Fraud Detection in Social Network

Yunjian Yang
Courant Institute of
Mathematical Science
New York University
New York City, New York
yunjian.yang@nyu.edu

Nan Liu
Courant Institute of
Mathematical Science
New York University
New York City, New York
nl1554@nyu.edu

Xinya Zhao
Courant Institute of
Mathematical Science
New York University
New York City, New York
xz1863@nyu.edu

***Abstract*—It is a common phenomenon that social media users buy crowds of fake followers or merchants forge fraud online reviews for their products. This project introduces a fraud detection tool for real-world social media platforms, which can be used to identify and distinguish fraud accounts and reviewsso that to assist platform providers to keep social network communities healthy.**

***keywords* — Analytics, Social Network, Fraud Detection**

I. Introduction

The prevalence of social media platforms drives people to be addicted to the information transmitted by these platforms, whereas there is no guarantee for the quality and fidelity of this information. For example, most of the followers of an online celebrity from Twitter or Google+ may be fake. With so many followers, which are bought from a third party, this seemingly famous celebrity is very attractive for advertisers who seek to utilize celebrity effects to sell their products. At the same time, politicians may buy fake followers on popular social media to put up a false front that they have legions of supporters. Also, some merchants are employing professional writers to post particular reviews on online stores, like Amazon and App Store, or online review publishing platforms, like Yelp and Airbnb, to give their audience a false impression that they enjoy good reputations among customers. Even some startup social network service providers, aiming at enlarging the number of platform users, create large amounts of fake accounts on their platforms, so that they can improve the company valuation and attract more investors attention.

This project is aimed at detecting fraud situations described above on several social media platforms. The input of this project is the network graph of users and their followers, or the network graph of products and their corresponding reviews from online stores. The data will be stored in the form of bipartite graphs and then leveraged to induce fraud cases of the social network. We will also make further efforts to optimize this analytic tool by identifying camouflaged users who add reviews or follows to honest targets in order to make themselves look normal. The output of this project is a subset of the input data, which is composed of the most likely fraud accounts or reviews.

II. Motivation

The manipulation of fraud accounts or reviews harms the credibility of information on social media and the interests of advertisers and investors. The detected fraud accounts or reviews can be utilized not only by social network providers to evict fake accounts and harmful information but also by advertisers and venture capital providers who are seeking trusty celebrity accounts to deliver advertisements or trustworthy setups to perform new investments.

III. Related Work

Different methods are proposed to detect fraud behaviors in social network. Most of these methods are based on how to partition dense graphs.

Moses Charikar[1] introduces the greedy approximation algorithm for finding the density of undirected and directed graphs, which is based on the definition that the density of the graph is the maximum value of its all induced subgraphs average degrees. The algorithm works iteratively to select a subgraph with the largest density compared to other subgraphs. In each iteration, it finds and evicts the vertices with the minimum degree, and then computes the density of the subgraph with rest vertex. When the set of vertices is finally empty, the iteration is done. The subgraph with the maximum value among these densities computed in iterations is the set of maximum average degree. The greedy approximation algorithm runs in linear time, which optimizes computing efficiency.

Karypis, George[2] propose another partition algorithm. He introduces a new (at least its new 20 years ago) way to partition unstructured graph as graph partition plays an important role in many applications. Besides, graph partition is also important on linear equation solving, matrix multiplication, and so on. The advantage of this graph partition algorithm is fast and high quality. This method can be used to find the dense subgraph of the original multi-graphs. Qiang Cao[3] finds a method called SybilRank to help social online services provider like Facebook, Google+, etc., find out fake accounts in large social networks in order to improve their services performance and advertising value. This method, which is based on the theory of undirected social graph and early-terminated random walk, were implemented by MapReduce framework. It performs power iterations to distribute trust value through initial trust

users to all the other users. The number of iteration steps is calculated from the mixing time of the social network graph and should reach the stationary distribution. It also partitions the whole network into communities and annotates the portion of fakes in different communities to decrease the probabilities of the situation which mistakes the real users with fewer connections for fake users. This approach largely improves the accuracy and efficiency of fake accounts identification automatically, which reduces the manual cost.

