The Twitter Rumor Network: Subject and Sentiment Cascades in a Massive Online Social Network

Sonal Gupta, Benjamin Slawski, Doris Xin, Wenqi Yao
Mentor: Adam Wierman
California Institute of Technology

Tendencies of individuals to behave like those around them leads to cascading phenomenon, in which an idea or behavior spreads quickly throughout a social network, being adopted by nearly all individuals in an area. We crawl the Twitter social graph and monitor users' posts, or 'tweets,' for several weeks, monitoring the spread of keywords, or 'hashtags,' along the graph structure. We simulate cascades on the Twitter graph using previous models and compare the results to the real cascade data. Additionally, we perform sentiment analysis on the tweets, determining whether a user has a positive or negative position on a hashtag. Finally, we isolate clusters in the network, examining the variance in sentiment within a cluster, observing the distribution of group sentiment divisions.

Introduction

As social creatures, human beings base their decisions not only on their personal goals and motives, but also largely take into account the decisions of those around them. The tendency of individuals to base their decisions on the decisions of others in a group leads to a cascading phenomenon [Bikchandani 1998], in which an idea or preference spreads more quickly based on the number of people who have been exposed to the idea, known as active nodes. A cascade occurs when nearly all of the nodes in a particular area become active in a relatively brief period of time. Cascading behavior is observed in things such as politics, economics, and even crime [Golub 2010; Moretti 2011]. This has led to much investigation into the behavior and modeling of information cascades. Most of this investigation has limited itself to simulated cascades in real networks [Kempe 2003], simulated cascades in random networks [Payne 2009], or laboratory testing of cascade behavior in experimental setups [Alevy 2005], as large social networks are difficult to observe in real-time.

This cascading behavior often occurs in various social networks — the graph of relationships and interactions within a group of individuals. The importance of social network as a medium for the spread of information, ideas, and influence among its members has increased drastically in the past few years. Thus, to understand the extent to which ideas and preferences are adopted in a social network, it is important to understand how the dynamics of adoption are likely to unfold within the underlying social network: the extent to which people are likely to be affected by decisions of their friends and colleagues, or the extent to which “word-of-mouth” effects takes hold. Recent popularity in large internet social networks has intensified interest in understanding cascades [Gonzalez-Bailon 2010]. Cascades in these systems often occur quickly, due to instant notifications of the changing states of others and the high connectivity of the graph. These social networks are interesting in the sense that they provide large, active data sets with information both about the structure of the social graph and the states of individuals in the network in real time.

We use data mined from the Twitter social network, specifically 'hashtags,' which are keywords identified in a users post, or 'tweet.' These hashtags allow us to easily track the spread of a subject across the network. We then use this data to test the accuracy of previous cascade models [Kempe 2003] and better understand cascading effects over large, real-world networks.

Additionally, through various natural language processing techniques and machine learning algorithms, we perform sentiment analysis on the tweets [Go 2009; Chen 2010], allowing us to label a tweet as positive or negative in regard to a hashtag. By identifying and isolating clusters in the network, we analyze the tendency for members of a group to conform to a common opinion by finding the percentage of the cluster with common sentiment towards a hashtag.

Twitter serves as an ideal social network for this analysis because unlike other social networks, Twitter is expressly devoted to disseminating information in that users “follow” other users to subscribe to broadcasts from other users. The 140 character limit to the tweet lengths simplifies the analization of cascading effects for hashtags and the performance sentiment analysis.

Methods

Data Gathering

We generate two large data sets for this project using Python. The first set is the the Twitter structure set. Beginning with a list of all users for which we gathered tweets, we then scan through a data file of the entire Twitter graph, stored as an edge list, writing only edges connecting two monitored users to the output file. The resulting data file represents the structure of only the subset of users which we monitored, containing 10 467 nodes and 487 948 directed edges.

The second large data set is the tweet data. To produce this data, we monitored RSS feeds of about four million Twitter users, organizing the tweets by both user and hashtag in seperate tables of a mySQL database, storing user ID, tweet content, tweet hashtags, and the post time in seconds since epoch. Figure 1 illustrates the entity-relationship model used for the database. The user IDs to monitor are gathered from the structure data. A daemon that iteratively visited each user’s RSS feed was run for 6 weeks to populate the database with tweets. Twitter reports that the average user tweets around five times per day, so we used a database for the tweet data instead of text files to handle the massive size of this data set.

