Early Fraud Detection with Augmented Graph Learning

Tong Zhao*, Bo Ni*, Wenhao Yu, Meng Jiang
Department of Computer Science and Engineering, University of Notre Dame, USA
{tzhao2,bni,wyu1,mjiang2}@nd.edu

ABSTRACT

Having an effective fraud detection system can help social media to identify suspicious behaviors or accounts. Early detection is crucial to minimize losses if the fraud is ongoing. Existing detection methods perform effectively when good amounts of observed behavior data are available (which sometimes has been too late); however, at an early stage when the observations are limited, the performance would not be satisfactory. In this work, we propose Alfrad, a novel self-training framework that uses behavior data augmentation for early fraud detection. It has a Seq2Seq-based behavior predictor that predicts (i) whether a user will adopt a new item or an item that has been historically adopted and (ii) which item will be adopted. Alfrad utilizes the prediction results of fraud detection methods to make better prediction of future behavior and uses the augmented graph to help fraud detection methods to achieve higher performance while not requiring any additional data. It explores the mutually beneficial relationship between fraud detection and behavior prediction. Experiments show that Alfrad improves the performance of different kinds of fraud detection methods. With Alfrad augmented methods, the performance of fraud detection at an earlier stage is comparable with and/or better than non-augmented methods on a greater amount of observed data.

1 INTRODUCTION

During the last twenty years, we have witnessed a boom in social networks and other web-based services. While it certainly makes people's life easier and more convenient, it also indirectly creates a market for malicious users. One can earn huge profits by selling fake followers on Instagram and Twitter, or fake reviews on Yelp and Amazon. Some malicious service providers could also help one disseminating information such as ads or fake news by manipulating botnets on social networks. It turns out that these behaviors have negative impact on our society: fake news could have tremendous effects on political activities; fake reviews constantly undermine customers' ability to make fair judgements; and fake followers will cause fake popularity, giving false credentials and breaking a competitive market. In this paper, we mainly focus on the suspicious behavior that is often being referred to as link-farming which involves creating false edges in a social network. For example, in a Facebook "who-likes-what-pages" graph, the fraudsters might create false edges that make certain pages look more popular or more legitimate [2].

Various efforts have been made in the data mining community to address the problem of link farming, including graph mining based methods such as Fraudar [9], LockInfer [12] etc., and graph machine learning based methods like Dominant [4]. Despite their effectiveness, we nevertheless witness a decline in performance when data is insufficient or incomplete. However, detecting the fraudsters after they have achieved their purpose is not ideal in real usage. This in turn poses a grim challenge for fraud detection at early stage: we want to prevent the negative impact incurred by fraudsters when observed data is not sufficient while existing fraud detection methods would inevitably underperform for the scarcity of available observations. Thus, in this paper, we aim to answer the following question: Is it possible to achieve a similar performance at an early stage when observed behavior data is incomplete? In other words, can we design a framework so that the performance of fraud detection at an earlier stage is comparable with and/or better than the performance on greater amount of data?

Present work. In this paper, we propose Early Fraud Detection (Alfrad), a novel self-training framework that is constructed by two components: a fraud detection method that detects fraudulent users (un)supervisedly and a Seq2Seq user behavior forecasting model that augments the graph. The two components interdependent on each other in the sense that the information derived from the first could be a useful input for the second, and vice versa. Shown in Figure 1 is a sample iteration: the behavior forecasting model consists of a two-step decoder to (i) predict whether the next item a user will adopt comes from his/her behavior history and to (ii) predict which item will be adopted through similarity matching. When making predictions, it takes advantage of the fraudster detection results from the previous iteration based on the assumption that malicious users tend to post more frequently. Also, we assume that the fraudulent users (often consist of bot accounts controlled by central servers) have a higher tendency to repeatedly adopt the same items while the normal users will have a consistent pattern of discovering new information. The augmentation model can hence update both the graph structure and attribute information simultaneously based on the newly predicted items.

