POSTER: Neural Network-based Graph Embedding for Malicious Accounts Detection

Ziqi Liu[†⋆], ChaoChao Chen[†⋆], Jun Zhou[†], Xiaolong Li[†], Feng Xu[†], Tao Chen[†], Le Song[†‡]

[†]Ant Financial Services Group, Hangzhou, China
[‡]Georgia Institute of Technology, Atlanta, USA
{ziqiliu,chaochao.ccc,jun.zhoujun,xl.li,fuyu.xf,boshan.ct,le.song}@antfin.com

ABSTRACT

We present a neural network based graph embedding method for detecting malicious accounts at Alipay, one of the world's leading mobile payment platform. Our method adaptively learns discriminative embeddings from an account-device graph based on two fundamental weaknesses of attackers, i.e. device aggregation and activity aggregation. Experiments show that our method achieves outstanding precision-recall curve compared with existing methods.

KEYWORDS

Malicious account detection; Graph embedding

1 INTRODUCTION

Large scale online services such as Gmail[1], Facebook[2] and Alipay[3] have becoming popular targets for cyber attacks. By creating malicious accounts, attackers can propagate spam messages, seek excessive profits, which are essentially harmful to the eco-systems.

Many existing security mechanisms to deal with malicious accounts have extensively studied the attack characteristics [1, 3–6] which hopefully can discern the normal and malicious accounts. To exploit such characteristics, existing research mainly spreads in three directions. First, *Rule-based methods* directly generate sophisticated rules for identification. For example, Xie et al. [5] proposed "spam payload" and "spam server traffic" properties for generating high quality regular expression signatures. Second, *Graph-based methods* reformulate the problem by considering the connectivities among accounts. This is based on the intuition that attackers can only evade individually but cannot control the interactions with normal accounts. For example, Zhao et al. [6] analyzed connected subgraph components by constructing account-account graphs to identify large abnormal groups. Third, *Machine learning-based methods* learn statistic models by exploiting large amount of historical data. For examples, Huang et al. [3] extracted features based on graph properties and built supervised classifiers for identifying malicious account. Cao et al. [1] advanced the usages of aggregating behavioral patterns to uncover malicious accounts in an unsupervised machine learning framework.

As attacking strategies from potential adversaries change, it is crucial that a good system could adapt to the evolving strategies [1, 6]. We summarize the following two major observations from attackers as the fundamental basis of our work. (1) *Device aggregation.* Attackers are subjected to cost on computing resources. That is, due to economic constraints, it is costly if attackers can control a large amount of computing resources. As a result, most accounts owned by one attacker or a group of attackers will signup or sigin frequently on only a small number of resources. (2) *Activity aggregation.* Attackers are subject to the limited time of campaigns. Basically, attackers are required to fulfil specific goals in a short term. That means the behaviors of malicious accounts controlled by single attacker could burst in limited time. We illustrate such two patterns from the dataset of Alipay in Figure 1.

The weaknesses of attackers have been extensively analyzed, however, it's still challenging to identify attackers with both high precision and recall[4]. Existing methods usually achieve very low false positive by setting strict constraints but potentially missing out the opportunities on identifying much more suspicious accounts, i.e. with a high false negative rate. The reason is that the huge amount of benign accounts interwined with only a small number of suspicious accounts, and this results into a low signal-to-noise-ratio. It is quite common that normal accounts share the same IP address with malicious accounts due to the noisy data, or the IP address comes from a proxy. Thus make it important to jointly consider the "Device aggregation" and "Activity aggregation" altogether.

In this work, we propose **G**raph **E**mbeddings for **M**alicious accounts, (GEM), a novel nueral network-based graph technique, which *jointly* considers "Device aggregation" and "Activity aggregation" in a unified model. Our proposed method essentially models the topology of account-device graph, and simultaneously considers the characteristics of activities of the accounts in the local strucuture of this graph. The basic idea of our model is that whether a single account is normal or malicious is a function of how the other accounts "congregate" with this account in the topology, and how those other accounts shared the same device with this account "behave" in timeseries. Unlike existing methods that one first studies the graph properties [3] or pairwise comparisons of account

[1]https://mail.google.com
[2]https://www.facebook.com
[3]http://render.alipay.com/p/s/download

*Equal contribution.

