Structural Analysis of Criminal Network and Predicting Hidden Links using Machine Learning

Emrah Budur
Stanford University
Stanford, California 94305
Email: emrah@stanford.edu

Seungmin Lee
School of Electrical Engineering
Stanford University
Stanford, California 94305
Email: smlee729@stanford.edu

Vein S Kong
Stanford University
Stanford, California 94305
Email: vskong@stanford.edu

***Abstract*—Analysis of criminal networks are inherently difficult because of the nature of the topic. Criminal networks are covert and most information is not publicly available. This leads to small datasets available analysis. The available criminal network datasets consists of entities, i.e. individual or organizations, which are linked to each other. The links between entities indicates that there is a connection between these entities such as involving in the same criminal event, having commercial ties, and memberships in the same criminal organization etc. Because of incognito criminal activities, there could be many hidden links from entities to entities, which makes the publicly available criminal networks incomplete. Revealing hidden links introduces new information, e.g. affliation of a suspected individual with a criminal organization, which may not be known with public information. *What will we be able to find if we can run analysis on a large dataset and use link prediction to reveal the implicit connections?* We plan to answer this question by using a dataset that is an order of magnitude more than most criminal networks analysis. And by using machine learning techniques, we will convert a link prediction problem to a binary classification problem. We plan to reveal hidden links and potentially hidden key attributes of the criminal network. With a more complete picture of the network, we can potentially use this data to thwart criminal organizations and/or take a pareto approach in targeting key nodes. We conclude our analysis with an effective destruction strategy to weaken criminal networks and prove the effectiveness of revealing hidden links when attacking to criminal networks.**

I. Introduction

This paper will be focusing on predicting hidden links in the context of criminal networks. Because of the nature of the subject: data and links for each node might be hard to get, or might in fact be hidden. For our project we surveyed a few papers: three of them deal with link prediction in the context of social networks, and two of them focusing on how social network analysis can be used to help understand crime and terror organizations.

A. Overview

The six papers we surveyed will give us a background on how link prediction is done for social networks and will give us a better idea on how to model and use network analysis to potentially find hidden links in crime networks.

In the Clauset et al. paper, it is focused on hierarchical structure and link prediction based on the hierarchical structure [3]. The paper selected three networks for analysis and using hierarchical random graph they were able to have similar average degree, clustering coefficient, and vertex-vertex as the selected networks. Most criminal networks will likely fit into the hierarchical category. It generates many hierarchical networks and use those randomly generated models to find high probable edges. It also has a comparison of area under curve (AUC) of the different link prediction methods which is really valuable.

The Hasan and Zaki paper focuses on surveying different link prediction models used in social networks [8]. They categorized them in four different categories: 1) Featured-based 2) Bayesian 3) Probabilistic 4) Linear Algebraic. They talk about the approaches in high level and does a brief review of all of them. They have many good feature suggestions, such as common neighbors, Jaccard coefficient, Katz, and shortest path distance.

Liben-Nowell and Kleinberg, takes Hasan and Zaki to the next level, by actually using different link prediction algorithms on actual data [9][8]. They selected research papers as the data and divide each into subject categories and ran a litany of link prediction algorithms on the dataset. They noted that some algorithms jump out as performing well, but that there were a lot of room for improvement. It also introduces us to a *score(x,y)* feature which helps us determine if there's a link by *embeddedness* or number of common neighbors. We more than likely will take this as one of the approaches to finding hidden links.

Medinas paper deals with using social network analysis (SNA) on terrorist networks, specifically al-Qaeda [10]. It uses basic SNA concepts. It uses small-world and scale-free network to model al-Qaeda. It tries to figure out whether al-Qaeda is hierarchical or decentralized. It also removes two key nodes in the network to see how robust the network is. Similarly in the Sparrow paper, it lays out some motivational points for using SNA in criminal networks [12].

In Backstrom and Leskovec, it talks about predicting links in a social network context, in this case, Facebook. The paper uses a *supervised random walk* approach to predict missing links. As mentined in the paper, research in security has recently recognized the role of social network analysis for this domain (e.g.,terrorist networks). In this context link prediction can be used to suggest the most likely links that may form in the future. Similarly, link prediction can also be used for prediction of missing or unobserved links in networks or to suggest which individuals may be working together even

though their interaction has yet been directly observed. [1]

These papers are very pertinent to the course. The topics of modeling using small-world or other models is prevalent in these papers and in our CS224W course. Also with link prediction and survey of link prediction algorithms, this will be covered as a future topic in class.

The common themes throughout these papers are link prediction and trying out many different link prediction algorithms. Also the attempt to model the network and getting attribute data on the network for future analysis is also prevalent throughout the paper.

B. Critique

Though the papers we reviewed gave us a good background on creating hidden links for criminal networks, it is not without fault. There are a few missing gaps in the papers. Most of them deal with the dataset selection and some gaps in the analysis.

Sparrow's work doesn't provide any mathematical approach or analysis [12]. It only suggests some concepts.

Medina's work although pertinent to our research lacks a large enough dataset to provide any substantial findings [10]. The dataset of 381 nodes and 690 undirected links is a small sample size. Since it is a public dataset of covert individuals, the dataset is incomplete. The analysis on density seems to be flawed since the data is sparse. It incorrectly uses the concept of small-world with just random graphs. We think it should be more intentional in creating it with less randomness. The analysis done on the network is pretty basic. We would like to see more advanced analysis done on the dataset for completeness.

Liben-Nowell and Kleinberg's paper uses a diverse set of numerical analysis but doesn't have much explanation of the data or analysis [9]. Also the dataset that they use is quite confined, the research paper's network would be different than a criminal network. The scores that are calculated in the paper are independent, we think we can combine them for better analysis, Which we plan to do in our research. The evaluation is performed over random networks, but we think it should use the test set error rate to evaluate it.

Clauset et al. focuses on hierarchical networks [3]. Though most criminal networks are hierarchical, some may be decentralized, which would make Clauset's paper not as relevant. The methods of common neighbors, shortest path length and product of degrees are correct in assortative networks, but can be misleading if our network is not assortative. The paper talks about p_r,which is probability that node r that a pair of right and left subtree of that node is connected ,but fails to give any real details of what p_r should be for hierarchical vs non-hierarchical.

