Invisible Edge
CS 526 Challenge Problem 3, Group 11

John Martin Matthew Urffer Lipeng Wan

***Abstract*—Communication companies have a large amount of information on their consumers. In order to accurately bill their costumers they need to collect the amount to calls they made (including the date and time), the length of the call, the texts the costumers sent and received. This data can then be mined in order to determine patterns between their costumer's in order to improve their service, investigate alternative products, or find new customers to pursue. Data provided by Link Analytics was mined for the invisible edge between costumers; i.e. at what probability can it be determined if two customers might communicate. This was completed using an LSA based approach, a Neural Network based approach, and a spectral graph. None of these methods were successful, so it is concluded that more advanced graph theory concepts are needed, but the run time of these algorithms are generally prohibitive on such a large problem size.**

I. Introduction

Communication companies have a large amount of information on their costumers. In order to accurately bill their costumers they need to collect the amount to calls they made (including the date and time), the length of the call, the texts the costumers sent and received. This data can then be mined in order to determine patterns between their costumers in order to improve their service, investigate alternative products, or find new customers to pursue.

One of the areas of active research is then the invisible edge problem. Simply put, at what confidence can you determine if two customers contact each other knowing the call patterns of the customers? This is equivalent to determining if and edge exists between the two vertices's. Link Analytics provided two sets of graph for large, yet different U.S. cities where each graph represents one month of communication between customers. The task is to then given a list of nodes, predict if an edge exists.

The layout of this paper is as follows. First the methods used to investigate this problem are discussed, as well as the methods for effectively coping with such a large problem Next the results of these methods discussed, along with the strengths and weakness of each approach. Finally, some conclusions about this project are discussed.

II. Methods

The call graph was provided as a labeled edge list, and was modeled as a weighted, undirected, multigraph, as shown in Fig. 1. The graph was organized as each of the nodes being the unique identifier to a customer, with edges between the customers representing:

- `days` - the number of distinct days that the two nodes communicated during the month,
- `calls` - the number of distinct calls that the the nodes made during the month,
- `secs` - the cumulative sum of all calls that the modes made during the month and,
- `texts` - the number of distinct texts that the two nodes made during the month.

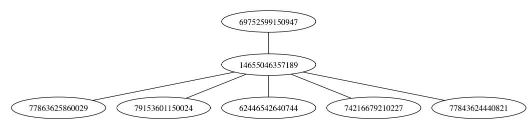

Fig. 1: Example Call Graph

The data was analyzed in order to determine the distribution of calls, texts, days, and seconds for each city in order to determine the size of the problem. It was immediately noticed that several of the callers where large outliers, these are probably robot callers in the network. In addition, the sum of the proprieties of the nodes were calculated by summing all of the edges into the nodes. It is then possible to see the distribution of how people communicated for the two cities. The degree is defined as the number of neighbors a given node has.

A. LSA Based Similarity Measure

Previous work conducted by J Martin suggested that personality profiles can be modeled with an LSA based system. Based on this premise, it was suggested that a node (person) could be effectively "described" by the set of other people they were connected to. This description could then be measured for similarity to other node descriptions and hopefully we would find that "similar" nodes were connected. In the past work we did there was

