Telecom Fraud Detection via Hawkes-enhanced Sequence Model

Yan Jiang, Guannan Liu, Junjie Wu, Hao Lin

Abstract—Detecting frauds from a massive amount of user behavioral data is often regarded as finding a needle in a haystack. While tremendous efforts have been devoted to fraud detection from behavioral sequences, existing studies rarely consider behavioral targets and companions and their interactions simultaneously in a sequence model. In this paper, we suggest extracting source and target neighbor sequences from the temporal bipartite network of user behaviors, and disclose the interesting correlation mode and repetition mode hidden inside the two types of sequences as important clues for fraudsters distinguishment. We then propose a novel Hawkes-enhanced sequence model (HESM) by integrating the Hawkes process into LSTM for historical influence learning. A historical attention mechanism is also proposed to enhance the strength of the long-term historical influence in response to the repetition mode. Moreover, in order to collectively model both types of neighbor sequences for capturing the correlation mode, we propose a correlation gate to control the information flow in sequences. We conduct extensive experiments on real-world datasets and demonstrate that HESM outperforms competitive baseline methods consistently in telecom fraud detection. Particularly, the abilities of HESM in historical influence leaning and sequence correlation learning have been explored visually and intensively.

Index Terms—fraud detection, Hawkes process, Long Short Term Memory (LSTM), sequence model, temporal bipartite network

1 Introduction

TELECOM fraud has been a pervasive type of fraudulent crimes since telephone becomes one of the most important communication channels in the society. With specially designed fraudulent scripts during several consecutive calls, fraudsters may induce calling targets to trust their fake stories and transfer money to designation accounts, which could bring huge economic losses for ordinary telecom users. It is reported that the number of telecom fraud instances has reached 537,000 per year in China, and the resulting economic loss is as high as 12 billion RMB. Though great efforts have been made to crack down varied types of telecom frauds, *e.g.*, using fraudulent voice templates as interceptor of telecommunications, such cases still occur frequently for the very low interception probability of voice templates and the rapidly evolving cheating tactics. Fraudsters today tend to act more like normal users and collaborate as highly organized groups, which makes it extremely challenging to detect telecom frauds from the massive amount of normal calling records.

Prior research work generally tackles fraud detection problem with respect to different behavioral characteristics. In recent years, one research mainstream regards the consecutive behaviors as behavioral sequences, and exploits sequence learning methods to distinguish fraudsters from normal users [1, 2, 3]. These studies mainly reconstruct normal sequential patterns to predict future behaviors, and the sequences with large prediction losses tend to be identified as fraudulent. Except for the sequential perspective, fraudulent behaviors are usually regarded as interactions between two different parties, *i.e.*, source users who launch fraudulent behaviors and target users who are potential victims. In this regard, many prior methods have also been proposed to tackle fraud detection from the network structure formed by interactive behaviors [4, 5, 6, 7]. However, fraudsters can dynamically manipulate the interaction structure to escape, and hence some extended studies adopt the idea of burst detection to detect abrupt changes of the interaction structure [8, 9]. Despite the great efforts devoted to fraud detection, rarely have they taken both the temporal information and network interaction perspectives into account, which indeed motivates our study.

In this work, we take a bipartite view of telecom calling records, with the callers as the source nodes and the callees the target nodes. Since the interactions between source and targets occur at different time, each interactive edge can be annotated with the calling time, giving rise to the *temporal bipartite network*. We then take a sequence view of the network and form the *target neighbor sequence* to describe the short-term massive calling behavior of a fraudster, and form the *source neighbor sequence* with the purpose of capturing the latent gang crimes behavior of fraudsters. We have found several notable characteristics in distinguishing fraudsters from normal callers from the observing the neighbor sequences in the real-world telecom data. On one hand, the repetitions of the calling targets in the sequence show distinctive patterns for normal callers and fraudsters. That is, the normal users usually maintain long-term contacts with their families and friends, and thus the repetitions of

- *Yan Jiang, Guannan Liu and Junjie Wu are with Beihang University. Junjie Wu is also with Beijing Key Laboratory of Emergency Support Simulation Technologies for City Operations, and MoE Key Laboratory of Complex System Analysis and Management Decision. Email: jyvip5211@163.com, {liugn, wujj}@buaa.edu.cn. Hao Lin is with Department of Informatics, Technical University of Munich. Corresponding authors: Guannan Liu, Junjie Wu.*

This work was partially supported by National Key R&D Program of China (2019YFB2101804). Dr. Guannan Liu's work was supported by National Natural Science Foundation of China (NSFC) under Grant 71701007, 92046025. Dr. Junjie Wu's work was supported in part by NSFC under Grants 72031001, 71725002, 72021001, and by Key Projects of Science and Technology Program of Beijing Municipal Education Commission under Grant KZ202110017025..

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2022.3150803, IEEE Transactions on Knowledge and Data Engineering

the calling target neighbors in the sequences would likely span over a long time period; while on the contrary, the fraudsters often make massive calls to find possible victims, which results in shorter-time repetitions. On the other hand, we have also found that the correlation between source neighbor sequences and target neighbor sequence shows contrastive trends for the two types of callers. That is, the average number of target neighbors shows negative correlation with the number of source neighbors for fraudsters but vice versa for the normal callers.

Therefore, the observed *repetition mode* in target neighbor sequences and the *correlation mode* between the two types of sequences can serve as important clues for distinguishing the calling behaviors, which however, have merely been considered in prior work on sequence modeling. Thus, how to model the unique sequential patterns, *i.e.*, the repetition and the correlations in these neighbor sequences remains to be a focal challenge. In particular, we make use of the *Hawkes process* to model the neighbor sequences, which allows past events to influence current events with a *decay gate* developed specifically to help model the long-term historical influences. In the meantime, a *historical attention mechanism* is proposed to jointly capture the historical influences with regards to the *repetition mode*. Furthermore, in order to collectively model both types of neighbor sequences for capturing the *correlation mode*, we propose a *correlation gate* to control the flow of information across different types of neighbor sequences. Based on the historical influence learning as well as the correlation learning, a new conditional intensity function is developed, which finally gives rise to a *Hawkes-enhanced sequence model* (HESM). The experimental results on real-world telecom datasets demonstrate the superior detection performance of our model in comparison with some state-of-the-art methods. Particularly, the abilities of HESM in historical influence leaning and sequence correlation learning have been explored visually and intensively, which further explains the effectiveness of the modelling components of HESM.

The remainder of this work is organized as follows. Section 2 introduces the related literature. Section 3 introduces the real-life telecom dataset and the two distinguishing behavioral modes, which defines our problem in this study. Section 4 describes the details of our HESM model and Section 5 presents the experimental results. We finally conclude our work in Section 6.

2 Related Work

In this paper, we aim to detect frauds by sequence prediction based on learned sequence pattern representations. Prior works along this line can be roughly summarized into three mainstreams as follows.

2.1 Sequence-based Fraud Detection

Sequence-based fraud detection generally identifies an entire sequence, subsequence or sequential pattern to be anomalous if it deviates significantly from normal sequences [10]. The most intuitive methods are similarity-based techniques, which compute the pairwise similarities between sequences using some specific similarity measure [11]. There also exists a stream of studies that have focused on window-based mining and locating some partial abnormalities in an entire sequence [12]. Another widely used family of approaches are Markov model-based techniques, which model the generative process of sequence data from a probabilistic perspective [13]. More recently, deep learning methods based on sequence data have become another important class of methods for fraud detection. Some scholars have studied abnormal sequence data detection using recurrent neural networks (RNNs) [2, 3]. In particular, Kieu et al. [14] proposed two solutions in time series based on recurrent autoencoder ensembles for anomaly detection. Su et al. [15] proposed Omnianomaly, a novel stochastic recurrent neural network for multivariate time series anomaly detection that can deal with explicit temporal dependence among stochastic variables to learn robust representations of input data. Bernardo et al. [16] proposed a complete RNN framework to detect fraud in real-time, presenting an efficient ML pipeline from preprocessing to deployment. Zhu et al. [17] proposed a hierarchical explainable network (HEN) to model users behavior sequences, improving the performance of fraud detection and making the inference process interpretable. Generally, these sequence-based approaches are closely related to our work because they share the common purpose of modeling complex normal patterns to reveal anomalous patterns underlying the observed behaviors. However, they have rarely taken both the sequence and network interaction perspectives into account simultaneously.

2.2 Hawkes Process

Temporal point processes [18] are mathematical abstractions for many different phenomena across a wide range of domains. In particular, temporal point processes are well suited to learn the event patterns of user behaviors from event sequence data [19, 20]. Hawkes process [21], a specific type of temporal point process, has been widely used to model event streams, including for constructing and inferring network structures [22] and discovering patterns in social interactions [23]. In these studies, the specified form of the point process limits its capability to capture the dynamics of data, and the historical events were generally thought to influence current events independently and additively. Recently, an increasing number of studies have focused on the combination of neural networks and Hawkes processes. Some work used recurrent neural network to approximate the conditional intensity function of the Hawkes process [24, 25, 26]. There also exist other work that have modeled continuous-time point processes [27, 19, 28]. Additionally, some work combined recurrent neural networks and Hawkes process to capture the temporal dependency and historical influence [29, 30, 31]. Particularly, Cai et al. [32] proposed a long and short term Hawkes process model, which models the short-term dependency between users' actions within a period of time via a multi-dimensional Hawkes process and the long-term dependency between actions across different periods of time via a one dimensional Hawkes process. Okawa et al. [33] proposed a deep mixture point process model, which uses the deep learning method and point process intensity to capture the complex effects of unstructured contextual features on the event occurrence.

