Classification of Conspirators in Social Networks based on Iterative Model using Support Vector Machine

Ruofei Du, Ge Hao, Chen Weijiong
Shanghai Jiao Tong University

Summary

This paper discusses how to identify, classify and prioritize conspirators and innocence in a social network. Firstly, we use the conventional social network model and filter the given information with parameters such as closeness. In addition, we establish a novel iterative model which synthesizes the information of both messages and labeled people. Afterwards, we apply supervised learning algorithms to our social network in order to get a synthesis results from features like distances, closeness and degree. Specifically, we use Support Vector Machine (SVM) algorithm with radial basis function as the kernel function to calculate each individual's probability of guilty. Eventually, we improve our model by semantic network analysis and provide several significant ideas.

Before the establishment of our own theory, we review and deal with the given information of messages and names. On one hand, we ignore some impossible information like self-talking. On the other hand, with the analysis of the original social network, we identify the known conspirator who shares the same name with another.

Considering the fact that both messages on suspicious topic and known conspirators play a vital role in the social network, we build an iterative model to judge how much an individual is suspicious. The highlights of this model are our special measures to establish a mathematics link between every two nodes and use iteration method to simulate the sustained interrelationship among people. Nevertheless, to avoid acting rashly and alerting the criminals, we should take the significance of leadership among the conspirators into consideration. Therefore, we use SVM algorithm to estimate an appropriate priority of the arresting order.

Moreover, we further improve our model by semantic network analysis. We discuss some possible methods of semantic network analysis such as counting keywords' frequencies, parsing and closeness analysis. To verify our analysis, we take the messages in the EZ case as an illustration. Last but not the least, we try to assign a more reasonable weight to each topic and make our prediction even more accurate. Also, our model is applied on the different conditions and the results seem stable.

Eventually, our iterative model and SVM model not only find out the most likely candidates for the unidentified co-conspirators and unknown leaders, but show high precision and efficiency as well.

Keywords

Social Network Analysis, Supervised Learning Algorithm, Iterative Model, Supported Vector Machine, Semantic Network Analysis

Contents

I. Introduction

As a problem of Criminology, it's a hot topic nowadays that how we could know about the structure and organization of criminal networks with the limited information we have [1].

However, conventional approaches remain primarily a manual process and cannot meet our request. The conventional approaches have three steps: first, make an association matrix; second, draw a link chart for visualization purposes; third, make judgment and find the center of the link chart. Though such a manual approach is helpful for crime investigation, it can hardly show us who is the conspirator and it depends on the investigator's judgment to a great extent. What is worse, for very large data, the conventional approaches become extremely ineffective and inefficient [2].

In this paper, we will first use iteration method to establish a model to prioritize the 83 nodes by likelihood of being part of the conspirator. Then we will use Support Vector Machine (SVM) algorithm to give another list and the nodes is prioritized by the order of being interrogated and the differences between the use of these two models will be discussed.

With the help of semantic network analysis, we can develop our model by giving messages more reasonable weight. Then a comparison will be made to see the effect of semantic network analysis.

II. Problem Review

2.1 Background

A group of conspirators are committing a criminal act in a company. Several members of the conspiracy as well as several innocent people have been identified. Besides, a small set of messages from a group of 83 workers in the company has been found.

2.2 Task

Two problems are provided for us to solve. One is to establish a model and offer a list of suspicious conspirators for law enforcement and intelligence agencies. It would be better if the model nominated the conspiracy leaders. The other one is to improve our model with the help of semantic network analysis and natural language processing.

Apparently, before we establish our models, we should deal with the raw data. Firstly, some useless information can be ignored. In addition, we should come over

the difficulty caused by the duplication of names with the help of preliminary analysis in the social network.

Thus our target in this paper is to establish a novel model to offer a list in some order for law enforcement, and then try to answer the following questions: How can we take the advantage of semantic network analysis? What effect does semantic analysis have on our models? Does semantic network analysis matters?

In order to show the use of semantic network analysis, we will make a comparison on EZ case. Can we find the right criminals after using network analysis? Let us go further in detail.

III. Assumptions

1. **The information we have is all-round and the 83 workers in the company didn't communicate in other ways.**

   It means that the raw messages can stand for all messages among the 83 workers. The messages we have can reflect the structure and organization of the network to a crumb.

2. **No argot is used in the messages we has recently found.**

   That means every message only has its literal meaning. We have no need to think about that criminals use some specific words to communicate with each other.

3. **The topics of the messages are classified accurately.**

4. **More assumptions will be described respectively in the models.**

IV. Definition and Terms

4.1 Definition

- SDM Suspicious degree of a member
- SDT Suspicious degree of a topic
- SVM Support Vector Machine

4.2 Terms

- G The original social network

- V The set of vertex, representing all individuals
- E The set of edges, representing all messages
- E' The subset of E, representing suspicious messages
- W The weight of edges
- $deg_{in}(k)$ The in-degree of vertex k
- $deg_{out}(k)$ The out-degree of vertex k
- $deg(k)$ The total degree of vertex k
- $\alpha_i(k)$ The number of edges between vertex i and k.
- $P_i(k)$ The largest SDT between topics talked by member i and k.
- $dist(i,j)$ The shortest distance between the vertex i and j
- $D_{min}(k)$ The minimum distance from vertex k to conspirators
- $C_{closeness}(k)$ The closeness of vertex k: the sum of shortest distances between k and other nodes. [1]
- $C_{betweenness}(k)$ The betweenness of vertex k: the total number of shortest path passing through k. [1]
- $\phi(r)$ The kernel function in the SVM model
- $P(k|\text{features})$ The posterior probability given by SVM model

V. The Filter of Raw Data

Before establishing mathematical models on the given information, we need to filter the data. After the analysis of raw data, we find out that there are three kinds of problematic data. Firstly, in messages.xls, some messages are sent from a person to himself or herself. Secondly, in messages.xls, a message's topic is out of the topics in topics.xls. Thirdly, because of the lists of known conspirators, known innocence and senior managers only give the name of people, some people with the same names (**Table 5.1**) need to be distinguished. For the first and the second kinds of problematic data, it is necessary to delete these mistaken data, because they are useless in establishing model.