Hooi and Bryan[4] invent new metrics and introduce a new algorithm which can detect camouflage fraud. As people are drowning in information and starving for knowledge, many people or organizations are trying to manipulate the information to control our mind. The data are stored in the form of a bipartite graph (also called a bigraph, is set of graph vertices decomposed into two disjoint sets such that no two graph vertices within the same set are adjacent). They invent a new metric and try to make an optimization. After training, the method can get a subgraph which may contain the fraud action of a certain social media. This algorithm is effective, and much more importantly, linearly scalable.

To compare different fraud users detect algorithm, Neil Shah and Alex Beutel[5] firstly gives a brief introduction of related works to detecting fraudulent users, including spectral methods, graph traversal methods, and feature-based methods. And then from its own perspective, the paper examines the above methods from an adversarial point-ofview and presents lemmas and theorems showing their vulnerability to intelligent attackers. Figuring out the condition, which, once met, will lead to evasions from spectral methods, the paper analyzes three types of attack strategies and evaluates the suitability of each attack for an adversary, and finally draws the conclusion that existing detecting techniques for catching fraud have firm effective detection thresholds and are entirely ineffective in detecting stealth attacks (small-scale attacks) that fall below this threshold. Then in the next part, the paper formally identifies the existing problem and proposes its own algorithm, FBOX, for addressing this problem. The algorithm uses a solely graph-based method, which will be able to complement existing fraud detection techniques by discerning previously undetectable attacks. At the last part, the paper presents the experiments they have implemented using FBOX on Twitter and Amazon graphs. In conclusion, this paper focused on spotting fraudsters and their customers in online social networks and web servicesit presents a complementary method to existing spectral approaches that detects stealth attacks that previous methods miss and proves the effectiveness of its method on real data.

IV. Design

We'll first provide an overview of the framework for our analytic project, then describe each part in more details.

A. an Overview of the Analytic Framework

The data flow design model for this project is shown in Fig. 1. The whole project is implemented based on Hadoop framework utilizing various big data storage and processing tools of the Hadoop ecosystem. According to the roles that different parts of the system play, the project construct could be partitioned into three layers, the bottom layer for data storage, the medium layer for data processing and analyzing and the top layer for project management. The raw data streamed in through various social media APIs and will be firstly stored in HDFS; then different data processing tasks from the middle layer, which follows the general standard diagnostic model of big data, will interact with the storage layer sequentially and finally generate the ready-to-output result data; lastly, the result data will stream out of the system and can be used for further analyzing such as visualization and so on. Whats more, the whole process will be automatically or manually supervised and managed by the third layer.

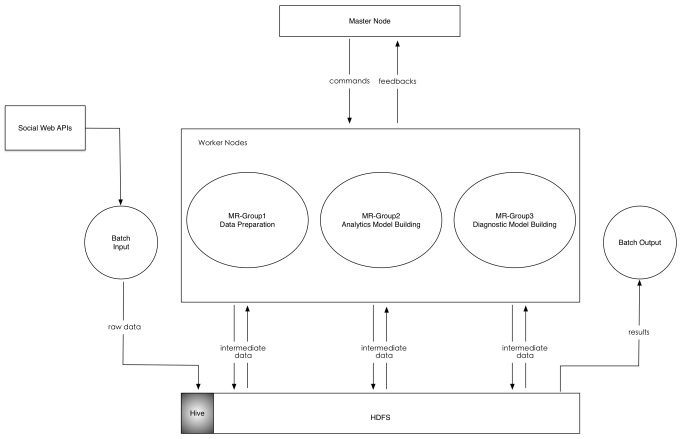

Fig. 1. Data Flow for our Project

B. Data Collection

Due to the restriction of API, we have three data source for our project: Twitter®, Facebook®, Google+®. The size of data set tested by our program is almost 100MB for each data source. With HBase and Other No-SQL database, a larger data set can also be used in this framework.

