Bitcoin Transaction Prediction

Michael Yitayew
Princeton University
`myitayew@princeton.edu`

Abstract

In this paper, we investigate graph based models for Bitcoin transaction prediction. Specifically, we treat the problem of determining future interactions in a bitcoin network from the current structure of the network as a *Link Predicition Problem*[1], where given a graph G at time t, the goal is to predict what the graph will be at some later time t'. We implement techniques from link prediction including *Common neighbors* and *Jaccard similarity* to predict future transactions in the bitcoin network, and achieve a max ROC area under curve of 0.68 for this binary classification problem.

I sense you have some ORFE background

neat

1 Introduction

Bitcoin is a decentralized virtual cryptocurrency. It has reached a high level of prominence in recent years, with a total market cap of seven billion dollars and the acceptance of Bitcoin payment by big name companies such as Amazon and CVS. While every Bitcoin transaction is publicly logged[3] on the *blockchain*, the Bitcoin system can be used as an anonymous way to transact money because only addresses are recorded. This anonimity has lead to use of Bitcoin by illicit groups online such as the drug marketplace SilkRoad and increased research in de - anonymizing Bitcoin transactions. BitCoin de anonymization is possible because of the reuse of addresses(sender/receiver identities). By observing transaction patterns of addresses, it is possible to predict the identity of a sender/receiver, so much so that projects are now underway such as *Zerocoin* at Johns Hopkins university, to make the BitCoin system completely anonymous. Our project is related to de anonymization: we seek to predict whether a transaction between address s and r in the BitCoin network will occur, by observing the current state of the BitCoin network. This goal is important to the overall aim of de-anonymization, because highly likely transactions can be assumed to have taken place over another channel and can be used as additional input to infer the identitfy of the sender/receiver.

Good introduction and related work section.

2 Related Work

Bitcoin de-anonymization has been studied since the deployment of the crypto-currency in 2008. Pfitzmann[2] defined Bitcoin anonymity as anonymity of a subject from a set in the Bitcoin network. [4] studied the anonymity weaknesses of the network and techniques for uncovering identities from Bitcoin addresses, and even identified addresses used with SilkRoad and Mt. Gox, an online Bitcoin exchange platform. On the techniques side, various results from inference on graphs and matrix completion can be used for the purposes of Bitcoin de-anonymization. E. Candes and B. Recht[5] came up with methods of optimal matrix completion that work as long as the adjacency matrix of a network is sampled with sufficient frequency. Liben-Nowell and Kleinberg studied the link prediction problem for social networks and proposed various methods to predict future links based on current network structure. Most of the techniques we use here are adapted from their paper, *The Link Prediction Problem for Social Networks*[1].

3 Methods

3.1 Data Exploration

The BitCoin transaction data is a set of transactions between pairs from among 444,075 users. There are 3,348,026 transactions, in which 443,652 senders and 439,602 receivers participate. The most salient feature of this data is the transaction sparsity; out of the almost $2\times\ 10^{11}$ transactions that can occur, only $0.001\ \%$ are given. This makes in-memory matrix based methods infeasible and requires computation to be done in sparse representation format. As an undirected graph, the BitCoin network is almost connected, with 443954 users belonging to the same connected component. The average transaction is about 5 bitcoins with comparable average in-trasactions and out-transactions. The test data is a set of user pairs from a later time with the binary label of whether a transaction occured between the two users by that later time. The test data has 10 % positive instances and 90 % negative instances; thus the ”percent of correct classifications” metric does not describe the accuracy of prediction methods. Other metrics must be used.

3.2 Transaction Prediction Methods

We model the BitCoin transaction prediction problem as the Link Prediction Problem[1]: given a social network graph G at time t where the edges are interactions, what interactions are likely to have occured by time t' in the near future. This is because the BitCoin network **is a social network**; the users are people who upon interaction become more ”similar”. 99.9 % of all users in the BitCoin network are in the same buyer/seller group (connected component) and like celebrities in a social network, few users in the Bitcoin network have very high degree. We use the following methods to predict Bitcoin transaction, given two users s and r and the current Bitcoin network G.

- *Simple Connectvity:* Predicts True if s and r are in the same connected component
- *Within K hops:* Predicts True if the distance between s and r is at most K in G Should be well motivated
- *N Common Neighbors:* Predicts True if s and r have at least N neighbors in common
- *Jaccard Similarity:* Predicts True if the jaccard similarity of s and r is above set threshold
- *Simple Drug Network:* Predicts True if s and r fit the roles of participants in a hypothetical drug marketplace assumed over the Bitcoin network
- *Drug Network:* Predicts True if s and r fit drug network roles or are jaccard similar.

Impressive - did you decide to do that to deal with the sparsity?