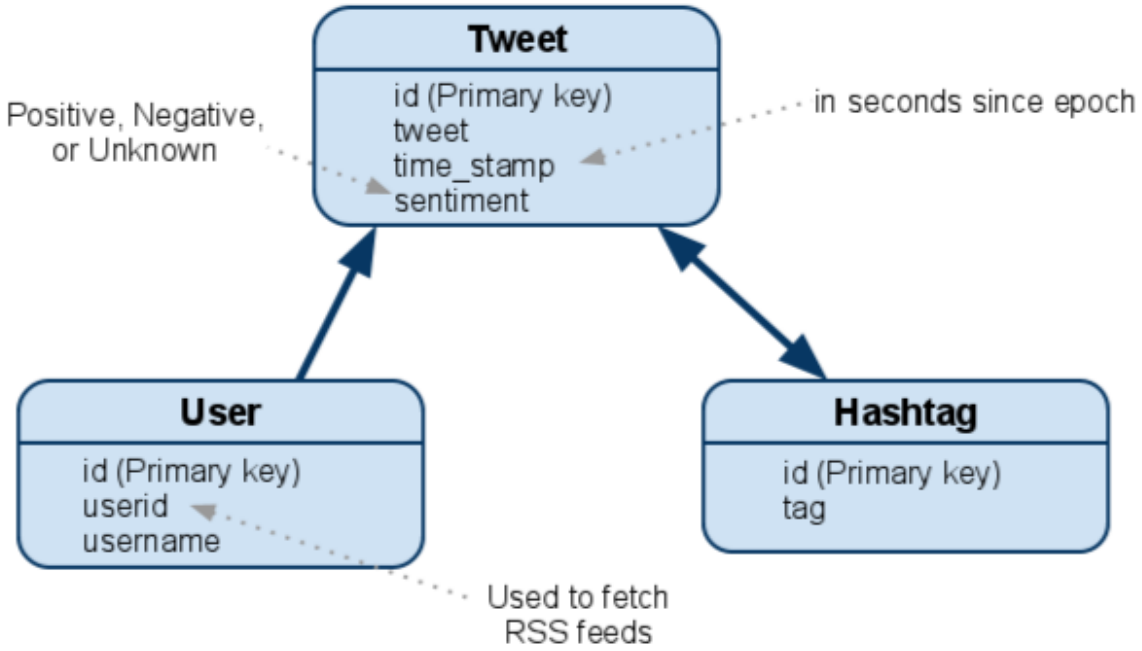

Figure 1. The entity relationship model for the database

For sentiment analysis, we used the training set as collected by Go et al 2009; a collection of 1.6 million tweets, each labeled as either positive or negative in sentiment. These tweets have been classified on the basis of emoticons that appear in the tweet - happy faces are labeled as positive, and sad faces as negative. The labels obtained in this fashion were somewhat noisy, but can provide a relatively high degree of accuracy. In the training set, tweets have been stripped of their emoticons.

Simulation and Cascade Tracking

We constructed a cascade simulator using Python capable of simulating two cascade models. It begins by reading a structure data file and creating a set of nodes, each corresponding to a user ID. Follower lists are stored in nodes. At each time step, all active nodes are iterated over, and their weighted contributions passed to follower nodes based on the cascade model. Each node sums the input passed to it by nodes it is following, and decides its state based on the model. The simulator outputs a list of user IDs paired with the timesteps in which they became active, stored as an ASCII text file. The simulator was implemented in Python.

We also constructed a data harvester capable of reading cascades in the tweet database. This harvester searches for a particular hashtag, returning a list of all tweets containing this hashtag in the database. The program then scans this list, noting the earliest appearance of the hashtag from each user, rounding to the minute. Then, these times are converted to minutes since the first appearance of the hasgtag, and can now

be considered as timesteps. These user, active timestep pairs are written to a data file in the same style as the simulator data.