In the data source from Twitter® and Google+®, there is a natural follower-followee relationship, which can be applied directly into our framework. Each user has a unique user ID, such as *@realDonaldTrump*. For convenience, we hash these unique user ID into natural numbers with a prefix i for the follower and j for the followee in this paper. For example, i_5 means a follower which is numbered as 5 in our program, while the original user ID is omitted for computation convenience.

In the data source from Facebook®, there is no natural follower-followee relationship. The *Page* for a certain organization plays the role of followee in the previous two data set. The user who *likes* the *page* plays the role for follower.

These data are stored as a bipartite graph in our storage solution. The followers constitute the first part and the followees constitute the second part of this bipartite graph, with edges from the first part points to the second part. So the relationship

of follower i_5 follows followee j_6 is expressed by an edge from vertex i_5 toward j_6 in this graph. The relationship of the whole social network is characterized by Fig. 2.

Although all the three data sources are from social networks service provider, the framework of our project can also be applied to local search, business ratings and reviews service. In this case, just replace the follower and followee in this paper by the reviewer and the reviewed object in these services.

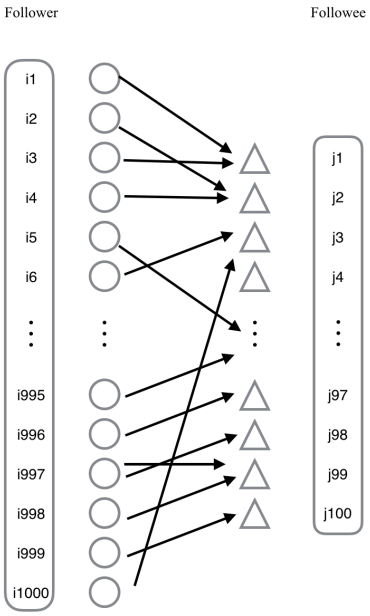

Fig. 2. follower-followee relationship in the bipartite graph

C. Algorithm

The process of how to analyze these data is inspired by the work of Cao[3]. Before introducing this algorithm, we need to know why the semantic analysis is not enough and why network analysis is difficult.

With the accuracy of classification algorithm improving, the behavior of fake users becomes more and more close to a normal user. There is no way to identify whether a review is written by a fake user if we only look at the semantic meaning of this review. How can we classify a review as a fake review if it looks like the same of other reviews? The only way to handle this problem is to consider the whole network as a single graph and find some patterns in this graph.

The remaining question is how to define such patterns. There is a topic named dense graph mining[2][7], aiming at finding patterns in a dense graph.With this idea, unexpected dense graphs have the high probability to be frauded networks. So we just need some data mining techniques to find the unexpected dense graph.

The work of Cao and Qiang[3] provides a novel model for this process, with theoretical bounds on the final outcome. Their model define the density of suspiciousness of a given graph S as $g(S) = \frac{f(S)}{|S|}$. where $|S|$ is the number of verteces in this graph and $f(S)$ is defined as below

$$f(S) = f_v(S) + f_e(S) = \sum_{i \in S} a_i + \sum_{i,j \in S,(i,j) \in e} c_{ij}$$

where a_i is the suspiciousness of a single vertex I, and c_{ij} is the suspicousness of the edge between vertex i and vertex j.

In this model, the suspiciousness of a graph contains two parts. The first part is contributed by the suspiciousness of vertices and the second part is contributed by the relation between these vertices. We can get the density of suspiciousness if we divide the suspiciousness by the scale of this graph. The subgraph has the largest density of suspiciousness can be unexpected dense subgraph in the original network.

The mathematical definition of a_i and c_{ij} can varies. In our analytic project, we have tried several different expressions for a_i and c_{ij}. The result can be very different depends on how you choose these expressions. The best result can be obtained when the c_{ij} is defined as

$$c_{ij} = \frac{1}{\log_2 d_j + Constant}$$

where d_j is the indegree of object j.

Having made it clear that our object is to find a subgraph with the most density of suspiciousness. We must discuss the algorithm for this job.