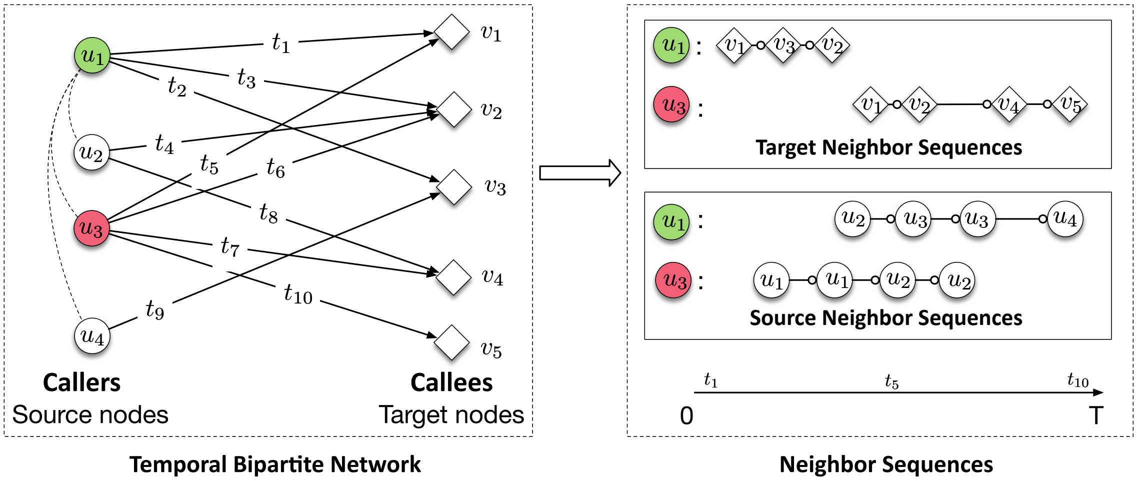

u_1
u_2
u_3
u_4
v_1
v_2
v_3
v_4
v_5
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t_9
t_{10}
Callers
Source nodes
Callees
Target nodes
Temporal Bipartite Network
u_1 : v_1 v_3 v_2
u_3 : v_1 v_2 v_4 v_5
Target Neighbor Sequences
u_1 : u_2 u_3 u_3 u_4
u_3 : u_1 u_1 u_2 u_2
Source Neighbor Sequences
t_1
t_5
t_{10}
0
T
Neighbor Sequences

Fig. 1. An Example of Temporal Bipartite Network and Neighbor Sequences

Vassøy et al. [34] used a temporal hierarchical recurrent neural network to model intersession relations and capture users' long-term preferences for inter-session and intra-session recommendations and return-time prediction. Zuo et al. [35] leveraged the self-attention mechanism to capture long-term dependencies and meanwhile enjoys computational efficiency. These methods are similar to our model philosophy in incorporating the Hawkes process with deep learning models. However, they have not emphasized the specific sequential patterns exclusively for telecom frauds, which calls for more delicate models in capturing the discovered characteristics in the sequences.

2.3 Graph-based Fraud Detection

Graph-based fraud detection has become popular and has received significant attention because many real-world fraudulent incidents can form graph structures. One category of studies along this line has focused on detecting anomalous nodes [36, 37] and particular anomalous subgraph structures [6, 38] in static networks. In general, these studies are mainly based on handcrafted node attributes or structures; however, fraudsters can easily modify their attributes and connection structures to avoid being detected. Another category of graph-based methods focuses on detecting anomalous structures in dynamic networks. In these studies, a scoring function is generally defined for the normality of nodes, edges or subgraphs [39, 40, 41, 42, 8, 43, 44] and then abrupt changes in the scoring values are detected with respect to the whole sequences in a dynamic network. These methods generally require long sequences, in which the behaviors remain relatively stable at most times and abnormalities are shown within a particular time window. However, these studies merely exploit the dynamic interactive behaviors to mine temporal behavioral patterns but are not able to discover other complex specific patterns.

3 Preliminaries and Problem Formulation

In this section, we first conduct an exploratory analysis to a real-world telecom dataset. We aim to unveil the unusual behavioral patterns of telecom fraudsters, which further motivates the model to propose. Finally, we give a formal problem definition.

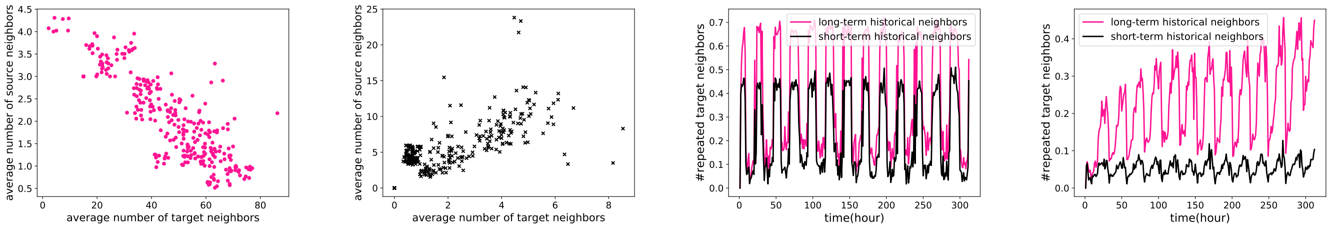
3.1 Real-world Telecom Dataset Description

We obtain a real-world telecommunication dataset from one of the largest telecommunication operators in China. The dataset contains millions of call detail records (CDRs), each recording the detailed information of a phone call such as the phone numbers of both the caller and callee, the starting time of a call, the call duration and the call ending time. For privacy concerns, all the identifiable information of individuals (*e.g.*, phone numbers and regions) is encrypted. With the crowdsourcing service provided by the telecom operator, receivers can mark each incoming call as either "fraudulent" or not, which enables us to gather 109,425 labeled callers from September 18th to October 1st, 2018. The callers labeled for more than 10 times as "fraudulent" are further marked as *fraudulent callers* and the number of fraudsters is 17,471, accounting for approximately 16% of the total labeled callers. We then gather all 109,425 callers' full calling sequences during this period, which involves 3,905,403 CDRs and 1,948,068 distinct callees.

3.2 Exploratory Analysis

In the real-world telecom dataset, the connections between callers and callees at different time form a *temporal bipartite network*, in which callers act as source users, callees act as target users, and a directed edge associate with a timestamp denotes that a caller calls a callee at a specific timestamp, as shown in Fig.1.

Definition 1. (Temporal Bipartite Network) A temporal bipartite network is a network with edges annotated by the chronological interaction relations between source users and target users. Specifically, a temporal bipartite network is denoted as $\mathcal{G} =< \mathcal{U}, \mathcal{V}, \mathcal{E} >$, where $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$ denotes the source users, $\mathcal{V} = \{v_1, v_2, \cdots, v_m\}$ denotes the target users, and $\mathcal{E}$ denotes the set of edges, *i.e.*, the interaction behaviors between $\mathcal{U}$ and $\mathcal{V}$ formed at different time t.



(a) Fraudsters (b) Normal Callers (c) Fraudsters (d) Normal Callers

Fig. 2. Analysis of Sequential Behaviors in Telecom Networks

Given a temporal bipartite network, the dynamic evolution of the network can be explored clearly. Specifically, the target neighbors of each caller, *i.e.*, the callees that directly connect with the caller, can also be derived. For example, as shown in Fig.1, caller u_1 has direct edges with callees v_1, v_2, v_3 at different time, which become the target neighbors of u_1. Furthermore, the target neighbors of each caller in the network can be organized as a sequence in chronological order. We formally define the *target neighbor sequence* as follows.

Definition 2. (Target Neighbor Sequence) Given a source user u in a temporal bipartite network and its target neighbor set $\mathcal{TN}(u)$, the target neighbor sequence of u can be represented chronologically as $\mathcal{TNS}(u) = [(t_1, v_1), (t_2, v_2), \cdots, (t_l, v_l)]$, with each tuple representing the interaction behavior between u and its target neighbor $v_i \in \mathcal{TN}(u)$ at time t_i.

It is generally understood that fraudulent callers tend to make scam calls to possible victims. We can therefore expect the target neighbor sequences could help to expose fraudulent callers with unusual calling targets and calling behaviors. It is also reported that many telecom frauds committed in recent years are *gang crimes*, which implies there might exist clues of fraudster groups in the relations among callers themselves. In reality, the relations between callers can be revealed by the *source neighbors* of each caller, *i.e.*, other callers that have common target neighbors with the caller in the temporal bipartite network. To avoid data sparsity, we set those callers who have common target neighbors within a certain time window (such as one hour) as the source neighbors of a caller at that time step. For example, as shown in Fig.1, caller u_1 has common target neighbors with callers u_2, u_3, u_4 at different time steps; therefore, the source neighbors of u_1 are u_2, u_3, u_4. These source neighbors of the caller are in chronological order to indicate the evolution of the relations at different time steps. We formally define the *source neighbor sequence* in chronological order below.

Definition 3. (Source Neighbor Sequence) Given a source user u in the temporal bipartite network and its source neighbor set $\mathcal{SN}(u)$, the source neighbor sequence can be chronologically represented as a series of target neighbors, *i.e.*, $\mathcal{SNS} = [(t_1, u_1), (t_2, u_2), \cdots, (t_l, u_l)]$, with each tuple representing that the source neighbor $u_i \in \mathcal{SN}(u)$ has at least one common target neighbor with source u at time step t_i.

Given the two types of neighbor sequences of a caller, in what follows, we showcase two sequential patterns, *i.e.*, the correlation mode and the repetition mode, that can help distinguish fraudsters from normal callers.