Plotter

We used python with Matlab (matplotlib and numpy) to generate line plots displaying cascade sizes over time for real and simulated cascades, and histograms displaying the distribution of cluster sentiment divisions. Further development could generate a spatial network plot which shows the real-time cascading over a network structure. Plots with different cluster sizes for different hastags or a set of hashtags can be obtained.

Models

We used two models in our cascade simulation [Kempe 2003].

1. Linear Threshold

Consider a network with nodes 1,...,n, s.t. node i has friends Fi, and size(Fi) = d (i). Let each node be in state 0 or 1, with most nodes starting in state 0. At the beginning of the simulation, each node i is given some threshold q (i) in [0, 1], chosen uniformly at random. Define the weight of the connection from node j to node i to be 1/d(i). If the weighted sum of the states of all connected nodes exceeds node i's threshold value q(i), then the node will change to the 1 state. Similarly, if the weighted sum is below q(i), then the node is in state 0. If there is no change between successive generations, the cascade is complete.

2. Independent Cascade

Consider a network with nodes 1,...,n, s.t. node i has friends Fi, and size(Fi) = d (i). At the beginning of the simulation, each node is given some small activation probability p(i) > 0. Let each node be in state 0 or 1, with most nodes starting in state 0. When a node is activated it is considered newly activated,(for the first generation, all active nodes are assigned as newly activated), and only during the next generation will possibly activate others. The probability of node j activating node i is p(i). When there are no newly activated nodes left, the cascade is complete.

Sentiment Analysis

Feature Extraction and Reduction

In the unigram model, each distinct word in a tweet is treated as a unique feature. A “word” in this context refers to a continuous string of alphanumerical characters not containing a space, with contractions such as “I’m” or “we’ve” counting as one word. For example, “I’m a computer scientist” contains the unigrams “I’m”, “a”, “computer”, and “scientist”. In the n-gram (for n >= 2) model, features are extracted from a tweet with a sliding window containing n words. E.g. the example sentence above has the bigrams “I’m a”, “a computer”, “computer scientist”.

In this paper, we considered each feature to be binary--a feature is either present or absent in a tweet--instead of counting the frequency of a feature in a tweet. Pang et al have obtained better results by using a term presence rather than its frequency [Pang et al., 2002].

A large challenge in sentiment analysis is feature reduction, and this problem is heightened since Twitter is a casual platform where users may not use standard English. Along with the large number of legitimate words, there exist tokens in the text that are not necessarily distinct. For example, words with many repeated letters (e.g. “yaaaaaay” vs. “yay”) and misspellings of other words (e.g. “great” vs graet”) can result in unnecessary features being checked.

We provided rudimentary feature reduction in replacing hyperlinks with the word “URL” and user names (prefixed by “@”) with “@”, as well as removing any words of length 2 or less from the list of words. Most punctuations are removed from the tweet with the exception of apostrophes in contractions as mentioned above. Hashtags prefixed by “#” are demoted to regular work status by removing the “#” prefix. This brings the number of features down to less than 50% of the original number [Go et al, 2009]. Stopwords such as “the” and “all” were also filtered out, as they should not impact the sentiment of a tweet [Chen et al, 2010].

We introduced certain bigrams that could change the sentiment of a sentence; namely, “no/not _”, “more _”, “less _”. Each of these bigrams were considered as a single feature.

Training Algorithm

We trained 3 algorithms - Naive Bayes, Maximum Entropy and SVM - on the data, and the most successful (SVM) was used to classify the tweets in the database. With each algorithm, the 15000 most common features (unigrams) in the data set was first found, and their occurrence in 80% of the tweets was used to determine the impact of these words on the overall sentiment classification.

We used the Python NLTK toolkit for the Naive Bayes and Maximum Entropy classifiers, and SVM-Light for the SVM classifier.

Naive Bayes

The Naive Bayes classifier finds the probability for a label given the features present using Bayes' rule, then makes the 'naive' assumption that all features are independent, in order to reduce the dimensionality of the data. The calculation hence takes the form of:

P(label | features) = (P(label) * P(f1|label) * … * P(fn|label)) / P(features)

Since P(features) is invariant across labels, the denominator is calculated for each label, and normalized to sum to one.

