Learning Social Circles in Networks

1. Introduction

Automating the creation of social circles is a unique and challenging task. Predicting social circles in networks is highly relevant and intriguing because of the current rising trend in social media usage. In a world where over one billion people have a social network account, automatically creating social circles on social sites would help site owners or advertisement agencies with providing targeted content and suggesting new, related material to specific user demographics, which benefits both the user and the operator of the website. It would also assist users with controlling and sharing information with relevant friends. Overall, these circles improve connectivity and organization of friends in the networks.

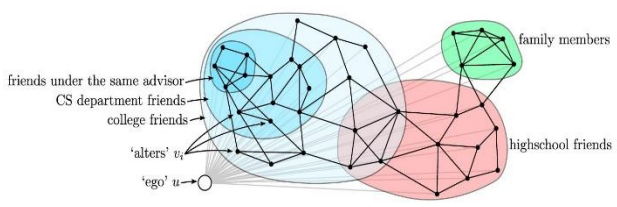
We try to predict social circles using various clustering algorithms – a modified version of K-Means clustering algorithm, Markov clustering algorithm and Infomap clustering algorithm. Using the evaluation provided by Kaggle, we conclude that topology information is more important than user's profile information for predicting social circles. We also conclude that predicting overlapping social circles is still an open problem.

The remainder of the report is organized as follows: Section 2 describes precise problem definition and the algorithms we tried, Section 3 provides evaluation and results, Section 4 and 5 talk about related and future work and Section 6 is the conclusion.

2. Problem Definition and Methods

2.1 Task Definition

Our goal is to automatically infer users' social circles. Social circles are groupings of a user's friends into categories such as "Cornell college friends", "Machine Learning friends", etc. The following diagram displays a representation of social circles: [1]



Given a Kaggle-provided list of the user's friends, anonymized Facebook profiles of each of those friends, and a network of connections between those friends, the main task is to find disjoint, overlapping, and hierarchical circles representing social circles within a person's social network. There are many interesting questions to consider: Which features will be most important in deciding these circles? Is connectivity of friends more important than shared features between friends in deciding which circles these users belong to? What feature set can be used from profile and connections to efficiently create these circles?

2.2 Algorithms and Methods

We chose to use three algorithms as part of our approach: a modified K-means clustering algorithm, a Markov clustering algorithm and an InfoMap clustering algorithm.

2.2.1 Modified K-Means

The K-Means clustering algorithm tries to partition data points into K different clusters so that the total distance between cluster members minimizes. We made two modifications to the standard K-Means algorithm to make it suitable for the clustering problem at hand:

- We maximize the similarity score between cluster members instead of minimizing Euclidian distance.
- Instead of choosing the mean of the cluster as the centroid, we choose a friend from the cluster which maximizes the similarity score between cluster members as the centroid.

In other words, our modified K-Means algorithm tries to solve the following objective function:

$$\underset{c}{argmax} \sum_{i=1}^{k} \sum_{x \in c_i} Similarity(x, x')$$

where c_i is the set of friends belonging to cluster i and x' is the friend within the cluster that maximizes similarity between cluster members.

We calculate similarities based on following three approaches: similarity based on common attributes, similarity based on topology as well as common

attributes, and similarity based on weighted attributes and topology.

- The similarity function using only common attributes simply counted the number of profile attributes two people had in common:

$$Similarity(x_i, x_j) = \sum_{k=1}^{l} \Delta(a_{i,k}, a_{j,k})$$

where $a_{i,k}$ denotes the value of k^{th} attribute of person x_i and Δ is defined as:

$$\Delta(a_i, a_j) = \begin{cases} 1 & if\ a_i = a_j \\ 0 & otherwise \end{cases}$$

- The similarity function that used topology as well as common attributes was similar to the previous similarity function except that it added an extra bonus of 5 if the two people were friends in the original network:

$$Similarity(x_i, x_j) = \sum_{k=1}^{l} \Delta(a_{i,k}, a_{j,k}) + F(x_i, x_j)$$

where

$$F(x_i, x_j) = \begin{cases} 5 & if\ x_i and\ x_j\ are\ friends \\ 0 & otherwise \end{cases}$$

- The similarity function using weighted attributes was similar to previous function, but instead of assigning equal weightage to all attributes, it gave different weightage to different attributes in similarity calculation:

$$Sim(x_i, x_j) = \sum_{k=1}^{l} w_i \cdot \Delta(a_{i,k}, a_{j,k}) + F(x_i, x_j)$$

where w_i is the weight of attribute a_i. These weights can be found in supplemental material. The weights were chosen arbitrarily, based on our thinking of which attributes were more important for clustering social circles. For example, having the same birthday may not be as important as sharing an employer while deciding whether to put the two persons in the same social circle.