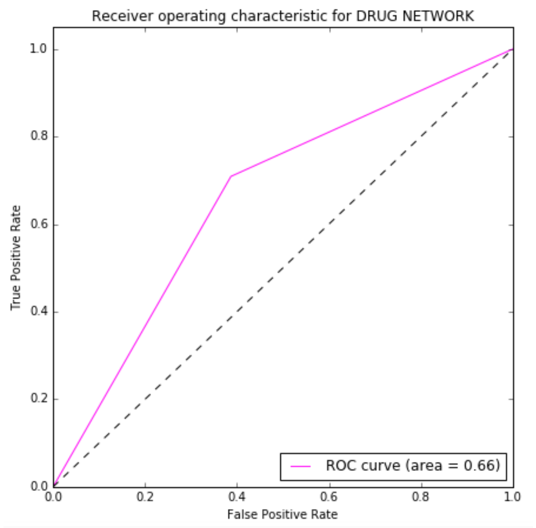
These methods were implemented from scratch in C++. Implementation details are in the code appendix. To make a prediction for any pair (s, r), a method in the above list needs to look at the current Bitcoin network G. In that sense, there is no training phase; that is, the training data is not used to create a model that predicts whether a transaction occured between s and r. A prediction method makes its prediction by looking at the relationship of s and r in the current Bitcoin network. Related methods include Pagerank and random walker based prediction. We omitted these methods because their computation times are very high and can only be used on graphs of about 100 edges in reasonable time.

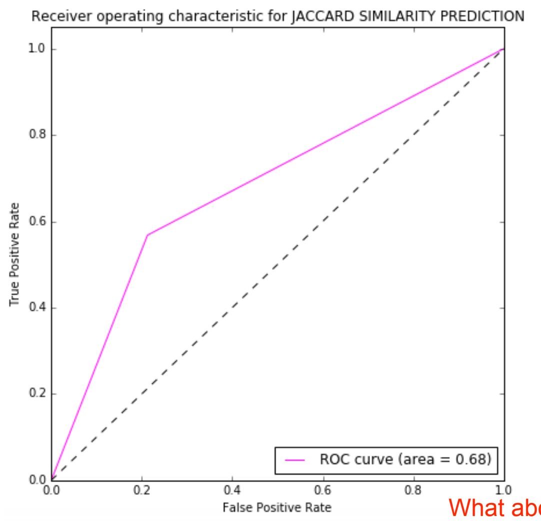
Have you tried, or got this from some reference? A statement like this should include a reference or explanation via theoretical run times

Of these methods, the first four ignore the directedness(sender/receiver distinction) and weightedness of the network to make predictions. **Within K hops** predicts true if s and r are at most K interactions apart. The **neighbors** of a user are users who have sent to or received from that user. Let ϕ(x) for a node x denote the neighbor set of x. **N Common Neighbors** predicts True if $|\ \phi$(s) $\cap\ \phi$(r) $|\ \geq$ N. Then, the **Jaccard similarity** of (s, r) is $|\phi(s)\cap\phi(r)|/|\phi(s)\cup\phi(r)|$. The **drug network model** assigns to each of s and r one of three roles $\{$Supplier, Middle-man, Customer$\}$ based on the user’s ratio of bitcoins sent to bitcoins received. It then predicts True if s is a Supplier and r is a Middle-man, if s and r are Middle-men or if s is a Middle-man and r is a customer.

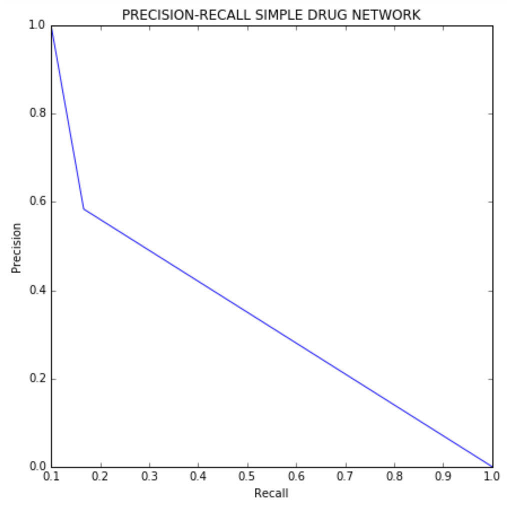
4 Results

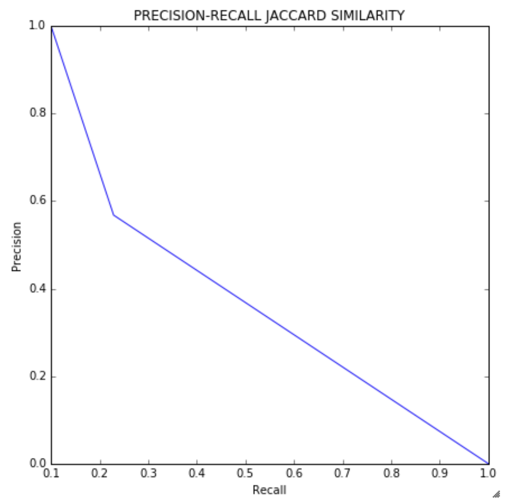
The results of our methods are below. Accuracy1 refers to the percentage of correct predictions on the test data. Accuracy2 is the percentage of correct predictions on 10 % of the test data that is all positive(transaction occured in all test instances). K hops was tested for **K = 3**. K





What about other methods?





was chosen based on the premise that interactions that are weaker than a direct send receive but stronger being in the same connected component can help predict future transactions. N nearest neighbors was tested for **N = 2**. The threshold for Jaccard similarity was set to 0.00001, chosen to be a magnitude smaller than median jaccard similarity of transacting pairs in the train data.

have you tried to tune the parameters in your models?