We thus summarize our contribution as follows:

- To the best of our knowledge, this is the first work that studies the problem of early-stage fraud detection on social media by learning for behavior forecasting.
- We propose a novel framework that improves the performance of early fraud detection by behavior forecasting and behavior graph augmentation.
- We conduct extensive experiments on a real-world dataset and obtain better performance than both unsupervised and supervised fraudulent detection methods when data is relatively insufficient and incomplete.

2 RELATED WORK

Our work aims to address the early-stage fraud detection problem using a two-step sequence predictor for data augmentation, so in this section, we will first review related works in suspicious behavior detection. Then, we will shift our focus to some of the relevant

* Equal contribution.

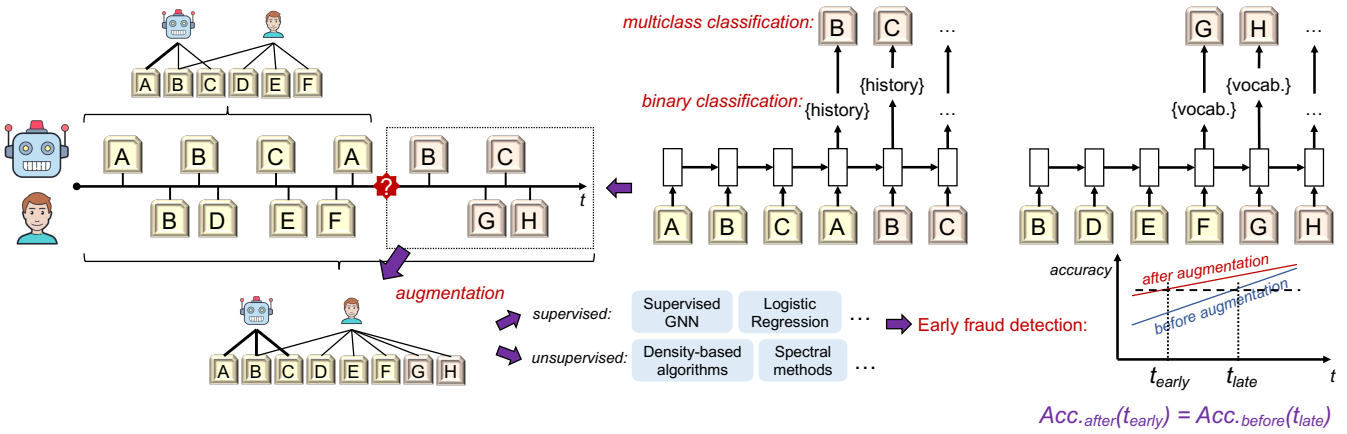


Figure 1: One pass for the early fraud detection framework: The Seq2Seq model first predict the items that a user will adopt in the future based on his/her behavior history. (1) The first step is to predict whether the item has been in the user's behavior history or is a new item in the global "vocabulary." The second step is to predict the item from the historical items or from the vocabulary. (2) The framework then uses the predicted items to augment user-item bipartite behavior graphs. The predicted items and links will improve the performance of a wide line of fraud detection methods including semi-supervised learning (e.g., GCN [13]) and unsupervised learning (e.g., Fraudar [9], LockInfer [12]). Our framework's performance of fraud detection at an earlier stage is comparable or better than non-augmented methods with more observed data.

works related to our framework, including sequence prediction, dynamic graph learning, and behavior data mining.

Unsupervised Fraud Detection. Suspicious behavior detection has received a great amount of academic interest in the past decade [2]. We categorize these methods into unsupervised and supervised methods. Unsupervised methods approach the problem by taking certain assumptions regarding fraudulent behaviors. LockInfer [11] investigates the *lockstep behavior* using a generalized SpokEn [21] method, which utilizes pairs of eigenvectors of graphs to detect the "eigen-spokes" patterns. Dominant [4] was a graph auto-encoder based deep model that detects anomalous nodes from attributed networks. Rev2 [14] was an iterative algorithm that detected outlier users with low fairness scores. Zhao *et al.* [30] proposed an actionable algorithm to block the dense subgraphs on bipartite graphs.

