MINT: Detecting Fraudulent Behaviors from Time-series Relational Data

Fei Xiao
Shopee Singapore
National University of Singapore
fei.xiao@shopee.com

Yuncheng Wu*
National University of Singapore
dcswuyu@nus.edu.sg

Meihui Zhang
Beijing Institute of Technology
meihui_zhang@bit.edu.cn

Gang Chen
Zhejiang University
cg@zju.edu.cn

Beng Chin Ooi
National University of Singapore
ooibc@comp.nus.edu.sg

ABSTRACT

The e-commerce platforms, such as Shopee, have accumulated a huge volume of time-series relational data, which contains useful information on differentiating fraud users from benign users. Existing fraud behavior detection approaches typically model the time-series data with a vanilla Recurrent Neural Network (RNN) or combine the whole sequence as a single intention without considering the temporal behavioral patterns, row-level interactions, and different view intentions. In this paper, we present MINT, a **M**ulti-view row-**IN**teractive **T**ime-aware framework to detect fraudulent behaviors from time-series structured data. The key idea of MINT is to build a *time-aware behavior graph* for each user's time-series relational data with each row represented as an *action node*. We utilize the user's temporal information to construct three different graph convolutional matrices for hierarchically learning the user's intentions from different views, that is, short-term, medium-term, and long-term intentions. To capture more meaningful row-level interactions and alleviate the over-smoothing issue in a vanilla time-aware behavior graph, we propose a novel *gated neighbor interaction* mechanism to calibrate the aggregated information by each action node. Since the receptive fields of the three graph convolutional layers are designed to grow nearly exponentially, our MINT requires many fewer layers than traditional deep graph neural networks (GNNs) to capture multi-hop neighboring information, and avoids recurrent feedforward propagation, thus leading to higher training efficiency and scalability. Our extensive experiments on the large-scale e-commerce datasets from Shopee with up to 4.6 billion records and a public dataset from Amazon show that MINT achieves superior performance over 10 state-of-the-art models and provides better interpretability and scalability.

PVLDB Reference Format:
Fei Xiao, Yuncheng Wu, Meihui Zhang, Gang Chen, and Beng Chin Ooi. MINT: Detecting Fraudulent Behaviors from Time-series Relational Data. PVLDB, 16(12): 3610 - 3623, 2023.
doi:10.14778/3611540.3611551

*Yuncheng Wu is the corresponding author.

Proceedings of the VLDB Endowment, Vol. 16, No. 12 ISSN 2150-8097.
doi:10.14778/3611540.3611551

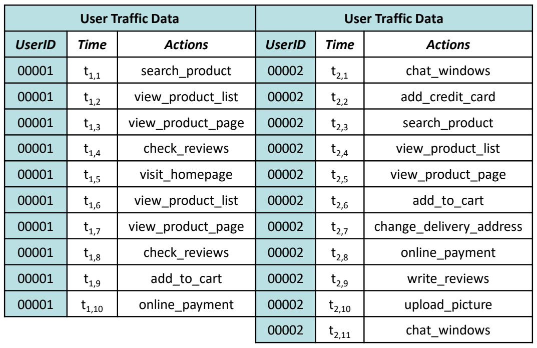
1 INTRODUCTION

With the flourishing development of online shopping, various forms of e-commerce fraud, such as account takeover, promotion abuse, fake review, and malicious transaction have become threats to e-commerce platforms and customers [7, 24, 28, 31, 44, 45]. A recent research report estimated that the cumulative merchant losses caused by online payment fraud globally will exceed \$343 billion between 2023 and 2027[1]. Clearly, the detection of fraudulent activities is crucial to the healthy development of e-commerce platforms and the digital economy.

Users' fraudulent behaviors are reflected in their daily activities [5, 25, 26, 52]. Figure 1 shows an example of two users' daily actions[2] in chronological order. User_00001 compares different items and their reviews to decide on the best product to purchase, which is a common habit exhibited by most genuine shoppers. By contrast, User_00002 has some unusual actions, that is, searching and purchasing the targeted product without comparing it to other items, exhibiting a suspicious behavioral pattern. To combat fraudulent activities, various deep learning-based frameworks [2, 5, 25, 26, 49, 52] have been proposed to exploit a user's behavioral sequence as they can capture the user's intention for fraud detection. For instance, [25, 26] adopt a tree structure to reorganize the sequential actions and use Long Short-Term Memory (LSTM) networks to learn the intentions of different branches. The fraud prediction is made based on the user's final intention, which is a fusion of the branch intentions. However, these solutions have two limitations. First, direct application of LSTM and one-sided view learning method cannot effectively exploit implicit user representations in complex scenarios [15, 36]. For example, fraudsters can imitate the behaviors of benign users by performing similar actions in the same chronological order, and therefore bypass the anti-fraud systems. Second, they depend on experts' domain knowledge to construct the tree structure, which requires extensive effort and is inflexible. For example, they assume that all of the intentions start from "visit-homepage". This may mistakenly divide a branch containing "visit-homepage" into many sub-branches, leading to inaccurate branch intentions.

To address these limitations, we propose a novel **M**ulti-view row-**IN**teractive **T**ime-aware (MINT) fraud detection framework. The key idea is to incorporate the temporal information between behavioral

[1]https://www.juniperresearch.com/pressreleases/online-payment-fraud-losses-to-exceed-343bn
[2]In the following sections, "action" represents users' behaviors and it includes "sign-in", "change-delivery-address", "add-credit-card", "online-payment" and so on.



User Traffic Data			User Traffic Data		
UserID	*Time*	*Actions*	*UserID*	*Time*	*Actions*
00001	$t_{1,1}$	search_product	00002	$t_{2,1}$	chat_windows
00001	$t_{1,2}$	view_product_list	00002	$t_{2,2}$	add_credit_card
00001	$t_{1,3}$	view_product_page	00002	$t_{2,3}$	search_product
00001	$t_{1,4}$	check_reviews	00002	$t_{2,4}$	view_product_list
00001	$t_{1,5}$	visit_homepage	00002	$t_{2,5}$	view_product_page
00001	$t_{1,6}$	view_product_list	00002	$t_{2,6}$	add_to_cart
00001	$t_{1,7}$	view_product_page	00002	$t_{2,7}$	change_delivery_address
00001	$t_{1,8}$	check_reviews	00002	$t_{2,8}$	online_payment
00001	$t_{1,9}$	add_to_cart	00002	$t_{2,9}$	write_reviews
00001	$t_{1,10}$	online_payment	00002	$t_{2,10}$	upload_picture
			00002	$t_{2,11}$	chat_windows

Figure 1: An example of two users' time-series relational data from the real-world dataset.

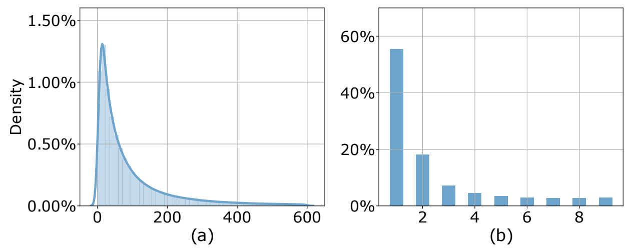


Figure 2: (a) Time interval distribution for consecutive actions (clipped to 600 seconds). (b) Distribution of time interval between the first-time click and the payment for one product (clipped to 9 days).

actions into fraud detection, so as to capture each user's intentions more accurately. Intuitively, two actions with smaller time intervals should have a greater impact on each other. To better illustrate the temporal patterns, we analyze users' actions in a public real-world dataset from Taobao [50] and show the time interval distribution for consecutive actions in Figure 2(a). We observe that the time interval varies from a few seconds to minutes, which is a significant feature for detecting fraudulent activities. Take the account takeover fraud as an example, cybercriminals may try to conceal their activity by performing multiple other actions in between "change-passwords" and "online-payment". This would cause the anti-fraud systems in [25, 26] to mistakenly identify the sequence as normal activity. However, cybercriminals typically need to change passwords and make a purchase in a short time, making the temporal behavioral pattern critical for fraud detection.

Specifically, MINT has four novel features. First, we model each user's behavioral sequence as a *time-aware behavior graph*, in which each row is represented by an action node, and the edge weights are determined by the time intervals and corresponding hyper-parameters. To the best of our knowledge, MINT is the first work that utilizes users' time-series behaviors with graph topology for fraud detection. Second, we design a *multi-view scheme* to capture users' intentions from short-term, medium-term, and long-term perspectives. This is motivated by the observation in Figure 2(b) that users may take several days to search for and purchase a product, highlighting the need to model intentions from different time periods to identify fraud users. For example, to detect Brushing fraud[3], we need to learn the user's long-term intentions since the smart Brushing process involves multiple actions in days [24, 26]. Third, we devise a *gated neighbor interaction* mechanism to calibrate the aggregated information from neighboring nodes. For instance, there are two "view-product-list" actions in User_00001's behavioral data. When we learn the local intention of the first "view-product-list" node, we can aggregate more information from the node "search-product" as this sub-sequence indicates the user's potential purpose of product purchase. However, the aggregation of "visit-homepage" to the second "view-product-list" node should be calibrated to reduce the impact of "visit-homepage" since it is a very common action. Fourth, the proposed framework does not involve extensive feature engineering and domain knowledge to build the behavior graph, and thus can be easily generalized to various relational data analytics.

We present an overview of Shopee's MINT fraud detection framework in Figure 3. The pre-processing module consists of a graph convolutional matrix constructor and a node embeddings constructor, which builds a time-aware behavior graph for each user and represents one time-stamped record with a node. The behavior graph has three different views, and in each view, the node features are the same but the edges are different. In the multi-view graph convolution module, we exploit the information of the three different views to learn the user's various intentions. In the prediction module, we fuse the user's intentions and calculate the probability that the user is classified as a fraudster. Moreover, we develop a fraudulent behaviors extractor to find the suspicious actions and the sub-sequences that contribute to their classification as fraud.