This is a greedy algorithm which will delete a most innocent vertex from this social network each time it iterates through this graph. Each deletion will result in a new subgraph with its own density of suspiciousness. We choose the subgraph with largest density of suspiciousness as the final output. This algorithm is written below:

Algorithm 1 Algorithm for finding the most suspicious subgraph

Input: A bipartite G graph for social network
Output: The most suspicious fraud sub network
Initialisation :
1: Get the indegree for each vertex in the followee set
LOOP Process
2: **while** G is not empty **do**
3: Get the value for a_i and c_{ij} from the indegrees of G;
4: Delete a vertex which will cause the suspiciousness of remaining graph as large as possible;
5: Store the suspiciousness of the new subgraph and the new subgraph in an array.
6: update the indegree for each vertex in the new graph
7: **end while**
8: **return** the graph with the largest density of suspiciousness.

This algorithm is guaranteed to find a subgraph whose density of suspiciousness is larger than the half of the optimum value[3].

V. Result

We designed experiments to answer the following questions:

- Can we discover fraud relationships in a social network?
- Can we detect camouflaged fraud users, e.g. a zombie fan or a review pretending to be normal?
- Can we use this model to process real-world big data whose volume may be a factor of PB or EB.

We tested our analysis system effectiveness from two perspectives, the choice of kernel functions and the implementation on real-world data sources. For the kernel function, because we test the impact of the system on synthetic data, the outcome of which is easy to observe, we did not introduce a systematic method to evaluate the correctness of the outcome. But for the data source experiments, which refer to real world big data, we performed a scientific evaluation for the system accuracy.

According to the design of our algorithm, we know that the kernel function has an impact on the first two questions above. We test our system firstly with function

$$c_{ij} = f(d_j)$$

where c_{ij} is the edge cost in the graph and $f(d_j)$ is a linear function of the in-degree of node j, the outcome of our experiment is the sub-graph with the largest edge density of the original graph. In particular, the outcome contains both fake users and innocent popular accounts, the latter of which is not what we want. Then we chose the function

$$c_{ij} = \frac{1}{\log d_j + Constant}$$

for the second experiment, the outcome of it is much closer to the true answer. The innocent popular accounts are eliminated from the outcome effectively while the fraud accounts, whose followers are much less than those popular accounts are preserved in the final outcome. Thus, according to the goodness of results from these experiments, we decided to take the second function as our kernel function for future experiments.

Before presenting system implementation and evaluation results on real-world big data sources, it is necessary to introduce how we performed the evaluation. We proposed two criteria for true fraud user judgment. The first is based on the following characteristics of users profile data, which are based on established criteria in former literature.

- links on profile associated with malware or scams
- clear bot-like behavior (e.g. replying to large numbers of status with identical messages)
- account deleted
- account suspended

The second one is based on the fact that a users status containing the URLs of two known follower-buying services, FollowME and FollowerGetter, which indicate that this user had advertised relevant services via this account.

For both of the criteria, we used hand-labeling to mark the true fraudulent users. And it is worthy of being noticed that our criteria are merely based on profile and status data but not follower-followee network data, whereas the analytics of the system uses only the network data, so the design of these criteria can be regarded as a fair estimate of the system results accuracy.

Because of the large scale of the whole detected graphs formed by data sources, we implemented the evaluation on sample data of the detected graphs for further investigation. We firstly run our system on the three data sources and got the detected suspicious sub-graphs respectively. Then we randomly sampled 200 followers and 200 followees in each graph detected by the system. Thirdly we hand-labeled the sample graphs according to the two criteria respectively to determine how many of them appear to be fraudulent. For comparison, we also built a control group of size 200, containing users that were not detected by the system, and did the same processing of hand-labeling towards this group.

The results of system evaluation on three real world data resources are shown in the Figure. 3 and Figure. 4. According to the results, we can present two pieces of evidence relevant to the previous criteria demonstrating the systems effectiveness for true fraud detection. The first one is that for the first criteria, the average percentage of the three data sources of being labeled to be fraudulent arrived at 59% among detected followers and 44% among detected followees. However, in the control group, the rate was much lower, which was only 10% for all users. The evidence for the second criteria is that users detected by the system are much more likely to be with status advertising the follower-buying services, with an average percentage to be 38% for detected followers, and 20% for detected followees. However, when we experimented this on the control group, there was only one mention of FollowerGetter for all users.