Graph Neural Networks. In recent years, following the initial idea of convolution based on spectral graph theory [1], many spectral GNNs have since been developed and improved by [3, 7, 13, 15, 17, 29]. As spectral GNNs generally operate (expensively) on the full adjacency, spatial-based methods which perform graph convolution with neighborhood aggregation became prominent [5, 6, 18, 19, 22], owing to their scalability and flexibility [26]. Moreover, several recent works proposed more advanced architectures which add residual connections to facilitate deep GNN training [16, 25]. More recently, GNN-based models were also proposed for tasks in varies fields of research such as natural language processing [27] and behavior modeling [23].

3 PROBLEM DEFINITION

Consider a bipartite graph $\mathcal{G}_t = (\mathcal{U}, \mathcal{V}, \mathcal{E}_t)$ at timestamp t, where $\mathcal{U}$ is a set of m users, $\mathcal{V}$ is a set of n items and $\mathcal{E}_t$ is the set of edges at time t. Let $\mathbf{X}_t \in \mathbb{R}^{(m+n)\times k}$ be the feature matrix at time t that contains k-dimensional raw features of all nodes. We also denote $\mathbf{y} \in \{0,1\}^m$ as labels for users where fraudsters get 1 and others get 0. Following the above notations, we first define the task of fraud detection, and then proceed to give a formal definition of early-stage fraud detection.

Definition 1. *(Fraudulent Behavior Detection) Given the bipartite graph $\mathcal{G}$ and feature matrix $X \in \mathbb{R}^{(m+n)\times k}$, find a function $g : \mathcal{G}, \mathbf{X} \rightarrow \mathbf{y}'$ that returns a vector of prediction logits $\mathbf{y}' \in [0,1]^m$.*

Next, we define early-stage fraud detection. At time t, we aim to design a framework that will achieve comparable or better performance with the current cross-sectional data than with the cross-sectional data at time T where $T > t$. Formally, our goal is to find a data augmentation framework satisfying the following criteria:

Definition 2. *(Early Stage Fraudulent Behavior Detection) Let $h : (\mathbb{R}^m, \mathbb{R}^m) \rightarrow \mathbb{R}$ be an evaluation metric such that a larger value is more desirable holding other conditions the same. Let g be a fraud detection method. Design a framework $f : \mathcal{G}_t, \mathbf{X}_t \rightarrow \mathcal{G}'_t, \mathbf{X}'_t$ that satisfies $h(g(f(\mathcal{G}_t, \mathbf{X}_t)), \mathbf{y}) \geq h(g(\mathcal{G}_t, \mathbf{X}_t), \mathbf{y})$.*

4 METHODS

In this section, we present our proposed Alfrad approach towards the above problem. We discuss the two key components of the Alfrad framework. We first focus on the fraud detection module; then we discuss the Seq2Seq behavior predictor module.

4.1 Fraud Detection Module

The first component of our proposed Alfrad framework is a fraud detection module. It is worth pointing out that this part of Alfrad is not model-specific: any fraud detection/node classification model(e.g., Fraudar [9], CatchSync [10], GCN[13], GraphSAGE [6]) suffices to be the this component of the framework and could have its performance improved with Alfrad. Without the loss of

generality, let the fraud detection model g be defined as:

$$g : \mathcal{G}, \mathbf{X} \rightarrow \mathbf{y}', \tag{1}$$

where $y' \in [0,1]^m$ is the predicted suspiciousness of the user nodes of being fraudsters.

For graph representation learning methods (graph neural networks) that is capable of learning low-dimensional node representations as well as giving predictions, we also take advantage of the learned representations for user nodes. Without the loss of generality, here we take the widely used Graph Convolutional Network (GCN) [13] as a fraudulent user nodes detection method, because it's capable of semi-supervised node classification. The graph convolution operation of each GCN layer is defined as:

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{2}$$

where l indicates the layer, $\mathbf{H}^l$ is the node embedding matrix generated by l-th layer, $\mathbf{W}$ is the weight matrix of the co-responding layer, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with added self-loops, $\tilde{\mathbf{D}}$ is the diagonal degree matrix $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, and $\sigma(\cdot)$ denotes a nonlinear activation such as the Rectified Linear Unit (ReLU).