As Shopee's business and data rapidly grow in Southeast Asia and Latin America, it needs an anti-fraud system to efficiently identify fraudulent user behavior. We duly develop a system based on MINT to collect user traffic data through Kafka[4] and efficiently generate a risk score for each user. In summary, we make the following contributions:

- We develop a novel fraud detection framework, which utilizes the temporal information between actions in the behavioral sequence to build a time-aware behavior graph with three different views for better detecting users' fraudulent behaviors.
- We propose a multi-view graph convolutional network, which can effectively capture the user's short-term and long-term intentions, and achieve higher training efficiency and scalability than conventional graph neural networks.
- We design a gated neighbor interaction mechanism to calibrate the aggregation information from neighboring nodes and learn sophisticated cross-action representation.
- We conduct extensive experiments on real-world data from Shopee[5] and Amazon[6]. The results confirm that MINT can learn both the global and local intentions of users, and it consistently outperforms 10 state-of-the-art baselines and provides good interpretability, demonstrating MINT's superior performance.

[3]https://en.wikipedia.org/wiki/Brushing_(e-commerce)

[4]https://kafka.apache.org/
[5]https://en.wikipedia.org/wiki/Shopee
[6]https://en.wikipedia.org/wiki/Amazon_(company)

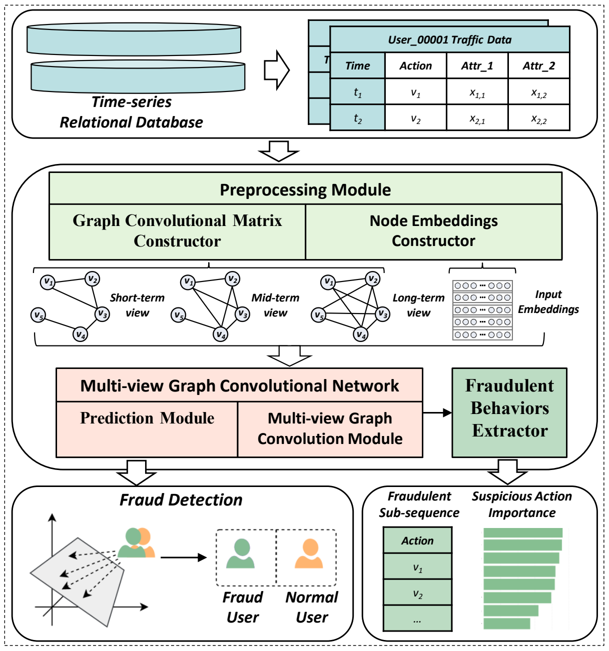


Figure 3: Overview of the MINT fraud detection framework.

Table 1: Summary of notations.

Notation	Description
t	The timestamp of one record
x_i	Numerical or categorical features
$\mathbf{x}$	The data in each record
$\mathcal{V}, v$	Set of actions and an action
$\mathcal{U}, u$	Set of users and a user
n	Length of behavioral sequence, same as $\lvert V \rvert$
m	Number of attributes in the table
ϵ	Edge weight threshold in behavior graph
ρ	A hyper-parameter to tune values of $\mathbf{A}$
$\mathcal{T}$	Logical tables
V	A behavior sequence or the vertices for a graph
G	Time-aware behavioral graph
E	All edges for a behavioral graph
$\mathcal{N}(v_i)$	Neighboring nodes of v_i
$\mathbf{h}_{v_i}^{(l)}$	Embeddings of action node v_i in l-th layer
$\mathbf{H}^{(l)}$	Embeddings of behavioral sequence in l-th layer
$\mathbf{A}^{(l)}$	Normalized graph convolutional matrix
$\mathbf{z}$	User representation
y	A fraud label

This paper is organized as follows. Section 2 introduces the preliminaries and formulates our fraud detection problem. Section 3 presents MINT with its modules and optimization schemes. The experimental evaluation of MINT is described in Section 4. Section 5 reviews related works. We conclude in Section 6.

2 PROBLEM DEFINITION

In this section, we first introduce the preliminaries of time-series relational data and formulate the problem statement of fraud detection on it. Then, we present two relevant techniques that are central to our proposed framework MINT: the time-aware behavior graph and graph convolutional matrix. The notations used in the remaining of this paper are summarized in Table 1.

Time Series Relational Data. Time series relational data is typically stored as tables in a relational database (RDBMS). Various classification and prediction analytics have been conducted over the relational data [6, 8, 23, 27, 29, 30, 33, 34], in which each record is a time-stamped action. We consider the behavioral structured data as a set of tables $\mathcal{T}$ of rows (records) and columns (attributes) with the corresponding timestamp. More specifically, each record is denoted as $\mathbf{x} = (v, t, x_1, x_2, ..., x_{m-2})$, in which v represents the time-stamped action, t is the timestamp and x_i represents other attributes, such as deviceID and session duration.

Fraud Detection over Time Series Relational Data. In a historical behavioral database that contains fraud labels, each row consists of a user's time-stamped actions and other numerical and categorical features. Given a user $u \in \mathcal{U}$, its behavioral data contains a behavioral sequence $V = \{v_0, v_1, \cdots, v_{n-1}\}$ and the corresponding attributes, where $v_i \in \mathcal{V}$ represents user's action, and n is the length of the sequence. The fraud detection task over the database is to identify whether the user conducts fraudulent activities.

A user's behavioral sequence is generally presented in chronological order, and the sequence pattern of a fraudulent user usually demonstrates abnormal characteristics compared to the vast majority of benign users. The aim is to determine whether the users have suspicious behaviors given the users' historical sequential behavior data and the task can be formulated as a binary classification task.

Time-aware Behavior Graph. Given a user with its time-stamped behavioral sequence $V = \{v_0, v_1, \cdots, v_{n-1}\}$ and the corresponding attributes, its time-aware behavior graph is $G = \{V, E, \mathbf{A}\}$, where V represents the action nodes, E is the edges, $\mathbf{A} \in \mathbb{R}^{n \times n}$ $(0 \leq \mathbf{A}_{i,j} \leq 1)$ is the graph convolutional matrix. Each node v_i in the graph represents one record and each edge $< v_i, v_j >$ has a weight inversely proportional to the time difference between v_i and v_j.

Graph Convolutional Matrix. The primary concept behind the traditional graph convolutional networks (GCN) [20] is to capture the relationships between nodes in a graph by leveraging the information from their neighboring nodes. Specifically, GCN computes the weighted average of node features for all neighboring nodes, including the node itself. The weight matrix is coined as a graph convolutional matrix, and in the context of GCN, it specifically refers to the symmetric normalized adjacency matrix.

In a time-aware behavior graph, we build a time-aware graph convolutional matrix to model the inter-dependency of the actions. More specifically, the normalized edge weight between i-th node and j-th node is:

$$\widetilde{\mathbf{A}}_{i,j} = \frac{\rho^{|t_i - t_j|}}{\sum_{k=0}^{n-1} \rho^{|t_i - t_k|}} \tag{1}$$

Algorithm 1: Construct Graph Convolutional Matrix in MINT

```
Input: Time-series structured data $\mathcal{T}$ for sets of users $\mathcal{U}$;
       receptive field hyper-parameter $\rho$; edge weight
       threshold $\epsilon$
Output: Graph convolutional matrices $\mathbf{A}$
1  for each user $u \in \mathcal{U}$ do
2      Extract the time-stamped records $V$ for $u$ from $\mathcal{T}$;
3      Get the pairwise time differences $\Delta T$ for the records;
4      for each record $v_i \in V$ do
5          Calculate $\rho^{\Delta T_i}$ and normalise it as $\widetilde{\mathbf{A}}_i$;
6          for each record $v_j \in V$ do
7              if $\widetilde{\mathbf{A}}_{i,j} < \epsilon$ then
8                  $\widetilde{\mathbf{A}}_{i,j} \leftarrow 0$
9          Normalise $\widetilde{\mathbf{A}}_i$ and obtain $\mathbf{A_i}$;
10 return $\mathbf{A}$;
```

where $0 < \rho < 1$ is the hyper-parameter that controls the range of the receptive field for each target node. t_i and t_j denote the timestamps of the target node and neighboring node, respectively. We assign zero weights to edges below a predefined threshold $\epsilon = 0.0001$. We summarize the graph convolutional matrix construction in Algorithm 1. As $\rho \rightarrow 0$, the range of the receptive field will be small and we can obtain the user's short-term intentions. As $\rho \rightarrow 1$, it will result in a larger receptive field, which is equivalent to applying a deep graph convolutional model. We can employ a graph convolutional layer with large ρ to capture the user's long-term intentions. On top of that, since the time interval is highly variable, a single graph convolutional matrix cannot effectively capture all valuable neighboring information for the target node.

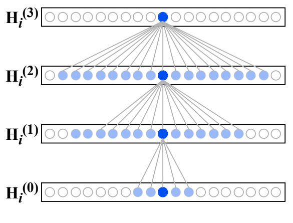
3 THE MINT FRAMEWORK

In this section, we present our **M**ulti-view row-**IN**teractive **T**ime-aware (MINT) fraud detection framework, which is designed to identify fraudulent behaviors from time-series structured data and provide explanations for the prediction. As illustrated in Figure 3, we first extract the user's temporal information to build a time-aware behavior graph with three different views and generate action node embeddings from the attributes (Section 3.1). For each user's behavior graph, we feed the corresponding graph convolutional matrices and node features into the graph convolution module to learn multi-view intentions (Section 3.2). Finally, we provide end-to-end training (Section 3.3) and interpretation for fraud behavior detection (Section 3.4).

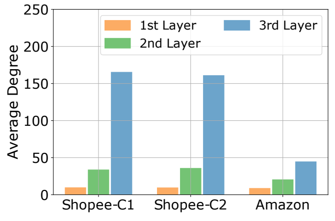
3.1 Data Preprocessing

As shown in Figure 3, the data preprocessing module of MINT is composed of a graph convolutional matrix constructor and a node embeddings constructor.