Hasan and Zaki's survey paper does just an overview of the methods without analyzing any real dataset [8].

II. Project Proposal

A. Problem

The common problem of the criminal network datasets is that they are incomplete and inherently covert by nature since public information doesn't hold complete information about the actors in the criminal events [12]. So, an existing direct relationship in the real world may not have been discovered yet, leading to hidden links in the network. The hidden links may evolve into critical actions of future events or they may play a key role to extending existing networks to operate in a broader range. Moreover, they may result in revealing critical links and help to capture critical people that were wanted for a long time, like Saddam Hussein [4]. Therefore, sophisticated techniques are required to reveal the hidden links in criminal networks with high precision to solve an array of mission critical problems.

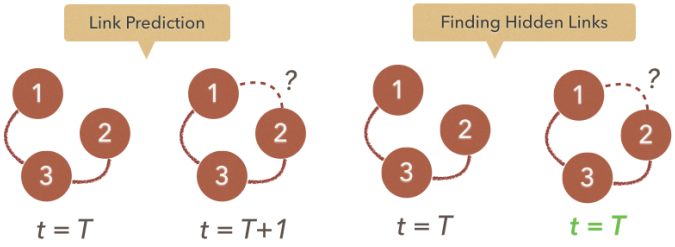

Fig. 1. Comparison between Link Prediction And Finding Hidden Links

B. Dataset

Most of the studies in criminal networks have used small sizes of datasets due to the scarcity of the accessible data in this field [12]. Throughout this study we will use an anonymized large dataset that consists of 1.5 millions nodes worldwide with 4 millions undirected edges. The nodes will include the publicly available sanction lists such as the famous US sanction list, namely OFAC [11]. Compared to datasets in the reviewed articles, our dataset has an order of magnitude more data, which will allow us to extract the real big picture of criminal networks worldwide.

C. Scope Definition

Our project consists of three main subproblems.

1) Link Prediction Model using GBM: We will apply a machine learning technique called Gradient Boosting Machine(GBM) on this problem. Unlike previous work for link prediction, we do not have transitional network data(network images at time t and t'), we will make a supervised link prediction model based on the current network image. Fig 1 depicts the concept.

In order to apply GBM, we regarded this problem as a binary classification problem. We transformed a network image to a feature matrix with various metrics that are famously used in current link prediction studies.

We will analyze our model from various viewpoints and use this model as a base model for the future experiments.

2) Finding Hidden Links in Criminal Network: We will create corrupted networks, which will have 5% to 50% of edges removed from the original network, and then regard removed edges as hidden links. Now, we will test using our model to see how well our model trained in corrupted networks can find hidden links. This experiment is critical to find out

whether we can find hidden links and the result would be important since we can detect hidden criminal connections or activities.

We will also analyze various aspects of this experiment.

3) Destroying Criminal Network: Finally, we will go on our study with advanced analysis of our criminal network using our model. Probability of being an edge can be obtained by prediction using our model. We will use that probability as a weight on a specific edge, and apply pagerank algorithm on weighted network to get scores for each node. Then, we will analyse what method can destroy the criminal network as fast as possible as a result of this advanced analysis. We will finally show that Weighted PageRank scores will help destruct the criminal network much smarter and much faster. We will conclude that Weighted PageRank score can also be used as a *suspiciousness index* to use to target *the vital few* of the network.

D. Background

This study requires a background on machine learning and social network analysis. Below are a brief explanation of the essential subjects that are meant to provide a firm foundation for the method section.

1) Terminology: The word *link* refers to an *edge* between two nodes in the network. The term E refers to the set of all *possible* edges regardless of the existence of the edge in the original network.

If there is an edge between two nodes in the original network, we call this edge a *positive edge*. The term E^+ refers to the set of all positive edges.

On the other hand if there is no edge between a particular pair of nodes in the original network, we use the term *negative edge* to refer to the absence of the edge. The term E^- refers to the set of all negative edges. The term E_d^- denotes the set of randomly chosen negative edges where each edge has shortest path length of d.

We used the term *hidden edge* or *hidden link* for a particular negative edge that we must *predict* it as a positive edge.

2) Link Prediction Metrics: Table I shows a number of well-known metrics that are commonly used in link prediction between two nodes. The formula $\varphi(i)$ represents the set of nodes that are neighbors of the node i and k_i represents the degree of the node i.

3) Gradient Boosting Machine(GBM): Gradient Boosting Machine is one of the most popular tree-based machine learning techniques. It is based on decision trees and it is used for both regression and classification problems. GBM applies boosting methods for decision trees to reduce variance. Hence, the output of GBM has low variance.

Compared to bagging method in other Tree-based methods, which creates many bootstrapped trees all at once and aggregating them, GBM keeps a long learner tree and grow it sequentially to avoid overfitting and high variance. Instead of growing a single large decision tree to the data which results in potentially overfitting, the boosting approach learns slowly by iteratively correcting the error of the previous iteration.

Metric	Definition
Common Neighbors	$S_{xy} = \lvert\varphi(x) \cap \varphi(y)\rvert$
Salton Index	$S_{xy} = \frac{\lvert\varphi(x) \cap \varphi(y)\rvert}{\sqrt{k_x \times k_y}}$
Leicht-Holme-Newman Index	$S_{xy} = \frac{\lvert\varphi(x) \cap \varphi(y)\rvert}{k_x \times k_y}$
Sorensen Index	$S_{xy} = \frac{\lvert\varphi(x) \cap \varphi(y)\rvert}{k_x + k_y}$
Jaccard Index	$S_{xy} = \frac{\lvert\varphi(x) \cap \varphi(y)\rvert}{\lvert\varphi(x) \cup \varphi(y)\rvert}$
Hub Index	$S_{xy} = \frac{\lvert\varphi(x) \cap \varphi(y)\rvert}{\min(k_x, k_y)}$
Preferential Attachment Index	$S_{xy} = k_x \times k_y$
Adamic-Adar Index	$S_{xy} = \sum_{z \in \varphi(x) \cap \varphi(y)} \frac{1}{\log k_z}$

TABLE I. LINK PREDICTION METRICS

Algorithm I Boosting for Regression Trees

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
   (a) Fit a tree $\hat{f}^b$ with d splits (d+1 terminal nodes) to the training dataset D_{train}.
   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$
   (c) Update the residuals,
   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$
3. Output the boosted model,
$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

Fig. 2. Algorithm : Boosting for Regression Trees [5].