Clustering Behavior

In a network, clusters are defined as subsets of nodes that are more tightly connected to each other than to the rest of the graph at large. It is intuitive that cascades would happen more rapidly in clusters due to increased influence, however, clusters tend to prevent large cascades, as information is less likely to spread between clusters [Payne 2009]

We used a variation on the netclust clustering algorithm, along with a similarity measurement, in order to isolate clusters. The similarity measurement is based on mutal friendships [Huang 2010]. For users A and B, let I be the intersection of the followers of A and B. The similarity between A and B is given as the the fraction of the followers of A that are in I multiplied by the fraction of the followers of B that are in I. In this way, not only mutual followers are considered, but follower importance as well. Given some cutoff value, netclust then scans the structure, eliminating edges with similarities under the cutoff, then using the remaining edges to find spanning trees uniting the nodes. In this way, we isolate clusters in the graph structure.

Once we isolated these clusters, we observed the distribution of sentiments about various hashtags, grouping the clusters by size. We used these ratios to create a histogram of percent positive sentiments over many clusters and hashtags. This plot demonstrates the tendencies of individuals in a cluster to carry similar sentiments as others in the cluster.

Results

We found hashtag cascades of various sizes in our tweet data, and by altering threshold and probability parameters, were able to produce similarly sized simulated cascades using both the linear threshold and independent cascade models (Figure 2).

Using clusters isolated by MCL, we were able to analyze the tendencies of individuals in a group to have similar sentiments to others in the group. For several clusters and hashtags, the percentage of cluster members that had a positive sentiment was calculated. The clusters were grouped by size, producing a distribution of cluster sentiments (Figure 3).

We also found that the cluster sentiment analysis generally gave us a 'W' graph which basically indicates that the clusters were either mostly positive, mostly negative or else were equally negative and positive. The graphs were generally positively skewed which implies that people generally tend to be more positive.

Discussion

Using the vast data set of Twitter, we have shown the behavior of cascades in large, real-world social networks. Additionally, we used sentiment analysis to study not only subject cascades, but cascades of opinion on those subjects. Our models, previously studied on co-authorship networks [Kempe 2003], produced similar results the original studies in the sense that they produced convex arcs on a log plot in small scale. However, in the total cascade, we see a plateau in the middle of the cascade. By fitting these models to actual cascade data, we see plateaus present in the real data plots, even in small scale. We were unable, however, to observe real cascades of any greater than a few hundred users, so it is unclear whether