CCS'17, October 30-November 3, 2017, Dallas, TX, USA,

ACM ISBN 978-1-4503-4946-8/17/10.
https://doi.org/10.1145/3133956.3138827

[4]https://en.wikipedia.org/wiki/Precision_and_recall

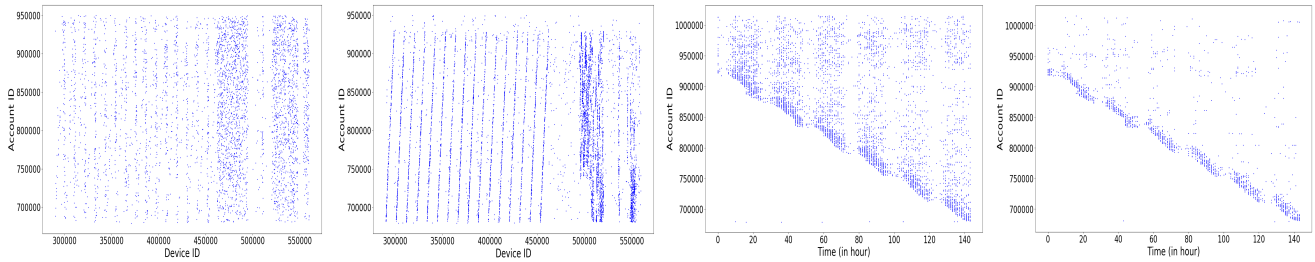

Figure 1: Device aggregation patterns: normal accounts (*first*) v.s. malicious accounts (*second*), malicious accounts tend to aggregate; Activity aggregation patterns: normal accounts (*third*) v.s. malicious accounts (*fourth*), new registered malicious accounts tend to be active only in a short term.

activities [1] then feeds into a machine learning framework, our proposed method directly learns a function for each account given the context of the local topology and other account activities nearby thanks to the representation power of neural networks.

We deploy the proposed work as the real system at Alipay. It can detect hundreds of thousands malicious accounts daily. We empirically show that the experimental results significantly outperform the results from a "Graph-based" system in terms of both precision and recall.

We summarize the contributions of this work as follows:

- We propose a novel neural network based graph embedding method for identifying malicious accounts by jointly capturing two of attackers' weaknesses, summarized as "Device aggregation" and "Activity aggregation".
- Our method is applied at Alipay, one of the largest third-party mobile and online payment platform serving hundreds of million users.

2 THE PROPOSED METHODS

2.1 Problem Definition

Assuming a set of N_c accounts $\{1, ..., i, ...N_c\}$ and associated N_d devices $\{1, ..., j, ...N_d\}$ in our dataset. We observe a set of M edges $\{(i, j)\}$ between accounts and devices over a time period $[0, T)$. Each edge denotes that the account i has activities, e.g. signup, login and so on, on device j. As such, we have a graph $G \in \{0, 1\}^{N_c, N_d}$ consists of accounts and devices as vertices , with edges connecting them. Note that the "device" here could be a much more broad concept. For example, the device could be an IP address, an IMEI ID of a cell phone, or even a like page in facebook. Furthermore, we do not limit the bipartite graph with only one type of devices. Instead, we support more than two types of devices.

Associated with this graph, we can further observe the activities of each account. Assuming a N_c by p matrix $X \in \mathbb{R}^{N_c, p}$, with each row x_i denotes activities of account i. In practice, the activities of account i over a time period $[0, T)$ can be discretized into p time slots, where the value of each time slot denotes the count of the activities in this time slot.