3.2.1 The Correlation Mode

In this section, we aim to explore the *correlation mode* between the target neighbor sequences and the source neighbor sequences of the callers. To this end, we serially calculate the average numbers of target neighbors as well as source neighbors of all the callers per hour on the chronological timeline. For the convenience of observation, we draw a scatterplot with the horizontal axis denoting the average number of target neighbors, and the vertical axis denoting the average number of source neighbors, and each dot represents a particular time step. As shown in Fig.2(a) and Fig.2(b), it is interesting to find that fraudsters generally exhibit negative correlations between their target neighbor sequences and source neighbor sequences, but normal callers exhibit mostly positive correlations.

To illustrate this, we recall that fraudsters now prefer to gang crimes, *i.e.*, work as a group to deceive target callees according to the following deception. First, *junior fraudsters* are relatively scattered and each makes massive calls to find as many potential victims as possible. At this stage, fraudsters have many target neighbors but few source neighbors. Then, when some potential victims are hooked, *senior fraudsters* form a persuasive group with different roles to call and induce the chosen preys to transfer money. During this phase, fraudsters have relatively few target neighbors but more source neighbors.

In addition, from the two subfigures, we can see the significant difference in neighbor scales. That is, fraudsters generally have much more target neighbors than normal callers, which coincides with the fact that fraudsters constantly make many calls to harvest more victims, but normal users dial more to callees whom they are familiar with. We can also see that fraudsters have fewer source neighbors than normal callers, implying that the number of fraudsters is limited by the size of a fraudulent group.

3.2.2 The Repetition Mode

In real life, callers may repeatedly call the contacts whom they have called in the past. We explore this *repetition mode* by computing the repeatability of the target neighbors of a source caller in different time steps. Specifically, we compare two types of repetitions of target neighbors, *i.e.*, the repeated

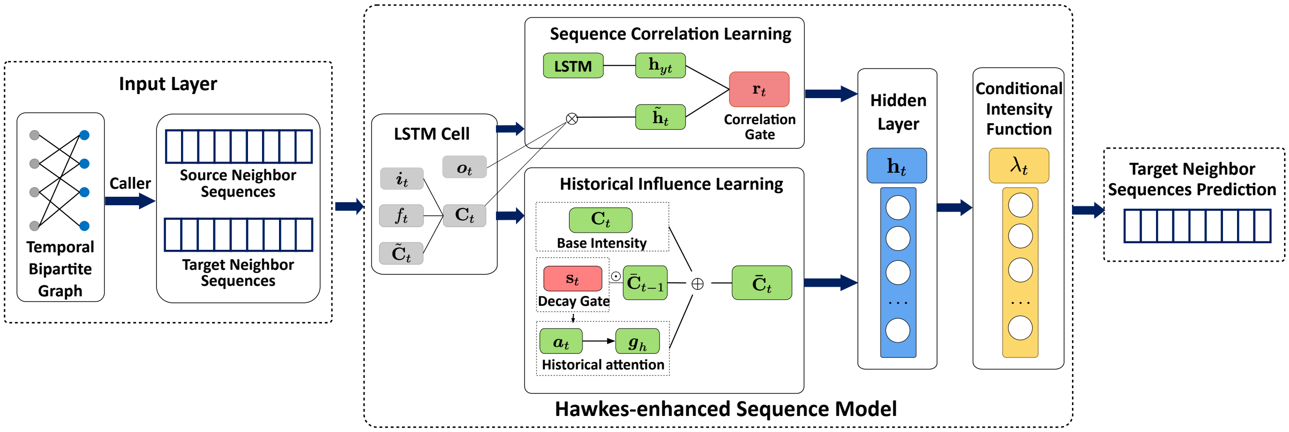


Fig. 3. An Overview of the Model Architecture

neighbors at the last time step referring to as *short-term historical neighbors*, and the repeated neighbors from the beginning of the observation till the last time step referring to as *long-term historical neighbors*. We show the neighbor repetitions for fraudsters and normal callers, respectively, in Fig.2(c) and Fig.2(d). The horizontal axis denotes the time step in hour and the vertical axis denotes the average number of repeated target neighbors at each time step.

It is obvious that while both fraudsters and normal callers repeat some of their historical calls periodically, their repetition modes are diverse. First of all, according to the short-term lines in black, the fraudsters generally have much more short-term historical neighbors than normal callers, and the peaks of their repetitive calls usually occur within a much shorter time period. This agrees with the fact that telecom fraudsters usually make massive scam calls to large but carefully selected sets of target neighbors intensively, which inevitably results in repeated calls. In contrast, most normal callers would not repeat calling their target neighbors, *e.g.*, the acquaintances, in a short period if with little information update.

We next compare the long-term lines in red. Let us focus on the gap between the long-term and short-term historical neighbors along the time line, which turns out to be much sharper for the normal callers than for the fraudsters. Indeed, normal callers are accustomed to maintaining long-term contacts with their families and acquaintances, which generates the wide gap between the numbers of long- and short-term historical neighbors. The case, however, is not for the fraudsters, who seeks more short-term contacts as victims rather than long-term contacts as friends.

3.3 Problem Definition

The correlation mode and repetition mode hidden inside callers' target as well as source neighbor sequences reveal the unusual behaviors of the fraudsters, which indeed motivates our study on building sequence model for telecom fraud detection.

Formally, given a temporal bipartite network $\mathcal{G} =< \mathcal{U}, \mathcal{V}; \mathcal{E} >$, for each source user $u \in \mathcal{U}$, $\mathbf{q}_t^u = [(t_1, v_1), (t_2, v_2), \cdots, (t_l, v_l)]$ denotes its target neighbor sequence, and $\mathbf{q}_s^u = [(t_1, u_1), (t_2, u_2), \cdots, (t_{l'}, u_l)]$ denotes its source neighbor sequence, where l (l') is the maximum number of time steps in the sequence. We aim to detect fraudsters accurately via learning the correlation mode and repetition mode from $\{\mathbf{q}_t^u\}$ and $\{\mathbf{q}_s^u\}$ simultaneously in an end-to-end manner.

4 The Proposed Model

Motivated by the observations from the real-world telecom data, we propose to model both the source and target neighbor sequences collectively for telecom fraud detection. Traditional sequence models such as RNN and LSTM can well capture the short-term time dependencies of consecutive behaviors. But the above-mentioned repetition mode contains long-term historical influences in sequences and the correlation mode involves the correlations between different types of behavioral sequences, both of which were not particularly considered by the traditional models. Therefore in this paper, we aim to integrate the observed clues into the sequence models for better distinguishing fraudsters from normal callers.

In particular, we can regard each call from the source to a target as an event, and the historical events would have an influence on the current event in terms of the repetition mode, which can be nicely modeled with a temporal point process. Thus, we make use of the *Hawkes process* to model the neighbor sequences, which allows past events to influence current events in a subtle way. Meanwhile, when each time period in the sequence is represented with a hidden state, we can further develop a *historical attention mechanism* to capture the repetition mode in the calling history. Furthermore, in order to collectively model both types of neighbor sequences for capturing the correlation mode, we propose a *correlation gate* to control the flow of information in sequences. These give rise to the so-called *Hawkes-enhanced sequence model* (HESM).

The model architecture is shown in Fig.3. The neighbor sequences of each caller are regarded as model input, and the LSTM cell is enhanced with the novel *historical influence learning* and *sequence correlation learning* modules. Finally, a novel *conditional intensity function* based on the hidden states of recurrent neural networks is formulated to enable the target neighbor prediction.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2022.3150803, IEEE Transactions on Knowledge and Data Engineering

4.1 Historical Influence Learning

Given the source and target neighbor sequences consisting of the timestamps and the neighbors, it is naturally appealing to capture the sequential patterns by modeling the sequences. LSTM [45] is typically a neural-based sequence model that can well model the short-term dependencies along the sequences. One major problem with the traditional LSTM is that the stored memories only transit between consecutive events in an accumulative way. In reality, however, consecutive events in a sequence might not have dependencies within a short time period. Instead, non-adjacent events in long-term might have interdependencies as revealed in the previous exploratory study.

Considering the distinctive historical influences of normal callers and the fraudsters, a more delicate sequence model that can capture the repetitions in a longer time hiorizon is in need. Thus, we integrate the Hawkes process [21], a well known temporal point process into LSTM by introducing a decay gate to learn the complex historical influences effectively. Additionally, we design a historical attention mechanism to capture the repetition mode in the neighbor sequences. In what follows, we firstly review the basics of the Hawkes process, and then propose a modified neural unit with the decay gate and the historical attentions.

4.1.1 Hawkes Process

The Hawkes process is a temporal point process that can capture the influences of historical events on current events with time decay effect. A core mechanism of the Hawkes process is the conditional intensity function, which represents the rate of occurrence for a new event conditioned on historical events and can be employed for predicting future events. In the traditional Hawkes process, the conditional intensity function is formulated as follows:

$$\lambda\left(t\right) = \mu\left(t\right) + \int_{-\infty}^{t} \exp\left(-\delta\left(t-s\right)\right) dN\left(s\right), \qquad (1)$$

where $\mu\left(t\right)$ is the base rate of an event, showing the spontaneous arrival rate of the event at time t; $\exp\left(-\delta\left(t-s\right)\right)$ is a decay function in the form of an exponential function that models the time decay effect of the historical events on the current event. Moreover, to handle different types of events, the Hawkes process can be extended to a multivariate case, where we can define the conditional intensity function for each event type. This gives rise to the self-exciting multivariate Hawkes process [21].