As it is seen in algorithm in Fig 2, GBM function has three tuning parameters:

- B: Is the number of trees. If we choose B too large, boosting can overfit the data. So, a proper value of B is chosen by validating the output of the function against the training dataset, which is known as *cross validation*.
- λ: Is the learning rate of the function. It is a small positive number which controls the rate at which boosting learns. There is a correlation between λ and B. Small λ requires large B in order to achieve good performance.
- d: Is the number of splits in each tree, which controls the complexity of the tree structure. It is also known as *the interaction depth*. The bigger d results in more interaction in each decision tree.

We used a set of parameters for GBM in Table II :

Parameters	Value
Maximum Tree Depth	5
Number of Trees	50
Fewest allowed observations in a leaf	10
Number of Bins in a histogram	20
Learning Rate	0.1

TABLE II. PARAMETERS USED IN GBM MODEL

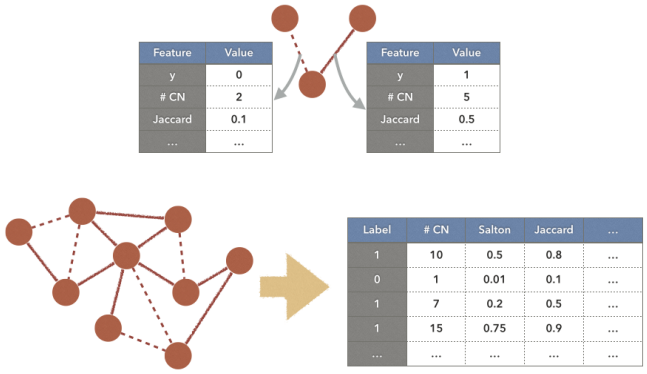

Fig. 3. How to convert Network into Feature Matrix

4) Base Data for Link Prediction Model: In order to build a link prediction model, we designed a training and test data set.

First of all, we converted the criminal network into a feature matrix to apply machine learning. The concept is described in Fig 3.

We denoted the set of positive edges as E^+, and the set of nodes as N. We denote the size of nodes and positive edges as $|N|$ and $|E^+|$ respectively. The number of all possible edges is $\binom{|N|}{2}$. It can be noted that $\binom{|N|}{2} \gg |E^+|$. Consequently, the number of negative edges are much greater than the number of positive edges. Hence we chose a total of $|E^+|$ negative edges by randomly undersampling from E^-. In this way, we avoided bias in our machine learning model. Let's denote this dataset as D(D has total $2|E^+|$ edges; $|E^+|$ positive edges and $|E^+|$ negative edges).

For the training phase, we prepared 75% of D to be used for training our model and we will refer it as D_{train}. We prepared the remaining 25% of D to be used for testing purpose and we will refer to it as D_{test} for convenience. Of course, each of D_{train} and D_{test} has equal number of positive and negative edges.

After training GBM model using D_{train}, we will evaluate our model using the test set D_{test} and calculate the resulting AUC.

5) Undersampling: Undersampling is a technique used in data analysis to adjust the class distribution of a specific data set. Undersampling randomly chooses some fraction of data from dominant classes to make equal numbers of class samples. For example, in our project, negative edges are overwhelmingly dominant, leading *class imbalance*, so we undersampled those negative edges to make their numbers the same as the number of the positive edges. If we did not undersample, the model would be biased seriously with a tendency of predicting towards negative.

We did an experiment to test whether our sampling method is accurate. First, since our dataset has so many nodes, considering all pairs is intractable, so we created a random graph of $10,000$ nodes following the power law with the same exponent that our criminal network has, $\alpha = 3.6346$. Now we compared the two model: one with undersampling negative edges to make

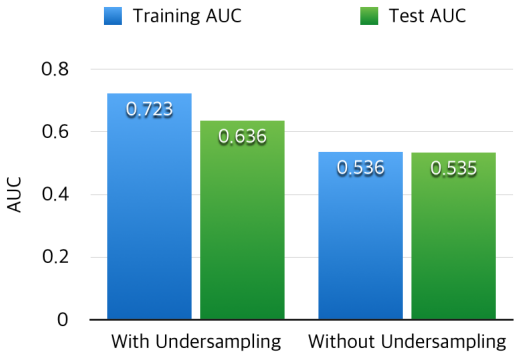

Fig. 4. Comparing Model's Training and Test AUC between With Undersampling and Without Undersampling

Fig. 5. Prediction Imbalance between With Undersampling and Without Undersampling

50:50=pos:neg, and the other without undersampling where the number of negative edges extremely dominates the number of positive edges.

The results are as we expected, undersampling made both training and test AUC higher than not using undersampling, shown in Fig 4. The strong need for using undersampling is shown in Fig 5. We defined

$$\text{Prediction Imbalance} = |err_+ - err_-|$$

to represent the magnitude of prediction imbalance where err_+ means error rate on predicting positive edges and err_- on negative edges. If we let negative examples dominates positive examples, then the model will be overfitted to predict negative examples which results in serious bias of the model.

So we decided to use undersampling technique throughout our project.

6) Area Under Curve (AUC): In order to evaluate the performance of our link prediction algorithm we will use one of the frequently used accuracy metrics which is known as Area Under Curve(AUC) statistics. AUC is calculated as the area under the Receiver Operational Curve (ROC).