Name	Identity	Node 1	Node 2
Gretchen	manager	4	32
Elsie	conspirator	7	37
Beth	unknown	14	38
Jerome	manager	16	34
Neal	unknown	17	31

Table 5.1 The nodes that share the same name

For the third kinds of problematic data, let's firstly give a definition of closeness:

$C_{closeness}(k)$ measures the sum of the shortest distance between a particular node k and all the other nodes in the social network. It actually measures how far away vertex i is from other nodes:

$$C_{\text{closeness}}(k) = \sum_{i=1}^{n} dist(k,i)$$

The smaller closeness is, the higher status of the node is in the social network. Then we can compare the closeness of people with the same names in particular social network. (The code is in **Appendix 4.1**)

Firstly, we identify whether the conspirator Elsie is node 7 or 37. Messages about suspicious topic 7, 11, 13 are selected out to establish a criminal social network. In this network, the closeness of Elsie 7 is 516 and the closeness of Elsie 37 is 523. The shorter closeness is, the closer person is with the network. And Elsie 7 sends and receives more suspicious messages than Elsie 37, so Elsie 7 is the real conspirator and Elsie 37 is unknown.

Then, we try to distinguish the managers who have the same names. We think that topic 1, 2, 4, 6, 10, 14, 15 in topic.xls concentrate on company's development, according their content, and managers should be the cores of these topics. Like identifying the conspirator, we select these topics to build a working social network. Members' closeness (**Table 5.2**) in this network is a very important evidence to support who is manager and who is not. Clearly, senior managers are Gretchen 32, Jerome 34, and Dolores 10.

Node	Name	Closeness
32	Gretchen	275
4	Gretchen	284
34	Jerome	295
10	Dolores	301
16	Jerome	319

Table 5.2 The closeness of some people in the network

Finally, we give the list of known identity of members in node number. (**Table 5.3**)

Conspirators	Innocence	Senior Manager
18	48	32
21	64	34
7	65	10
43	68	
54	74	
67	0	
49	2	
	78	

Table 5.3 The analysis result of identities of key members

VI. Iterative Model

The main idea in this model is that the SDM of a member depends on both the SDM of other members whom he or she talked with and the SDT of the topics they talked about.

The steps to calculate the SDM_k of member k are as follows: Firstly, pick node i which is directly connected to k in the network. The initial value $(P_i(k), \alpha_i(k))$ should be produced, which represents the SDM relation between member k and i. $P_i(k)$ represents the largest SDT between topics talked by member i and k. $\alpha_i(k)$ represents the number of messages between vertex i and k. It is the number of different topics they talked about. Then $(P_i(k), \alpha_i(k))$ from different member i is sorted by $P_i(k)$ in descend order. Afterwards, we select the first N pairs to calculate member k's SDM in the following function (if the number of pairs is smaller than N, all the pairs need to be selected to calculate).

If a member said a lot of message on low-suspicious topics to others, his or her SDM would decrease quickly, even though he or she has said some high-suspicious topics to members with high SDM. Therefore, we use the boundary N to avoid such events. N can be set by the number of conspirators supposed and the number of members in the case. For this case, we set N to 5.

$$SDM_k = \frac{\sum_{i=1}^{N} P_{ik} \times \alpha_{ik} \times SDM_i}{\sum_{i=1}^{N} \alpha_{ik}}$$

Initially, SDM_i is set to 1.0 for known conspirators and 0.0 for innocence which won't be changed in the procedure of iteration. Other nodes' SDM are assigned 0.1 as the initial value. Considering the suspicious extent to the topics, we assign the initial SDT in **Table 6.1**

Topic	Weight	Topic	Weight	Topic	Weight
1	0	**6**	0	**11**	0.9
2	0	**7**	1	**12**	0
3	0	**8**	0	**13**	0.8
4	0	**9**	0	**14**	0
5	0	**10**	0	**15**	0

Table 6.1 The weight of edges estimated for the case

Then the program calculates the SDM of each node according to the function iteratively. The SDM converges rapidly as expected. When everyone's SDM no longer changes, the program halts. The results are listed and analyzed in **Part VIII**. The code is listed in **Appendix 4.3**

VII. Support Vector Machine Model

In order to get a synthesis results from all possible features, we use machine learning algorithms [3] for the classification of conspirators and non-conspirators. Considering that there exist known conspirators and innocence, we choose Support Vector Machine (SVM) for this supervised learning.

The goal of the SVM algorithm is to construct a hyperplane called decision boundary to separate the vectors consisting of different features like closeness and betweenness in a high-dimensional space apart, which can be used for classification or regression. When the decision boundary has the largest distance to the nearest training data points of any class, the best separation is chosen. [4]

Let w denote a vector orthogonal to the decision boundary, and b denote a scalar "offset" term. The decision boundary can be written as follows:

$$\frac{\mathrm{w}^{\mathrm{T}}}{|w^T|}x + \frac{b}{|w^T|} = 0 \text{ [3]}$$

However, the nodes in the social network are not linearly separable. Therefore, we need to map the original finite-dimensional space into a much higher-dimensional space. To keep the algorithm efficient, the mapping should ensure that the dot products be computed with low time complexity by defining a kernel function. In this case, we use radial basis function as kernel function:

$$r = |x - x_i|$$

$$\phi(r) = e^{-(\varepsilon r)^2}$$

So the vectors defining the hyperplanes could be selected to be linear combinations with parameters α_i of images of feature vectors that occur in the data base. With this choice of a hyperplane, the points x in the feature space that are mapped into the hyperplane are defined by the relation:

$$\sum_{\mathrm{i}} \alpha_i K(x_i, x) = constant$$

Then, by using SVM training algorithm, which builds a model with given labeled vectors, we can get a decision boundary which classifies the test data into different categories. Feature 1, 3, 4 is built on $\mathrm{G} = (\mathrm{V}, \mathrm{E}, \mathrm{W})$, which denotes the whole social network of the company while Features 2 and 3 are built in $\mathrm{G}' = (\mathrm{V}, \mathrm{E}', \mathrm{W}')$, which denotes the social network with edges that contain suspicious messages.