The interdependency between historical and current events modeled in the traditional Hawkes process is deemed desirable for capturing the historical influences in neighbor sequences. We therefore adopt the Hawkes process to model the complex historical influences. Particularly, the *excitation effect* is set as a sum over all the historical target neighbors events of different types, captured by an excitation rate $\alpha_{z,y}$ between historical target neighbor z and current target neighbor y. Hence the sequences can be modeled as follows:

$$\lambda_{y|x}\left(t\right) = \mu_{x,y} + \sum_{t_z<t} \alpha_{z,y} \exp\left(-\delta\left(t-t_z\right)\right), \qquad (2)$$

where $\mu_{x,y}$ represents the base connection rate between a neighbor y and a source node x, while z is the historical neighbor event that occurred prior to time t. $\alpha_{z,y}$ represents the degree to which a historical neighbor z can excite the current neighbor y, and $\exp\left(-\delta\left(t-t_z\right)\right)$ is the time decay effect function, illustrating that historical neighbors can influence current neighbors in different intensities over time.

Though the traditional Hawkes process is appropriate for modeling the neighbor sequences, it is inefficient in handling large-scale event types, and the parameter inference could be very time consuming [46]. In addition, the conditional intensity function of the Hawkes process generally has the restriction that historical events only have independent and additive influences on the current event, but the influences of historical events on future events might be superadditive or even subtractive. Therefore, rather than directly modeling the sequences with Hawkes process, we remove the restrictions by designing an intensity function conditioned on the hidden states of LSTM. We then enhance the neural units of LSTM with regards to the pervasive traits discovered in the above observations.

4.1.2 Decay Gate

The basic neural unit for the hidden state at each time step of LSTM consists of an input gate $\boldsymbol{i}_t \in \mathbb{R}_+^d$, a forget gate $\boldsymbol{f}_t \in \mathbb{R}_+^d$, an output gate $\boldsymbol{o}_t \in \mathbb{R}_+^d$, a candidate cell state $\widetilde{\boldsymbol{C}}_t \in \mathbb{R}^d$, and a cell state $\boldsymbol{C}_t \in \mathbb{R}^d$, with d denoting the hidden layer size, all of which can control the flow of memory between consecutive time steps. Obviously, such an information flow fails to capture the crucial historical influences, especially the long-term influences in the repetition mode for discovering fraudsters.

Specifically, as shown in Eq.(2), we can see that the excitation rates of the historical neighbors on the current neighbor are determined by the interval lengths between the historical time and current time, *i.e.*, the excitation rate decays over time exponentially. Therefore, in analogy to the time decay factor δ of historical events on the current event in Eq.(2), we design a new decay gate $\boldsymbol{s}_t \in \mathbb{R}_+^d$ to control the time decay effect of historical neighbor influence on the current neighbor over time in the neural unit, which can be formulated as follows,

$$\boldsymbol{s}_t = softplus(\boldsymbol{W}_{sx}\boldsymbol{x}_t + \boldsymbol{W}_{sh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_s), \qquad (3)$$

where $\boldsymbol{x}_t \in \mathbb{R}^v$ is the input vector at current time step t, and v denotes the input size. $\boldsymbol{W}_{sh} \in \mathbb{R}^{d\times d}, \boldsymbol{W}_{sx} \in \mathbb{R}^{d\times v}$ are the matrices of weight parameters, and $\boldsymbol{b}_s \in \mathbb{R}^d$ is the corresponding bias vector. Particularly, $softplus(x) = \log(1 + \exp(x))$ is the activation function of the neural network, which can be regarded as a smooth version of the $relu(x) = \max(0, x)$ function. With this activation function, Each value of vector $\boldsymbol{s}_t$ is ensured to be positive, and we can guarantee that the time decay of historical influence is proportional to the time interval between the historical time and the current time.

In sum, the decay gate $\boldsymbol{s}_t$ can mimic the decay effect of the historical influences. In particular, we can employ the decay gate $\boldsymbol{s}_t$ and the last cell state to express the short-term historical influence, deterministically controlling the decay effect of short-term influence from the last time step to the current time step t. Thus, the cell state and hidden state of the neural network can account for the decaying influences of the historical neighbors.

4.1.3 Historical Attention

According to the analysis in Section 3.2.2, fraudsters can hardly maintain long-term and stable relationships with targets, which on the contrary is the commonplace for normal callers. Therefore, except for the decay effect and the short-term influences by considering the last cell state, it is necessary to capture the long-term historical influences by modeling the complete historical neighbor sequence of each source caller. We apply a historical attention mechanism [47] by computing the attention weights of historical events with respect to the current event based on all the historical states. In this regard, for the current event at each time step t, we track all the historical events at time step $k \in \{t_1, t_2, \cdots, t_p\}$, where t_p denotes the preceding time step of the current time t. Finally, the attention weights are calculated through Eq.(4) and Eq.(5) as follows:

$$\boldsymbol{a}_{t,k} = \boldsymbol{W}_a(tanh(\boldsymbol{W}_{ah}\widetilde{\boldsymbol{h}}_t + \boldsymbol{W}_{ak}\boldsymbol{h}_k + \boldsymbol{b}_a)), \tag{4}$$

$$\boldsymbol{\alpha}_{t,k} = softmax(\boldsymbol{a}_{t,k}), \tag{5}$$

where $\widetilde{\boldsymbol{h}}_t \in \mathbb{R}^d$ represents the candidate hidden state given by Eq.(7) below. To implement the attention mechanism, we score each historical state $\boldsymbol{h}_k \in \mathbb{R}^d$ by comparing it with the current candidate hidden state $\widetilde{\boldsymbol{h}}_t$ and further normalize the scores as the final output of attention weights. $\boldsymbol{a}_{t,k} \in \mathbb{R}$ denotes the raw attention weight vector, and $\boldsymbol{\alpha}_{t,k} \in \mathbb{R}_+$ denotes the normalized weight vector. $\boldsymbol{W}_a \in \mathbb{R}^{1\times d}$, $\boldsymbol{W}_{as} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{W}_{ak} \in \mathbb{R}^{d\times d}$ are the weight parameters matrices, and $\boldsymbol{b}_a \in \mathbb{R}^d$ is the attention bias vector.

Finally, we obtain the historical context vector $\boldsymbol{g}_h \in \mathbb{R}^d$ by considering both the decay effect and the long-term historical influences. Specifically, as shown in Eq.(6)

$$\boldsymbol{g}_h = \sum_{k=1}^{t-1} \boldsymbol{\alpha}_{t,k} \odot \boldsymbol{h}_k \odot \exp(-\boldsymbol{s}_t(t - t_k)), \tag{6}$$

it can be computed by the weighted average of all the historical states $\boldsymbol{h}_k, k \in \{t_1, t_2, \cdots, t_p\}$. Meanwhile, the decay effect can be represented by an exponential decay factor $\exp(-\boldsymbol{s}_t(t - t_k))$, which is analogous to decay effect of the conditional intensity function of the Hawkes process.

4.2 Sequence Correlation Learning

As seen in the above exploratory analysis, the correlation mode between target and source neighbor sequences is crucial for distinguishing fraudsters from normal users. However, prior work on sequence models merely addresses the correlations between different types of sequences. We therefore further incorporate a new correlation gate $\boldsymbol{r}_t \in \mathbb{R}^d$ into the neural unit of the traditional LSTM and plug it into the conditional intensity function.

Particularly, we introduce a weight parameter $\boldsymbol{W}_r \in \mathbb{R}^{d\times d}$ to learn the correlation mode between the source and target neighbor sequences. As shown in Eq.(7) and Eq.(8)

$$\widetilde{\boldsymbol{h}}_t = \boldsymbol{o}_t \odot \boldsymbol{C}_t, \tag{7}$$

$$\boldsymbol{r}_t = \tanh(\widetilde{\boldsymbol{h}}_t - (\boldsymbol{W}_r\boldsymbol{h}_{yt} + \boldsymbol{b}_r)), \tag{8}$$

at each time step, we derive the candidate hidden state $\widetilde{\boldsymbol{h}}_t \in \mathbb{R}^d$ from the target neighbor sequences based on the output gate $\boldsymbol{o}_t$ and the current cell state $\boldsymbol{C}_t \in \mathbb{R}^d$. In addition to learning from the target neighbor sequences, we also incorporate the current hidden state $\boldsymbol{h}_{yt} \in \mathbb{R}^d$ learned from the source neighbor sequences with respect to another LSTM model. In this regard, the weight parameter can be regarded as the correlation between $\widetilde{\boldsymbol{h}}_t$ and $\boldsymbol{h}_{yt}$ with a bias vector $\boldsymbol{b}_r \in \mathbb{R}^d$, which is further transformed with an activation function $\tanh(\cdot)$ to obtain the correlation gate $\boldsymbol{r}_t$.