We reorganize the user's time-stamped behavioral sequence using graph topology, representing each action as a node with corresponding attributes as node features. This allows us to build a comprehensive model that captures the connections between different actions and their underlying structures. To achieve this, the graph convolutional matrix constructor follows Algorithm 1 to build three graph convolutional matrices with different receptive fields. As shown in Figure 4(a), the deeper graph convolutional layer will aggregate more neighborhood (light blue nodes in $\mathbf{H}^{(l)}$) information to the target node. Accordingly, Figure 4(b) shows the average degree of nodes in different layers, where the deeper the layer, the higher the degree. The expanded receptive fields enable us to capture a wider range of actions and interactions between different row data, leading to more accurate predictions and better insights.



(a) Graph convolution view



(b) Average degree of nodes

Figure 4: (a). The l-th graph convolutional layer takes $\mathrm{H}^{(l-1)}$ as input to generate $\mathrm{H}^{(l)}$ and the receptive field of three layers expands progressively. (b). The average degree of the action nodes in different graph convolutional layers for three real-world datasets.

We process action node features by projecting numerical features to a latent space and transforming categorical attributes into a one-hot representation. Then, we employ a multilayer perceptron (MLP) to model the dependencies and correlations between attributes. While there are other feature interaction methods that have been proposed [6, 11, 42], such as factorization machines or deep neural networks, we find that the MLP performs well for our task and is efficient in terms of training time and memory usage. For each user, its initial behavioral sequence embeddings are denoted as $\mathbf{H}^{(0)} \in \mathbb{R}^{n \times d}$, in which n is the number of nodes and d represents the dimension of input embeddings.

3.2 Multi-view Graph Convolutional Network

Existing RNN-based fraud detection methods [26, 32, 52] on users' behavioral data typically focus on a short sequence of user actions, which disregards the temporal information and long-term intention. Similarly, conventional graph neural networks [20, 39] will also fail to capture users' global interests from the behavioral graph due to their limited effective receptive fields, which only expand linearly instead of exponentially. This limitation arises because the receptive field in a behavioral graph can only expand along the corresponding sequence. To learn about long-term intentions, traditional graph neural networks must utilize more graph convolutional layers [9], which require additional memory and training time. To resolve this issue, the receptive fields of the three graph convolutional matrices in MINT are designed to grow nearly exponentially, as shown in Figure 4 (b). As a consequence, we can use much fewer layers to guarantee sufficient feature aggregation from neighbors, making it suitable for large-scale fraud detection tasks.

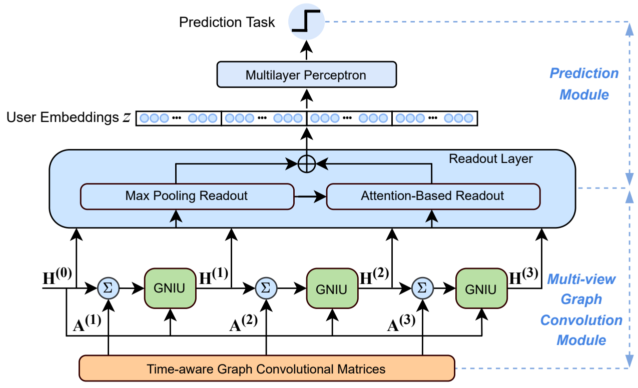


Figure 5: Multi-view graph convolutional network consists of a multi-view graph convolution module and a prediction module.

3.2.1 Multi-view Graph Convolution. In each graph convolutional layer, the feature aggregation is performed as follows:

$$\mathbf{h}_{\mathcal{N}(v_i)}^{(l)} = \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{A}_{v_i,v_j}^{(l)} * \mathbf{h}_{v_j}^{(l-1)} \tag{2}$$

where $\mathbf{h}_{\mathcal{N}(v_i)}^{(l)}$ represents the aggregated neighbor representations for action node v_i in l-th layer, and $\mathbf{A}_{v_i,v_j}^{(l)}$ denotes the normalized aggregation coefficient for action node v_j to node v_i in the l-th layer. Then, we perform feature transformation for the aggregated neighbor representations as follows:

$$\mathbf{h}_{v_i}^{(l)} = LeakyReLU(\mathbf{W}^{(l)}\mathbf{h}_{\mathcal{N}(v_i)}^{(l)}) \tag{3}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d\times d}$ is the trainable parameter matrix for transformation function in the l-th layer. For each behavior graph, it has one input embedding matrix $\mathbf{H}^{(0)}$ and three different-view node embedding matrices: $\mathbf{H}^{(1)}, \mathbf{H}^{(2)}$, and $\mathbf{H}^{(3)}$, where $\mathbf{H}^{(l)} = [\mathbf{h}_{v_0}^{(l)}, \mathbf{h}_{v_1}^{(l)}, \cdots, \mathbf{h}_{v_{n-1}}^{(l)}]$. Subsequently, we can use them to obtain the corresponding intention representations. There are two main advantages of the multi-view graph convolution.

- *Capturing multi-view intentions.* Conventional GNN models that use a single graph convolutional matrix are limited to learning a single-fold intention, similar to RNN-based approaches [25, 26]. However, behavioral sequences exhibit diverse temporal characteristics, leading to an enormous variation in the graph convolutional matrix. The proposed multi-view graph convolution technique can alleviate the problem by employing different graph convolutional matrices to complement each other. Furthermore, the multi-view graph convolutional model enables hierarchical aggregation of multi-hop neighboring features across deeper layers, capturing both local and global intentions.
- *Low space and time complexity.* Recent works [14, 21] have shown that the feature propagation and transformation in GNN can be separated without sacrificing the model efficiency. In our model, the hyper-parameter ρ for the last graph convolutional layer is set to be 0.9999 for Shopee datasets, leading to a larger receptive field. This allows the last layer to propagate node information to multi-hop neighbors like deep GCN [9, 14, 21], but with fewer layers, reducing memory requirements and training time. Additionally, since the first two graph convolutional matrices are still sparse, we can perform efficient graph training, in which computational complexity is linear to the number of nodes and edges [20, 40]. As a consequence, the proposed multi-view graph convolution can deal with long user behavioral sequences.

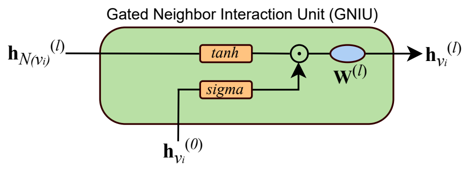


Figure 6: Gated neighbor interaction unit calibrates the aggregated information from neighboring nodes.

3.2.2 Gated Neighbor Interaction. Typically, in a behavior graph, an information aggregation method that relies only on the time interval information will result in a serious over-smoothing problem. That is, some common actions, such as "visit-homepage", appear far more often than others in the user's behavior data. As a consequence, the user's representations will be dominated by information about common actions, degrading the fraud detection performance.

To resolve the over-smoothing issue and aggregate more useful information from neighboring nodes to the target node, we design a gated neighbor interaction mechanism to calibrate the information that flows from neighboring nodes and learn feature interactions in node level. As illustrated in Figure 6, the aggregated neighbor representation will be fed into the neighbor interaction layer to do mutual relation modeling rather than a direct feature transformation as in Equation (3):

$$\widehat{\mathbf{h}}_{\mathcal{N}(v_i)}^{(l)} = \mathbf{LayerNorm}(\sigma(\mathbf{h}_{v_i}^{(0)}) \odot tanh(\mathbf{h}_{\mathcal{N}(v_i)}^{(l)})) \tag{4}$$

$$\mathbf{h}_{v_i}^{(l)} = LeakyReLU(\mathbf{W}^{(l)}\widehat{\mathbf{h}}_{\mathcal{N}(v_i)}^{(l)}) \tag{5}$$

where $\odot$ denotes the Hadamard product and LayerNorm represents the layer normalization [1]. The function of the gated neighbor interaction is similar to but more efficient than the LSTM cell, where three gates are required to regulate the flow of information into or out of the cell. In the proposed gated aggregation unit, the initial representation of each node controls the local intention information from the short-term view to the long-term view. Specifically, each node will filter out the extraneous neighboring information which may perturb the intention of the current action and propagate valuable information to the next layer. For instance, one user has two records "change-delivery-address" and "visit-homepage" stored in the table. The action node "change-delivery-address" will block the information from the neighboring node "visit-homepage" even though their time difference is very small. That is because the record "visit-homepage" is a pretty common action for users and it does not contain relevant semantics to "change-delivery-address".

Note that the neighbor interaction will give rise to a feature value shift for the action node and the negative effect will accumulate if more layers are stacked. Layer normalization is adopted to normalize the features for each layer, thus stabilizing the training process and saving training time. The superiorities of the proposed gated neighbor interaction techniques over conventional GNN-based and LSTM-based models are detailed as follows:

- *Over-smoothing mitigation.* The nodes interaction cell is designed to filter out redundant information from neighboring nodes and to mitigate the over-smoothing issue. Without the proposed gated neighbor interaction, both the global and local intentions will be dominated by the action nodes with high frequency, resulting in indistinguishable user representations.
- *Sophisticated node interaction.* When neighboring features are aggregated to one target node, the user's local intention centered on this action node will be learned. The explicit interaction between the target node and its neighboring nodes can help to capture the cross-row patterns for the user, leading to more sophisticated representation learning than a straightforward feature aggregation.

Readout Layer. To generate the intention embedding vectors from the action embedding matrices ($\mathbf{H}^{(0)}, \mathbf{H}^{(1)}, \mathbf{H}^{(2)}$, and $\mathbf{H}^{(3)}$), we design a max pooling and attention-based readout layer, as shown in Figure 5. Specifically, we employ max pooling in the behavioral sequence dimension to keep the most conspicuous features. Then we can get four *embedding-related intention representations*: $\mathbf{h}_e^{(0)}, \mathbf{h}_e^{(1)}, \mathbf{h}_e^{(2)}$, and $\mathbf{h}_e^{(3)}$. This will extract the relevant attribute information (i.e., column data in a table) that are closely related to fraudulent behavior detection. We also apply an attention-based fusion in the embedding dimension to keep the most conspicuous actions. Since the action embeddings are aggregated from neighboring nodes in the previous layer, it represents a sub-sequence representation. For each view, the *action-related intention representation* is obtained as follows:

$$\boldsymbol{\alpha}^{(l)} = \phi_{att}(\mathbf{h}_e^{(l)},\ \mathbf{H}^{(l)}) = \mathbf{h}_e^{(l)\top}\mathbf{W}^{att}\mathbf{H}^{(l)} \tag{6}$$

$$\mathbf{h}_a^{(l)} = \sum_{i=0}^{n-1} \alpha_i^{(l)} \cdot \mathbf{h}_i^{(l)}, \alpha_i^{(l)} \in \boldsymbol{\alpha}^{(l)}, \mathbf{h}_i^{(l)} \in \mathbf{H}^{(l)} \tag{7}$$

where $\mathbf{W}^{att} \in \mathbb{R}^{d\times d}$ is the bilinear attention weight matrix and shared by all the intention learning. For each view, the representation is $\mathbf{h}^{(l)} = \mathbf{h}_e^{(l)} + \mathbf{h}_a^{(l)}$.