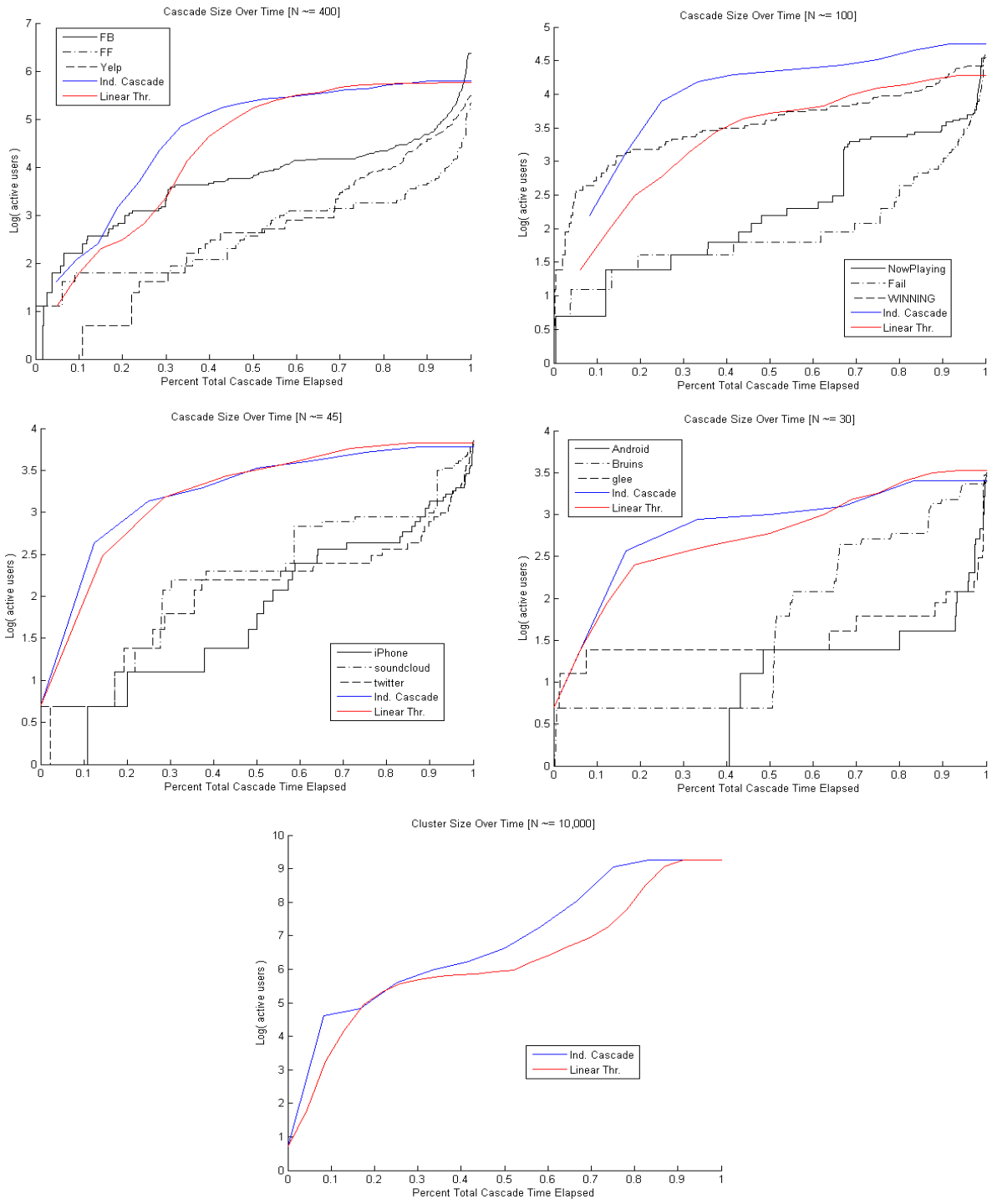

Figure 2. Cascades size over time for cascades of approximately 400, 100, 45, 30, and 10 000. Colored lines represent simulated cascades, black lines represent cascades observed in the real data. No 10 000 node cascades were observed in the real data, so only the simulation results are plotted for this size

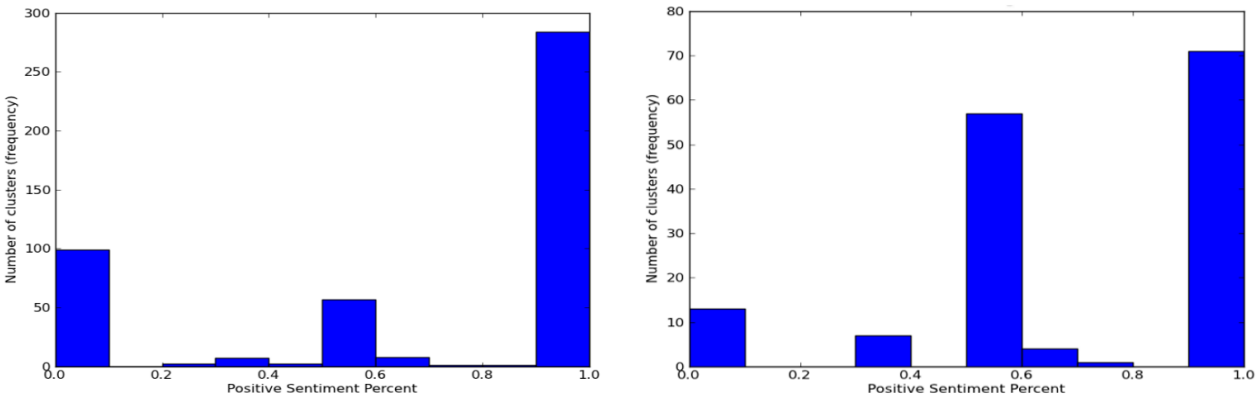

Figure 3. Distribution of group sentiment for small (left) and large (right) clusters. The distribution takes a W shape.

this plateau is present in the real data at large scale as well.