4.3 Conditional Intensity Function

In our model, the dynamics of time-varying intensity are controlled by the hidden state $\boldsymbol{h}_t$, which depends on the newly designed memory cell state $\overline{\boldsymbol{C}}_t \in \mathbb{R}^d$. The key of $\overline{\boldsymbol{C}}_t$ is to extend the basic neural units of the traditional LSTM with the decay gate and the historical attention mechanism. As a result, the original cell state $\boldsymbol{C}_t$ of the traditional LSTM can act as the base intensity of occurrence for an event at time t, and the new cell state $\overline{\boldsymbol{C}}_t$ can be employed to account for the self-exciting effects of the Hawkes process. In sum, $\overline{\boldsymbol{C}}_t$ can be formulated as follows:

$$\overline{\boldsymbol{C}}_t = \boldsymbol{W}_{c1}\boldsymbol{C}_t + \boldsymbol{W}_{c2}(\overline{\boldsymbol{C}}_{t-1} \odot \exp(-\boldsymbol{s}_t(t - t_p))) + \boldsymbol{W}_{c3}\boldsymbol{g}_h, \tag{9}$$

where t_p denotes the last time step prior to the current time step, and $\boldsymbol{W}_{c1}, \boldsymbol{W}_{c2}, \boldsymbol{W}_{c3} \in \mathbb{R}^{d\times d}$ are weight parameters matrices. $\boldsymbol{C}_t$ denotes the original cell state at the current time step t, which is similar to the base intensity of occurrence for the next event. $\overline{\boldsymbol{C}}_{t-1} \odot \exp(-\boldsymbol{s}_t(t - t_p))$ denotes the self-exciting influences of the historical neighbor events on the occurrence of the current neighbor with an exponentially decaying rate. The merit of this formulation lies in that we can explicitly represent the event history by a latent vector with a nonlinear mapping of the conditional intensity function, without specifying a fixed parametric form for the dependency structure over the historical events.

We then derive the hidden state $\boldsymbol{h}_t$ of the proposed model by employing the output gate $\boldsymbol{o}_t$ of the traditional LSTM, and the correlation gate $\boldsymbol{r}_t$ and the newly designed updated cell state $\overline{\boldsymbol{C}}_t$ given by Eq.(8) and Eq.(9), respectively, as follows:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \boldsymbol{r}_t \odot \tanh(\overline{\boldsymbol{C}}_t). \tag{10}$$

We can see that $\overline{\boldsymbol{C}}_t$ deterministically controls the hidden state $\boldsymbol{h}_t$, and thus can affect indirectly the conditional intensity function vector $\boldsymbol{\lambda}(t) \in \mathbb{R}^v_+$ via $\boldsymbol{h}_t$, which can be regarded as the intensity of future neighbors' occurrence to predict the target neighbor sequence. With the enhanced neural unit of LSTM, we can extend the conditional intensity function vector by conditioning on the hidden states output in each time step of LSTM, which can be formulated as,

$$\boldsymbol{\lambda}(t) = f(\boldsymbol{W}_{hq}\boldsymbol{h}_t + \boldsymbol{b}_q), \tag{11}$$

where $\boldsymbol{W}_{hq} \in \mathbb{R}^{v\times d}$ is weight parameters matrix and $\boldsymbol{b}_q \in \mathbb{R}^v$ is the bias vector. f is a nonlinear function, and we can choose the $softplus(\cdot)$ function to ensure a positive intensity. Intuitively, $\boldsymbol{h}_t$ summarizes the historical behaviors $\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_{t-1}$ and includes the representations of the current behavior. In short, we can predict the node that most likely becomes the next target neighbor of the caller.

Algorithm 1 Training Algorithm of HESM

```
Input: The training samples: the target neighbor sequences $\boldsymbol{X}_{target}$, the source neighbor sequences $\boldsymbol{X}_{source}$; The maximum number of epochs: epoch; The number of mini-batches: batch.
Output: The network parameters: $\Theta$.
Prepare the training dataset $\{\boldsymbol{X}_{target}, \boldsymbol{X}_{source}\}$;
Initialize the network parameters $\Theta$;
for each $i \in 1, 2, \cdots, epoch$ do
    Shuffle the training dataset;
    for each $j \in 1, 2, \cdots, batch$ do
        for each $\boldsymbol{q}^j_{target} \in \boldsymbol{X}_{target}, \boldsymbol{q}^j_{source} \in \boldsymbol{X}_{source}$ do
            Compute $\boldsymbol{h}_{yt}$ w.r.t. traditional LSTM;
            Compute $\boldsymbol{h}_t$ w.r.t. Eq.(10);
            Compute $\boldsymbol{\lambda}(t)$ w.r.t. Eq.(11);
        end for
        Compute the loss function of the training samples w.r.t. Eq.(12) and Eq.(13);
        Update the parameters $\Theta$ by Adam algorithm;
    end for
end for
```

4.4 Target Neighbor Sequence Prediction

Based on the learned representations of neighbor sequences, we then propose an unsupervised fraud detection framework by sequentially predicting the next calling target in a target neighbor sequence.

Without ground-truth labels for the real-life telecom frauds in the training phase, we can rank the callers according to the loss value between the predicted neighbor sequences and the real target sequences. Since we assume that normal callers have more stable contacts in their neighbor sequences, their sequential patterns should be more easily learned than those of fraudsters and hence result in a lower prediction loss. In contrast, fraudsters do not have stable contacts and often change their temporal calling modes fiercely. This implies that to predict their neighbor sequences is more difficult and may result in a larger loss value.

Formally, the loss function of this framework can be formulated as follows:

$$\boldsymbol{y}_t = softplus(\boldsymbol{W}_{hq}\boldsymbol{h}_t + \boldsymbol{b}_q), \tag{12}$$

$$L = -\frac{1}{N}\frac{1}{T}\sum_{n=1}^{N}\sum_{t=1}^{T}(\boldsymbol{y}_t^m - \ln(\sum_{v=1}^{V}\exp(\boldsymbol{y}_t^v))), \tag{13}$$

where m is the real calling target at the t-th time step and $\boldsymbol{y}_t \in \mathbb{R}_+^v$ is the predicted target vector based on the conditional intensity function vector $\boldsymbol{\lambda}(t)$. V is the number of targets. T is the maximum sequence length, and N is the total number of sequences. We use *cross-entropy* [48] with softmax to obtain the loss for the prediction. Algorithm 1 illustrates the training process of our model. We employ the Adam [49] algorithm to minimize the loss function.

5 Experiments

In this section, we conduct fraud detection experiments on real-world telecom datasets. All experiments are implemented on a server with Intel Xeon E5-2609 v4 8 1.7GHz CPUs and 4 GeForce GTX 1080 Ti GPUs for fair comparison.

TABLE 1
Statistics of the Experimental Datasets

	Recording	**Genuine**	**Complete**
# Fraudulent callers	2199	776	17471
# Normal callers	13356	9986	91954
# Total callers	15555	10762	109425
# Interactions	1770240	941176	3905403

5.1 Experimental Setup

5.1.1 Real-world Telecom Datasets

We use a real-world large-scale telecom network for our experiments, the details of which has been given in Section 3.1. Since different fraudulent types have their own characters, we additionally construct two datasets, each having only one type of fraudsters, to test the robustness of HESM in different fraudulent scenarios. The statistics of all the three datasets are given in Table 1.

The two types of fraudsters include *recording* and *genuine* callers. Recording callers first record their voices as pre-designed scripts and replay those scripts when calling targets for intentional deceptions. Genuine callers pretend to be officials working in public sectors such as police, procurator, tax bureau, *etc.*, or senior but unfamiliar superiors knowing some private information, and speak directly to possible victims to gain higher credibility. It is reported that genuine callers have become the major criminal force in telecom frauds in recent years. The ability of HESM in detecting genuine callers is thus the concern of our experimental study. We randomly extract partial data of two fraudulent types from the *Complete* dataset and form the *Recording* and *Genuine* datasets, respectively.

Given the real-world telecom datasets, we construct each caller's target and source neighbor sequences to train and test our model and the baseline models. Since we have the labels for both fraudulent and normal callers, we use *Precision* (P), *Recall* (R), *F-measure* (F). Since the precision and recall may be largely influenced by the threshold, we also adopt the *Area Under the Curve* (AUC) as validation measures to evaluate the model performance which can better demonstrate the overall performances. All the experiments are repeated for five times to obtain the average performances for reliable evaluation. Given the focus of fraudsters and limited text space, we only report the classification performance of the positive class in the experiments.

5.1.2 Baseline Methods

Table 1 shows that normal callers dominate in the datasets, and therefore it is reasonable to assume sequence learning-based baseline models can well capture normal sequential patterns in the data. Based on this assumption, we first train the neighbor sequences in an unsupervised setting by predicting the future target neighbors. Then, the prediction error shown in Eq.(13) is employed as the anomalous score, given that fraudulent behaviors usually diverge from normal sequential patterns and are more difficult to predict.

In the experiments, we compare our HESM with several sequence learning methods, including variants of RNNs, neural Hawkes methods, *etc.*, as follows.