Thus the GCN used for fraud detection can be notated as:

$$g_{gcn} : \mathcal{G}, \mathbf{X} \rightarrow \mathbf{Z}, \mathbf{y}', \tag{3}$$

where g_{gcn} is a multi-layer GCN model, $\mathbf{Z}$ is the node embedding matrix generated by the second last layer, and $\mathbf{y}'$ is still the predicted user suspiciousness. We use a standard binary cross entropy loss for training the GCN.

4.2 Behavior Predictor

We aim to explore the user-level sequential data for behavior modelling. In other words, given the item history of each user $u \in \mathcal{U}$, we want to be able to forecast what item this user will post next. For this purpose, we employ an Seq2Seq model as our building block for behavior modelling in order to capture the information embedded in user's item history.

Encoder. We can regard each user's item history as a sequence of features that are constructed as a *prior* knowledge. For example, for content-based items, it could be the embedded representation of the texts, but it could also be other features like in/out degrees. Suppose user u_i has an item history of j items, we can denote the sequence of item history for user u_i as $x^{(i)} = \{\mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_j^{(i)}\}$, of which each item $\mathbf{x}_j^{(i)} \in \mathbb{R}^k$ stands for the feature vector of the corresponding item node. If we have a fraudster prediction model that gives user embedding (GCN etc.), say z_i for user u_i, the item history could also be constructed as $x^{(i)} = \{\mathbf{x}_1^{(i)} \oplus \mathbf{z}_i, \ldots, \mathbf{x}_j^{(i)} \oplus \mathbf{z}_i\}$, where $\oplus$ stands for concatenation. For the ease of reading we omit the user index i in the following of this section, denoting $\mathbf{x}_j^{(i)}$ as $\mathbf{x}_j$. and index i will be mainly referred to as the current time stamp.

We adopt Long Short Term Memory network (LSTM) [8] as encoder to capture contextualized representation for each item in the sequence. The encoder contains only a forward LSTM as we want to make predictions based on the history items. Hence the hidden state for each item is calculated by:

$$\mathbf{h}_i, \mathbf{c}_i = \overrightarrow{\text{LSTM}}(\mathbf{x}_i, \mathbf{h}_{i-1}, \mathbf{c}_{i-1}), \tag{4}$$

where $\mathbf{h}_i$ and $\mathbf{c}_i$ refer to the hidden state and cell state in i-th step respectively.

Decoder. When decoding the hidden features, we build a *two-step* decoder for behavior pattern mining: first, we decide if the user is going to select an item that has already been selected; then we want to minimize the distance between the predicted item and the real item in the feature space. The first step is motivated by our observation that fraudulent users tend to have more repeated items, which could be explicitly modelled through learning. These two tasks are jointly optimized through our *two-step* decoder.

We first decode the hidden states and cell states for each item, the readout operation is defined as following:

$$\hat{p}_i = \phi(\mathbf{W}_{rh} \cdot \mathbf{h}_i + \mathbf{W}_{rc} \cdot \mathbf{c}_i + \mathbf{b}_r), \tag{5}$$

where $\mathbf{W}_{rh} \in \mathbb{R}^{|\mathbf{h}| \times k}$, $\mathbf{W}_{rc} \in \mathbb{R}^{|\mathbf{c}| \times k}$, $\mathbf{b}_r \in \mathbb{R}^k$ are trainable parameters, ϕ is the sigmoid function, and $\hat{p}_i$ is the probability of repeating item, i.e, larger $\hat{p}_i$ indicates that the user if more likely to perform a repeat behavior (e.g., repost a message that the user has posted in the past). Let p_i be the ground truth of repeating behavior where $p_i = 1$ if the next item is a recurring item, and $p_i = 0$ otherwise. Then we define the loss for prediction of repetition as:

$$\mathcal{L}_{rep} = -\sum_{i=1}^{j} \big(p_i \log(\hat{p}_i) + (1 - p_i) \log(1 - \hat{p}_i)\big). \tag{6}$$