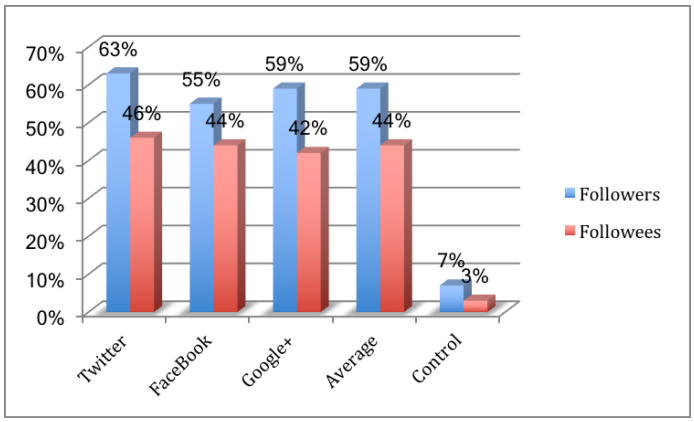

Fig. 3. Fraudulence Percentage of Detected Graph Using Criterion 1

VI. Future Work

In our method, the most suspicious account subset in a given user group, we could expand our analytic by combining our method with some potential approaches.

One potential way is to analyze users relationships. Some social platforms pay special attention to users with no followers as they think it is an abnormal phenomenon under

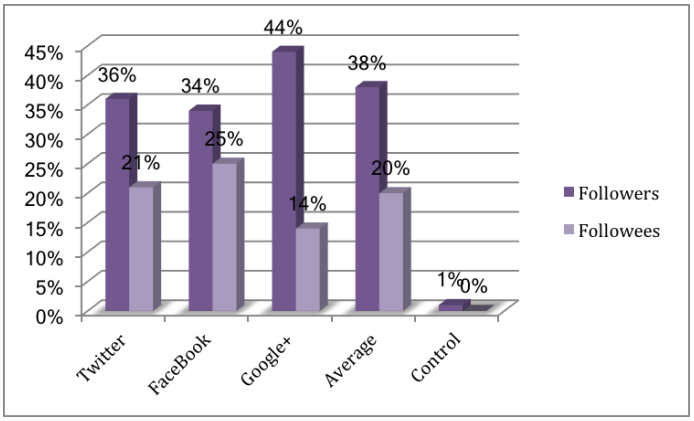

Fig. 4. Fraudulence Percentage of Detected Graph Using Criterion 2

the social network pattern. To avoid this kind of detection, fake accounts in the same group will build relationships with each other. Thus, a suspicious user is more likely to be a fake account when he/she has a nontrivial number of friends or followers who is also suspicious. Based on this connection, we could perform further analysis to classify accounts outputted by our method in different suspicious layers.

We could check user and his friends addresses, phone number and other personal information. In the realistic environment, a user who lives in New York City has the low possibility that all his/her friends come from Buenos Aires. Also, it is reasonable for us to deduce that users become more suspicious and need to be further inspected if his/her friends phone numbers belong to some countries without real-name requirements, or the user and his/her friends are scattered around the world but their phone number belong to the same region.

Detecting users IP address will get some useful information. This method has a similar concept with the previous one, as a group of fake accounts always have the same IP addresses. Hence, if we have found some user on an IP address is fake, other accounts on this IP address should be suspected.

Another approach is to analyze the frequency of users actions. Via observation on the fake account action patterns, we find that the users with following features are possible to be fake: adding considerable number of friends, following lots of accounts, and continuously publishing contents on various kinds of topics during a short period, or a group of users who always log in into a platform and publish similar contexts on same topics at the same time.

And we can verify users profile picture. Fake accounts usually use pictures obtained online to pretend that they are real users. We could distinguish suspicious fake accounts by using image search tools combined with picture comparison technology.