Long Short-Term Memory (LSTM) [45]: This method was originally designed to solve the gradient vanishing and explosion problems of RNNs. Its recurrent unit consists of a

TABLE 2
Performances of Fraud Detection on Complete Telecom Dataset

Methods	Training Proportion 30%				Training Proportion 60%				Training Proportion 90%			
	P	R	F	AUC	P	R	F	AUC	P	R	F	AUC
LSTM	0.5502	**0.9546**	0.6581	0.6357	0.5323	0.9809	0.6527	0.6321	0.6330	***0.8668***	0.6635	0.7827
GRU	0.4297	0.7860	0.4872	0.5874	0.4709	**0.9939**	0.5581	0.6396	0.6111	**0.9444**	0.6524	0.7394
RMTPP	0.2170	***0.8829***	0.3413	0.5313	0.2811	0.8922	0.4078	0.5515	0.3908	0.7932	0.4729	0.5884
ERNN	0.1810	0.9265	0.3029	0.5662	0.2107	***0.9178***	0.3427	0.5977	0.3000	0.6000	0.4000	0.6753
DeepHawkes	0.7925	0.8476	***0.8060***	0.5147	0.7807	0.9043	***0.8098***	0.5120	0.8717	0.9247	***0.8868***	0.6527
NeuralHawkes	0.9275	0.7103	0.8045	***0.8851***	0.9322	0.4472	0.6044	***0.8622***	***0.9371***	0.8543	0.8901	***0.9194***
HAInt-LSTM	0.7376	0.7236	0.7297	0.8142	0.8510	0.6638	0.7457	0.8276	0.8544	0.6666	0.7870	0.8298
NHA-LSTM	0.8123	0.6611	0.7286	0.8117	0.8583	0.7401	0.7922	0.8295	0.8658	0.7433	0.7696	0.8328
THP	0.6480	0.4327	0.6038	0.6208	0.6667	0.6154	0.6400	0.6591	0.9286	0.6334	0.7530	0.8070
AnomRank	0.9120	0.6085	0.7098	0.7083	***0.9097***	0.6580	0.7279	0.7160	0.9367	0.7118	0.8098	0.8591
Omnianomaly	***0.9253***	0.7345	0.8180	0.9296	0.9048	0.8571	0.8803	0.9517	0.9402	0.8055	0.8676	0.9529
HESM (ours)	**0.9466**	0.7847	**0.8573**	**0.9358**	**0.9514**	0.8305	**0.8854**	**0.9519**	**0.9708**	0.8567	**0.9094**	**0.9641**

TABLE 3
Performances of Fraud Detection on Telecom Datasets of Different Fraudulent Types

Methods	Recording				Genuine			
	P	R	F	AUC	P	R	F	AUC
LSTM	0.5711	0.8043	0.6296	0.7626	0.5493	0.6945	0.5787	0.7937
GRU	0.4562	**0.9174**	0.5392	0.7130	0.4533	0.7764	0.5154	0.7105
RMTPP	0.3532	0.7050	0.4053	0.6433	0.4028	0.2166	0.2719	0.4924
ERNN	0.2083	***0.8333***	0.3333	0.5991	0.5000	0.3333	0.4000	0.6318
DeepHawkes	0.7298	0.8373	0.7454	0.4046	0.7580	**0.8986**	0.7805	0.6060
NeuralHawkes	0.9181	0.6796	0.7762	***0.9022***	0.9239	0.6845	***0.7836***	***0.9063***
HAInt-LSTM	0.8410	0.6590	0.7388	0.8272	0.8085	0.7275	0.7635	0.8129
NHA-LSTM	0.8530	0.7376	0.7885	0.8279	0.8435	0.6688	0.7460	0.8143
THP	0.8334	0.8077	0.8200	0.8185	0.8466	0.4702	0.6044	0.7915
AnomRank	***0.9193***	0.6966	0.7874	0.8410	0.8980	0.6544	0.7560	0.8065
Omnianomaly	0.9310	0.7297	***0.8182***	0.9204	***0.9167***	***0.7875***	0.8462	0.9197
HESM (ours)	**0.9333**	0.8130	**0.8658**	**0.9559**	**0.9321**	0.8602	**0.8927**	**0.9495**

Note: Training proportion is 90%.

memory cell and three gates, namely, the input gate, forget gate and output gate, which help it to effectively model sequential dependencies.

Gated Recurrent Unit (GRU) [50]: This method is another widely used variant of RNNs. It replaces the forget gate and input gate with a single update gate and passes the hidden state directly to the next unit, while LSTM uses the output gate to wrap the hidden state. It also has a reset gate to control the information from the previous moment.

Recurrent Marked Temporal Point Processes (RMTPP) [51]: This method simultaneously models the event time stamps and the markers, views the intensity function of a temporal point process as a nonlinear function of the history, and uses a recurrent neural network to automatically learn representations of the influences of historical events.

Modeling the Intensity of Point Processes via Recurrent Neural Networks (ERNN) [52]: This method models the background by a RNN whose units are aligned with time series indexes, while the historical effect is modeled by another RNN whose units are aligned with asynchronous events to capture the long-range dynamics of the data. The whole model, with event type and timestamp prediction output layers, can be trained in an end-to-end manner.

Continuous-time LSTM (NeuralHawkes) [26]: This generative model allows past events to influence future events in complex and realistic ways by conditioning future event intensities on the hidden state of a recurrent neural network that has consumed the stream of past events.

DeepHawkes [53]: This model leverages end-to-end deep learning to make analogies between the interpretable factors of the Hawkes process.

HAInt-LSTM [54]: This model forms representations of the behavioral sequences necessary for fraud detection. In designing the interaction module, it only takes the original IDs of the source and target users as input and concatenates them with the feature vectors as the output of the model.

NHA-LSTM [47]: This model augments the traditional LSTM with a modified forget gate, where the interval time is the duration between consecutive time steps, and designs a self-historical attention mechanism to allow for long-term dependencies. In addition, an enhanced network embedding method, FraudWalk, is considered to construct embeddings for the nodes in the interaction network with regard to higher-order interactions and particular time constraints for revealing potential group fraud.

THP [35]: The model leverages the self-attention mechanism to capture long-term dependencies and meanwhile enjoys computational efficiency.

AnomRank [43]: The model uses a two pronged approach defining two novel metrics for anomalousness. Each metric tracks the derivatives of its own version of a 'node score' (or node importance) function, which can detect two different types of anomalies: sudden weight changes along an edge, and sudden structural changes to the graph.

Omnianomaly [15]: The model uses a stochastic recurrent neural network for multivariate time series anomaly detection. Its core idea is to capture the normal patterns of multivariate time series by learning their robust representations, and use the reconstruction probabilities to determine anomalies.

5.1.3 Parameter Settings

In the experiments, we set the size of all the hidden state vectors, also can be referred to as embedding size, to 256, and set the sequence length for prediction to 100 for all the neural-based methods. For our method HESM, we set the learning rate to 0.001, and apply gradient clipping during the training process with the value of the gradient clip being 0.05. Other parameters for each baseline method are tuned to achieve the best performance. For the sake of fair comparisons, we separate a common validation set from the whole samples and the parameters of all the methods are tuned in the validation set. We have conducted experiments to show the influences of the parameters including the embedding size, the sequence length, which are presented in the supplemental materials.

5.2 Experimental Results

5.2.1 Performance Comparison on Complete Dataset

For the large-scale, complete telecom dataset, we vary the size of the training set among $30\%, 60\%, 90\%$ of the total samples, and randomly leave out $20\%, 20\%, 5\%$ from the data as the validation set respectively, then the remaining data is used as the test set. After training the model as presented in Algorithm 1, we apply the trained model on test set to obtain the prediction loss of each individual caller.

In order to determine whether a caller is a fraudster or not, we need to set a threshold value for the prediction loss and the threshold value that yields the largest F-measure in the validation set can be used as the classification threshold value for the test dataset. We consider the callers whose prediction loss is greater than the threshold value are classified as fraudsters. Based on the threshold value, HESM classifies the callers and reports the classification performances.

Table 2 shows the results, with the best performances in **bold**, the second best underlined and the third best in ***bold italic***. As can be seen, our model achieves the best performances with all training set sizes in terms of the general metrics F-measure and AUC. In addition, Omnianomaly and NeuralHawkes in some cases are the second best or the third best baseline methods, respectively, with classification performances comparable to HESM. The performances of the rest baseline methods, however, are poorer than the above three methods, although AnomRank, HAInt-LSTM, NHA-LSTM and THP show relatively comparable performances with Omnianomaly and Neuralhawkes. As the best-performed method, HESM also demonstrates its robustness to different training set sizes. Indeed, HESM performs almost the same on 30%, 60% and 90% training sets in terms of AUC, and the gap between 30% and 60% training sets in terms of F-measure is merely 3%.

The excellent performance of Omnianomaly is not unusual. It captures complex temporal patterns of multivariate time series, which are right the key tasks of HESM in historical influence learning as well as the sequence correlation learning. The success of NeuralHawkes is worth noting. It indicates that the time dependency information and historical influences are necessary for modeling behavioral patterns, which indeed supports the historical influence learning of HESM. Finally, it is interesting that LSTM and GRU achieve the near-to-perfect recall values in 30% and 60% cases, which however are at the cost of low precision values. This, in turn, illustrates why we design HESM and introduce the Hawkes process to the LSTM model.

TABLE 4
Ablation Study Results

Methods	Precision	Recall	F-measure	AUC
HESM	**0.9708**	**0.8567**	**0.9094**	**0.9641**
w/o Hawkes	0.8963	0.8056	0.8360	0.8976
w/o corr	0.9000	0.7212	0.8004	0.9154
w/o att	0.9330	0.8136	0.8684	0.9417

5.2.2 Performance Comparison on Special-type Datasets

To validate the effectiveness of detecting different types of telecom frauds, we also conduct fraud detection experiments by employing all the competitive methods on the *Recording* and *Genuine* datasets, with 90% samples as training set, 5% samples as the validation set and the remaining as the test set. Table 3 shows the results.

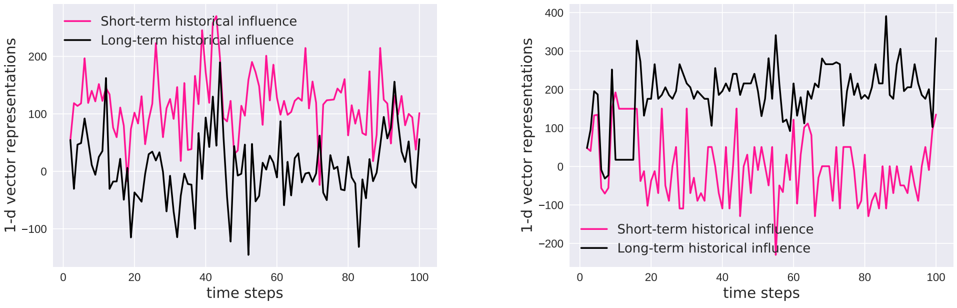
As shown in Table 3, with the best performances in bold, the second best underlined and the third best in bold italic, HESM still consistently outperforms all the baselines on the two datasets, which well demonstrates the effectiveness of HESM in detecting special types of telecom frauds. Omnianomaly and NeuralHawkes in most cases are the second best and the third best baseline methods, respectively. These results indeed agree with that of the complete dataset and indicate the robustness of HESM in different fraud detection scenarios.