The details of features are as follows:

1. $\mathcal{P}$ This probability measures the suspicion degree that the Iteration Method gives.
2. $\deg_{\text{effect}}(k)$ The effective degree measures the active level of a particular person in the crime graph. On one hand, $\deg_{out}(k)$ measures the number of suspicious messages vertex i transmitted. On other hand, $\deg_{in}(k)$ measures how many suspicious messages vertex i received. Define

$$\deg_{\text{effect}}(k) = \alpha \deg_{in}(k) + \beta \deg_{out}(k)$$

Considering that transmitting a message is more suspicious than receiving one, in our model,

$$\alpha = 1.2, \qquad \beta = 1.0$$

3. $dist'(i,j)$: The shortest distance between i,j measures the length of the shortest path between i,j in the criminal graph G'. Floyd algorithm is used to compute the shortest distance in $O(|V|^3)$. The algorithm is a simple dynamic programming procedure:

$$dist'(i,j) = min\big(dist'(i,k) + dist(k,j)\big), i,j,k \in V$$

$$\mathrm{D}_{\min}(k) = \min\big(dist'(k,i)\big), i,k \in V$$

4. $\mathrm{C}_{\text{closeness}}(k)$ The closeness (also called farness) measures the sum of the shortest distance between a particular node k and all the other nodes in the social network. It actually measures how far away vertex i is from other nodes. The smaller closeness is, the higher status of the node is. The code is in **Appendix 4.1**
5. $\mathrm{C}_{\text{betweenness}}(k)$ The betweenness measures the extent to which a particular vertex lies between others in the network. [1]

$$\mathrm{C}_{\text{betweenness}}(k) = \sum_{(i=1)}^{n} \sum_{j=1}^{n} (dist(i,k) + dist(k,j) = dist(i,j))$$

An individual with high betweenness may act as an important person for smooth communication in the criminal network. The code is in **Appendix 4.2**

After calculating all these features, we use an open-source tool called LIBSVM [5] to train the model. Then we test all individuals in the network with the model.

VIII. Analysis of the Result

By iterative model, the list of partly most suspicious members ($SDM > 0.1$) except known conspirators is presented in **Table 8.1**. The complete suspicious list is in **Appendix 1.1**

Node	SDM	Node	SDM	Node	SDM
81	0.550	37	0.271	13	0.175
51	0.506	28	0.268	16	0.175
10	0.406	33	0.266	56	0.132
60	0.367	30	0.228	3	0.127
17	0.331	36	0.223	32	0.118
38	0.280	57	0.206	22	0.113
50	0.271	6	0.199	11	0.109

Table 8.1 The result of Iterative Model corresponding to Requirement 1

Comparing the list with **Figure 8.1** of criminal social network, we separate these high-suspicious members into three groups. The first group consists of 10, 17, 38, 50, 37, 28, 33, 30, 36, 6, 13, 16, 3, 32, 22, 11. They are suspicious, because they are cores in the criminal social network which connect with many members or connect with some members and known conspirators. The second group consists of 81, 51, 60, 37. They only connect to one member in the network, but the only member they connect is known conspirators. There is no low-suspicious member to help them lower their suspiciousness, so it's reasonable to suspect them. The third group consists of 57, 56. They have no connection with other members in the large network. They are suspected because they talked about suspicious topics between each other.

Additionally, we can find two senior managers in this list of high suspiciousness, 10 and 32. 10 has higher SDM than 32, because 10 connects more members with high SDM than 32 does. If the police want to interrogate the most suspicious senior manager in the company, 10 is the best choice.