In this study, the y axis of ROC curve will show the ratio of correctly identified true-positive types. On the other hand, the x axis of ROC will show us the ratio of correctly identified true-negative types. The range of y axis will be [0-1] since it is a ratio. Similarly, the range of x axis is also [0-1]. So, the

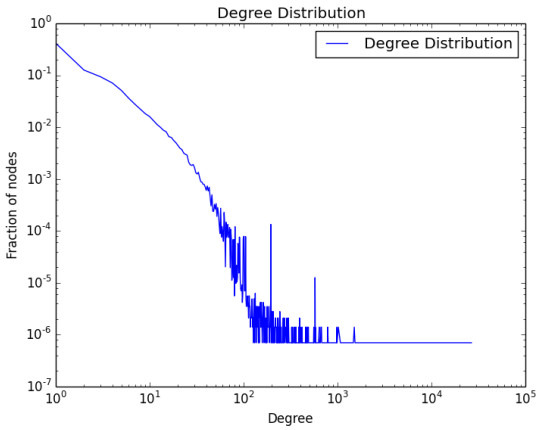

Fig. 6. Degree Distribution

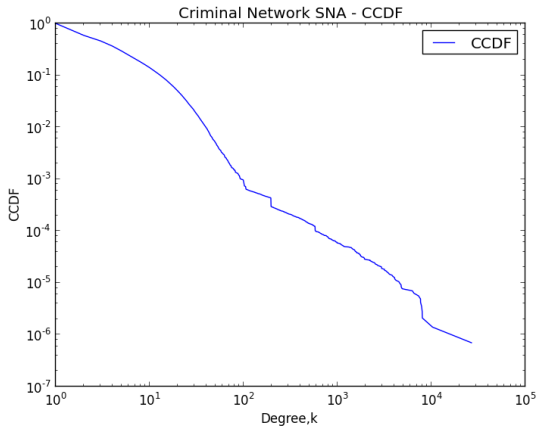

Fig. 7. Complementary Cumulative Distribution Function(CCDF)

area of a curve in this plot will also be in the range of [0-1]. Therefore, AUC will have a range of [0-1].

The values of AUC which are above 0.5 will indicate how the algorithm performs better than chance. In the ideal case, the algorithm is expected to produce output where AUC = 1.

III. Experiments And Analysis

A. Basic Analysis of Network

We employed basically a number of SNA and machine learning tools, i.e. *Snap.py, NetworkX, Gephi, IBM i2, H_2O etc*.

As an initial work, we extracted degree distribution and complementary cumulative distribution function(CCDF) plot as shown in Figure 6 and Figure 7 respectively. It is clearly seen on the plot that degree distribution follows power law with heavy tail structure. We figured out the parameters of the power law fitting in our dataset as $\alpha = 3.6345887074$ and $x_{min} = 29.0$ which tell us that the criminal network follows Preferential Attachment Model.

In addition, we extracted a number of fundamental figures about the network properties as shown in the following table.

Characteristics	Value
Nodes	1428002
Edges	3811872
Bidirectional Edges	7623744
Closed triangles	18235211
Open triangles	897960146
Fraction of closed triads	0.019903
Maximum WCC Size	0.538763
Maximum SCC Size	0.538763
Approximate full diameter	29
90% effective diameter	11.736947
Number of WCCs in the network	160840

TABLE III. Characteristics of a Criminal Network

B. Analysis on Link Prediction Model

1) How to choose negative examples for undersampling: The set of negative examples in the criminal network is too large that we may not guarantee that choosing $|E^+|$ number of negative edges from the set E^- which can be representative for the whole E^-.

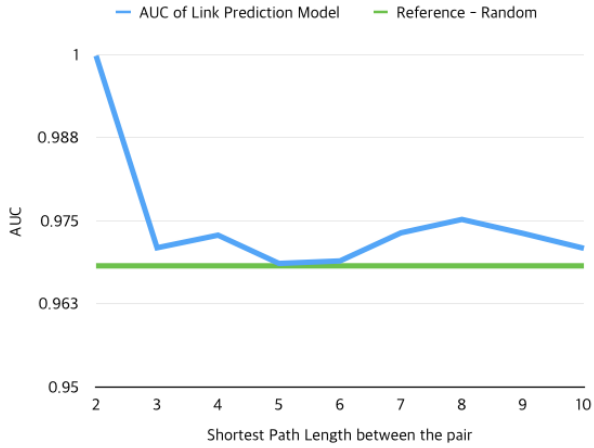

Fig. 8. Performance of Link Prediction Model vs. Sampling of Negative Examples according to Pair's Shortest Path Length

So, we designed an experiment such that comparing randomly downsampled negative edges from E^- with random sampled negatives from E_d^- where $2 \leq d \leq 10$. E_d^- denotes the set of negative edges where the pair's shortest path length is d.

The result is shown in Fig 8. Interestingly, we can make a really accurate classifier if we use negative examples from E_2^-. Since the green reference line is the lower bound of the blue line, we could conclude that randomly downsampling of E^- guarantees the minimum performance of the link prediction model and not overestimate the performance. So, we decided to downsample randomly from E^- for our standard data sampling method for future experiments.

2) Building Prediction Model and Feature Selection: For the training phase of the classification task, we extracted some relevant features of our entities. While reviewing some relevant features that are commonly used in the literature, we realized that Hasan et. al. proposed a variety of good features such as *Common Neighbors, Jaccard Coefficient, Adamic/Adar, Katz* and so on [8].

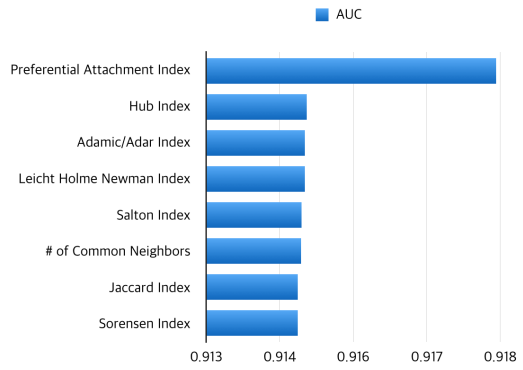

Fig. 9. AUC when used one feature

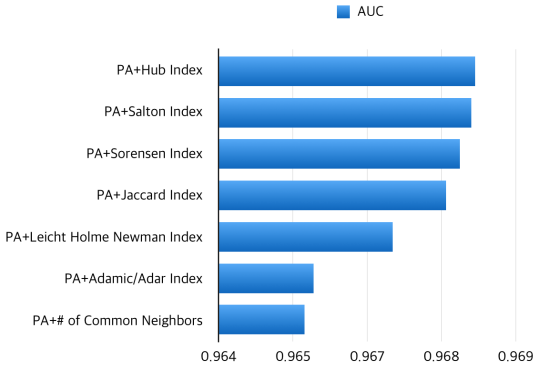

Fig. 10. AUC when used two feature

We trained our model by computing various features on the dataset D_{train}. We tested our model on D_{test} and obtained an AUC value. We obtained an AUC value for each feature we applied. These AUC values can be seen in in Figure 9. The results show that discriminating power of preferential attachment index outweighed others. So we decided to choose *preferential attachment index* as first feature for possible combination of features since it has the highest AUC value.