Our goal is to discriminate between malicous and normal accounts. That is, we want to learn a function f, such that given any account i, the graph G and activities X, the learned function f can correctly identify whether the account i is malicious or not, i.e. $f(i, G, X) = y_i \in \{-1, 1\}$, where y_i denotes the true label of account i.

2.2 A Motivating Example

We demonstrate how we can identify malicious accounts intuitively based on the ideas of "Device aggregation" and "Activity aggregation" in this section.

Device aggregation. The basic idea of device aggregation is that if an account connects with a large number of other accounts by sharing the same device or a set of devices, then such accounts would be suspicious. One can simply calculate the size of the connnected subgraph components as a measure for each account as in [6].

Activity aggregation. The basic idea of activity aggregation is that if accounts sharing the common devices behave in batches, then those accounts are suspicious. One can simply define the inner product of activities of two accounts sharing the same device as a measure of affinity, i.e. $S^a_{i,i'} = \langle x_i, x_{i'} \rangle$. Apparently the consistent behaviors over time between account i and i' mean high degrees of affinity. Such measures of affinity between two accounts can be further used to split a giant connected subgraph to improve the false positive rate [6].

The measures in this section are intuitive, however, they are sensitive to noisy data as discussed in section 1.

2.3 Our Methods

As discussed in above sections, to judge whether an account i is malicious or not, it depends on how the other accounts "congregate" around the same device together with account i, i.e. local topology, and also depends on the behaviors of those accounts in time series.

Inspired by a recent work from Dai et al. [2] that it builds connections between neural networks and graphical models [2] for the modeling of relational data. They show that one can effectively learn the sufficient statistics (embeddings) of each structured data (itself a graph) by considering the attributes of each node and their relations in the graph.

In our problem, we treat each vertex (i.e. account or device) in graph G as a data point, with edges in G denote the relations among those data points. We hope to learn effective embeddings μ_i for each vertex i based on "Device aggregation" and "Activity aggregation" by propagating transformed activities X in local topology of graph G:

$$\mu_i^0 \leftarrow 0 \quad \text{for all } i \tag{1}$$
$$\text{for} \quad t = 1, ..., C$$
$$\mu_i^t \leftarrow \mathcal{T}\left(W x_i + V \sum_{j \in \mathcal{N}(i)} \mu_j^{t-1}\right)$$

where $\mathcal{T}$ denotes a nonlinear transformation, e.g. a rectifier linear unit activation function, $\mathcal{N}(i)$ means the set of i's neighbors, $W \in \mathbb{R}^{k,p}$ and $V \in \mathbb{R}^{k,k}$ are parameters to control the "shape" of the desired function given the connectivities and related activities of accounts, with the hope that they can automatically capture more effective patterns compared with the intuitive patterns discussed in section 2.2. We let k denote the embedding size, and empirically set $k = 16$ and $C = 5$.

To effectively learn W and V, we link those embeddings to a standard logistic loss function:

$$\min_{W,V,u} \mathcal{L}(W,V,u) = -\sum_{i=1}^{N_c} \log \sigma\Big(y_i \cdot (u^\top \mu_i)\Big) \tag{2}$$

where σ denotes logistic function $\sigma(x) = \frac{1}{1+\exp -x}$, and $u \in \mathbb{R}^k$.

Our algorithm works interatively in an Expectation Maximization style. In e-step, we compute the embeddings based on current parameters W, V as in Eq (1). In m-step, we optimize those parameters in Eq (2) while fixing embedings.

3 EXPERIMENTS

3.1 Datasets

We depoly our method at Alipay[5], the world's leading mobile payment platform served more than 450 millions of users since 2013. Our system targets on millions of new registered accounts daily. We will not reveal some detailed numbers due to information sensitivity.

In our system, to predict new registered accounts daily, we train our model using data from past 7 days. This leads to several millions of accounts, tens of million associated devices, and billions of edges between those accounts and devices having activities like "signin" and "signup".