It is also interesting that while the three baseline models LSTM, GRU and RMTPP show much poorer performances on the *Genuine* dataset than on the *Recording* dataset, the rest models generate results of much comparability on the two datasets. It is generally believed that genuine callers are more like normal callers and thus are harder to be caught than recording callers. But the introduction of the Hawkes process and the ability of historical influences modelling and sequence correlation modelling make HESM a clever detector of crafty genuine callers. The other models like NeuralHawkes, HAInt-LSTM, NHA-LSTM and Omnianomaly, share more or less commonplaces with HESM and thus also perform well on the *Genuine* dataset.

5.2.3 Ablation Study of Modelling Components

Our model has three major components, *i.e.*, historical influence learning with the Hawkes process, historical attention mechanism, and sequence correlation learning. We here conduct an ablation study on the complete telecom dataset by removing these components respectively. We have 90% samples as the training set, 5% samples as the validation set and the remaining as the test set. HESM represents the full model with all the components, $\text{HESM}_{\text{w/o Hwakes}}$ removes the Hawkes-enhanced neural unit, $\text{HESM}_{\text{w/o att}}$ removes the historical attentions, and $\text{HESM}_{\text{w/o corr}}$ removes the sequence correlations.

As shown in Table 4, with the best performances in bold, we can easily see that HESM with all the components performs the best. Among the other models with partial components, the performance of $\text{HESM}_{\text{w/o Hwakes}}$ without the Hawkes process degrades the most, suggesting that the historical influences captured by the Hawkes process is the



(a) Fraudsters (b) Normal Users

Fig. 4. Illustration of Repetition Mode

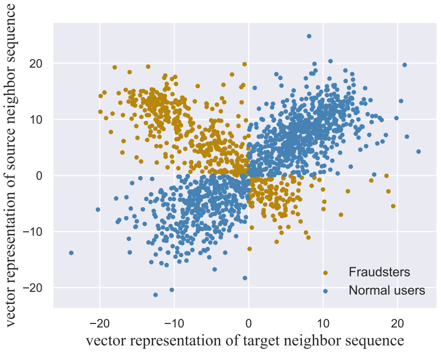


Fig. 5. Illustration of the Correlation Mode

most critical module in HESM. In contrast, the performance of HESM$_{\text{w/o att}}$ is the closest to that of the full model, illustrating that the historical attention mechanism contributes less than the other two components to the success of HESM.

5.2.4 Effectiveness Analysis of Modelling Components

To better understand how different modelling components can enhance sequence learning of HESM, we further analyze the effectiveness of these components individually.

(1) Historical influence learning. In our model, we integrate the Hawkes process into the LSTM to capture the historical neighbors' influences on the occurrence of the current neighbor. To validate the effectiveness of learning historical influence, we extract the short-term historical influence $\overline{\boldsymbol{C}}_{t-1}$ and long-term historical influence $\boldsymbol{g}_h$ in Eq.(9), and then employ the tSNE method [55] to project the vectors to 1-D space.

In Fig.4, the horizontal axis represents the time steps, and the vertical axis denotes the projected values of the vectors. As witnessed in Fig.4, the projected values of short-term historical influence of fraudsters are generally higher than that of long-term historical influence. This is due to the fact that fraudsters maintain short-term contacts and frequently switch between calling targets to harvest more victims, which exhibits short-term historical influence patterns. In contrast, the projected values of long-term historical influence of normal users are higher than that of short-term historical influence, since normal users have long-term and stable contacts with others and thus exhibit the significant long-term historical influence patterns. These results validate the effectiveness of learning the historical influence.

(2) Correlation learning. As observed in the exploratory analysis, the correlation mode between the target neighbor sequences and source neighbor sequences plays a vital role in distinguishing fraudsters from normal callers. To demonstrate whether we have correctly learned the correlations between the neighbor sequences, we extract the hidden-state vectors of the target and the source neighbors sequences, respectively, and then employ the tSNE method to project the vectors to 1-D space.

In Fig.5, the horizontal axis represents the projected value of hidden state vector of the target neighbor sequences, and the vertical axis denotes the projected value of hidden state vector of the source neighbor sequences, and the circle points denote the projected values at each time step. As witnessed in Fig.5, normal users show obvious positive correlations, while in contrast, fraudsters show negative correlations. This proves that we can effectively distinguish fraudsters from normal users based on their correlation modes.

(3) Historical attention mechanism. To evaluate the effectiveness of the historical attention mechanism with regard to fraud detection, we examine the performances of HESM and HESM$_{\text{w/o att}}$ (without the attention mechanism) by varying the sequence length among $50, 100, 150, 200$ during the training procedure. In Fig.6, we can see that HESM not only beats HESM$_{\text{w/o att}}$ with varying sequence lengths in terms of precision and F-measure, but also shows much more stable performances than HESM$_{\text{w/o att}}$. This indicates that the attention mechanism indeed can help our method achieve improved performances by learning the enhanced long-term historical neighbor influences in the data.

We further provide interpretability to the learned sequence representations using the attention mechanism. To that end, we randomly sample 30 callers from the sets of normal users and fraudsters, respectively, and draw the attention weights with a heatmap. Each row in Fig.7 represents the attention weights at each time step, where a darker color in the grid denotes a larger attention weight and vice versa. Generally, in the same scale of color range, we can find that the attention weights differ tremendously for normal users and fraudsters. Most grids of the normal users are light blue except for a small amount being deep blue at several time steps. This indicates that the behaviors of the normal users can be mainly influenced by several historical neighbors; meanwhile, different normal users have varied

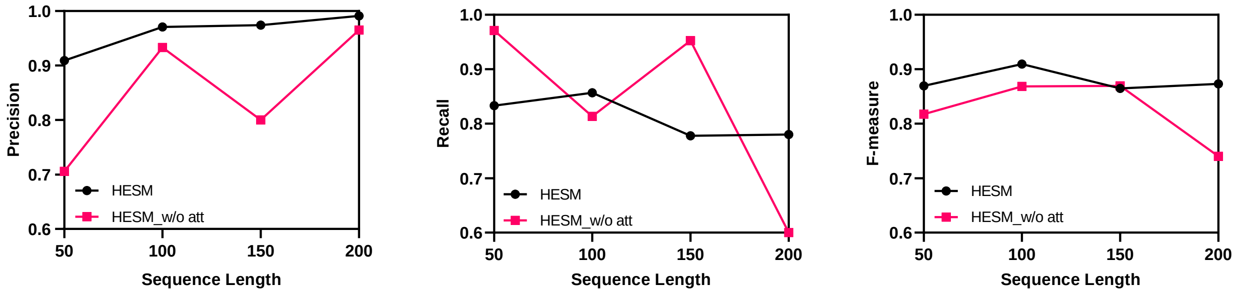
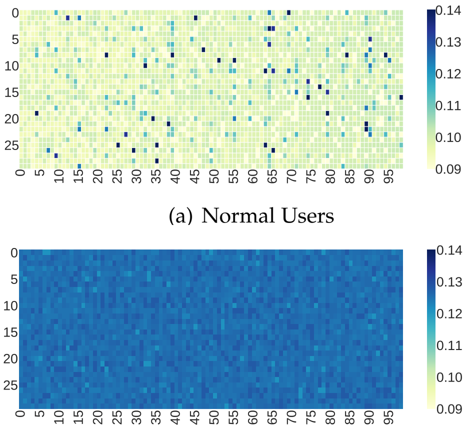


Fig. 6. Performances with or without Historical Attentions given Varying Sequence Lengths



(a) Normal Users

(b) Fraudsters

Fig. 7. Interpretability of Historical Attentions

cyclical characteristics of attention weights, showing that they generally have distinctive historical influence patterns. On the contrary, the attention weights for the fraudsters concentrate on a narrow scale range without significant differences, which corresponds to our previous observations that fraudsters are less likely to maintain long-term contacts.

6 Conclusion

Fraud detection is generally a challenging task since fraudsters usually hide their illegal behaviors in a large number of normal behaviors. In this paper, we argued that fraudulent behaviors can be manifested in a temporal bipartite network where both the consecutive and interactive behaviors are considered. Along this line, we conducted an exploratory analysis on real-world telecom data and discovered two types of distinguishing behaviors for frauds, namely the correlation mode and repetition mode. Inspired by these observations, we proposed a novel Hawkes-enhanced sequence model (HESM) to learn the sequential patterns from the neighbor sequences for the purpose of fraud detection. HESM integrated the Hawkes process into the neural units of traditional LSTM by designing a decay gate to allow for the long-term historical influences, and a historical attention mechanism was developed to account for the repetition mode. In addition, the correlations of different types of neighbor sequences were modeled with a correlation gate. Extensive experiments on real-world telecom datasets demonstrated the superiority of our method over the state-of-the-art sequence-based methods, and the effectiveness of the major modeling components have been validated.

References

[1] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[2] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] T. Ergen, A. Mirza, and S. Kozat, "Unsupervised and semi-supervised anomaly detection with lstm neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, 10 2017.