Furthermore, we tested all features again as the second feature with preferential attachment index evaluated by AUC. As a result, we obtained the bar chart shown in Figure 10. It is clearly seen that the inclusion of hub index as a second feature improves AUC. We tried the inclusion of each feature as the third feature as well, but from adding the third feature, we observed little improvement by inclusion of additional features on AUC result. So, preferential attachment index and hub index were the most important features and we trained and finalized our prediction model with AUC of 0.96822.

3) Probability of Hidden Link vs. Shortest Path Lengths: In order to find out the relationship between the probability of possible hidden link and shortest path length between two nodes, we had to randomly sample 9 sets of E_d^- where $2 \leq d \leq 10$. In other words, we made 9 different datasets of negative edges clustered by shortest path length of range from 2 to 10. These data sets were independently drawn from E^- to the negative edges used in D.

We expected a large probability of a hidden link when the pair has small d, due to triadic properties. Naturally, we

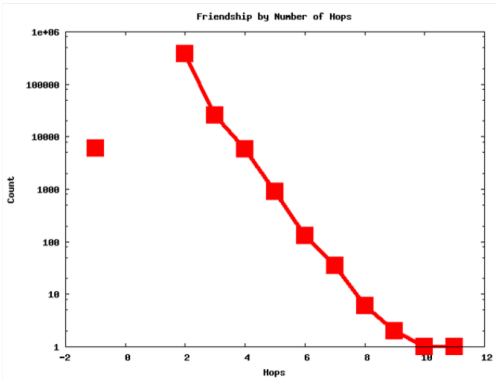

Fig. 11. Triad Closure Behaviour of Facebook Network: Distance x=2 denotes the friends of friends while x=-1 denotes isolated nodes[1]

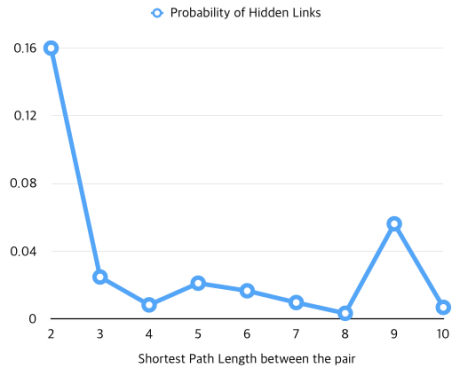

Fig. 12. Probability of Hidden Links vs Shortest Path Length

expected low probability of a hidden link when the pair has large d. The probabilities can be calculated by applying our prediction model on all datasets denoted previously as E_d^-.

As we expected, Figure 12 shows that the negative edges with the shortest path length of 2 have the highest probability of being a hidden link. The result is very well aligned with the triadic closure property of Facebook network revealed by Leskovec et.al. in Fig 11 [1]. Generally speaking, the tendency is the shorter the shortest path length, the higher the probability of hidden links as shown in Fig 13.

C. Finding Hidden Links on Corrupted Networks

For preparing the corrupted networks, we randomly chose edges of portions $5\%, 10\%, ..., 50\%$ and remove them from the original network. Let's denote a particular corrupted

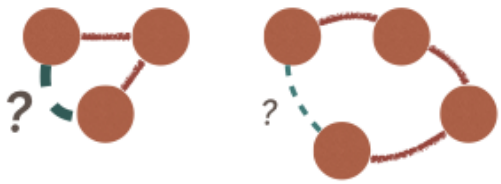

Fig. 13. Concept diagram for Probability of Hidden Links vs. Shortest Path Length

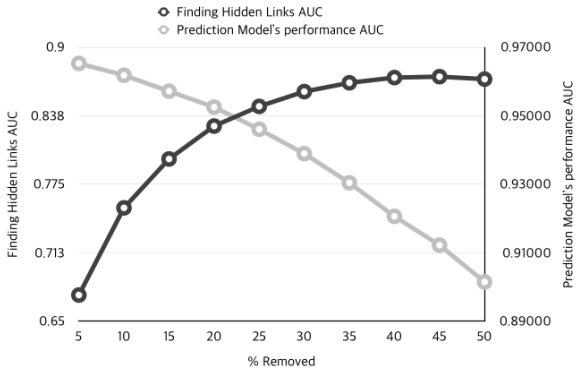

Fig. 14. AUC vs % Edges Removed

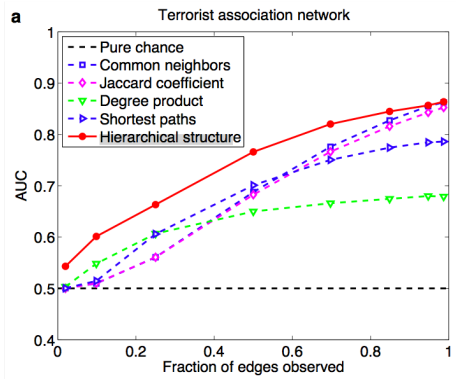

Fig. 15. AUC vs Fraction of Edge Observed [3]