Moreover, we use another readout function to decode the prediction of next item by

$$\hat{\mathbf{x}}_{i+1} = \mathbf{W}_p \cdot \mathbf{h}_i + \mathbf{b}_p, \tag{7}$$

where $\mathbf{W}_p \in \mathbb{R}^{|\mathbf{h}| \times k}$, $\mathbf{b}_p \in \mathbb{R}^k$ are trainable parameters and $\hat{\mathbf{x}}_{i+1}$ is the predicted feature vector for the next item. Then the loss function of item prediction can be defined as

$$\mathcal{L}_{pred} = \frac{1}{j} \sum_{i=1}^{j} ||\hat{\mathbf{x}}_i - \mathbf{x}_i||_1^2. \tag{8}$$

Therefore, we can train the behavior predictor by the following loss function:

$$\mathcal{L} = \mathcal{L}_{rep} + \alpha \cdot \mathcal{L}_{pred}, \tag{9}$$

where α is a hyperparameter.

Inference. After the model is well trained, it can be easily used to predict the future items for each user node. However, one question remains, that is, *how many* items should we predict for each user? With the assumption of fraudulent users tend to have higher degree to achieve their goals (e.g., hot topic boosting), we aim to make more predictions for the users that are more likely to be fraudsters. Hence we utilize the predicted suspiciousness scores $\mathbf{y}'$ for each user given by the fraud detection module (Eq. 1/3) to decide the number of predictions we make for each user u_i:

$$n_i = \lfloor y_i' \cdot \kappa \rfloor, \tag{10}$$

where $\kappa \in \mathbb{Z}^+$ is a hyperparameter to control the maximum number of predictions.

During inference, teacher forcing is not used. Hence when each $\hat{\mathbf{x}}_{i+1}$ is predicted by Eq.(7), the next input item is determined by cosine-similarity:

$$\mathbf{x}_{i+1} = \underset{\mathbf{x} \in \mathbf{X}_i}{\text{argmin}} \left(\frac{\hat{\mathbf{x}}_{i+1} \cdot \mathbf{x}}{||\hat{\mathbf{x}}_{i+1}|| \times ||\mathbf{x}||} \right). \tag{11}$$

where $\mathbf{X}_i$ is the current vocabulary calculated by

$$\mathbf{X}_i = \begin{cases} \mathbf{X} & \hat{p}_i \leq 0.5 \\ \{\mathbf{x}_1, ..., \mathbf{x}_i\} & \hat{p}_i > 0.5 \end{cases} \quad (12)$$

We thus obtain a new edge list $\mathcal{E}'$ and are able to update the graph by adding these new edges.

Self-training. As the fraud detection module and the behavior predictor can mutually enhance each other, i.e., the fraud detection results can help behavior predictor to make better predictions and the augmented graph enriched with predicted behaviors can also help fraud detection methods. We make this an iterative process: we first run the fraud detection method on the original graph, use the results together with a trained behavior prediction model to forecast future user behaviors, and augment the graph with the predicted user behaviors. Then we re-run the fraud detection method on the updated graph and repeat the above process for several iterations.

5 EXPERIMENTS

In this section, we evaluate the proposed ALFRAD framework for early-stage fraud detection on a real-world dataset.

5.1 Experimental Settings

5.1.1 Datasets.

Overview. We use a real world dataset constructed from Tencent Weibo, one of the most popular Chinese micro-blog sites between 2010 and 2015. Particularly, the dataset is crawled in November 2011, consisting posts and public user profiles. We construct the graph as a "who-post-what" graph where the user nodes are registered users and item nodes are micro-blog posts. Edges occur between users and posts when the user (re)posted the post. The features for each post is generated from its text with GLOVE[20]; the features for each user is the average features of all connected posts.