Users nickname should also be paid attention to. Many fake users never change their nicknames. It is worth to check the account with the initial nickname that automatically assigned by social platform.

Additionally, the Natural Language Processing technology can be used to analyze users published context. Fake users try to organize their contexts by learning from real users to make them look like normal accounts. This is more commonly seen in review platforms such Yelp and Amazon. Fake users published praises to convince other customers. However, there are still some patterns is those kinds of contexts. For example, it is more convenient for a fake account use similar context for different restaurants on Yelp, so the context will be more general and the specific feature of restaurants will be blurred. Via this analytic technique, it could be easier to detect fake account whose context share same patterns.

Combined with these or part of these approaches, the found suspicious accounts will be more accurate which make the manual verification of accounts more efficient. Our analytic method could be further implemented in a website or an APP to provide search and clear functions for related parties to look up whether a user is a fraud and delete them if so.

VII. Conclusion

In this project, we designed and implemented a fraud detection algorithm to distinguish suspicious accounts on social media platforms. Our analysis system runs on Spark with Scala, which realizes both the effectiveness and efficiency for the analysis and comparison for graph density, which is the major component of our algorithm. The reasonableness and robustness of the algorithm can be demonstrated in the following aspects. It is taken into consideration during the algorithm design stage that suspicious users may camouflage themselves in multiple ways, which leads to the utilization of directed graph for input data. Furthermore, the resilience of the algorithm to the effect of fraud users' factitiously added relationships with honest users enables the system to be able to effectively distinguish the camouflaged accounts that are pretending to be normal by building up connections with legitimate users. What's more, the optimized selection of the kernel function for the algorithm, which effectively limits edge weight of the graph, works to eliminate the possibility of misjudgment for honest users who are with a large amount of followers and so caused big contribution to graph density. Apart from illustration of the algorithm, we also elaborated our system implementation, performance and evaluation on real world big data, and proposed some potential optimization approaches for the future research in this paper. This fraud detection tool has been proved to be applicable and practical for various network data sources, including three currently most popular social media platforms, Twitter, FaceBook, and Google+. We believe that this system could be used to effectively detect and exclude fraud accounts and make a contribution to the construction of healthy social network communities.

Acknowledgment

We thank Bryan Hooi, Neil Shah, Meng Jiang, Qiang Cao, Moses Charikar, George Karypis for introducing us the theories and algorithms associated with this topic. We also would

like to thank Twitter, Facebook, and Google+ for providing API. We are grateful to Professor McIntosh for her feedbacks on our project and HPC team for their support during our use of HDFS and Spark.

REFERENCES

[1] Charikar, Moses. "Greedy approximation algorithms for finding dense components in a graph." *International Workshop on Approximation Algorithms for Combinatorial Optimization. Springer Berlin Heidelberg, 2000.*

[2] Karypis, George, and Vipin Kumar. "METIS–unstructured graph partitioning and sparse matrix ordering system, version 2.0." (1995).

[3] Cao, Qiang, et al. "Aiding the detection of fake accounts in large scale social online services." *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12). 2012.*

[4] Hooi, Bryan, et al. "Fraudar: bounding graph fraud in the face of camouflage." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2016.*

[5] Shah, Neil, et al. "Spotting suspicious link behavior with fbox: An adversarial perspective." *2014 IEEE International Conference on Data Mining. IEEE, 2014.*

[6] Jiang, Meng, et al. "Inferring strange behavior from connectivity pattern in social networks." *Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer International Publishing, 2014.*

[7] Giatsidis, Christos, Dimitrios M. Thilikos, and Michalis Vazirgiannis. "Evaluating cooperation in communities with the k-core structure." Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on. IEEE, 2011.

[8] T.White.Hadoop: The Definitive Guide. OReilly Media Imc., Sebastopol, CA, May 2012

[9] Apache Hadoop. http://hadoop.apache.org/

[10] Twitter Developers Platform. https://dev.twitter.com/

[11] Facebook For Developers. https://developers.facebook.com/

[12] Stanford Large Network Dataset Collection. http://snap.stanford.edu/data/index.html