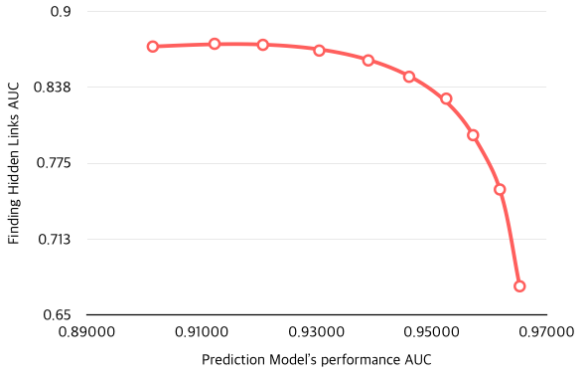

Fig. 16. AUC on Finding Hidden Links vs. Prediction Model's AUC

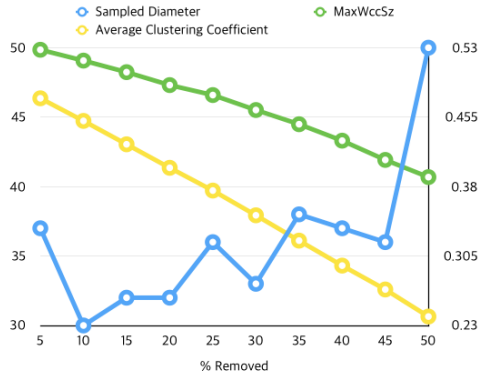

Fig. 17. Various Network Measurements vs. % Edges Removed

network as C_p and removed positive edges as R_p, where $p \in \{5\%, 10\%, 15\%..., 50\%\}$ denotes portion of removal.

We designed two experiments. One is to find out how well machine learning models can be built in corrupted networks with respect to the portion of removal while the latter one is to test how well the built model can reconstruct the original network. We thought the latter is important since we have an incomplete criminal network and there might be lots of hidden links and our goal is to find those hidden links. So the performance of reconstruction may be the measure of how well we can reveal hidden links from incomplete information (network).

1) Performance of Link Prediction Model from Corrupted Network: From each corrupted network, C_p, we made a training and test data sets, namely $D_{\text{train},p}, D_{\text{test},p}$ where $p \in \{5\%, 10\%..., 50\%\}$ using similar ways to train the original network. So, we have distinct link prediction model for each corrupted network.

For each corrupted network C_p, we obtained an AUC value on $D_{\text{test},p}$ using a model trained on $D_{\text{train},p}$. Gray line in Fig 14 shows that AUC values of models in corrupted networks have decreasing tendency when p gets larger. This result shows similarity to the result of past research which can be seen in Fig 15. Clauset et. al. used just one link prediction metric to obtain the best AUC of their study[3]. Since we used multiple prediction metrics in our model, we achieved higher AUC than their results.

2) Performance on Finding Hidden Links: In this part of the experiment, we tested the performance of our model with a series of analysis as follows. The result is quite interesting.

As we mentioned in the previous section, we tested how well each model from a corrupted network can be reconstructed the corresponding removed edges. The result was shown in the black line in Fig 14. The resulting plot shows increasing accuracy in finding hidden links when more edges were removed. This result is the exact opposite to one of model's AUC in gray line in the same figure. The opposite trends can be clearly shown when we draw additional graph in Fig 16.

Since removal of edges make the graph more incomplete, we expected decrease in AUC on finding hidden links when the ratio of removed edge increase. However, the observed behavior is surprisingly different from our expectation.

We guessed there must be more clues for this result such as the network structures must be related to the performance of finding hidden links. So, we investigated three representative measurements of the network: Sampled Diameter, Maximum WCC Size, and Average Clustering Coefficient. Fig 17 shows measurements of the network with respect to the portion of edges removed in percentage. Sampled diameter fluctuates, but its moving average increases as more edges are removed. On the other hand, Maximum WCC Size and Average Clustering Coefficient decreased steadily, as more edges are removed. It can be intuitively validated that the more edges are removed, the more sparse the network becomes, hence, leading to a

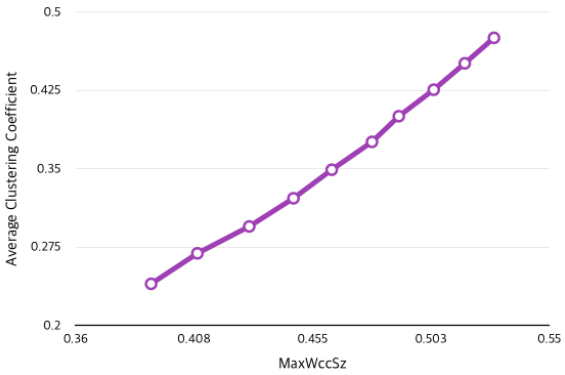

Fig. 18. Average Clustering Coefficient vs. Maximum WCC Size

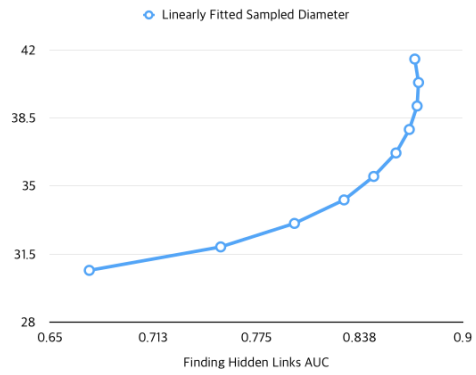

Fig. 19. Linearly Fitted Sampled Diameter vs. Finding Hidden Links AUC

decrease both in clustering coefficient and the size of the largest WCC. We can further explain this tendency by Fig 18, which shows that there is a linear dependency between average clustering coefficient and maximum WCC size and removal of edges reduces both of them.

And finally, the interesting result in Fig 16 can now be further explained by characteristics of the network as shown in Fig 19 and 20. Those three plots of performance on finding hidden links versus linearly fitted sampled diameter, average clustering coefficient, and maximum WCC size gives us consistent result and let us analyze with the characteristics of the network. Our criminal network has property that if maximum WCC size decreases, then the average clustering coefficient will also decrease. And since decreased maximum WCC size means more segmentation of WCC's throughout the network, diameter would increase as maximum WCC size decreases. Fig 19 and 20 shows that in this case, the AUC of finding hidden links increases while the AUC of prediction model decreases. We can explain this phenomenon using this with Fig 21:

- If the network has a larger core and denser structure (large maximum WCC size, large average clustering coefficient, and small diameter), this will make the prediction model learn more about the current network(higher performance of prediction model).
- However, in this case, since we can say the prediction model has been overfitted to the current image of network, the performance of finding hidden links(which are not real edges on corrupted networks) would decrease.