Labeling. The data does not come with golden labels, so we utilized rule-base labeling strategy that validated by human labeling. We first sampled 1000 users from the full dataset, along with links to their public profiles. Then several Data Labeler manually labeled these users by reading their post and profile information, and observing their temporal behavior pattern. Specifically, we conclude the following criteria that we followed when labelling users:

(i) *Suspicious Timestamp:* The major conspicuous characteristics of suspicious users is their bot-controlled behavior. It is truly hard for bots to be programmed to have human-like posting pattern. As we observe, suspicious users usually post in a fixed time interval. For example, an user with many posts is identified as suspicious since if 2/3 of her posts are posted with an interval of 10 ± 2 seconds.

(ii) *Deactivated Accounts:* We identify those users whose QQ account and Wechat account are deactivated as suspicious users since Wechat and QQ are two major online communication tools in China. Researchers have shown that over 90 percent of Chinese network users are also Wechat users [24].

(iii) *Malicious post/repost content:* If an account post certain content or malicious links multiple times, we will identify it as a malicious account candidate.

(iv) *Suspicious Username*: We also employ suspicious username as an important clue. For example, if a user has a highly random name, we consider it as a candidate. Then combining other factors, we can usually identify a fraudster confidently.

With these criteria, we coded rules that can reach over 96% accuracy in the sampled 1000 users and labeled the whole dataset with the rules. The final dataset consists of 40235 users in total, within which 3283 users are identified as malicious and 36952 users are identified as benign.

Early-Stage Detection. We manually modify the dataset in order to fit our task. We divide the dataset based on the percentage of user's posts to recreate the temporal evolvement of the graph. For example, a 10% graph is to select the first 10% posts of each user. Table 1 briefly summarizes the statistics of the divided graphs.

Table 1: Graph Statistics (Total # of Users: 40235)

Cut Percentage	Total Posts	Total Edges
20	3126	45024
40	3163	51784
60	3218	59020
80	3250	66866
100	3288	75285

5.1.2 Baselines.

We compare the improvement of our ALFRAD framework over the following fraud detection methods without graph augmentation:

- **FRAUDAR** [9]: An unsupervised fraud detection method with graph mining that aims to address the issue of camouflage.
- **LOCKINFER** [11]: An unsupervised fraud detection method that uses the *lockstep pattern* of fraudsters as a clue for detection.
- **GCN** [13]: An semi-supervised graph neural network. The neural structure has been empirically demonstrated to be useful in the area of fraudster detection [28].
- **GRAPHSAGE** [6]: An inductive graph neural network model that can also be used on semi-supervised node classification talk.

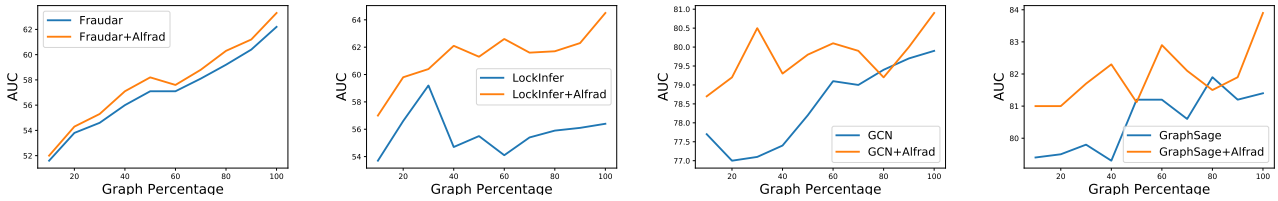
5.2 Experiment Results

Table 2 summarizes the performance of ALFRAD over the baseline methods. We use average precision (AP) and area under ROC curve (AUC) as our evaluation metric. As we can clearly see, our proposed ALFRAD framework is able to significantly improve all the baseline methods. Both graph-mining based methods and supervised graph learning methods are improved. Specifically, ALFRAD improves 1.1%(FRAUDAR), 6.0%(LOCKINFER), 1.2%(GCN) and 1.4%(GRAPHSAGE) on average. We observe that GNN-Based methods perform better than other fraudster detection methods in general for the ablility of graph machine learning models to learn both the topological structure as well as node embedded representations.