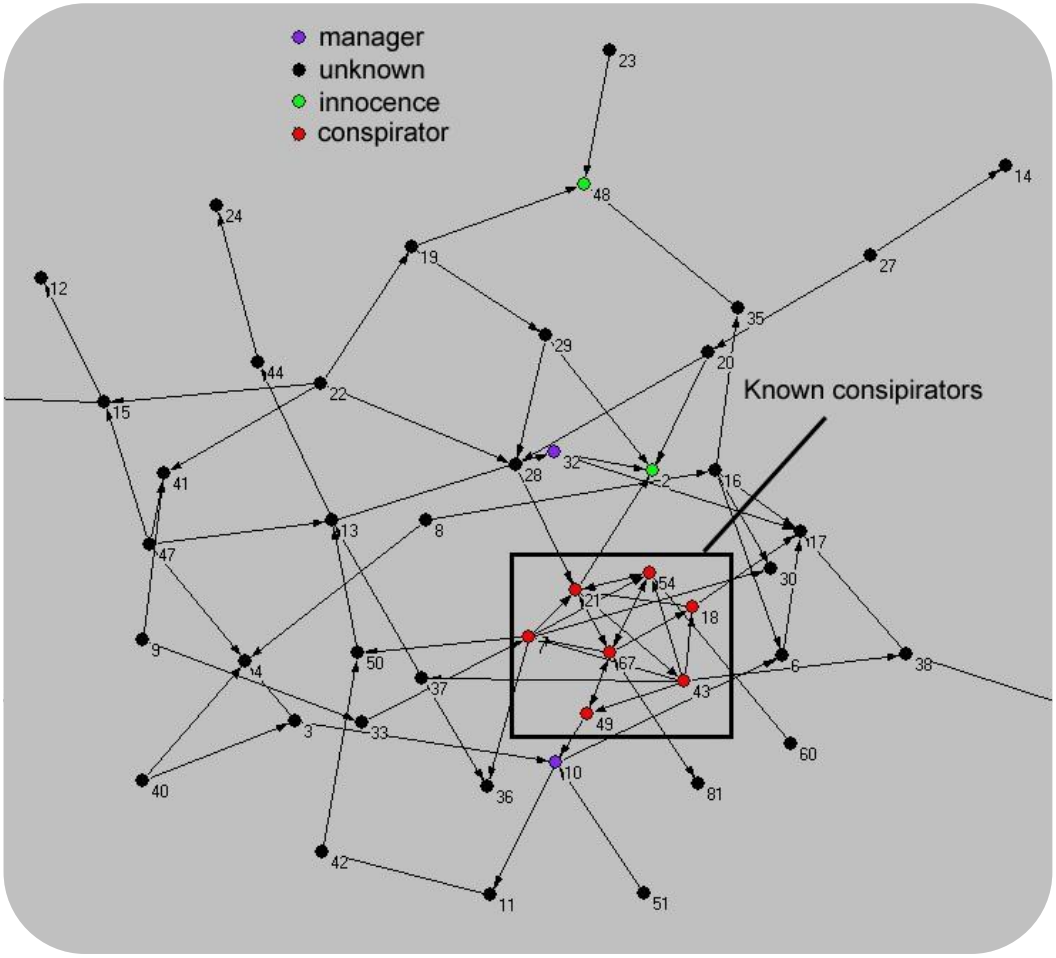


Figure 8.1 Graph corresponding to Requirement 1 (Drawn by Pajek[6])

By iterative model, the list of partly most suspicious members (SDM > 0.1) except known conspirators is presented in **Table 8.2**. The complete suspicious list is in **Appendix 2.1**

Node	SDM	Node	SDM	Node	SDM
81	0.550	**30**	0.228	**32**	0.148
51	0.514	**36**	0.224	**3**	0.145
10	0.414	**57**	0.219	**15**	0.145
60	0.367	**20**	0.219	**22**	0.143
17	0.340	**6**	0.203	**47**	0.136
28	0.300	**69**	0.195	**9**	0.125
38	0.290	**13**	0.192	**31**	0.123
50	0.281	**16**	0.178	**27**	0.123
33	0.275	**45**	0.160	**11**	0.118
37	0.274	**56**	0.149	**41**	0.101

Table 8.2 The result of Iterative Model corresponding to Requirement 2

If Member 0 is one of the conspirators and Topic 1 is connected to the conspiracy, Member 20, 69, 45, 15, 47, 9, 31, 27, 41 are added to high suspiciousness list, besides a few orders change a little in members who have already been in this list. Many of these newcomers are given a low SDM, compared with members in the list. That means that a member will not be suspected a lot if he or she only takes part in one suspicious topic. Member 20 and 69 are special. They are newcomers, but they are located at a comparatively high position **Figure 8.2** in the list, because they are not low enough in original complete SDM list and have much connection in newly additive topic.

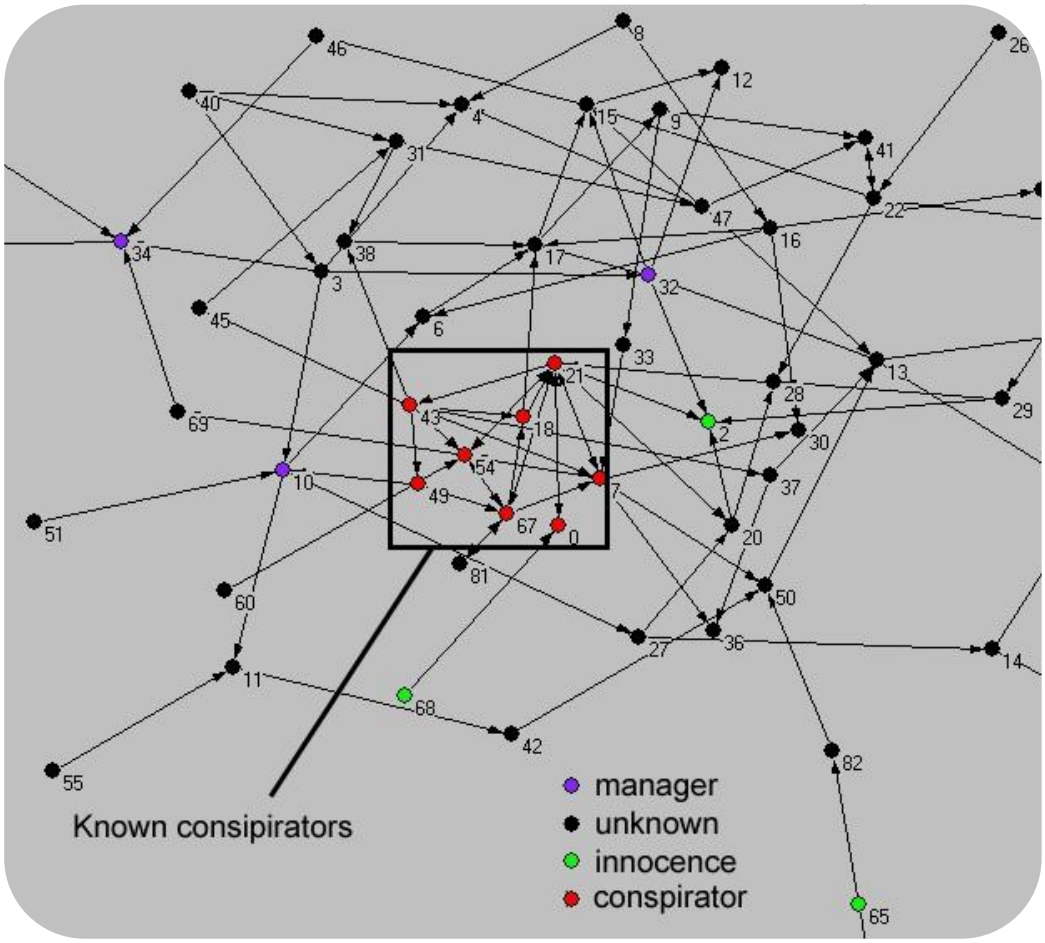


Figure 8.2 Graph corresponding to Requirement 2

The given conspirators and innocence and corresponding features are used as training data for SVM algorithm to learn. The test data and the corresponding posterior probabilities are as follows (top 30):