We also did some post pruning and removed those friends from a cluster who had their similarity score below than a particular threshold. Deciding the value of K is in general quite challenging. However, if the person has many friends, then there are more chances of having more circles. From our experiments we found that the following relation provided a good estimate for the number of social circles a person would have:

$$K = 2 + \frac{Number\ of\ Friends}{65}$$

With all of the above mentioned approaches, the best Kaggle rank that we could achieve was 122. Since variations of the K-Means clustering algorithm did not perform well, it gave us an intuition that profile information is probably not as important as topology for predicting social circles. So we decided to try clustering algorithms that use topology information for generating clusters.

2.2.2 Markov clustering algorithm

The Markov clustering algorithm [8] is an unsupervised clustering algorithm for networks and is based on the fact that there will be more links within a cluster and less links across clusters. Therefore, if we start a random walk from a node, there will be more chances that we travel within a cluster than across clusters. Random walks on clusters are calculated using Markov chains. A Markov chain works on transition probability matrices where probabilities for the next step only depend on current probabilities. Hence, given an adjacency matrix, each step of the algorithm consists of multiplying the matrix by itself, raising each entry to a power (expansion), and then normalizing the columns (contraction). This process continues until convergence or until a given number of iterations has been reached. After this process, non-zero rows correspond to cluster.

Algorithm Markov Clustering

Input: An undirected graph, power parameter e and inflation parameter r.
Steps:
1. Create adjacency matrix representing graph
2. Normalize the matrix
3. Expand by taking the eth power of the matrix
4. Inflate by taking inflation of the resulting matrix with parameter r
5. Repeat steps 5 and 6 until convergence or given number of iterations

The theory behind the algorithm is that if one simulates many random walks over a network, the random walks will tend to stay within clusters. To aid this claim, the expansion step of the algorithm makes stronger edges strong and weaker edges weaker. After many iterations, distinct, non-overlapping clusters should remain.

In our approach, we only ran the Markov clustering algorithm using an adjacency matrix corresponding to the original topology (0 if an edge was present, 1 otherwise) of the social network. The reason was that we could not come up with a way to account for

profile similarity that survived the expansion and contraction steps of the algorithm.

2.2.3 InfoMap Clustering Algorithm

The InfoMap [7] is a multi-level community detection algorithm that attempts to minimize the hierarchical map equation over all possible hierarchical partitions of a network. There are three main equations for the InfoMap algorithm:

$$L(M) = q \curvearrowleft H(Q) + \sum_{i=1}^{m} L(M^i)$$

$$L(M^i) = q^i \circlearrowright H(Q^i) + \sum_{j=1}^{m^i} L(M^{ij})$$

$$L(M^{ij\ldots k}) = p^{ij\ldots k} H(P^{ij\ldots k})$$

The first equation is the general hierarchical map equation, the second refers to the description length of a module's partitions, and the third equation is the base case for the final modular level.

Essentially, the hierarchical map equation sums the frequency-weighted average length of code words ($H(Q^i)$) used to encode a random walker in the network and the recursive application of this equation to a network's hierarchical partitions, $L(M^i)$. q is the rate of the code word used for entering or exiting a module, $p^{ij..k}$ is the rate of the code word for visiting submodules or exiting to the higher hierarchical level, and $H(P^{ij..k})$ is the frequency weighted average length of the code words in the submodule codebook.

The algorithm starts by assigning each person to its own cluster. Then each person is moved to the cluster that results in the largest decrease of the above map equations. If no move can decrease map equations, the user remains in its original cluster. This process is repeated until convergence.

The weighted adjacency matrix used for the InfoMap calculations was formed by initially taking the original topology of the network (represented as 0 if there was no edge and 1 otherwise) and then adding the fraction of attributes two nodes in the network have in common, if more than 5 attributes were shared.

3. Experimental Evaluation

3.1 Methodology

The Kaggle-provided data set was comprised of 110 users and their corresponding social networks. 60 of the social networks were for training and were accompanied by the actual circles the owner of the network created. The other 50 social networks were for prediction in order to evaluate any algorithms in the competition.

To evaluate the correctness of our predictions, Kaggle computed the edit distance between our proposed solution and the correct solution. Each of the following operations cost one unit:

- Adding a user to an existing circle
- Creating a circle with one user
- Removing a user from a circle
- Deleting a circle with one user

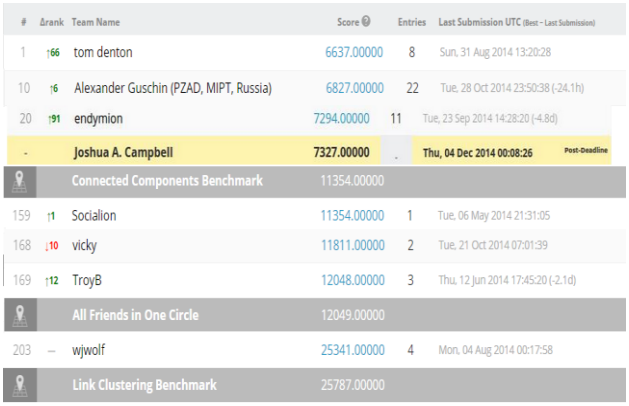
The minimum number of these operations required to transform our solution into the correct solution is the edit distance, which we are supposed to minimize.