3.2.3 Prediction Module. After learning the final intention representation with the readout layer, we feed the learned embeddings into an MLP classifier to generate the risk score of the user.

$$\mathbf{z} = CONCATE([\mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}]) \tag{8}$$

$$p = \phi_{MLP}(z), \phi_{MLP} : \mathbb{R}^{4*d} \mapsto \mathbb{R} \tag{9}$$

In the fraud detection task, p represents prediction value, and $\sigma(p)$ is the probability that one user is classified as a fraudster.

3.3 Model Training

The task of fraud detection involves binary classification, and for this purpose, we choose binary cross-entropy (BCE) as the objective function. BCE is specifically designed to quantify the difference between the predicted probability distribution and the true binary labels of a dataset. It is defined as follows:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{\mathcal{D}} ylog(\sigma(p)) + (1-y)log(1-\sigma(p)) + \lambda\,\|\theta\|^2 \tag{10}$$

where y is the ground truth, θ is the set of training parameters, λ is the regularizer parameter, $\mathcal{D}$ is the training sample and N is the size of training data. The gradient descent method is adopted to compute the parameters of the model for optimization.

In summary, MINT introduces multi-view graph convolutional matrices to hierarchically aggregate neighboring nodes' features and mitigates over-smoothing with a novel gated neighbor interaction technique.

3.4 Fraudulent Behaviors Extraction

In contrast to other classification tasks, identifying fraudulent transactions requires great caution to avoid negative impacts on the customer experience and platform credibility. To address this, we design a fraudulent behaviors extractor in MINT that offers global and local interpretation for model predictions. Based on these explanations, we can make more informed decisions by understanding how MINT identifies a fraud user.

Global Interpretation. There are hundreds of action types in the Shopee dataset, all of which have very different importance to fraud detection, and their proportion P_{action} in user behavior varies greatly. Among them, "visit-homepage" is the most common type of behavior, but it is not important for fraud detection. To align with human experts' insights and provide global interpretation, we design a new metric, called Normalized Reciprocal Rank@k (NRR@k) to evaluate the importance of each action type to fraud behavior detection. For each user that is labeled as a fraud, we extract its top-k important actions based on the attention weights $\alpha = \alpha^{(0)} + \alpha^{(1)} + \alpha^{(2)} + \alpha^{(3)}$ in the readout layer and compute the NRR@k as follows:

$$NRR = \frac{1}{N * P_{action}} \sum_{i=1}^{N} \left(\frac{1}{rank_{action}}\right) \tag{11}$$

where N is the number of users, P_{action} is the proportion of the action in the whole data, and $rank_{action}$ is the rank of action in the extracted suspicious behaviors. Note that an action may appear more than one time in one user's extracted fraudulent behaviors, and we will sum the reciprocal of its two ranks.

Local Interpretation. Additionally, to provide local interpretation, we extract the sub-sequences that have the highest importance when we derive the short-term action-related representation for the user. To this end, we use the attention weight $\boldsymbol{\alpha}^{(1)}$ to quantitatively evaluate the importance of each sub-sequence that was learned from the short view. For instance, when the short-view graph convolutional layer aggregates the neighborhood information for the target action node "login-with-password", the neighboring actions can be regarded as a behavioral sub-sequence centered on "login-with-password". Then we can use the corresponding attention weight in $\boldsymbol{\alpha}^{(1)}$ to represent the importance of each sub-sequence. Therefore, we can generate local interpretability for the identification of each fraud entity.

Table 2: Statistics of datasets used in the experiments. The *Pos. rate* is the ratio of fraud samples.

Dataset	#Users	#Action Type	#Actions	#Fields	Pos. rate
Shopee-C1	4.65M	727	1.40B	11	1.62%
Shopee-C2	0.63M	683	19M	11	1.11%
Amazon	18k	719820	13.27M	3	6.86%

4 EXPERIMENTS

In this section, we present the experimental setup and the evaluation of different models for the fraud detection task. We also conduct ablation studies on each technique of MINT, and learn the sensitivity of hyper-parameters over our model.

4.1 Experimental Setup

4.1.1 Datasets. We conduct experiments on real-world datasets collected from Shopee, the largest online e-commerce platform in Southeast Asia. The fraud labels are provided by Shopee's operation teams. We generate the risk score for each user based on its behavioral sequence and output the suspicious sub-sequence that results in a fraud prediction. As the Shopee users' traffic data (i.e., **Shopee-C1** and **Shopee-C2**) are private, we further conduct experiments on a public fake reviewer detection data, **Amazon dataset** [3, 4, 19] to demonstrate the effectiveness of our work. Fake reviewer detection is similar to Brushing fraud and is a special case of fraudulent behavior detection, where purchasing one product corresponds to one action.

Shopee. We collect large-scale industrial datasets from Shopee and the experiments utilize historical activity records from two different geographical regions (denoted as **Shopee-C1**, **Shopee-C2**) spanning a specific period. The statistics of data are summarized in Table 2. For each user, we collect 300 records and 11 attributes for fraud behavior detection. Each record stores one action, the corresponding timestamp, and other behavior-related attributes. The number of action types is different for the two regions, but the majority of actions are similar, i.e., "visit-homepage", "search-product", and "add-to-cart".

Amazon. The Amazon dataset includes product reviews under the categories of electronics, books, CDs, and movies[7]. The use case analysis in [3, 4, 19] shows that fraudulent groups usually exhibit over-consistent suspicious behaviors, purchasing the same group of products and writing reviews within a short period. To get the labels for each user, we follow the filtering algorithm in [13, 46] to label users with less than 20% helpful votes as fake users and users with more than 80% helpful votes as normal users. The remaining users are excluded from model training. It is important to note that the "helpfulness votes" attributes are not utilized in the model; instead, they serve solely to determine the labels of the users.

4.1.2 Baseline Methods. We compare MINT against 10 sequence-based and graph-based models. Sequence-based models treat each user's time-stamped activities as a sequence and utilize the behavioral sequence information to predict the fraud label of each user. For graph-based methods GCN, GAT, and GCNII, we build a behavior graph for each user with ρ set to 0.6 (same as the first layer in MINT) and we reset all the edge weights to 1. For TextGCN and IHGAT, we follow the original paper to construct a large-scale graph for all users and actions. Moreover, we construct ablation experiments over $\texttt{MINT}_{wo\_mv}$ and $\texttt{MINT}_{wo\_ri}$ to learn the multi-view and row-interactive functions, respectively. $\texttt{MINT}_{wo\_mv}$ is the variant of MINT with ρ set to 0.99 for all graph convolutional layers. $\texttt{MINT}_{wo\_ri}$ is the variant of MINT without gated neighbor interaction mechanism. The baselines are briefly introduced as follows:

(1) Sequence-based Methods:

- BiLSTM [35] is a variant of LSTM and applies bidirectional LSTM to capture the dependency information among sequential data.
- Time-BiLSTM [51] equips BiLSTM with time gates to model time intervals.
- Transformer [38] is the first sequence-based model which only utilizes self-attention to model the dependencies between input and output.
- LIC Tree-LSTM [26] takes tree-structure data as input and utilizes LSTM and self-attention mechanism to model user behaviors.
- HEN [52] uses a field-level extractor and action-level extractor to hierarchically learn users' representations and apply transfer learning to enhance the prediction performance.

(2) Graph-based Methods:

- GCN [20] is a widely used transductive graph neural network which aggregates the neighboring nodes' information based on the predefined normalized Laplacian matrix.
- GAT [39] introduces masked self-attention layers to learn specifying different weights to different nodes in a neighborhood.
- TextGCN [43] builds a heterogeneous graph for the items and the sequences, whose weights are determined by TF-IDF and PMI.
- GCNII [9] builds a deep graph convolutional network by introducing residual connection and identity mapping.
- IHGAT [25] introduces a hierarchical graph neural network to utilize LSTM and attention scheme to generate the node embeddings for user nodes.

4.1.3 Implementation Details. We implement the proposed model with PyTorch and tune hyper-parameters using the validation set. For the datasets, the split ratio of the training/validation/testing set is 80%/10%/10%. We select Adam as the optimizer and randomly initialize the model parameters with the Xavier initializer. We employ three graph convolutional layers in our proposed model, in which the values of hyper-parameter ρ are searched in respective ranges and set to 0.6, 0.99, and 0.9999 for Shopee datasets. For the Amazon dataset, it only provides the review date information and we set a different set of ρ, which are 0.7, 0.9, and 0.999, respectively. For other critical hyper-parameters in our model, the node representation dimension is set to 64, batch size to 512, and the learning rate to 0.0001. We also use L2 regularization with $\lambda = 0.00001$ to prevent over-fitting. For other compared methods, the hyper-parameters may be different from the proposed model to guarantee their performance. We adopt early stopping and terminate training if the validation performance does not improve for 10 epochs. All experiments are conducted with a single Tesla V100 GPU.

[7]https://jmcauley.ucsd.edu/data/amazon/.

Table 3: Fraud detection performance. The best-performing method in each column is boldfaced, and the best baseline in each column is underlined.