Node	Probability	Node	Probability	Node	Probability
18	87.46%	28	65.23%	3	25.34%
49	87.46%	38	65.14%	22	25.32%
54	87.46%	50	65.08%	11	25.08%
7	87.46%	37	64.99%	42	24.89%
67	87.46%	33	64.76%	20	24.84%
21	87.46%	30	64.19%	31	24.78%
43	87.46%	17	41.20%	23	17.58%
81	68.03%	51	27.33%	16	15.97%
10	67.94%	6	25.68%	32	15.77%
60	65.86%	13	25.59%	24	15.71%

Table 8.3 The top 30 result of SVM Model corresponding to Requirement 1

As for the question two, after new information comes to light that Topic 1 is also connected to the conspirator and Chris is one of the conspirators, things become:

Node	Probability	Node	Probability	Node	Probability
43	91.89%	3	53.09%	13	52.78%
67	91.89%	81	53.09%	32	52.50%
7	91.88%	10	53.09%	22	52.42%
0	91.88%	60	53.08%	30	52.32%
49	91.88%	28	53.04%	36	52.15%
18	91.88%	38	53.03%	20	51.67%
54	91.88%	33	52.97%	57	51.58%
21	91.88%	50	52.97%	15	51.27%
51	53.09%	37	52.96%	6	51.14%
17	53.09%	16	52.81%	69	50.75%

Table 8.4 The top 30 result of SVM Model corresponding to Requirement 2

IX. Semantic Analysis

If we could obtain the original messages, it would certainly be a great help for developing a better model and categorizations of the office personnel.

Firstly, by semantic network analysis, many powerful features can be added to the SVM to classify conspirators and innocence. For instance, the words' and the phrases' frequencies might improve the result a lot. For one thing, the group of conspirators might use certain words to transform the latest news. For another, the name of very important person among the conspirators might be mentioned frequently. Let's analysis the semantic network of the sample:

Two kinds of words seem suspicious in the sample: One is "budget", the others are words like "tired, stressed, exhausted". Let's take "budget" as an illustration:

	Anne	Bob	Carol	Dave	Ellen	Fred	George	Harry	Inez	Jaye
Budget(from)	0	0	0	2	2	1	0	2	0	0
Budget(to)	0	0	1	0	1	2	1	2	0	0

Table 9.1 The frequency of the word "budget" in the EZ case

From the table above, we can figure out that **Dave, Ellen, Fred and Harry** have something to do with the budgets. Afterwards, by parsing and semantic analysis, only **Dave, Ellen and George** are busy with budgets.

Furthermore, by parsing the messages, more details of the relationship in the social network can be revealed. That is to say, whether a person likes or dislikes another person helps a lot in our model. Large amount of communication does not necessarily equal to a good relationship. Instead, the relationship might be rather bad. For instance, in the sample,

Anne to Carol: Who is supposed to watch Bob? He is goofing off all the time. (1)

Carol to Anne: Leave him alone. He is working well with George and Dave. (1)

It can be inferred that Anne seems to have trouble with Bob though they have a lot of communication, but deal well with Carol. Meanwhile, from Carol's message, **Bob** maintains a rather good relationship with **George and Dave**. By adding two edges between Bob and George, Bob and Dave and applying the Iterative Model, we have the following result:

Name	SDM	Name	SDM
Dave	1.00	Inez	0.25
George	1.00	Carol	0.13
Bob	0.48	Harry	0.13
Ellen	0.38	Anne	0.00
Fred	0.25	Jaye	0.00

Table 9.2 The results of Iterative Model corresponding to the EZ case

Therefore, by semantic analysis, people like Bob would not have the opportunity to get reduced sentences.

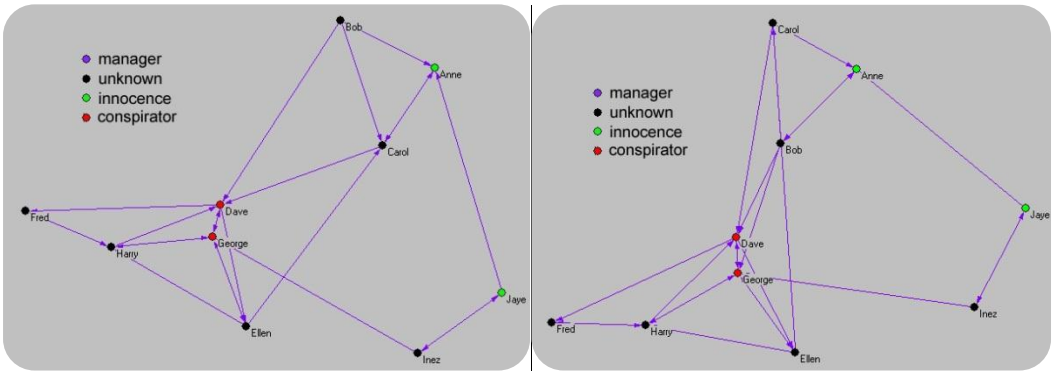


Table 9.2 The left one is the original network, the right one adds Bob to George

In addition, intimation degree can be calculated between two guys. By applying the intimation degree to the edge in the network, more accurate shortest distances can be calculated, thus closeness becomes more accurate.

Last but not the least, with semantic analysis, we can figure out those people who are really tired or exhausted or stressed in the company:

	Anne	Bob	Carol	Dave	Ellen	Fred	George	Harry	Inez	Jaye
Times	0	0	0	0	0	-1	3	1	2	0

Table 9.3 The times of tired /exhausted /stressed (minus means opposite idea)

Since the work in this company is rather light, George and Inez seem to be suspicious characters. Thus, people like Inez would hardly get off based on semantic analysis. Nevertheless, being tired can hardly be a direct evidence to arrest Inez, so more evidence is badly in need.

Nevertheless, we don't have the contents and context of the messages in the

network. However semantic analysis on topic descriptions still helps a lot. It can never be neglected that Spanish has a high word frequency in the description text. Thus topic 2 and 13 might be suspicious. And topic 8 might be the most suspicious among all the topics. By this argument, we assign new weights to edges in Iterative Model and SVM Model:

Topic	Weight	Topic	Weight	Topic	Weight
1	0.6	**6**	0	**11**	0.9
2	0.3	**7**	1	**12**	0.3
3	0	**8**	0	**13**	0.8
4	0	**9**	0	**14**	0
5	0	**10**	0.1	**15**	0

Table 9.4 The revised weight of edges corresponding to Requirement 3

The results are listed in **Appendix 3.1.**

X. Executive Summary

Attn: Chief of DA

From: MCM Team 14274

Subject: A Novel Idea for Network Analysis by Iterative Model and Support Vector Machine (SVM) Model.

