin, Telecom fraud detection via hawkes-enhanced sequence model, IEEE Trans. Knowl. Data Eng. 35 (5) (2023) 5311–5324.
[5] V. Chadyšas, A. Bugajev, R. Kriauzienė, O. Vasilecas, Outlier analysis for telecom fraud detection, in: Proceedings of the 15th International Baltic Conference on Digital Business and Intelligent Systems, 2022, pp. 219–231.
[6] R. Brause, T. Langsdorf, M. Hepp, Neural data mining for credit card fraud detection, in: Proceedings of the 11th International Conference on Tools with Artificial Intelligence, 1999, pp. 103–106.
[7] S. Rosset, U. Murad, E. Neumann, Y. Idan, G. Pinkas, Discovery of fraud rules for telecommunications—challenges and solutions, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 409–413.
[8] Q. Zhao, K. Chen, T. Li, Y. Yang, X. Wang, Detecting telecommunication fraud by understanding the contents of a call, Cybersecurity 1 (8) (2018) 1–12.
[9] A. Ravi, M. Msahli, H. Qiu, G. Memmi, A. Bifet, M. Qiu, Wangiri fraud: pattern analysis and machine learning-based detection, IEEE Internet Things J. 10 (8) (2023) 6794–6802.
[10] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection for discrete sequences: a survey, IEEE Trans. Knowl. Data Eng. 24 (5) (2010) 823–839.
[11] R. He, J. McAuley, Fusing similarity models with Markov chains for sparse sequential recommendation, in: Proceedings of the 16th International Conference on Data Mining, ICDM, 2016, pp. 191–200.
[12] S. Rayana, L. Akoglu, Less is more: building selective anomaly ensembles, ACM Trans. Knowl. Discov. Data 10 (4) (2016) 1–33.
[13] J. Jurgovsky, M. Granitzer, K. Ziegler, S. Calabretto, P.-E. Portier, L. He-Guelton, O. Caelen, Sequence classification for credit-card fraud detection, Expert Syst. Appl. 100 (2018) 234–245.
[14] S. Wang, C. Liu, X. Gao, H. Qu, W. Xu, Session-based fraud detection in online e-commerce transactions using recurrent neural networks, in: Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2017, pp. 241–252.
[15] B. Branco, P. Abreu, A.S. Gomes, M.S. Almeida, J.T. Ascensão, P. Bizarro, Interleaved sequence rnns for fraud detection, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 3101–3109.
[16] Y. Zhu, D. Xi, B. Song, F. Zhuang, S. Chen, X. Gu, Q. He, Modeling users' behavior sequences with hierarchical explainable network for cross-domain fraud detection, in: Proceedings of the Web Conference, 2020, pp. 928–938.
[17] J. Zhang, H. Chen, X. Yao, X. Fu, Cpfinder: Finding an unknown caller's profession from anonymized mobile phone data, Digit. Commun. Netw. 8 (3) (2022) 324–332.
[18] M.U. Togbe, Y. Chabchoub, A. Boly, M. Barry, R. Chiky, M. Bahri, Anomalies detection using isolation in concept-drifting data streams, Computers 10 (1) (2021) 13.
[19] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, G. Chen, Ecod: unsupervised outlier detection using empirical cumulative distribution functions, IEEE Trans. Knowl. Data Eng 35 (12) (2023) 12181–12193.
[20] Z. Li, Y. Zhao, N. Botta, C. Ionescu, X. Hu, Copod: copula-based outlier detection, in: Proceedings of the 2020 IEEE International Conference on Data Mining, ICDM, 2020, pp. 1118–1123.
[21] Y. Zhao, X. Hu, C. Cheng, C. Wang, C. Wan, W. Wang, J. Yang, H. Bai, Z. Li, C. Xiao, et al., Suod: accelerating large-scale unsupervised heterogeneous outlier detection, Proc. Mach. Learn. Syst. 3 (2021) 463–478.
[22] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S.A. Siddiqui, A. Binder, E. Müller, M. Kloft, Deep one-class classification, in: Proceedings of the 37th International Conference on Machine Learning, 2018, pp. 4393–4402.
[23] T. Pevnỳ, Loda: lightweight on-line detector of anomalies, Mach. Learn. 102 (2) (2016) 275–304.
[24] F.T. Liu, K.M. Ting, Z.-H. Zhou, Isolation forest, in: Proceedings of the 8th IEEE International Conference on Data Mining, 2008, pp. 413–422.
[25] S. Bhatia, A. Jain, S. Srivastava, K. Kawaguchi, B. Hooi, Memstream: memory-based streaming anomaly detection, in: Proceedings of the ACM Web Conference, 2022, pp. 610–621.
[26] X. He, T.-S. Chua, Neural factorization machines for sparse predictive analytics, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 355–364.
[27] R. Steffen, Factorization machines, in: Proceedings of the 10th IEEE International Conference on Data Mining, ICDM, 2010, pp. 995–1000.
[28] K.D. Doan, P. Yadav, C.K. Reddy, Adversarial factorization autoencoder for look-alike modeling, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2803–2812.
[29] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Massachusetts, 2016.
[30] P. Burge, J. Shawe-Taylor, Detecting cellular fraud using adaptive prototypes, in: Proceedings of the AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management, 1997, pp. 9–13.
[31] D.E. Denning, An intrusion-detection model, IEEE Trans. Software Eng. SE-13 (2) (1987) 222–232.
[32] T. Fawcett, F. Provost, Adaptive fraud detection, Data Min. Knowl. Discov. 1 (3) (1997) 291–316.
[33] T.F. Lunt, A survey of intrusion detection techniques, Comput. Secur. 12 (4) (1993) 405–418.
[34] J. Tang, K. Wang, Personalized top-n sequential recommendation via convolutional sequence embedding, in: Proceedings of the 11th ACM International Conference on Web Search and Data Mining, 2018, pp. 565–573.
[35] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, G. Sun, xdeepfm: combining explicit and implicit feature interactions for recommender systems, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1754–1763.
[36] R. Wang, B. Fu, G. Fu, M. Wang, Deep & cross network for ad click predictions, in: Proceedings of the ADKDD'17, 2017, pp. 1–7.