[4] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos, "Copycatch: Stopping group attacks by spotting lockstep behavior in social networks," in *Proceedings of the 22Nd International Conference on World Wide Web*, ser. WWW '13. New York, NY, USA: ACM, 2013, pp. 119–130.

[5] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Inferring strange behavior from connectivity pattern in social networks," vol. 8443, 05 2014.

[6] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "Fraudar: Bounding graph fraud in the face of camouflage," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 895–904.

[7] H. Lin, G. Liu, J. Wu, Y. Zuo, X. Wan, and H. Li, "Fraud detection in dynamic interaction network," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 10, pp. 1936–1950, 2020.

[8] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra, "Spotlight: Detecting anomalies in streaming graphs," in *the 24th ACM SIGKDD International Conference*, 2018.

[9] M. Hu, G. Xu, C. Ma, and M. Daneshmand, "Detecting review spammer groups in dynamic review networks," in *the ACM Turing Celebration Conference - China*, 2019.

[10] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Transactions on Knowledge & Data Engineering*, vol. 24, no. 5, pp. 823–839, 2012.

[11] V. Chandola, V. Mithal, and V. Kumar, "Comparative evaluation of anomaly detection techniques for se-

quence data," in *Eighth IEEE International Conference on Data Mining*, 2008.

[12] Shebuti, Rayana, Leman, and Akoglu, "Less is more: Building selective anomaly ensembles," *Acm Transactions on Knowledge Discovery from Data*, 2016.

[13] I. Melnyk, A. Banerjee, B. Matthews, and N. Oza, "Semi-markov switching vector autoregressive model-based anomaly detection in aviation systems," 08 2016, pp. 1065–1074.

[14] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19*, 2019.

[15] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 28282837.

[16] B. Branco, P. Abreu, A. S. Gomes, M. S. C. Almeida, J. a. T. Ascensão, and P. Bizarro, "Interleaved sequence rnns for fraud detection," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20. Association for Computing Machinery, 2020, p. 31013109.

[17] Y. Zhu, D. Xi, B. Song, F. Zhuang, S. Chen, X. Gu, and Q. He, "Modeling users behavior sequences with hierarchical explainable network for cross-domain fraud detection," in *Proceedings of The Web Conference 2020*, ser. WWW '20. Association for Computing Machinery, 2020, p. 928938.

[18] D. J. Daley and D. Vere-Jones, *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

[19] M. Farajtabar, Y. Wang, M. Gomez-Rodriguez, S. Li, H. Zha, and L. Song, "Coevolve: A joint point process model for information diffusion and network evolution," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1305–1353, 2017.

[20] Y. Wang, N. Du, R. Trivedi, and L. Song, "Coevolutionary latent feature processes for continuous-time user-item interactions," in *Advances in Neural Information Processing Systems*, 2016, pp. 4547–4555.

[21] A. G. Hawkes, "Spectra of some self-exciting and mutually exciting point processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971.

[22] E. Choi, N. Du, R. Chen, L. Song, and J. Sun, "Constructing disease network and temporal progression model via context-sensitive hawkes process," in *2015 IEEE International Conference on Data Mining*. IEEE, 2015, pp. 721–726.

[23] N. Du, M. Farajtabar, A. Ahmed, A. J. Smola, and L. Song, "Dirichlet-hawkes processes with applications to clustering continuous-time document streams," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 219–228.

[24] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1555–1564.

[25] H. Jing and A. J. Smola, "Neural survival recommender," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 515–524.

[26] H. Mei and J. M. Eisner, "The neural hawkes process: A neurally self-modulating multivariate point process," in *Advances in Neural Information Processing Systems*, 2017, pp. 6754–6764.

[27] L. Li, H. Deng, A. Dong, Y. Chang, and H. Zha, "Identifying and labeling search tasks via query-based hawkes processes," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 731–740.

[28] T. Bai, L. Zou, W. Zhao, P. Du, W. Liu, J. Nie, and J. Wen, "Ctrec: A long-short demands evolution model for continuous-time recommendation," in *the 42nd International ACM SIGIR Conference*, 2019.

[29] S. Xiao, J. Yan, M. Farajtabar, L. Song, X. Yang, and H. Zha, "Joint modeling of event sequence and time series with attentional twin recurrent neural networks," *arXiv preprint arXiv:1703.08524*, 2017.

[30] G. Yang, Y. Cai, and C. K. Reddy, "Recurrent spatio-temporal point process for check-in time prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ser. CIKM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 22032211.

[31] A. Trkmen, Y. Wang, and A. J. Smola, *FastPoint: Scalable Deep Point Processes*. Machine Learning and Knowledge Discovery in Databases, 2020.

[32] R. Cai, X. Bai, Z. Wang, Y. Shi, P. Sondhi, and H. Wang, "Modeling sequential online interactive behaviors with temporal point process," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ser. CIKM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 873882.

[33] M. Okawa, T. Iwata, T. Kurashima, Y. Tanaka, H. Toda, and N. Ueda, "Deep mixture point processes: Spatio-temporal event prediction with rich contextual information," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 373383.

[34] B. Vassøy, M. Ruocco, E. de Souza da Silva, and E. Aune, "Time is of the essence: A joint hierarchical rnn and point process model for time and item predictions," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, ser. WSDM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 591599.

[35] S. Zuo, H. Jiang, Z. Li, T. Zhao, and H. Zha, "Transformer hawkes process," in *the 37th International Conference on Machine Learning*, 2020.

[36] L. Akoglu, M. McGlohon, and C. Faloutsos, "Oddball: Spotting anomalies in weighted graphs," 07 2010, pp. 410–421.

[37] N. Shah, A. Beutel, B. Hooi, L. Akoglu, S. Günnemann, D. Makhija, M. Kumar, and C. Faloutsos, "Edgecen-

tric: Anomaly detection in edge-attributed networks," *CoRR*, vol. abs/1510.05544, 2015.

[38] K. Shin, B. Hooi, and C. Faloutsos, "Fast, accurate, and flexible algorithms for dense subtensor mining," *ACM Transactions on Knowledge Discovery from Data*, vol. 12, no. 3, pp. 1–30, 2018.

[39] J. Sun, C. Faloutsos, S. Papadimitriou, and P. Yu, "Graphscope: Parameter-free mining of large time-evolving graphs," 01 2007, pp. 687–696.

[40] S. Liu, B. Hooi, and C. Faloutsos, "Holoscope: Topology-and-spike aware fraud detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: ACM, 2017, pp. 1539–1548.

[41] C. Chelmis and R. Dani, "Assist: Automatic summarization of significant structural changes in large temporal graphs," in *the 2017 ACM*, 2017.

[42] K. Shin, B. Hooi, J. Kim, and C. Faloutsos, "Densealert: Incremental dense-subtensor detection in tensor streams," 2017.

[43] M. Yoon, B. Hooi, K. Shin, and C. Faloutsos, "Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 647657.

[44] S. Bhatia, B. Hooi, M. Yoon, K. Shin, and C. Faloutsos, "Midas: Microcluster-based detector of anomalies in edge streams," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 3242–3249, 2020.

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

[46] B. I. Godoy, V. Solo, J. Min, and S. A. Pasha, "Local likelihood estimation of time-variant hawkes models," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[47] G. Liu, J. Guo, Y. Zuo, J. Wu, and R. yong Guo, "Fraud detection via behavioral sequence embedding," *Knowledge & Information Systems*, no. 2, 2020.

[48] P. T. D. Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[49] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.

[50] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *Computer Science*, 2014.

[51] N. Du, H. Dai, R. Trivedi, U. Upadhyay, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*, 2016.

[52] S. Xiao, J. Yan, S. M. Chu, X. Yang, and H. Zha, "Modeling the intensity function of point process via recurrent neural networks," 2017.

[53] Q. Cao, H. Shen, K. Cen, W. Ouyang, and X. Cheng, "Deephawkes: Bridging the gap between prediction and understanding of information cascades," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 11491158. [Online]. Available: https://doi.org/10.1145/3132847.3132973

[54] J. Guo, G. Liu, Y. Zuo, and J. Wu, "Learning sequential behavior representations for fraud detection," in *2018 IEEE International Conference on Data Mining (ICDM)*, 2018.

[55] G. E. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2, pp. 2579–2605, 2008.

Yan Jiang is currently working toward the Ph.D. degree in the School of Economics and Management at Beihang University, China. Her research interests generally lie in the areas of data mining and machine learning, with special interests in anomaly detection.

Guannan Liu is currently an Associate Professor in the Department of Information Systems with Beihang University, Beijing, China. He received the Ph.D. degree from Tsinghua University, China. His research interests include data mining, business intelligence, and anomaly detection. His work has been published in the journal of IEEE TKDE, ACM TKDD, ACM TIST, Decision Support Systems, etc., and also in the conference proceedings such as KDD, ICDM, SDM etc.

Junjie Wu received his Ph.D. degree in Management Science and Engineering from Tsinghua University. He is currently a full Professor in Information Systems Department of Beihang University, the director of the Research Center for Data Intelligence (DIG), and the director of the Institute of Artificial Intelligence for Management. His general area of research is data mining and complex networks. He is the recipient of NSFC Distinguished Young Scholars award and MOE Changjiang Young Scholars award in China.

Hao Lin received his bachelor and Ph.D. degree from Beihang University, Beijing, China, in 2013 and 2020 respectively. He is currently a postdoc researcher in Department of Informatics, Technical University of Munich. His research interests generally lie in the areas of data mining and machine learning, with special interests in temporal data analysis and heterogeneous data fusion. His work has been published in refereed journals and conference proceedings, including IEEE TKDE, ACM TOIS, and AAAI.