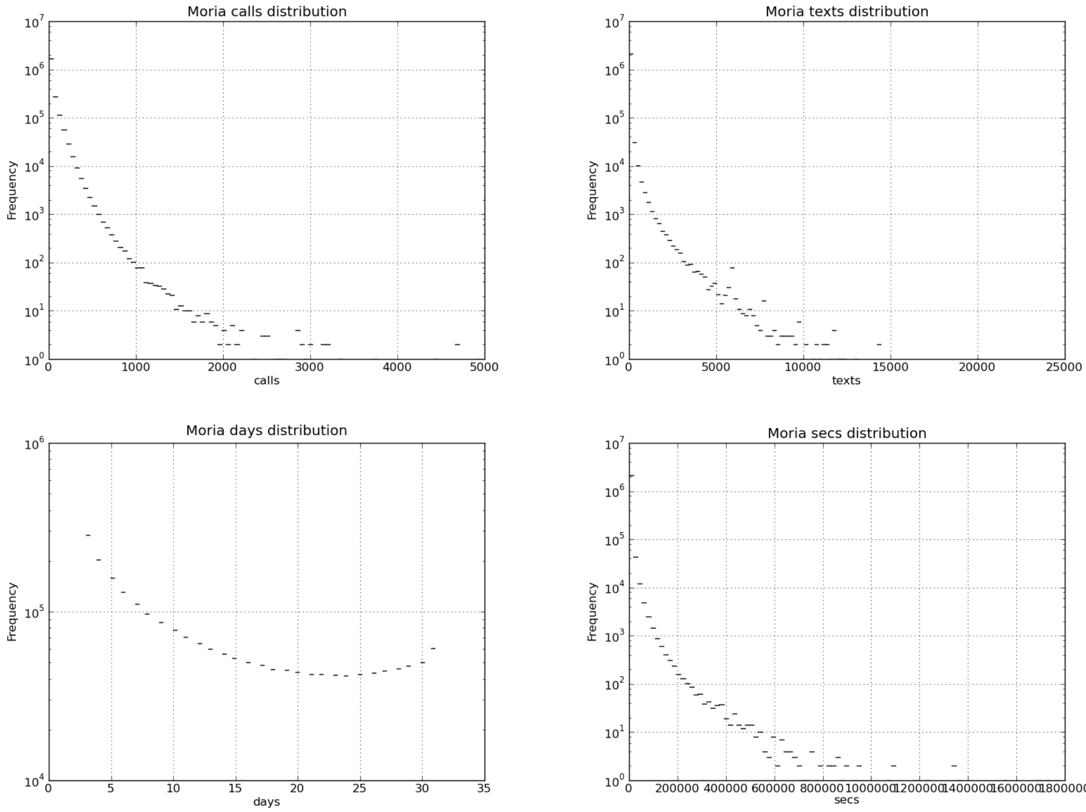

Fig. 2: Distribution of Moria Edges

some promise in the initial results for group analysis, but it was not pursued very far and there were other data items that were being considered as well (and it had nothing to do with telephone usage patterns).

This hypothesis was tested by initially constructed a sparse matrix for each of the graph files for both the Moria and Standelf. For each connected node, the connection weight was determined by taking each of the 4 attribute values, dividing by their respective standard deviation, and summing the results into a single weight for each connection. This experiment was also completed by using a simpler weighting scheme giving a 1 if any call was made and another 1 if any text was made.

This sparse matrix in any case was extremely sparse, being around 0.00051% nonzero for Moria and 0.00059% for Standelf. We usually deal with matrices that have a nonzero rate of 0.001% to 0.01% for text mining applications. I computed a truncated SVD for these sparse matrices, forming an LSA space at approximately 250 dimensions for each. Then using these 250 dimension spaces I looked at comparing the first 1,000 nodes of each graph to all the other nodes in the graph computing their vector cosine similarity and noting if the nodes were connected or not. This takes a while to run so there has not been much opportunity to tweak any of the parameters, therefore it cannot be conclude with any certainty, but the initial results do not look promising. The first sets that did not show any clear indication of connectivity determined by the similarity measure.

B. *Artificial Neural Network Classification*

The next approach implemented was an artificial neural network classification system. Using the `PyBrain` tool module a feed forward neural network was constructed with 8 inputs representing the edge between node `u` and `v`:

- the degree of node `u`,
- the closeness of node `u`,
- `days` of the edge,
- `calls` of the edge,
- `secs` of the edge,
- `texts` of the edge,
- the closeness of node `v`, and,
- the degree of node `u`.

The closeness of node was calculated using the `networkx` module with `closeness_centrality`,