Dear Sir or Madam,

We reviewed the conventional approaches for network analysis and found the methods limited in the small data and they depended on the investigator's judgment to a great extent. It is true that these measures are useful to some extent when deducing a small network. However, when the number of nodes is large and the links become very complex, these conventional approaches may offer little help. Therefore, we would like to think about this problem from a new perspective and create a new model to decrease the error as well as increase the stability.

Before the establishment of our new-concept model, we introduced some parameters such as closeness and betweenness. Then we established a mathematics link between every two nodes and calculate these parameters of each node. With the parameters, we can deal with the raw data easily and solve the trouble led by the duplication of names. In order to simulate the sustained interrelationship among people, we decided to use iteration method and the parameters converge rapidly as expected. What is more, we changed the condition a little and the result changed little, which shows the model's stability. Additionally, this model doesn't need too much time to get a good result, so it's very suitable to deal with a large social network with many nodes and complex relations.

At first, we were satisfied with the results of the EZ case by our model. Then, we applied this model to the case. Nevertheless, we noticed an important factor that may

have a great influence on our model. If we were policemen, in order to avoid acting rashly and alerting the criminals, we would like to catch the leader of criminals first instead of catching others in case the leaders escape in advance. So it is necessary to build a model considering not only suspicious degrees of nodes but also status of nodes in the criminal social network.

After reading some papers on social network analysis, we found that closeness and betweenness can measure the importance of nodes in social network. We used machine learning algorithms for the classification of conspirators and non-conspirators. Considering that there exist known conspirators and innocence, we chose Support Vector Machine (SVM) for this supervised learning.

The nodes have high possibility to be the leaders among the criminals must have both high suspicious degree as well as high status in criminal social network. Consequently, SVM is rather effective to provide an arresting-prior list to the police.

However, the social network we involved in the model is established only by messages. We believe that if we have text information, with the help of semantic network analysis, we could gain new relations and detailed suspicious messages to optimize the prior list by SVM. We tried this on the EZ case. The result is satisfactory that Bob is more suspicious than Coral and Inez is ranked high in the prior list.

In the end, our models, Iterative Model and SVM Model are very adaptable. Some parameters are added in our methods, which could be adjusted to match different types of crime network. Our models could be applied to other types of network as well, such as biological network, because in another aspect, infected cells are like known criminals and the sustained interrelationship among cells can also be simulated by iteration method.

Yours Cordially,
MCM Team 14274

XI. References

[1] Xu, J., and Chen, H. CrimeNet explorer: A framework for criminal network knowledge discovery. To appear in ACM Trans. on Info. Systems.

[2] McAndrew, D. 1999. The structural analysis of criminal networks. In The Social Psychology of Crime: Groups, Teams, and Networks. D. Canter and L. Alison, Eds. Dartmouth Publishing, Aldershot, UK, 53–94.

[3] M. Bishop, M. Jordan, J. Kleinberg, B. Scholkopf, Pattern recognition and Machine Learning, Second Edition, Chapter 7

[4] Wikipedia, http://en.wikipedia.org/wiki/Support_vector_machine

[5] The official website of LIBSVM: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[6] Pajek, http://vlado.fmf.uni-lj.si/pub/networks/pajek/

XII. Appendix

The solution based on Iterative Model to question 1:

Node	SDM	Node	SDM	Node	SDM	Node	SDM
49	1.000	**13**	0.175	**79**	0.068	**46**	0.033
18	1.000	**16**	0.175	**72**	0.068	**45**	0.032
67	1.000	**56**	0.132	**40**	0.055	**24**	0.028
43	1.000	**3**	0.127	**19**	0.054	**14**	0.027
21	1.000	**32**	0.118	**44**	0.053	**39**	0.027
7	1.000	**22**	0.113	**35**	0.051	**62**	0.025
54	1.000	**11**	0.109	**53**	0.050	**71**	0.020
81	0.550	**73**	0.100	**75**	0.050	**82**	0.020
51	0.506	**61**	0.100	**63**	0.050	**26**	0.020
10	0.406	**58**	0.100	**76**	0.050	**25**	0.020
60	0.367	**59**	0.100	**80**	0.040	**23**	0.020
17	0.331	**47**	0.097	**27**	0.040	**1**	0.020
38	0.280	**42**	0.088	**5**	0.040	**68**	0.000
50	0.271	**4**	0.084	**34**	0.039	**65**	0.000
37	0.271	**9**	0.078	**12**	0.034	**64**	0.000
28	0.268	**41**	0.078	**69**	0.034	**74**	0.000
33	0.266	**20**	0.076	**66**	0.033	**48**	0.000
30	0.228	**31**	0.075	**70**	0.033	**78**	0.000
36	0.223	**15**	0.072	**55**	0.033	**2**	0.000
57	0.206	**29**	0.071	**52**	0.033	**0**	0.000
6	0.199	**8**	0.068	**77**	0.033		

Appendix 1.1 The results of iterative model corresponding to Requirement 1

The solution based on Support Vector Machine Model to question 1 (with normalization) :

Node	Probability	Node	Probability	Node	Probability
18	100.00%	**53**	7.75%	**80**	7.72%
49	100.00%	**63**	7.75%	**5**	7.72%
54	100.00%	**32**	18.03%	**34**	7.71%
7	100.00%	**24**	17.96%	**69**	7.70%
67	100.00%	**9**	14.28%	**66**	7.70%
21	100.00%	**44**	13.61%	**70**	7.70%
43	100.00%	**45**	13.59%	**55**	7.70%
81	77.79%	**47**	11.85%	**52**	7.70%
10	77.68%	**4**	11.80%	**77**	7.70%
60	75.30%	**15**	11.78%	**12**	7.69%