To get the activities x_i, we discretize the activities in hours, i.e. $p = 7 \times 24 = 168$ slots, with the value of each slot as the counts of i having activities in the time slot.

As a result, we get the million by tens of million sparse matrix G with billions of non-zero entries. After we discretize the activities of each account, we get the million by 168 matrix X.

3.2 Experimental Settings

We describe our experimental settings as follows.

Evaluation. Alipay first identifies suspicious accounts and observes those accounts in a long term. Afterwards, Alipay is able to give labels to those accounts with the benefit of hindsight. In the following sections, we will evaluate the precision and recall curve on such "ground truth" labels.

Comparison Methods. We compare our methods with a baseline method, connected subgraph components, which is similar to the methods in [6]. The method first builds an account-account graph, and we define the weight of each edge as the inner product of two accounts x_i and $x_{i'}$ described in section 2.2. The measure of such affinity can help us split out normal accounts in a giant connected subgraph, to further balance the trade-off between precision and recall.

[5]https://en.wikipedia.org/wiki/Ant_Financial

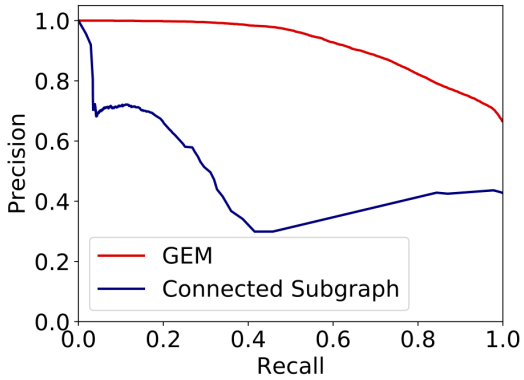

Figure 2: Precision-Recall curves on test data.

3.3 Results

We report the comparison results of all the methods in Figure 2. As we can see, our proposed method GEM significantly outperforms the connected subgraph methods in terms of both precision and recall. One of the largest connected subgraph consists of a total of 800 accounts aggregating together in our experimental dataset, that leads to high precision at low recall from the results of the connected subgraph method. However, it is extremely hard for such methods to retain consistent high precision/recall curves when the size of identified connected subgraphs tends to be small. On the other hand, GEM can automatically learn topological and activity patterns from data and results into a consistent high precision/recall curve.

4 CONCLUSION

In this paper, we present a real system deployed at Alipay, for detecting malicious accounts. We summarize two fundamental weaknesses of attackers, namely "Device aggregation" and "Activity aggregation", and naturally propose a neural network method based on account-device graphs. Our methods achieve promising precision-recall curves compared with existing methods. In future, we are interested in building a real-time malicious account detection system based on dynamic graphs instead of the proposed daily detection system.

REFERENCES

[1] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering large groups of active malicious accounts in online social networks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 477–488.

[2] Hanjun Dai, Bo Dai, and Le Song. 2016. Discriminative embeddings of latent variable models for structured data. In *International Conference on Machine Learning*. 2702–2711.

[3] Junxian Huang, Yinglian Xie, Fang Yu, Qifa Ke, Martin Abadi, Eliot Gillum, and Z Morley Mao. 2013. Socialwatch: detection of online service abuse via large-scale social graphs. In *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*. ACM, 143–148.

[4] Gianluca Stringhini, Pierre Mourlanne, Gregoire Jacob, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. 2015. Evilcohort: detecting communities of malicious accounts on online services. USENIX.

[5] Yinglian Xie, Fang Yu, Kannan Achan, Rina Panigrahy, Geoff Hulten, and Ivan Osipkov. 2008. Spamming botnets: signatures and characteristics. *ACM SIGCOMM Computer Communication Review* 38, 4 (2008), 171–182.

[6] Yao Zhao, Yinglian Xie, Fang Yu, Qifa Ke, Yuan Yu, Yan Chen, and Eliot Gillum. 2009. BotGraph: Large Scale Spamming Botnet Detection.. In *NSDI*, Vol. 9. 321–334.