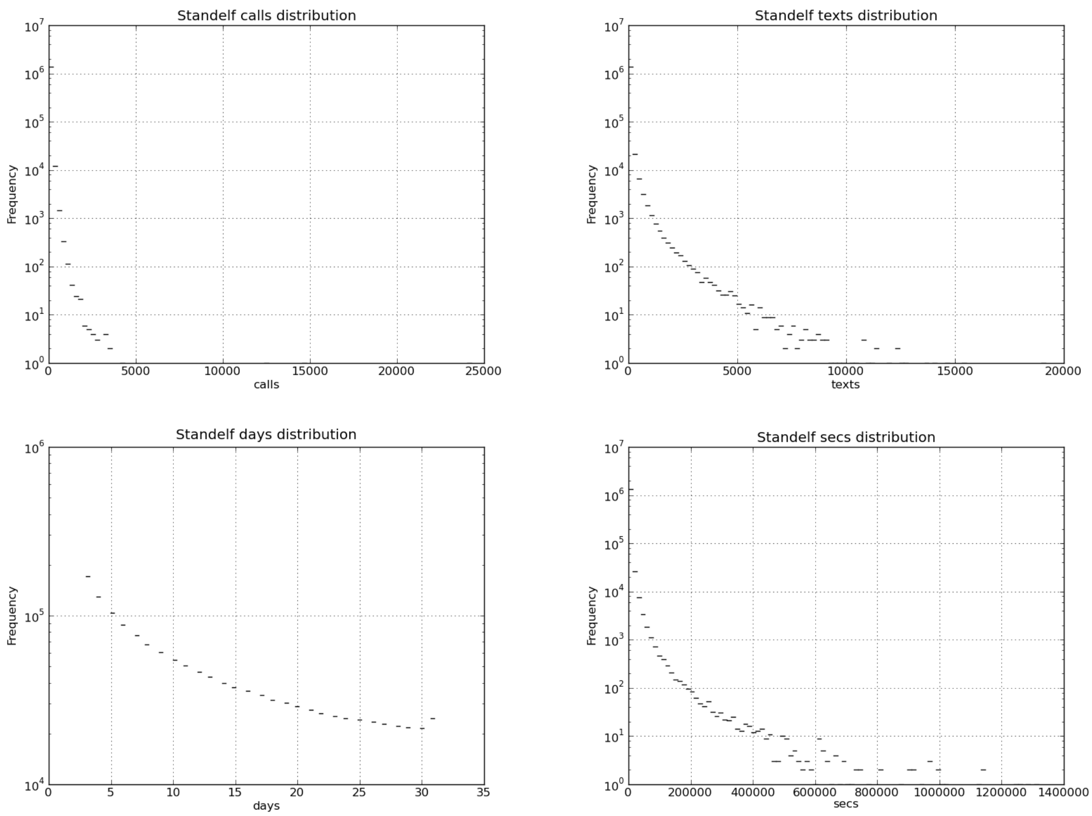

Fig. 3: Distribution of Standelf Edges

defined to be 1 over the average distance to all other nodes. The distances where not weighted, and all distances where normalized by the graph size. The degree was calculated with `degree` as the the sum of the edge weights of adjacent nodes for a particular node (completed for all nodes). The other elements of the training vector were simply filled with the values from the edge.

Training data for the neural network was then the set of all example nodes and edges presented in the target class of 1 (the edge exists). This was ultimately a flaw in the design of the neural network (or any classification based system) as discussed in Section III.

C. Spectral Approach for Finding Possible Communities in Graph

For this call graph data set, an interesting question comes to our mind which is how to find the community structure in the graph. The cell phone users in the same community will have stronger relationship between each other than that between uses from different communities. Therefore, if we can find such communities in the call graph, it might be very helpful for us to understand the cell phone users' behavior.

The approach we use to find community structure in call graph is named spectral graph. The basic idea of this approach is to use the eigenvectors of the Laplacian matrix of the graph to partition the vertices into different clusters. Now we are going to introduce the definition of the Laplacian matrix of a graph.

In graph G, let d_v denote the degree of vertex v, the Laplacian of G is defined as the matrix

$$L(u,v) = \begin{cases} 1 & \text{if } u = v \text{ and } d_v \neq 0, \\ -\frac{1}{\sqrt{d_u d_v}} & \text{if } u \text{ and } v \text{ are adjacent,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Let T denote the diagonal matrix with the (v,v)-th entry having value d_v. The Laplacian of G can be written as

$$L = I - T^{-1/2} A T^{1/2} \tag{2}$$

where A is the adjacency matrix of G (i.e., $A(x,y) = 1$ if x is adjacency to y, and 0 otherwise) and I is an identity matrix. All matrices here are $n \times n$ where n is the number of vertices in G.