Model Class	Model	C1 (Large) AUC	C1 (Large) $R@P_{0.9}$	C1 (Large) F1	C2 (Medium) AUC	C2 (Medium) $R@P_{0.9}$	C2 (Medium) F1	Amazon (Small) AUC	Amazon (Small) $R@P_{0.9}$	Amazon (Small) F1
Sequence-based Baselines	**BiLSTM**	0.8991	0.3334	0.5911	0.9163	0.4552	0.5772	0.9045	0.3109	0.5198
	Time-BilSTM	0.9015	0.3379	0.6165	0.9213	0.4568	0.5914	0.9153	0.3281	0.5219
	Transformer	0.8914	0.3078	0.5874	0.9132	0.4419	0.5467	0.8803	0.2328	0.4967
	LIC Tree-LSTM	0.8819	0.3058	0.5744	0.8891	0.4213	0.5413	0.8831	0.2173	0.4355
	HEN	0.9011	0.3436	0.6023	0.9236	0.4718	0.5765	0.8966	0.2575	0.4754
Graph-based Baselines	**GCN**	0.8974	0.3251	0.6012	0.9137	0.4372	0.5687	0.9044	0.3149	0.5117
	GAT	0.8926	0.3179	0.5965	0.9133	0.4364	0.5654	0.8956	0.2895	0.5009
	TextGCN	0.8191	0.1981	0.4798	0.8301	0.3989	0.4533	0.8645	0.2074	0.3867
	GCNII	0.9003	0.3295	0.5988	0.9226	0.4722	0.5962	0.9174	0.3211	0.5216
	IHGAT	0.8976	0.3142	0.5921	0.9118	0.4441	0.5645	0.8987	0.2813	0.4988
MINT's	$\texttt{MINT}_{wo\_mv}$	0.9208	0.4025	0.6453	0.9393	0.5419	0.6215	0.9287	0.3446	0.5485
	$\texttt{MINT}_{wo\_ri}$	0.9206	0.3916	0.6398	0.9403	0.5453	0.6278	0.9243	0.3305	0.5561
	MINT	**0.9321**	**0.4512**	**0.6781**	**0.9515**	**0.5781**	**0.6485**	**0.9361**	**0.3671**	**0.5722**
Performance Gain over Baselines	**Sequence-based**	3.39%	31.32%	9.99%	3.02%	22.53%	9.66%	2.27%	11.87%	9.64%
	Graph-based	3.53%	36.93%	12.79%	3.13%	22.43%	8.77%	2.04%	14.33%	9.70%

4.1.4 Evaluation Metrics. In our experiments, we evaluate all methods with three widely used metrics: AUC (Area Under ROC curve), $R@P_k$, and F1-score. AUC represents the probability that the prediction of the positive cases is ranked before negative cases. $R@P_k$ is defined as the recall rate when the precision rate equals k. F1-score measures the balance between precision and recall in binary classification tasks. In our fraud detection scenario, our goal is to detect as many fraudulent users as possible without negatively impacting the regular operations of legitimate entities. k is set to be 0.9 for all experiments. Therefore, an efficient method should be able to achieve higher AUC, $R@P_{0.9}$ and F1-score than others. We also introduce two new metrics based on Equation 11 to measure the importance of the users' actions in fraud behavior detection: NRR@1 and NRR@5, which measure the degree to which an action is associated with the fraud.

4.2 Fraud Detection Comparison

Table 3 shows the performance comparison of the proposed models with state-of-the-art baselines. The AUC, $R@P_{0.9}$ and F1-score of MINT have displayed superior performance to baselines, consistently demonstrating its effectiveness in fraud detection.

In more detail, the reported AUC value for Shopee datasets is at least 3.02% higher than the sequence-based methods, $R@P_{0.9}$ gets a 31.32% improvement, and F1 gets a 9.66% improvement at the same time. The usage of the time-aware graph convolution enables us to capture the behavior-related temporal patterns and relationships between different action nodes. Meanwhile, the gated neighbor interaction scheme can calibrate the different-view intentions and avoid over-smoothing, enhancing the discrimination ability of the system. The improvement of the AUC for the Amazon dataset is a little smaller than that of the Shopee dataset because the Amazon dataset only has date information for actions, which is not as precise as the timestamp information used in Shopee-C1 and Shopee-C2. Among these sequence-based models, since Time-BiLSTM utilizes the time interval information to tune the three gates during training, it achieves more precise feed-forward propagation in two directions than other sequence-based models. The attention-based model (Transformer) ignores the time interval information and introduces the noisy position information, which degrades the effectiveness of the models.

Furthermore, compared to the best baselines of graph-based methods, MINT has gained at least 2.04% improvement in AUC, 14.33% in $R@P_{0.9}$, and 8.77% in F1 for all datasets. It indicates the capability of our model to distinguish suspicious sequential behaviors from normal ones. Compared to other graph-based approaches, MINT can capture both the short-term and long-term representations of the behavioral sequence and achieve multi-view extraction of the sequence information. At the same juncture, the proposed gated neighbor interaction mechanism can learn the cross-action relations and efficiently alleviate the over-smoothing issue. In comparison, conventional GCN and its variants which employ the same graph convolutional matrix in all layers are not capable of effectively capturing global information. Also, Text-GCN misses the valuable local intention representation by formulating the co-occurring actions as connected nodes in the graph because the employed point-wise mutual information (PMI) results in severe over-smoothing for the high-frequency action nodes, thus causing the fraudsters and benign users to be indistinguishable. Although IHGAT leverages a behavior tree to obtain local intention and obtain better user representations than Text-GCN, the importance of each intention is not learned effectively, resulting in its inferior performance. GCNII performs superior to other graph-based models since it can partially learn the global intention representation with more stacked layers and avoid severe over-smoothing with the residual connection.

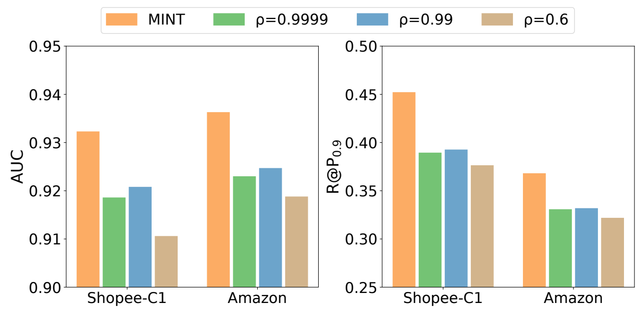


Figure 7: Comparison of MINT and its variants without multi-view graph convolution modules on different datasets. Each variant employs three graph convolutional layers but with the same graph convolutional matrix.

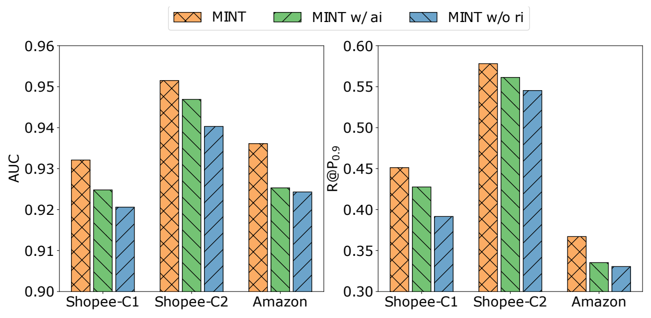


Figure 8: Impact of neighboring nodes interaction. The experimental settings of MINT and its variant are the same except for the nodes interaction layer.

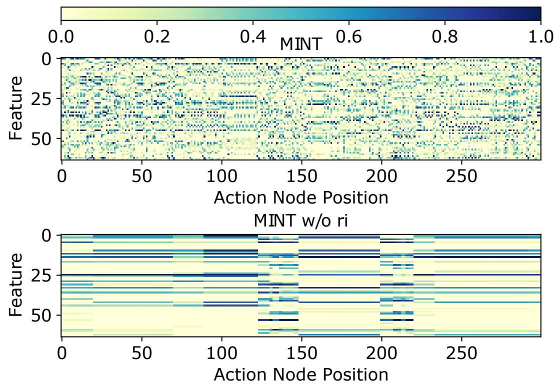


Figure 9: Visualizations of normalized action nodes representation. For each feature, we divide its value by the maximum one in that dimension. The node representations in MINT$_{wo\_ri}$ show serious oversmoothing.

4.3 Ablation Studies and Effectiveness Analysis

We perform ablation studies on MINT to show how multi-view graph convolution and gated neighbor interaction affect its performance. To investigate the effect of reducing over-smoothing as stated in Section 3.2.2, we compare the action nodes representation of MINT with its variant.

4.3.1 Multi-view Graph Convolution. To evaluate whether multi-view graph convolution can effectively learn target entities' intentions from different perspectives, we build three new graph models, which apply the same graph convolutional matrix in all the feature propagation layers with the other components similar to MINT. The comparison results shown in Figure 7 indicate that the integration of different convolutions can consistently enhance the representation capabilities of graph modeling and improve performance by a great margin. The improvements of AUC and R@P$_{0.9}$ are at least 1.25% and 11.06%, respectively.

It's worth noting that the performance of the new graph model with $\rho = 0.99$ performs slightly superior to the other two models. It is because a small ρ will lead to an insufficient receptive field, thus losing the capability of capturing important global intention representation. While graph convolutional layers with a large ρ will learn more neighboring information but stacked layers may cause trivial over-smoothing. Although there may exist an optimal ρ value for each sequence data, it will require more parameters and the model is difficult to converge. By contrast, we use multiple graph convolutional matrices to compensate for each other, without degrading the model training efficiency.

4.3.2 Gated Neighbor Interaction. To further assess the effectiveness of the proposed gated neighbor interaction method, we conduct ablation tests by removing this method (i.e., MINT$_{wo\_ri}$) or replacing it with attention-based row interaction (i.e., MINT$_{wi\_ai}$). Specifically, for MINT$_{wi\_ai}$, we modify the graph attention scheme in [39] and integrate it with the multi-view GCN models. Different from [39], where feature aggregation is only determined by the attention weights of neighboring nodes, MINT$_{wi\_ai}$ aggregates the neighboring features based on both the temporal information and attention weights, so that it can capture users' temporal patterns.