Accuracy			
Method	**Accuracy1 %**	**Accuracy2 %**	**ROC area**
Simple Connectivity	90	0	0.5
Within K hops	49.6	-	0.61
N Common Neighbors	82.24	49.31	0.66
Jaccard Similarity	76.53	56.8	0.68
Simple Drug Network	66.6	58.4	0.63
Drug Network	62.3	70.9	0.66

5 Discussion

The perfomance of the prediction methods in general shows us that treating the Bitcoin network as a Social network is acceptable, at least for predicting future transactions. We see that roughly, transactions are more likely to occur between two users s and r if there is already some existing relationship between s and r, such as common senders/receivers, common sending/receiving patterns and proximity in the Bitcoin network. To start with, the results from *Simple Connectvity* and the connected component analysis in data exploration tell us that every user knows almost every other user through several immediate acquaintances. We hypothesize that if we observed the graph at an earlier time, say when the number of transactions was some small constant times the number of users(e.g 800,000 transactions instead of the current 3.3 million), its links will be such that the largest connected component of the network will have over 75% of all users. We say this because of the existence of a few distinct type of users in the network with similar transaction volumes and send to receive ratios; we tried to model this aspect of the network in the *Drug Network* technique. Similarly, we hypothesize that with the passage of time, certain nodes will still remain isolated from the largest connected component. These are a few user/address pairs that transacted among each other a few times and stopped. The relatively high prediction success of *Common Neighbors* and *Jaccard Similarity* tells us that other local methods i.e methods that look at only a user and its immediate neighbors may be valuable for prediction; we interpret this as the pairs of users that are likely to interact in the near future are those that have a high number of immediate acquintances in common. The precision-recall curves for both methods show us both methods err more on the side of classifying pairs that transacted as not. We surmise such misclassified pairs are users who have some connection that is not detected by these local methods.

6 Summary and Conclusion

To summarize this work, we modelled the problem of Bitcoin transaction prediction as the link prediction problem on a social network and used different graph based methods to predict whether a transaction occurs between two users in the near future. All our methods achieved a better than chance prediction on a random pair, with a maximum ROC area under curve of 0.68. The success of these methods imply certain facts about the Bitcoin network, discussed above.

There are a number of possible extensions to this work: one is **Visualization** of the Bitcoin network as it would be informative to get a picture/simulation of the Bitcoin network graph. Because of the size of the network ($>$ 1 million edges), visualization packages such as Gelphi would require $>$ 8GB ram to display the graph. Another is **Random walk methods** including prediction via hitting time and Pagerank. These also require immense computing resources for the Bitcoin network.

7 Code description

C++ was used to implement the link prediction methods and **Python** was used for the data analysis; here is some description of the code and the workflow.
- **combine.txt** is a text file with the training data and the test data.
-**linkPredict.cpp** is the c++ code that reads the input, computes the link prediction metrics and outputs predictions/accuracy values. Most of the methods in linkPredict.cpp have been commented out. As it is now, linkPredict can be compiled and run on combine.txt to give the accuracy1 of Jaccard similarity.
Sample execution
- % g++ -o linkPredict linkPredict.cpp **Compiles linkPredict**
- % ./linkPredict < combine.txt **Runs on combine.txt, currently outputs accuracy1 for Jaccard prediction**
- To use another prediction method, comment out Jaccard in the code and uncomment the other method
- To compute ROC and precision recall curves, comment out the code to print accuracy and uncomment the line to print predictions.
- Then run as % linkPredict < combine.txt > roc.txt. roc.txt will now store the predictions which is used by the python code we provide to compute AUC and precision and recall.

8 Acknowledgments

I would like to thank the COS 424 teaching staff, specifically Mehmet Basbug and Dr. Xiaoyan Li for their helpful explanations in precept and on piazza. I would also like to take Abhinav Khanna, another student in the class with whom I had helpful discussions about Bitcoin transaction prediction methods.

9 Bibliography

[1] David Liben-Nowell, Jon Kleinberg, The link prediction problem for social networks in Proceeding CIKM '03 Proceedings of the twelfth international conference on Information and knowledge management
[2] A.Pfitzmann, M.Hansen, A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity
[3] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. A fistful of bitcoins: Characterizing payments among men with no names. pages 127140, 2013.
[4] Androulaki, E.; Karame, G.; Roeschlin, M.; Scherer, T.; Capkun, S. Evaluating User Privacy in Bitcoin; IACR Cryptology Print Archive, vol. 2012:596
[5] E. Candes, B. Recht, Exact Matrix Completion via Convex Optimization; 2008