28	74.59%	**41**	11.77%	**46**	7.69%
38	74.48%	**19**	11.69%	**39**	7.68%
50	74.41%	**27**	11.66%	**14**	7.68%
37	74.30%	**8**	9.50%	**62**	7.68%
33	74.04%	**40**	9.48%	**82**	7.67%
30	73.40%	**35**	9.46%	**71**	7.67%
17	47.10%	**29**	9.35%	**26**	7.67%
51	31.25%	**57**	9.31%	**25**	7.67%
6	29.36%	**56**	8.35%	**1**	7.67%
13	29.26%	**58**	8.06%	**2**	7.66%
3	28.97%	**59**	8.06%	**68**	7.65%
22	28.95%	**61**	8.06%	**65**	7.65%
11	28.67%	**73**	8.06%	**74**	7.65%
42	28.46%	**79**	7.84%	**64**	7.65%
20	28.40%	**72**	7.84%	**48**	7.65%
31	28.34%	**36**	7.84%	**78**	7.65%
23	20.10%	**76**	7.75%	**0**	7.65%
16	18.26%	**75**	7.75%		

Appendix 1.2 The SVM results corresponding to Requirement 1(normalize to 1)

The solution based on Iterative Model to question 2:

Node	SDM	Node	SDM	Node	SDM	Node	SDM
0	1.00	**20**	0.22	**4**	0.10	**80**	0.04
43	1.00	**6**	0.20	**42**	0.09	**71**	0.04
49	1.00	**69**	0.20	**34**	0.09	**26**	0.04
67	1.00	**13**	0.19	**29**	0.08	**39**	0.04
21	1.00	**16**	0.18	**40**	0.08	**62**	0.03
7	1.00	**45**	0.16	**72**	0.07	**66**	0.03
18	1.00	**56**	0.15	**8**	0.07	**77**	0.03
54	1.00	**32**	0.15	**79**	0.07	**52**	0.03
81	0.55	**3**	0.15	**12**	0.07	**70**	0.03
51	0.51	**15**	0.14	**44**	0.06	**24**	0.03
10	0.41	**22**	0.14	**19**	0.06	**25**	0.03
60	0.37	**47**	0.14	**63**	0.06	**1**	0.02
17	0.34	**9**	0.13	**55**	0.06	**23**	0.02
28	0.30	**31**	0.12	**82**	0.05	**2**	0.00
38	0.29	**27**	0.12	**14**	0.05	**74**	0.00
50	0.28	**11**	0.12	**35**	0.05	**68**	0.00
33	0.28	**41**	0.10	**5**	0.05	**78**	0.00
37	0.27	**73**	0.10	**76**	0.05	**65**	0.00
30	0.23	**58**	0.10	**75**	0.05	**64**	0.00
36	0.22	**61**	0.10	**53**	0.05	**48**	0.00

57	0.22	**59**	0.10	**46**	0.05		

Appendix 2.1 The results of iterative model corresponding to Requirement 2

The solution based on Support Vector Machine Model to question 2 (with normalization) :

Node	Probability	Node	Probability	Node	Probability
43	100.00%	**6**	55.65%	**14**	21.93%
67	100.00%	**69**	55.23%	**26**	21.16%
7	100.00%	**56**	53.40%	**39**	20.66%
0	100.00%	**45**	50.86%	**12**	19.28%
49	100.00%	**47**	49.35%	**24**	17.67%
18	100.00%	**1**	41.39%	**79**	16.98%
54	100.00%	**9**	40.41%	**25**	16.83%
21	100.00%	**27**	39.85%	**35**	14.94%
51	57.77%	**72**	39.83%	**82**	14.21%
17	57.77%	**75**	39.38%	**5**	13.22%
3	57.77%	**11**	38.32%	**63**	13.05%
81	57.77%	**31**	35.22%	**76**	12.67%
10	57.77%	**23**	32.95%	**53**	12.67%
60	57.77%	**41**	31.00%	**46**	12.42%
28	57.72%	**80**	29.10%	**71**	10.67%
38	57.71%	**73**	28.09%	**62**	10.18%
33	57.65%	**61**	28.09%	**52**	10.10%
50	57.64%	**59**	28.09%	**77**	10.10%
37	57.63%	**58**	28.09%	**70**	10.10%
16	57.47%	**34**	27.88%	**66**	8.53%
13	57.44%	**40**	27.72%	**65**	8.24%
32	57.13%	**4**	27.56%	**68**	8.24%
22	57.04%	**42**	26.73%	**48**	8.24%
30	56.93%	**8**	26.42%	**74**	8.24%
36	56.75%	**55**	24.75%	**2**	8.24%
20	56.23%	**29**	24.11%	**64**	8.23%
57	56.13%	**19**	23.75%	**78**	8.23%
15	55.80%	**44**	23.62%		

Appendix 2.2 The SVM results corresponding to Requirement 2(normalized to 1)