Figure 2 presents the results in line plots for clearer analysis. We can observe that in general, ALFRAD is able to accomplish the early-stage fraudster detection task as we defined in section 3. Particularly, for FRAUDAR, we achieve roughly the same performance in the 50% graph with ALFRAD as in the 70% graph without ALFRAD; for LOCKINFER, since the performance without ALFRAD is not monotonically increasing, we cannot draw a comparative conclusion; for GCN, ALFRAD achieves the same level of performance in 10% graph as the baseline GCN in the 50% graph; similarly, despite some irregular behavior in larger graphs, GRAPHSAGE achieves the same level of performance in 20% graph with ALFRAD as in 80%

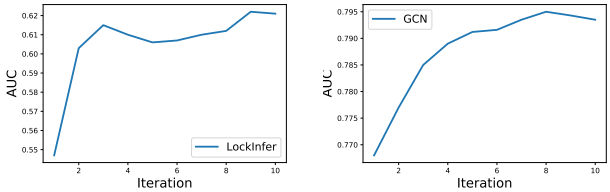
Table 2: ALFRAD performance across different methods and different amount of data.

Graph Percentage	FRAUDAR Original AUC	FRAUDAR Original AP	FRAUDAR +ALFRAD AUC	FRAUDAR +ALFRAD AP	LOCKINFER Original AUC	LOCKINFER Original AP	LOCKINFER +ALFRAD AUC	LOCKINFER +ALFRAD AP	GCN Original AUC	GCN Original AP	GCN +ALFRAD AUC	GCN +ALFRAD AP	GRAPHSAGE Original AUC	GRAPHSAGE Original AP	GRAPHSAGE +ALFRAD AUC	GRAPHSAGE +ALFRAD AP
10	51.6	13.9	**52.0**	**14.3**	53.7	8.5	**57.0**	**9.5**	77.7	32.9	**78.7**	**33.3**	79.4	34.8	**81.0**	**37.7**
20	53.8	16.1	**54.3**	**16.9**	56.6	8.8	**59.8**	**10.4**	77.0	31.6	**79.2**	**33.1**	79.5	35.9	**81.0**	**36.2**
30	54.6	16.2	**55.3**	**17.2**	59.2	9.9	**60.4**	**12.3**	77.1	32.3	**80.5**	**35.9**	79.8	**37.5**	**81.7**	37.3
40	56.0	17.4	**57.1**	**18.5**	54.7	9.4	**61.3**	**12.4**	77.4	32.9	**79.3**	**36.9**	79.0	34.2	**82.9**	**40.1**
50	57.1	18.4	**58.2**	**19.2**	55.5	10.3	**61.3**	**12.4**	78.2	33.8	**79.8**	**38.1**	**81.4**	**40.7**	81.0	38.0
60	57.1	18.4	**57.6**	**19.2**	54.1	10.0	**62.6**	**17.3**	79.1	35.2	**80.1**	**38.5**	81.2	40.8	**82.4**	**41.2**
70	58.1	19.0	**58.8**	**15.4**	55.4	10.0	**61.6**	**15.2**	79.0	37.0	**79.9**	**37.7**	80.5	**39.0**	**82.1**	36.5
80	59.2	18.9	**60.3**	**20.2**	55.9	9.9	**61.7**	**12.9**	**79.4**	**37.4**	79.2	34.2	**82.1**	**41.7**	81.2	33.1
90	59.5	19.4	**60.3**	**20.2**	56.1	9.8	**62.3**	**12.3**	79.7	**39.4**	**80.0**	39.1	81.2	39.5	**82.3**	**40.3**
100	60.4	20.0	**61.2**	**21.3**	56.4	10.4	**64.5**	**13.7**	79.9	**40.2**	**80.9**	39.4	81.4	39.3	**84.0**	**41.5**



(a) FRAUDAR (b) LOCKINFER (c) GCN (d) GRAPHSAGE

Figure 2: Performance measued in AUC change as graph size increases



(a) LOCKINFER + ALFRAD (b) GCN + ALFRAD

Figure 3: Performance measured in AUC change as iteration continues

without ALFRAD. Although for methods like LOCKINFER, the baseline is not monotonically increasing, it is also worth pointing out that for all of the baseline fraudster detection methods, we are able to observe substantial improvements in absolute performance on average, which are evidence for ALFRAD's powerful ability to model the evolution of the graph structure and user-item interactions.