As a result of this series of analysis, it is observed that pruning out some fraction of edges from the original network would help achieve better performance in finding hidden links.

D. Destroying Criminal Networks

One of our final goals is to find out a smart way of destroying criminal networks. So, we focused on what the most effective way is for arresting criminals to destroy the network in shorter time. The intuitive way to do that is to choose random node and remove it. The better way is to choose node with highest degree first. More sophisticated way is to choose node with highest pagerank score first.

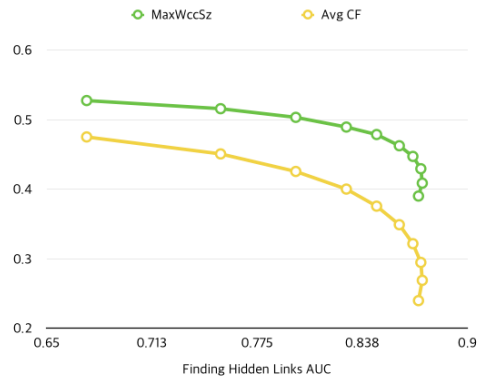

Fig. 20. Maximum WCC Size and Average Clustering Coefficient vs. Finding Hidden Links AUC

At this point, we will use our prediction model to give weights on existing edges by probabilities of being a link between two nodes. More suspicious connections will have higher weights(close to 1) and less suspicious ones will have lower weights(close to 0). Using those weights we could create weighted network and get the weighted pagerank scores.

We will do the robustness test using those weighted pagerank scores. We expect a faster collapse of the criminal network using our method.

1) Performance of Weighted PageRank method in Destroying the Network: We calculated the classical PageRank scores of each node based on unweighted edges on the original network. In addition, we calculated *Weighted PageRank* scores based on the weights we obtained. We analysed the change of diameter and the ratio of largest connected component under a random failure scenario and 3 targeted attack scenarios based

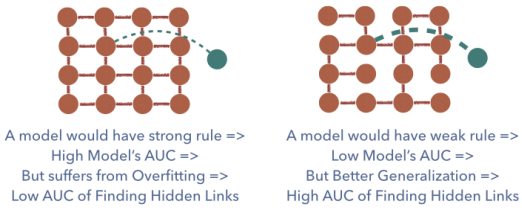

Fig. 21. Concept diagram for Network Characteristics and Performance on Finding Hidden Links

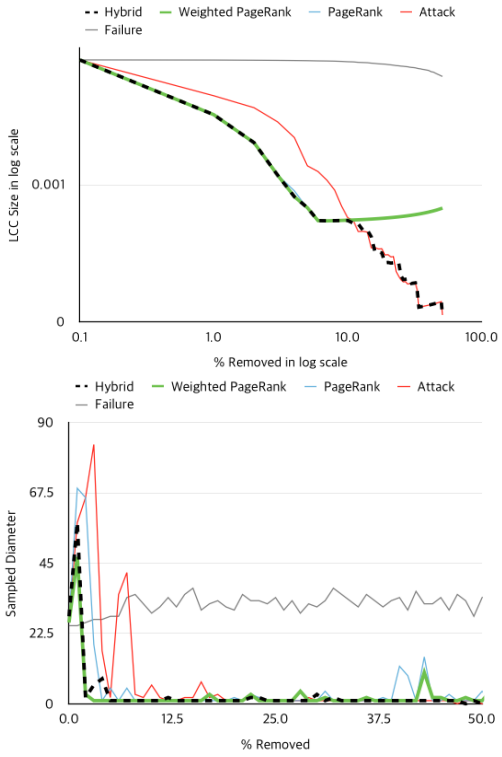

Fig. 22. Destruction Characteristics of Criminal Network in Different Attack Scenarios

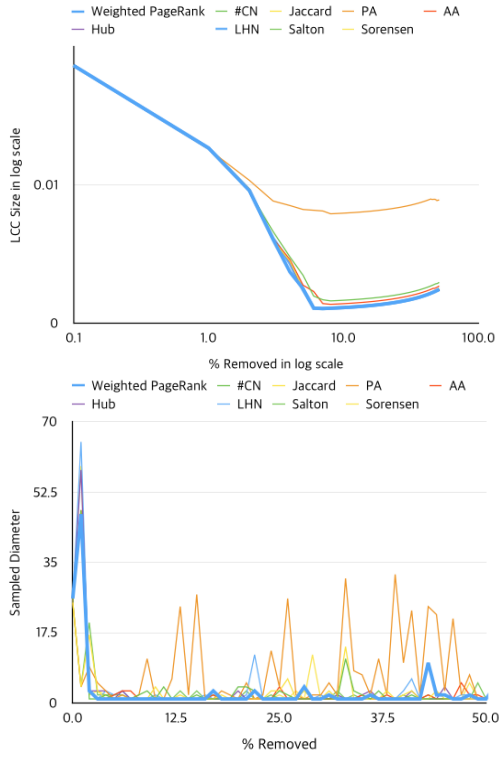

Fig. 23. Comparing Weighed PageRank Method to Single Index PageRank Methods

on degree of nodes, PageRank, and Weighted PageRank scores. The results are shown in Figure 22.

We observed in the first plot of Figure 22 that removal of nodes having maximum Weighted PageRank score reduces the largest component of the network much faster than removal of nodes having maximum degree. It means the Weighted PageRank scores we obtained identifies the most influential nodes of a network much precisely. So, in the criminal network context, we called the Weighted PageRank score as *suspiciousness index*. Consequently, suspiciousness index can be used to take a pareto approach to maximize preventive risk reduction of possible criminal activities by watching the *vital few* of the network much more closely.

On the other hand we observed no clear difference between effects of PageRank and Weighted PageRank scores on the destruction characteristics of criminal network in the LCC size plot of Fig. 22. However, we observed clear difference on the influence of PageRank and Weighted PageRank in the diameter plot of Fig. 22. We also validated our expectation by observing attack scenarios outweighed the random failure scenario.