The comparison results are shown in Figure 8. We can observe that MINT$_{wo\_ri}$ is clearly inferior to MINT with gated neighbor interaction in all datasets, which indicates that the cross-neighbor features captured by MINT are complementary to the multi-view representation and the integration leads to better intention modeling. Moreover, MINT$_{wi\_ai}$ outperforms MINT$_{wo\_ri}$, which reinforces the necessity of learning cross-nodes interaction. However, the calibration for the aggregated feature in MINT$_{wi\_ai}$ is performed on a single neighboring action, rather than the whole view representations. On the contrary, MINT calibrates the representations of each view, capturing more meaningful semantics than MINT$_{wi\_ai}$. As a result, it achieves better intention modeling.

To better analyze the effectiveness of the gated neighbor interaction method in over-smoothing mitigation, we provide a visualization of the node representation for the training sequence in Figure 9. The figures illustrate the normalized node representation in each dimension for the 3rd-layer output of MINT and MINT$_{wo\_ri}$, respectively. We remove the layer normalization since it will change the feature distribution within a sequence and affect the visualization results. In Figure 9, the x-axis represents the position of the action node in the sequence, and the y-axis denotes the normalized

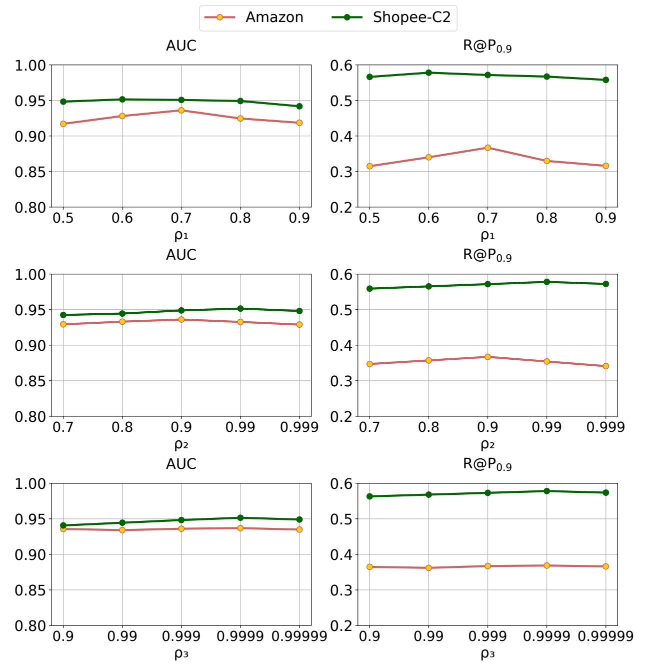


Figure 10: Sensitivity analysis of MINT on critical hyper-parameters ρ_1, ρ_2 and ρ_3.

features in each dimension. In fraud behavior detection, the more action feature information retained in the behavior graph learning, the better the performance. We observe that two neighboring nodes in MINT may have similar feature values in some dimensions but their action embeddings still have quite a few unique characteristics. On the contrary, a lot of connected action nodes in $\texttt{MINT}_{wo\_ri}$ have the same feature values in all dimensions. This is because the target node in MINT can calibrate the aggregated neighboring node representation in the feature propagation stage, allowing each node to retain more unique features and avoid the domination of some common actions. However, $\texttt{MINT}_{wo\_ri}$ propagates node features to multi-hop neighbors without considering the sophisticated feature selection, leading to feature concentration within a few dimensions. Consequently, it cannot obtain efficient node representation and differentiate between various node intentions.

4.4 Hyper-parameters Analysis

Receptive Field Hyper-parameter. The hyper-parameter ρ is denoted as ρ_1, ρ_2, and ρ_3, which determine the receptive fields of the three graph convolutional layers, respectively. We vary their values and evaluate the impacts they have on the model performance. When one of them varies, the other two maintain their original values as specified in the settings outlined in 4.1.3. The results on the datasets Shopee-C2 and Amazon are shown in Figure 10. We observe that the performance in terms of AUC and R@P$_{0.9}$ is relatively stable when the three hyper-parameters vary. Note that the Amazon dataset is more sensitive to the value of ρ_1 than the Shopee dataset. It is because the temporal information of the Amazon dataset is not as precise as Shopee-C2, and thus the edge weights for the behavioral graphs are not very accurate. For the Amazon dataset, when ρ_1 is very small, the effective receptive field of the first graph convolution layer will be too sharp to sufficiently capture the neighborhood information. On the other hand, a large ρ_1 for the Amazon dataset will lead to over-smoothing of locality features and degrade the final user representations learning. By contrast, both Amazon and Shopee-C2 datasets are relatively stable with the variation of ρ_2 and ρ_3. The reason is that their effective receptive fields are smoother so they are insensitive to the inaccuracy caused by a few neighboring nodes. By analyzing the impact of the three hyper-parameters on the model performance, we can find the necessity of multi-view graph convolution. The model performance will be sensitive to the variation of ρ value if the graph convolution layers can only learn the one-fold intention. It is worth noting that all of the MINT variants with different ρ settings achieve better results than all the baselines for Shopee-C2.

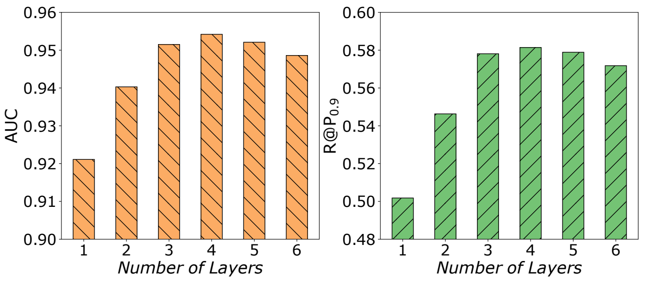


Figure 11: The performance of MINT with different numbers of graph convolutional layers on dataset Shopee-C2.

Number of Graph Convolutional Layers. We further conduct experiments over the Shopee-C2 dataset using MINT with different numbers of graph convolutional layers. For each variant, we select the best hyper-parameter settings. For instance, the value of ρ is set to (0.9999) for $\texttt{MINT}_{1L}$ (i.e., MINT with one graph convolutional layer) and (0.5, 0.8, 0.9, 0.99, 0.9999, 0.99999) for $\texttt{MINT}_{6L}$. Figure 11 shows that the AUC and R@P$_{0.9}$ of MINT increases with the number of graph convolutional layers before the layer depth is less than four. However, when the number of layers is larger than three, the AUC improvement of MINT variants over MINT is insignificant. That is because a hierarchical graph convolutional model with three layers can sufficiently learn users' behavioral patterns from a sequence length of 300.

4.5 Efficiency and Scalability

Next, we investigate the training efficiency and scalability of the proposed model compared to baselines.

Efficiency. We measure the training time for one epoch and the overall time required for each model to converge. For a fair comparison, LSTM layers are all coded in Python rather than the modules provided by PyTorch. The training efficiency results on Shopee-C2 are shown in Figure 12 (the performance of the other two datasets are similar). We only select four representative baselines which achieve comparable performance in Table 3. It can be observed that the GNN-based models (i.e., MINT and GCNII) achieve much higher training speed (time for one epoch) than others. Since the

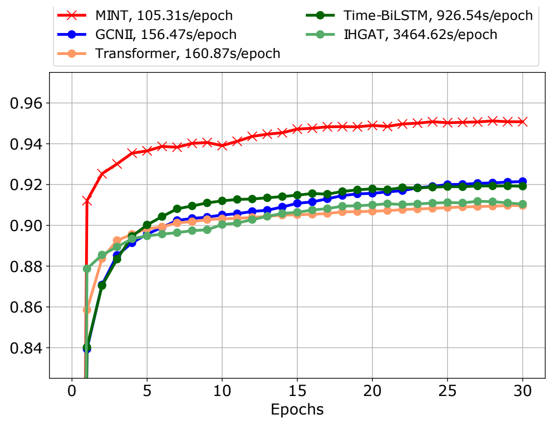


Figure 12: Training efficiency on dataset Shopee-C2. MINT not only converges faster than baselines but also takes much less time for each epoch.

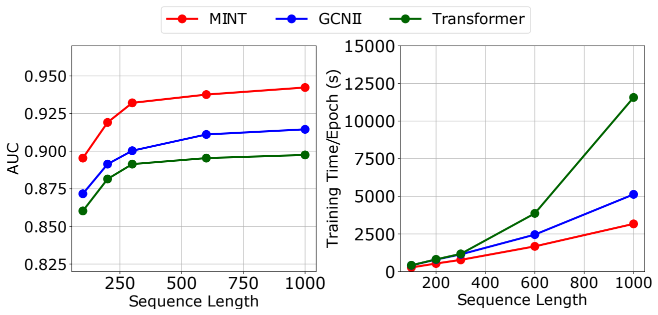


Figure 13: Training speed and performance of MINT and two baselines over Shopee-C1 with a different number of action nodes for each sequence.

proposed model requires much fewer layers than GCNII to capture multi-hop neighboring information, we can achieve more efficient forward propagation. By contrast, the application of the attention mechanism in Transformer takes more time than in a fast graph neural network. In addition, we can observe that MINT achieves an AUC of higher than 0.91 after one epoch, showing the effectiveness of gated neighbor interaction and multi-view learning. On the other hand, Time-BiLSTM and IHGAT need to do recurrent feedforward propagation for the purpose of capturing long-term information, thus lowering the training speed. Additionally, IHGAT also requires information propagation along the user graph after learning the behavioral sequence pattern via RNN, leading to worse training efficiency than Time-BiLSTM.