3.2 Results

The following table shows how the algorithms we selected placed in the official Kaggle rankings (Only the best-ranking implementations for each group are shown):

Algorithm	Kaggle Rank
Infomap Clustering	21
Markov Clustering	91
K-Means Clustering	122

The following diagram shows how our best algorithm (InfoMap) placed in the competition. Also shown are various Kaggle-provided benchmarks.



3.3 Discussion

- K-Means: We observed that K-Means and its variations did not perform well. One of the reasons for modified K-Means failure is the difficulty to predict the number of circles for each friend (i.e. value of k). Moreover, it seems

that the topology information is more important than profile information for predicting social circles. Since K-Means highly relies on profile information for predicting circles, it fails to produce good results.

- MCL: Markov clustering does not incorporate attributes and just uses topology information. It outperforms K-means because it dynamically determines the number of clusters by simulating random walks and grouping the clusters based on probability of random walks.
- Infomap: InfoMap clustering algorithm performed the best because it was able to dynamically determine the number of clusters using both the topology of the network and the attributes of profiles. By using the notion of entropy, the InfoMap algorithm reflected the idea that people are using social circles to organize friends in logical group, which is the effect of minimizing entropy.

Note that the difference between 1st and 21st place is about 700 points. Given that there were 50 social networks in the prediction set, this means on average each of our predictions is only an edit distance of 14 away from the 1st place solutions. One conclusion we can draw from this is that none of the submissions is actually accounting for overlapping or containing circles since we were able to get so close without accounting for them either.

4. Related Work

There are not many works in this area. The closest methods are those that find clusters in a network, such as [3], [4], [5], [7], and [8]. However, none of these methods account for overlapping circles, which is an anomaly found in real-life social networks. Recently some topic modeling techniques have been used to uncover overlapping clusters [13].

5. Future Work

The biggest shortcoming of all the methods presented is that none of them support overlapping or containing social circles. As shown previously, even one of the simplest training social circles contained users that were in multiple circles. Future work includes determining overlapping circles as well as determining a metric to conclude when hierarchical circles should be included.

There are various metrics that could be tested for overlapping circles. There are a few sources we considered ([9],[10],[11]) to incorporate into our current method. However, all of these methods are relatively new and untested in this setting. A simple heuristic that could easily be incorporated is searching for pairs of similar users that are in separate circles and including one of them into the other's circle. Another simple heuristic is randomly merging friends into clusters and applying the InfoMap equations to minimize or prune the additional entropy added into the system.

To handle hierarchical circles, we could run InfoMap recursively on the final clusters we find. We could experimentally find some threshold of when we should include containing clusters. Based on analyses of the training data, containing clusters are relatively few compared to the occurrences of overlapping clusters.

6. Conclusion

The goal of this research was to reliably predict social circles. Our results show that we are able to compete with the best algorithms submitted to Kaggle without taking overlapping social circles into account. We also provide the algorithm for constructing the adjacency matrix necessary for the calculations. Based on the results from Kaggle, we concluded that topology information is more important than profile attributes in deciding social circles. We also conclude that the InfoMap algorithm is a very good approximation to the problem because it mimics the idea that users are using social circles to organize their friends, which results in a decrease in entropy in the resulting network topology. Although not commercial ready, our results are a promising next step towards full automation of creating social circles on networks.

7. References

[1] Learning to Discover Social Circles in Ego Networks http://i.stanford.edu/~julian/pdfs/nips2012.pdf

[2] Learning Social Circles in Networks https://www.kaggle.com/c/learning-social-circles

[3] K-Means Clustering Algorithms http://en.wikipedia.org/wiki/K-means_clustering

[4] Single-link and Complete-link Clustering http://nlp.stanford.edu/IR-book/html/htmledition/hierarchical-agglomerative-clustering-1.html

[5] A Tutorial on Clustering Algorithms. http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/index.html

[6] scikit-learn Clustering library in Python. http://scikit-learn.org/stable/modules/clustering.html#clustering

[7] Rosvall, Martin, and Carl T. Bergstrom. "Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems." *PloS one* 6.4 (2011): e18209.

[8] Van Dongen, Stijn Marinus. "Graph clustering by flow simulation." (2000)

[9] Palla, Gergely, et al. "Uncovering the overlapping community structure of complex networks in nature and society." *Nature* 435.7043 (2005): 814-818.

[10] Banerjee, Arindam, et al. "Model-based overlapping clustering." Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. ACM, 2005.

[11] Cleuziou, Guillaume, Lionel Martin, and Christel Vrain. "Poboc: an overlapping clustering algorithm." *Application to rule-based classification and textual data*(2004): 440-444.

[12] Markov Clustering Algorithm: https://www.cs.ucsb.edu/~xyan/classes/CS595D-2009winter/MCL_Presentation2.pdf

[13] E. Airoldi, D. Blei, S. Fienberg, and E. Xing. Mixed membership stochastic blockmodels. JMLR, 2008