2) Comparing Weighted PageRank Method to Single Index PageRank Methods: Using our prediction model, we can obtain probabilities of each edge being a link. The probabilities are the output of our machine learning algorithm which uses an ensemble of indexes. We use these probabilities to turn our original network into weighted network. Then, we used the weights as an input to the pagerank algorithm, which we called *weighted pagerank method*, to obtain *weighted pagerank scores* of the nodes.

We came up with the idea that since our prediction model uses ensemble of multiple indices, it must outperform any single index pagerank methods which uses single index, i.e. Jaccard index. As it can be seen in Fig 23 with a blue bold line, the weighted pagerank method outweighs other pagerank methods which uses single index. So, we achieved the maximum performance in destroying criminal network by using the weighted pagerank method.

3) Hybrid Method for the Best Performance in Destroying the Network: As it can be seen in Fig 22, all pagerank methods have a performance bottleneck after some period of attacks. This may be because we calculate the pagerank score only at the very first time and the subsequent snapshots of attacked network were not reflected in the pagerank scores. If we can calculate pagerank score at every iteration step, it would yield different results and may even be the maximum performance. However, since calculating pagerank scores for each node in every snapshot is very expensive, we were not able to observe that result. As another practial solution, we designed a new method which can combine the advantages of both weighted pagerank method and one that chooses a node with highest degree first at each round. The result of this hybrid method is shown in Fig. 22 with black dashed line. Its performance in terms of maximum WCC size exactly meets our expectation.

$$W_{\text{hybrid}} \approx \min\left(W_{\text{Weighted Pagerank}}, W_{\text{Attack}}\right)$$

Also, in terms of sampled diameter, hybrid method has heavy tail and fast decreasing tendency at the first attack phase which means it has the advantages of both pagerank and highest degree first method, hence, achieves the maximum performance.

As a result, we achieved the maximum performance on destroying the criminal network, by using the hybrid method.

IV. Conclusion

A. Link Prediction Model

We applied GBM to make a machine learning model which can describe the current network. We used indices which are famous in link prediction algorithms as features of the model. Instead of employing single index, we ensembled multiple indices to improve prediction performance.

B. Finding Hidden Links

We tested the performance of our prediction model to find hidden links in the corrupted networks. From our experiments, we concluded that the more decentralized and less clustered network has better performance on finding hidden links. This is because our machine learning model avoids overfitting to our dataset and has low variance as the dataset becomes less clustered and more decentralized by removing more edges.

C. Destroying the Network

Finally, we used the probabilities predicted by the model on current edges as weights of the network, and obtained weighted pagerank scores from the weighted network. We tested various attacking methods and the two methods, weighted pagerank method and removing the one with highest degree method, had both advantages and disadvantages. At the end we combined advantages of both methods and achieved the best performance on destroying network.

V. Future Research

A. Applying various Machine Learning Techniques

We only used GBM as our machine learning technique. However, there are many other potential techniques we could apply. We can apply logistic regression, random forest, SVM, and neural network to get the various results. Also, GBM has parameters to be tuned. So we could do a grid search to find the best parameters for link prediction model.

B. Finding additional indices to be used for features

One of the most important issues in machine learning problem is finding useful features. We used various indices to our model, but there might be useful additional features such as clustering coefficients, pagerank scores, and triadic ratio. We may apply those indices to improve link prediction model.

C. Comparison of our Link Prediction Model to Existing Methods

We could not do an experiment on temporal change of our data set since there was only a snapshot of the criminal network data set at the time of the study. If we could obtain the temporal changes of our data set, we would be able to compare the performance of our predictions to real data. In addition, we would be able to compare our prediction performance with the prediction performance of the study carried by Leskovec et.al.[1].

D. Using Actual Weights of the Network

The network only has information whether there is an edge between the pair of nodes or not. If we have additional information such as the amount of phone calls, transactions, and number of suspicious meetings, we can use them as additional weights for the edges in the network. Using this weighted network, we will be able to use also regression models instead of classification model. Also, we can compare the network destruction performance of our weighted pagerank method by using the new weights and existing weights, respectively.

References

[1] Lars Backstrom and Jure Leskovec. Supervised random walks: Predicting and recommending links in social networks. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 635–644, New York, NY, USA, 2011. ACM.

[2] Lars Backstrom and Jure Leskovec. Supervised random walks: Predicting and recommending links in social networks. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 635–644, New York, NY, USA, 2011. ACM.

[3] Aaron Clauset, Cristopher Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98–101, 2008.

[4] Farnaz Fassihi. Two novice gumshoes charted the capture of Saddam Hussein, December 2003.

[5] Trevor Hastie Robert Tibshirani Gareth James, Daniela Witten. *An Introduction to Statistical Learning with Applications in R*. Springer.

[6] Rohan Gunaratna. Icpvtr - international centre for political violence and terrorism research, 2014.

[7] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[8] Mohammad Al Hasan and MohammedJ. Zaki. A survey of link prediction in social networks. In Charu C. Aggarwal, editor, *Social Network Data Analytics*, pages 243–275. Springer US, 2011.

[9] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, pages 556–559, New York, NY, USA, 2003. ACM.

[10] R. M. Medina. Social network analysis: A case study of the islamist terrorist network. *Secur J*, 27(1):97–121, Feb 2014.

[11] Office of Foreign Assets Control. Specially designated nationals list (sdn), October 2014.

[12] Malcolm K. Sparrow. The application of network analysis to criminal intelligence: An assessment of the prospects. *Social Networks*, 13(3):251 – 274, 1991.

[13] Wikipedia. Decision tree learning, 2014.

[14] Wikipedia. Oversampling and undersampling in data analysis — wikipedia, the free encyclopedia, 2014. [Online; accessed 5-December-2014].

Emrah: Data acquisition from providers, teaming up, literature survey, coding for structural and supplementary analysis, reviewing and writing the report, visualization of data.

Seungmin: Problem formulation, parsing data into edgelist, writing up the report, coding up the algorithm, running tests, tabulating final results, plotting graphs and analyzing results

Vein: Reviewed paper grammer and organization, wrote part of the report, surveyed and critique pertinent papers, supplemental analysis of results

---