Iteration Convergence. In figure 3, we randomly select two graphs and plot the evolution of the ALFRAD with LOCKINFER and GCN (one unsupervised graph-mining method and one semi-supervised machine learning method). The first is the 8% graph and the second is the 4% graph. Qualitatively, they generally have the same behavior: the performance quickly increases in the first couple of iterations, and oscillates around a value as more iteration continues. We can observe a more smooth line for GCN than for LOCKINFER, which we think is because GCN's capability of learning low-dimension node representations, which are also used in the fraudster detection module. In contrast, the graph-mining based methods like LOCKINFER relies more on the graph topological structure and do not learn any node representations, making the improvement less stable. But still, it is largely consistent with our original supposition that the iterative process will reach a steady state as more iteration happens, demonstrating the mutually beneficial relationship between the two main tasks - fraud detection and behavior forecasting.

6 CONCLUSION

In this work, we propose a novel early-stage fraud detection framework ALFRAD that consistently improve the existing fraudster detection methods in the task of early-stage detection when graph is *temporally* incomplete. In summary, our work manages to model both the node representation and graph topology evolution through behaviour modelling. The experiment results based on the real world dataset demonstrates the effectiveness of ALFRAD on the task of early-stage fraud detection.

ACKNOWLEDGMENTS

This work is supported by National Science Foundation IIS-1849816.

REFERENCES

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[2] Peng Cui, Meng Jiang, and Christos Faloutsos. 2016. Suspicious Behavior Detection: Current Trend and Future Directions. *IEEE Computer Society* (2016).

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[4] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. 2019. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, 594–602.

[5] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. 2018. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1416–1424.

[6] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[7] Mikael Henaff, Joan Bruna, and Yann LeCun. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163* (2015).

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[9] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. Fraudar: Bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 895–904.

[10] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Catchsync: catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 941–950.

[11] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Inferring strange behavior from connectivity pattern in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 126–138.

[12] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2016. Inferring lockstep behavior from connectivity pattern in large graphs. *Knowledge and Information Systems* 48, 2 (2016), 399–428.

[13] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[14] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 333–341.

[15] Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. 2018. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing* 67, 1 (2018), 97–109.

[16] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. 2019. Deepgcns: Can gcns go as deep as cnns?. In *Proceedings of the IEEE International Conference on Computer Vision*. 9267–9276.

[17] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. 2018. Adaptive graph convolutional neural networks. In *Thirty-second AAAI conference on artificial intelligence*.

[18] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5115–5124.

[19] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*. 2014–2023.

[20] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *In EMNLP*.

[21] B. Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Faloustsos Christos. 2010. Eigenspokes: Surprising patterns and scalable community chipping in large graphs. *Advances in knowledge discovery and data mining* (2010).

[22] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[23] Daheng Wang, Meng Jiang, Munira Syed, Oliver Conway, Vishal Juneja, Sriram Subramanian, and Nitesh V Chawla. 2020. Calendar Graph Neural Networks for Modeling Time Structures in Spatiotemporal User Behaviors. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[24] Wikipedia contributors. 2020. Wechat— Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/WeChat [Online; accessed 14-June-2020].

[25] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. *arXiv preprint arXiv:1806.03536* (2018).

[26] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 974–983.

[27] Wenhao Yu, Mengxia Yu, Tong Zhao, and Meng Jiang. 2020. Identifying referential intention with heterogeneous contexts. In *Proceedings of The Web Conference 2020*. 962–972.

[28] 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. *SIGIR (Information retrieval) 20* (2020).

[29] Tong Zhao, Yozen Liu, Leonardo Neves, Oliver Woodford, Meng Jiang, and Neil Shah. 2020. Data Augmentation for Graph Neural Networks. *arXiv preprint arXiv:2006.06830* (2020).

[30] Tong Zhao, Matthew Malir, and Meng Jiang. 2018. Actionable objective optimization for suspicious behavior detection on large bipartite graphs. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 1248–1257.