Now let's show how to use spectral graph approach to find possible communities in graph. For example, we first import the raw data of call graph of Moria city in

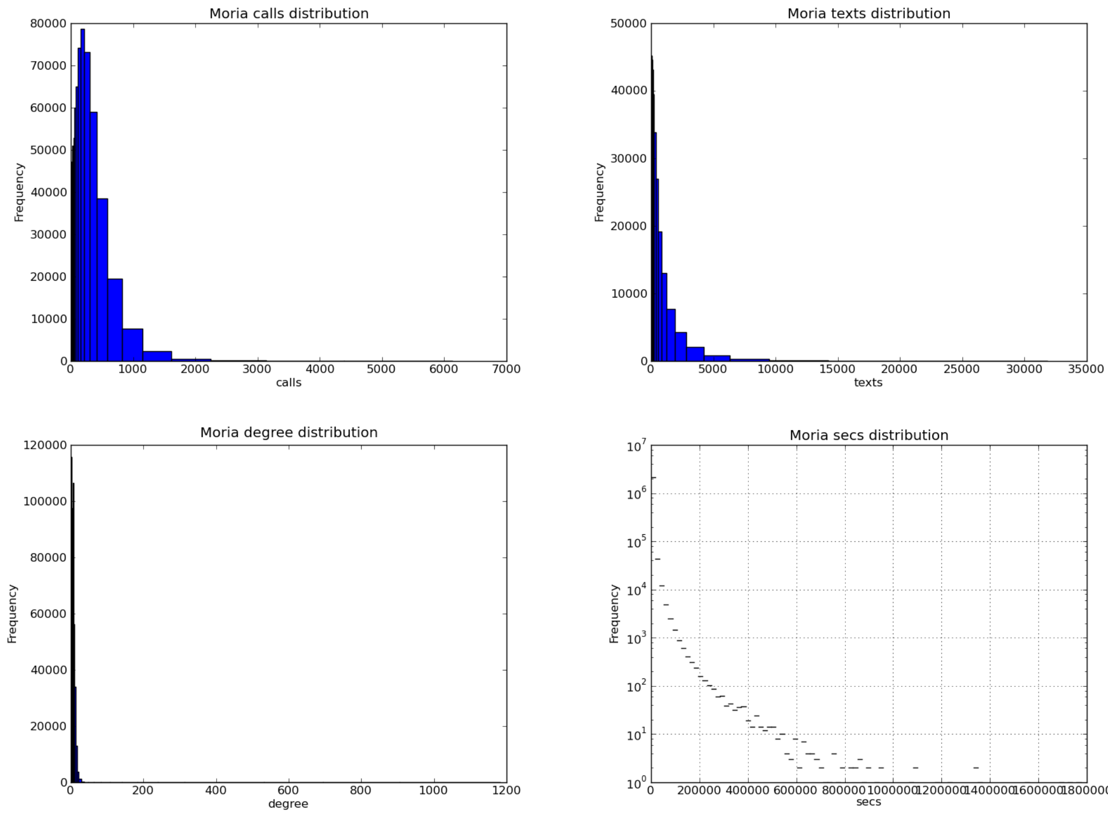

Fig. 4: Distribution of Moria Nodes

month 1 and generate the adjacency matrix of this call graph as shown in Fig. 6.

From Fig. 6, we can observe that this call graph is very dense and it is impossible for us to find any community structures in this figure because the vertices of this call graph are randomly arranged in the adjacency matrix.

Now let's investigate what the second smallest eigenvector of the Laplacian matrix tells us about this graph. The following MATLAB code is used to compute the eigenvalues and eigenvectors of the Laplacian matrix of call graph of Moria city in month 1.

```
Moria_1_I = speye(size(Moria_1_adj,1));
Moria_1_T = diag(sum(Moria_1_adj,2).^(-0.5));
Moria_1_L = Moria_1_I-Moria_1_T*Moria_1_adj*
    Moria_1_T;
[Moria_1_V,Moria_1_D] = eigs(Moria_1_L);
```

Then we sort the second smallest eigenvector and permute the vertices of this call graph based on the order of components in the sorted second smallest eigenvector.

```
[ignore Moria_1_p] = sort(Moria_1_V(:,2));
spy(Moria_1_adj(Moria_1_p,Moria_1_p));
```

In the end, the **spy()** function will plot the rearranged adjacency matrix of the call graph, which is shown in Fig. 7.