Scalability. To evaluate the effect of the behavioral sequence length on the performance and training speed of MINT, we collect more behavior actions for each user in Shopee-C1. Due to significantly poorer training efficiency observed in Time-BiLSTM and IHGAT, our comparison in Figure 13 focuses mainly on MINT with GCNII and Transformer models, considering various sequence lengths. We can see that the three models have better performance with a larger behavioral graph, and MINT still outperforms the two baselines. Also,

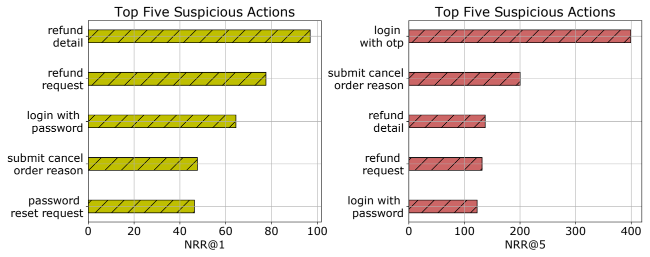


Figure 14: The NRR@1 and NRR@5 of the five most important actions for fraud behavior identification.

the training speed of MINT presents greater advantages when the sequence length is large and its training time increases almost linearly with the sequence length. This is because the short-term view and mid-term view of the behavior graph in MINT are very sparse, whose average degrees are about 8 and 35 for Shopee data as shown in Figure 4. The training time of a sparse graph neural network is mainly determined by its number of edges [20, 40], which linearly increases with the sequence length in MINT. As a consequence, the training time of MINT increases linearly with sequence length when the length is very large. We also implement GCNII with a sparse graph convolutional matrix, resulting in much better performance than the Transformer in terms of training speed. Hence, our model can easily scale to sequences of up to more than 1000 actions, which is suitable for typical traffic data in the e-commerce platform.

4.6 Fraudulent Behaviors Verification

Global Interpretability. We validate the extracted fraudulent behaviors with the human expert in the business unit to verify the interpretability of the developed MINT. Figure 14 illustrates the NRR@1 and NRR@5 of five action types that have the highest NRR values across the whole dataset for Shopee-C2. We ignore those action types whose frequency in Shopee-C2 is less than 10. By checking with the risk control experts in Shopee, the actions in Figure 14 appear far more frequently in the behaviors of fraudulent users than in normal users. For example, the action types "refund-detail" and "refund-request" are closely related to refund fraud and chargeback fraud, in which fraudsters request a refund and claim that they have paid an excessive amount or cancel the order. The actions "login-with-password", "login-with-otp", and "password-reset-request" often appear in account takeover fraud scenarios. On the contrary, the NRR@1 of "visit-homepage" and "search-product" are 0.25 and 0.36, respectively.

Local Interpretability. The complexity of human behavior makes it impossible for our risk control experts to create a comprehensive list of all the suspicious behavior sequences. Instead, our system detects and extracts suspicious behavior sequences, providing the most accurate and appropriate verification method. In the following, we present two examples to illustrate how our system identifies and examines these sequences, and how our risk control experts interpret the findings. The first suspicious sub-sequence output by the fraudulent behaviors extractor is: "delink-phone ⟶ verify-phone ⟶ check-new-phone ⟶ set-new-phone ⟶ edit-my-profile ⟶ change-delivery-address ⟶ checkout-product". This is flagged as an account takeover fraud by the human expert. The cybercriminal gains unauthorized access to a benign user's account, changes the victim's phone number to their own, and uses the account to make a purchase. The second suspicious example is: "update-creditcard → online-payment → check-my-order → chat → check-payment-option → refund-request → refund-detail → check-payment-option". This is flagged as refund fraud, in which users claim an issue with the transaction, requesting a refund and keeping the purchased item without canceling the order. The interpretability that MINT offers can provide the fraudulent subsequences that are closely related to the final identification.

5 RELATED WORK

In this section, we review existing literature related to fraud detection systems on structured data and graph neural networks.

Fraud Detection Systems on Structured Data. Traditional fraud detection systems typically rely on expert knowledge or supervised models for statistical feature training [17, 28, 31]. However, they are limited in their applicability and scalability due to the need for careful feature engineering. In line with technological development, several deep learning-based fraud detection models have been proposed, learning users' transaction history and behavioral data from structured data [7, 10, 32, 41, 48]. According to the information source utilized, existing fraud detection systems on structured data can be categorized as transaction-based methods [7, 10, 32, 48] or behavior-based methods [5, 25, 26, 41, 52].

Transaction-based anti-fraud systems extract transaction-related records and attributes from the database to determine the risk score of each transaction record [7, 10, 18, 32, 48]. For instance, [7, 32] transform tabular data into graph-structure data and adopt graph neural networks to learn user and transaction representations for fraud probability prediction. To learn the logical relationships between attributes in historical transaction records, [48] proposes a logical graph of behavior profiles (LGBP) model, which uses a path-based transition probability to characterize users' transaction behaviors and calculate the acceptance probability of the incoming transactions based on the latest k transaction records. [10] proposes a 3D-convolutional neural network for fraud transaction detection, modeling the temporal and spatial aggregation of fraudsters in transaction records. However, these transaction-based fraud detection methods cannot perform well at an early stage when the available transaction data are fairly limited [37, 47].

Behavior-based fraud detection methods involve extracting the daily activities of each user as a behavioral sequence from structured data, which can be modeled using various techniques. For instance, recurrent neural networks (RNN) or attention-based mechanisms have been used for this purpose [2, 5, 25, 26, 49, 52]. In [47], researchers employ gated recurrent unit (GRU) to learn the mapping function between time-varying covariates and survival probabilities to predict fraud probability [49]. Of particular note is the utilization of a behavior tree to extract users' intentions from sequence data in [26], in which the branch representation is captured by long short-term memory (LSTM) neural network with an attention scheme. The Tree-LSTM approach is further improved in [25] by building connections between different users and updating the user embeddings by aggregating the representations of neighboring users and intention nodes. In addition, a neural factorization machine (NFM) is deployed in a risk management system to capture high-order feature interaction between different events [41]. However, all of the existing behavior-based fraud detection methods disregard the temporal information of the user's behaviors, making them vulnerable to well-camouflaged fraudsters. Additionally, the one-sided view learning methods typically fail to capture the user's long-term intentions in complex scenarios.

Graph Neural Network. Graph convolutional network [20] extends traditional graph-based signal processing work [12] to scalable semi-supervised learning, which is widely used in node classification and graph classification. The GraphSAGE model proposed in [16] introduces inductive learning to graph convolutional network (GCN), which allows batch learning and can be easily applied on a large graph. Masked self-attention is leveraged in [39] to refine GCN and assign different importance to neighbor nodes for feature propagation. Various deep graph convolutional networks have been proposed to enlarge the receptive field and minimize over-smoothing [9, 22]. To enable GCN to express higher-order polynomial filters, [9] adds initial residual connection and identity mapping into vanilla GCN and achieves state-of-the-art performance in several small open graph datasets. But most existing GNN models have only one graph convolutional matrix, which limits them to a very one-sided picture of graph representation. In this work, a multi-view graph learning approach is proposed, which allows us to capture hidden representations from different perspectives.

6 CONCLUSIONS

In this paper, we propose a novel multi-view row-interactive time-aware fraud behavior detection framework MINT targeting time-series structured data. MINT reorganizes each user's behaviors as a time-aware behavior graph and constructs three different views for each user, which allows the model to capture user intentions from multiple perspectives. Also, we devise a novel gated neighbor interaction mechanism to learn the row-interactive information and mitigate the over-smoothing issue. Furthermore, we generate an explainable fraudulent behavior sub-sequence so that human experts can interpret the detection findings. To the best of our knowledge, MINT is the first to utilize the temporal information in users' time-series behaviors with graph topology for fraud detection. Extensive experiments on the Shopee and Amazon datasets validate that MINT achieves superior effectiveness, efficiency, and scalability compared to 10 state-of-the-art baselines, demonstrating its usefulness on production platforms.

ACKNOWLEDGMENTS

This research is supported by Shopee Singapore Private Limited, Singapore Economic Development Board, and the National Research Foundation, Singapore under its Emerging Areas Research Projects (EARP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. The work of Meihui Zhang is supported by National Natural Science Foundation of China (62072033).

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *arXiv:1607.06450v1* (2016). arXiv:arXiv:1607.06450v1

[2] Dmitrii Babaev, Maxim Savchenko, Alexander Tuzhilin, and Dmitrii Umerenkov. 2019. E.T.-RNN: Applying Deep Learning to Credit Loan Applications. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* ACM, New York, NY, USA, 2183–2190. https://doi.org/10.1145/3292500.3330693

[3] Yikun Ban, Xin Liu, Yitao Duan, Xue Liu, and Wei Xu. 2019. No place to hide: Catching fraudulent entities in tensors. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019* 2 (2019), 83–93. https://doi.org/10.1145/3308558.3313403 arXiv:1810.06230

[4] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. CopyCatch: Stopping group attacks by spotting lockstep behavior in social networks. *WWW 2013 - Proceedings of the 22nd International Conference on World Wide Web* (2013), 119–129.

[5] Bernardo Branco, Pedro Abreu, Ana Sofia Gomes, Mariana S.C. Almeida, João Tiago Ascensão, and Pedro Bizarro. 2020. Interleaved Sequence RNNs for Fraud Detection. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2020), 3101–3109. https://doi.org/10.1145/3394486.3403361 arXiv:2002.05988

[6] Shaofeng Cai, Kaiping Zheng, Gang Chen, H. V. Jagadish, Beng Chin Ooi, and Meihui Zhang. 2021. ARM-Net: Adaptive Relation Modeling Network for Structured Data. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2021), 207–220. https://doi.org/10.1145/3448016.3457321

[7] Shaosheng Cao, Xin Xing Yang, Cen Chen, Jun Zhou, Xiaolong Li, and Yuan Qi. 2018. TitAnt: Online realtime transaction fraud detection in ant financial. *Proceedings of the VLDB Endowment* 12, 12 (2018), 2082–2093. https://doi.org/10.14778/3352063.3352126 arXiv:1906.07407

[8] Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating Embeddings of Heterogeneous Relational Datasets for Data Integration Tasks. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2020), 1335–1349. https://doi.org/10.1145/3318464.3389742

[9] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and Deep Graph Convolutional Networks. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 1725–1735. https://proceedings.mlr.press/v119/chen20v.html

[10] Dawei Cheng, Sheng Xiang, Chencheng Shang, Yiyi Zhang, Fangzhou Yang, and Liqing Zhang. 2020. Spatio-Temporal Attention-Based Neural Network for Credit Card Fraud Detection. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (2020), 362–369. https://doi.org/10.1609/aaai.v34i01.5371

[11] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. 2020. Adaptive factorization network: Learning adaptive-order feature interactions. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence* (2020), 3609–3616. https://doi.org/10.1609/aaai.v34i04.5768 arXiv:1909.03276