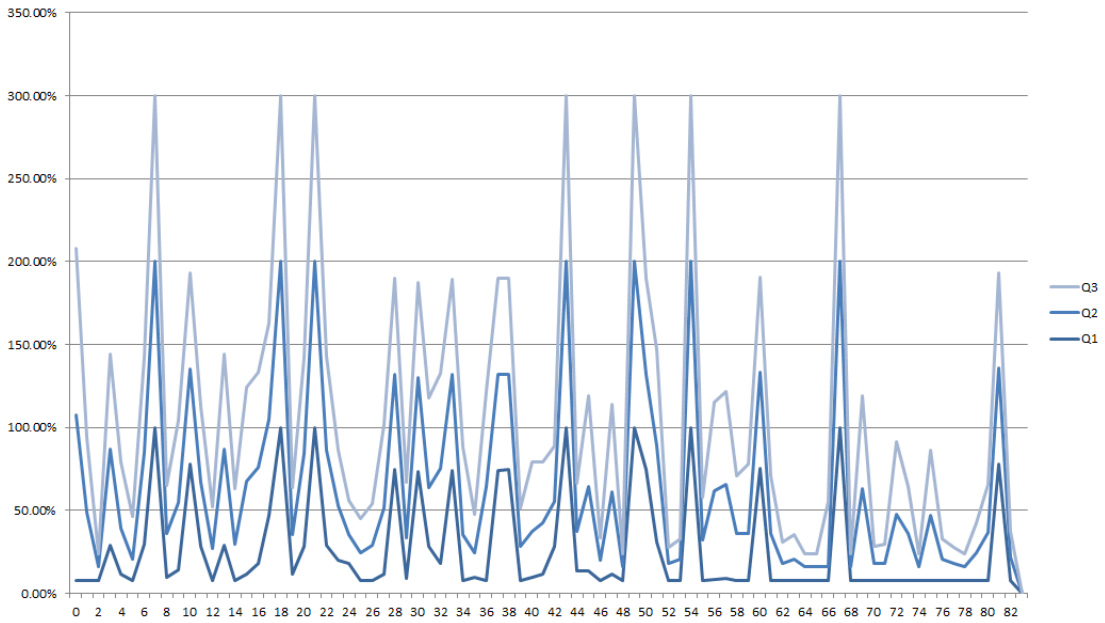
Based on Support Vector Machine and Iterative Model with new weights by semantic analysis, we have the following results:

Node	Probability	Node	Probability	Node	Probability
43	100.00%	**15**	56.39%	**73**	28.09%

67	100.00%	**57**	56.18%	**55**	25.86%
7	100.00%	**45**	54.41%	**12**	25.50%
0	100.00%	**31**	54.41%	**5**	25.35%
49	100.00%	**56**	53.51%	**26**	25.34%
18	100.00%	**47**	52.94%	**39**	23.49%
21	100.00%	**34**	52.44%	**35**	23.38%
54	100.00%	**27**	50.11%	**25**	20.78%
81	57.77%	**9**	49.84%	**24**	20.71%
3	57.77%	**11**	45.69%	**79**	17.83%
51	57.77%	**1**	44.66%	**82**	15.80%
10	57.77%	**72**	43.69%	**63**	14.52%
17	57.77%	**40**	42.34%	**62**	13.41%
60	57.77%	**59**	41.97%	**46**	13.33%
28	57.74%	**4**	39.76%	**53**	12.67%
38	57.74%	**75**	39.38%	**76**	12.67%
50	57.71%	**66**	39.28%	**71**	11.01%
33	57.70%	**41**	36.36%	**70**	10.61%
37	57.70%	**58**	34.48%	**77**	10.10%
16	57.60%	**61**	34.47%	**52**	10.10%
13	57.57%	**14**	33.80%	**65**	8.24%
20	57.39%	**42**	33.51%	**68**	8.24%
32	57.31%	**29**	33.32%	**48**	8.24%
30	57.31%	**23**	32.95%	**74**	8.24%
22	57.25%	**80**	29.20%	**2**	8.24%
6	57.13%	**8**	29.10%	**64**	8.23%
36	56.89%	**44**	29.01%	**78**	8.23%
69	56.42%	**19**	28.47%		

Appendix 3.1 The SVM results corresponding to Requirement 3(normalized to 1)

Comparing the relationship between SVM solutions to Question 1, 2 and 3(with topic weight changed based on semantic analysis), it is clear that the tendencies look similar which proves that our model is rather stable. (Only node 0 is captured in question 2 and 3, but 0 talked few about suspicious topic).



Appendix 3.2 The comparison of SVM results corresponding to 3 requirements

```
void closeness(double para, int id){
    for (int i = 0; i < n; ++i ) for (int j = 0; j < n; ++j) dist[i][j] = 00;
    for (int i = 0; i < n; ++i ) dist[i][i] = 0;
    for (int i = 0; i < n; ++i ) for (int j = 0; j < n; ++j )if (e[i][j].weight >= para)
        dist[i][j] = 1;

    for (int k = 0; k < n; ++k )
    for (int i = 0; i < n; ++i ) if (i != k)
    for (int j = 0; j < n; ++j ) if (i != j && j != k) {
        dist[i][j] = min(dist[i][k] + dist[k][j], dist[i][j]);
    }

    for (int i = 0; i < n; ++i ){
        feature[i][id] = 0;
        for (int j = 0; j < n; ++j ){
            feature[i][id] += dist[i][j];
        }
    }
}
```

Appendix 4.1 Main Code of Closeness

```
void between(double para, int id){
    for (int i = 0; i < n; ++i ) for (int j = 0; j < n; ++j) dist[i][j] = 00;
    for (int i = 0; i < n; ++i ) dist[i][i] = 0;
    for (int i = 0; i < n; ++i ) for (int j = 0; j < n; ++j )if (e[i][j].weight >= para)
        dist[i][j] = 1;
```

```
    for (int k = 0; k < n; ++k )
    for (int i = 0; i < n; ++i ) if (i != k)
    for (int j = 0; j < n; ++j ) if (i != j && j != k) {
        dist[i][j] = min(dist[i][k] + dist[k][j], dist[i][j]);
    }

    for (int k = 0; k < n; ++k ){
         for (int i = 0; i < n; ++i ) if (i != k)
         for (int j = 0; j < n; ++j ) if (i != j && j != k) {
              if (dist[i][k] + dist[k][j] == dist[i][j]) ++feature[k][id];
         }
         feature[k][id] /= 2;
    }
}
```

Appendix 4.2 Main Code of Betweenness

```
for (int k = 0; k < itertaion; ++k ){
    for (int i = 0; i < n; ++i) {
        if (p[i] > 0.99 || p[i] < 0.01) continue;
        p[i] = 0.1;
        r = 0;
        for (int j = 0; j < n; ++j ) if (e[i][j].connect){
            q[r++] = e[i][j].weight * p[j];
        }
        sort(q, q+r, descend);
        int limit = min(LIMIT, r);
        for (int j = 0; j < limit; ++j ){
            p[i] += q[j];
        }
        if (limit != 0) p[i] /= limit;
    }
}
```

Appendix 4.3 Main Code of Iterative Model