In Fig. 7, we can observe some community structure, for example, the vertices at the upper left corner of the matrix have stronger connection between each other. Similarly, we can also generate the rearranged adjacency matrix of call graph of Moria city in month 2, as shown in Figure 8

From Figure 8 we can observe that though some edges have been removed in the call graph of Moria city in month 2, the community structures in the call graph of Moria city haven't been changed much. In Figure 8, we still can find the community structures which are similar to those in Figure 7. For instance, the vertices at the lower right corner in Figure 8 and the vertices at the upper left corner in Figure 7 are in the same community.

III. Results

A. LSA Based Similarity Measure

B. Artificial Neural Network Classification

The artificial neural network trained to a very low (less than 1%) error using 5-fold cross validation and back-propagation in 20 iterations. However, when tested the accuracy of predicting whether a node existed or not

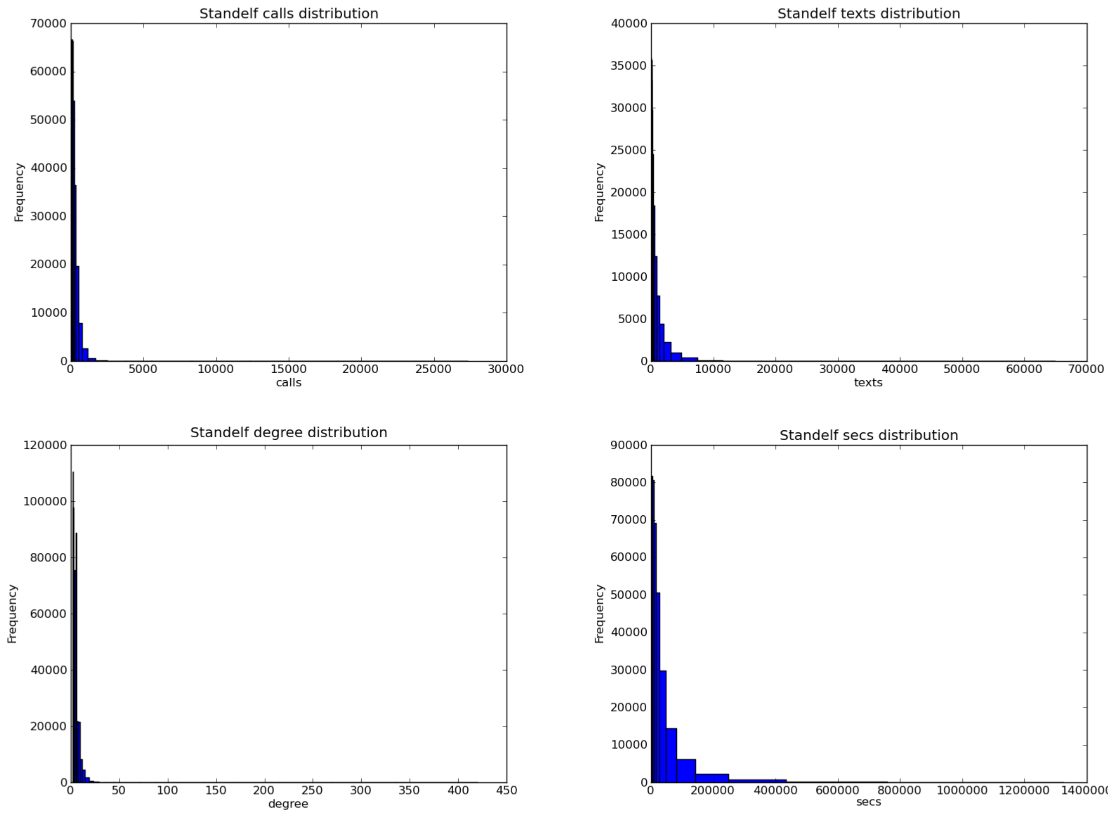

Fig. 5: Distribution of Moria Nodes

was 51%. This low accuracy is no better than flipping a coin. This is because the neural network was not feed any negative class examples; all the network had to learn was that an example input pattern corresponded to an edge. Negative input patters could not be created as network input because there is no guarantee that the input pattern would not exists, thus the network could be taught incorrect data. I

IV. Conclusions

The invisible edge problem was investigated for a large call graph representing the communication between customers in two large U.S. cities. Of the methods investigated (LSA, ANN, and spectral graph theory) none of them provided significant probability of correctly detecting the missing edge. The LSA fails because it tries to complete the matrix in the SVD stage. It is concluded that due to the lack of negative training examples (edges that should not be there) any classification based system will preform with an accuracy of 50%. It is unclear why the spectral graph methods failed. Further work might be to employ graph theory methods (such as a min-cut problem), but the run time of these algorithms is why they were discarded as methods for this work.