[12] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in Neural Information Processing Systems* Nips (2016), 3844–3852. arXiv:1606.09375

[13] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. 2020. Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters. *International Conference on Information and Knowledge Management, Proceedings* (2020), 315–324. https://doi.org/10.1145/3340531.3411903 arXiv:2008.08692

[14] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. 2020. Graph Random Neural Network for Semi-Supervised Learning on Graphs. NeurIPS (2020), 1–18. arXiv:2005.11079 http://arxiv.org/abs/2005.11079

[15] Qingyu Guo, Pengrui Hui, Jiaming Huang, Zhao Li, Long Zhang, Bo An, and Mengchen Zhao. 2019. Securing the deep fraud detector in large-scale e-commerce platform via adversarial machine learning approach. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019* (2019), 616–626. https://doi.org/10.1145/3308558.3313533

[16] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, Vol. 2017-Decem. 1025–1035. arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[17] Ariel Jarovsky, Tova Milo, Slava Novgorodov, and Wang Chiew Tan. 2018. GOLDRUSH: Rule sharing system for fraud detection. *Proceedings of the VLDB Endowment* 11, 12 (2018), 1998–2001. https://doi.org/10.14778/3229863.3236244

[18] Jiaxin Jiang, Yuan Li, Bingsheng He, Bryan Hooi, Jia Chen, and Johan Kok Zhi Kang. 2022. Spade: A Real-Time Fraud Detection Framework on Evolving Graphs. *Proceedings of the VLDB Endowment* 16, 3 (nov 2022), 461–469. https://doi.org/10.14778/3570690.3570696 arXiv:2211.06977

[19] Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2016. Spotting Suspicious Behaviors in Multimodal Data: A General Metric and Algorithms. *IEEE Transactions on Knowledge and Data Engineering* 28, 8 (2016), 2187–2200. https://doi.org/10.1109/TKDE.2016.2555310

[20] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings* (2017), 1–14. arXiv:1609.02907

[21] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then propagate: Graph neural networks meet personalized PageRank. *7th International Conference on Learning Representations, ICLR 2019* (2019), 1–15. arXiv:1810.05997

[22] Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. 2021. Training Graph Neural Networks with 1000 Layers. (2021). arXiv:2106.07476 http://arxiv.org/abs/2106.07476

[23] Side Li, Lingjiao Chen, and Arun Kumar. 2019. Enabling and optimizing non-linear feature interactions in factorized linear algebra. *Proceedings of the ACM SIGMOD International Conference on Management of Data* degree 2 (2019), 1571–1588. https://doi.org/10.1145/3299869.3319878

[24] Xiang Li, Wen Zhang, Jiuzhou Xi, and Hao Zhu. 2017. HGsuspector: Scalable Collective Fraud Detection in Heterogeneous Graphs. 17 (2017). https://doi.org/10.475/123_4

[25] Can Liu, Li Sun, Xiang Ao, Jinghua Feng, Qing He, and Hao Yang. 2021. Intention-aware Heterogeneous Graph Attention Networks for Fraud Transactions Detection. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2021), 3280–3288. https://doi.org/10.1145/3447548.3467142

[26] Can Liu, Qiwei Zhong, Xiang Ao, Li Sun, Wangli Lin, Jinghua Feng, Qing He, and Jiayu Tang. 2020. Fraud Transactions Detection via Behavior Tree with Local Intention Calibration. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2020), 3035–3043. https://doi.org/10.1145/3394486.3403354

[27] Zhaojing Luo, Shaofeng Cai, Yatong Wang, and Beng Chin Ooi. 2023. Regularized Pairwise Relationship based Analytics for Structured Data. *Proc. ACM Manag. Data* 1, 1 (2023), 82:1–82:27.

[28] Tova Milo, Slava Novgorodov, and Wang Chiew Tan. 2015. Rudolf: Interactive rule refinement system for fraud detection. *Proceedings of the VLDB Endowment* 9, 13 (2015), 1465–1468. https://doi.org/10.14778/3007263.3007285

[29] Beng Chin Ooi, Kian-Lee Tan, Sheng Wang, Wei Wang, Qingchao Cai, Gang Chen, Jinyang Gao, Zhaojing Luo, Anthony K. H. Tung, Yuan Wang, Zhongle Xie, Meihui Zhang, and Kaiping Zheng. 2015. SINGA: A Distributed Deep Learning Platform. In *ACM MM.* 685–688.

[30] Jiarui Qin, Weinan Zhang, Rong Su, Zhirong Liu, Weiwen Liu, Ruiming Tang, Xiuqiang He, and Yong Yu. 2021. *Retrieval & Interaction Machine for Tabular Data Prediction.* Vol. 1. Association for Computing Machinery. 1379–1389 pages. https://doi.org/10.1145/3447548.3467216 arXiv:2108.05252

[31] Xiafei Qiu, Wubin Cen, Zhengping Qian, You Peng, Ying Zhang, Xuemin Lin, and Jingren Zhou. 2018. Real-time constrained cycle detection in large dynamic graphs. *Proceedings of the VLDB Endowment* 11, 12 (2018), 1876–1888. https://doi.org/10.14778/3229863.3229874

[32] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2021. xFraud: Explainable Fraud Transaction Detection. *Proceedings of the VLDB Endowment* 15, 3 (2021), 427–436. https://doi.org/10.14778/3494124.3494128 arXiv:2011.12193

[33] Babak Salimi, Harsh Parikh, Moe Kayali, Lise Getoor, Sudeepa Roy, and Dan Suciu. 2020. Causal Relational Learning. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2020), 241–256. https://doi.org/10.1145/3318464.3389759 arXiv:2004.03644

[34] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 10843 LNCS. Springer Verlag, 593–607. https://doi.org/10.1007/978-3-319-93417-4_38

[35] Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681. https://doi.org/10.1109/78.650093

[36] Fengzhao Shi, Yanan Cao, Yanmin Shang, Yuchen Zhou, Chuan Zhou, and Jia Wu. 2022. H2-FDetector: A GNN-based Fraud Detector with Homophilic and Heterophilic Connections. *WWW 2022 - Proceedings of the ACM Web Conference 2022* (2022), 1486–1494. https://doi.org/10.1145/3485447.3512195

[37] Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. 2020. Hierarchical propagation networks for fake news detection: Investigation and exploitation. *Proceedings of the 14th International AAAI Conference on Web and Social Media, ICWSM 2020* (2020), 626–637. arXiv:1903.09196

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 2017-Decem (jun 2017), 5999–6009. arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[39] Petar Veličković, Arantxa Casanova, Pietro Liò, Guillem Cucurull, Adriana Romero, and Yoshua Bengio. 2018. Graph attention networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* (2018), 1–12. arXiv:1710.10903

[40] Jun Wu, Jingrui He, and Jiejun Xu. 2019. Demo-Net: Degree-specific graph neural networks for node and graph classification. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2019), 406–415. https://doi.org/10.1145/3292500.3330950 arXiv:1906.02319

[41] Dongbo Xi, Bowen Song, Fuzhen Zhuang, Yongchun Zhu, Shuai Chen, Tianyi Zhang, Yuan Qi, and Qing He. 2020. Modeling the Field Value Variations and Field Interactions Simultaneously for Fraud Detection. (2020), 1–11. arXiv:2008.05600 http://arxiv.org/abs/2008.05600

[42] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. *Ijcai* (aug 2017), 3119–3125. https://doi.org/10.5555/3172077.3172324 arXiv:1708.04617

[43] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph Convolutional Networks for Text Classification. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (jul 2019), 7370–7377. https://doi.org/10.1609/aaai.v33i01.33017370 arXiv:1902.05575

[44] Chang Ye, Yuchen Li, Bingsheng He, Zhao Li, and Jianling Sun. 2021. GPU-Accelerated Graph Label Propagation for Real-Time Fraud Detection. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2021), 2348–2356. https://doi.org/10.1145/3448016.3452774

[45] Ge Zhang, Zhao Li, Jiaming Huang, Jia Wu, Chuan Zhou, Jian Yang, and Jianliang Gao. 2022. eFraudCom: An E-commerce Fraud Detection System via Competitive Graph Neural Networks. *ACM Trans. Inf. Syst.* 40, 3 (2022), 47:1–47:29.

[46] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, and Lizhen Cui. 2020. GCN-Based User Representation Learning for Unifying Robust Recommendation and Fraudster Detection. Dl (2020), 689–698. https://doi.org/10.1145/3397271.3401165 arXiv:2005.10150

[47] Tong Zhao, Bo Ni, Wenhao Yu, and Meng Jiang. 2020. Early Anomaly Detection by Learning and Forecasting Behavior. 1 (2020). arXiv:2010.10016 http://arxiv.org/abs/2010.10016

[48] Lutao Zheng, Guanjun Liu, Chungang Yan, and Changjun Jiang. 2018. Transaction fraud detection based on total order relation and behavior diversity. *IEEE Transactions on Computational Social Systems* 5, 3 (2018), 796–806. https://doi.org/10.1109/TCSS.2018.2856910

[49] Panpan Zheng, Shuhan Yuan, and Xintao Wu. 2019. SAFE: A neural survival analysis model for fraud early detection. *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019* (2019), 1278–1285. https://doi.org/10.1609/aaai.v33i01.33011278 arXiv:1809.04683

[50] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2018), 1079–1088. https://doi.org/10.1145/3219819.3219826 arXiv:1801.02294

[51] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to do next: Modeling user behaviors by Time-LSTM. *IJCAI International Joint Conference on Artificial Intelligence* 0 (2017), 3602–3608. https://doi.org/10.24963/ijcai.2017/504

[52] Yongchun Zhu, Dongbo Xi, Bowen Song, Fuzhen Zhuang, Shuai Chen, Xi Gu, and Qing He. 2020. Modeling Users' Behavior Sequences with Hierarchical Explainable Network for Cross-domain Fraud Detection. *The Web Conference 2020 - Proceedings of the World Wide Web Conference, WWW 2020* (2020), 928–938. https://doi.org/10.1145/3366423.3380172