V. Acknowledgments

The work was distributed as follows. John completed the analysis of the data with the LSI based similarity measure. Matthew completed the distributions of the data as well as the neural network classification system. He also implemented the probability based approach. Lipeng completed the work on the spectral graph density. Many thanks to Link Analytics for providing this unique data set.

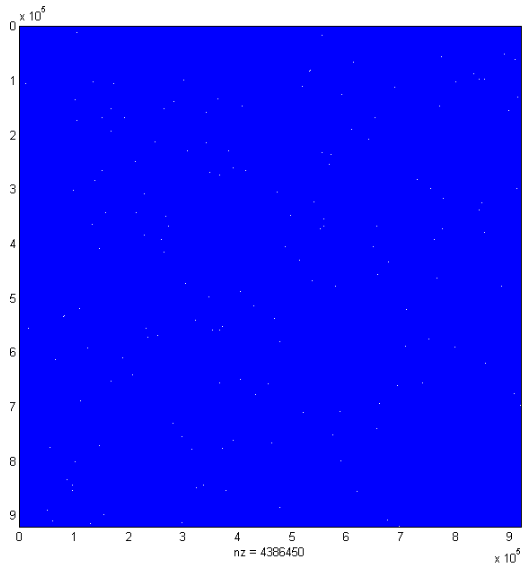

Fig. 6: Adjacency Matrix of Call Graph of Moria City in Month 1

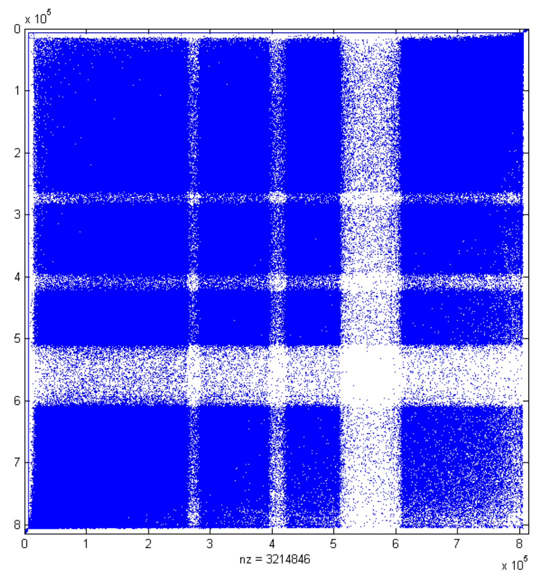

Fig. 8: Rearranged Adjacency Matrix of Call Graph of Moria City in Month 2

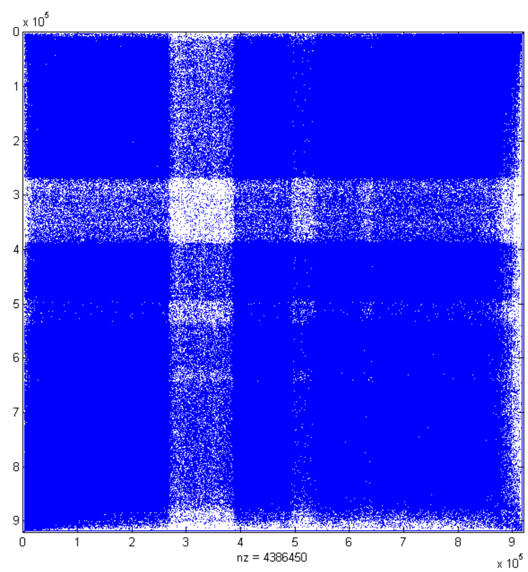

Fig. 7: Rearranged Adjacency Matrix of Call Graph of Moria City in Month 1