We also used Netclust to isolate clusters in the Twitter graph. By isolating a large number of clusters and observing them over several hashtag cascades, we produced a distribution of percent positive sentiments in a cluster for different sized clusters, and found that the distribution takes a W shape, with peaks at zero, fifty, and one hundred percent. Previous work by
Friedkin et al. 2010 suggests such a structure, where the middle peak results from Reasoned Actions, and the sides from Attitude and Behavioral links in the social network. This demonstrates that groups tend to either agree or be evenly divided on a subject. Small, dissenting subgroups within a group are rarely observed. Additionally, it was observed, though not shown here, that the relative size of the peaks varied based on hashtag, with some having larger middle peaks and others having only side peaks. All hashtags, however, demonstrated a bias towards the positive end.

In conclusion, by studying cascade behavior in large networks such as Twitter, we can better understand the trending behavior of humans. Being able to understand cascades of opinion, as well as subject, can have important ramifications for areas such as politics and advertising. Future study on other social networks, both online and real world, is necessary to determine how accurately Twitter cascades reflect cascade behavior in other networks. Longer, more comprehensive study of Twitter and other networks needs to be done in order to observe the cascade behavior of super-massive cascades, which encompass a large percentage of the Twitter graph and were not seen in out study. Also, the sentiment distribution of various categories of hashtags may differ greatly, but were not examined here. Further examination of such cascading behavior helps us not only to predict it in future events, but helps us better understand the interconnected nature of human society.

References

Alevy, J., Haigh, M., List, J., 2005. Information Cascades: Evidence from a Field Experiment with Financial Market Professionals. Journal of Finance 62: 151-180.

Bikchandani, S., Hirshleifer, D., Welch, I., 1998. Learning from the Behavior of Others: Conformity, Fads, and Informational Cascades. Journal of Economic Perspectives 12: 151-70.

Chen, M., Lin, H., Shih, C., Hsu, Y., Hsu, P., Hsieh, S., 2010. Classifying mood in plurks. Proceedings of the 22nd Conference on Computational Linguistics and Speech Processing: 172-183.

Domingos, P., Richardson, M., 2002. Mining the Network Value of Customers. In Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining: 57-66. ACM Press.

Friedkin, N., 2010. The Attitude-Behavior Linkage in Behavioral Cascades. Social Psychology Quarterly 73: 196-213.

Go, A., Bhayani, R., Huang, L., 2009. Twitter sentiment classification using distant supervision.

Golub, B., Jackson, M., 2010. Naive Learning in Social Networks and the Wisdom of Crowds. American Economic Journal: Microeconomics 2, 112-149.

Gonzalez-Bailon, S., Kaltenbrunner, A., Banchs, R., 2010. The Structure of Political Discussion Netowrks: A Model for the Analysis of Online Deliberation. Journal of Information Technology 25, 230-243.

Heal, G., Kunreuther, H., 2010. Social Reinforcement: Cascades, Entrapment, and Tipping. American Economic Journal: Microeconomics 2, 86-99.

Huang, J., Sun, H., Huan, J., Deng, H., Sun, Y., Liu, Y., 2010. SHRINK: A Structural Clustering Algorithm for Detecting Hierarchical Communities in Networks. Proceedings of the 19th ACM International Conference on Information and Knowledge Management. ACM Press.

Kempe, D., Kleinberg, J., Tardos, E., 2003. Maximizing the Spread of Influence through a Social Network. In Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining: 137-146. ACM Press.

Malcom, G., 2000. The Tipping Point: How Little Things Can Make a Big Difference. Little, Brown and Company, New York.

Mei, C., Jiang, H., Jenness, J., 2009. Markov Clustering (MCL) based Thread Grouping and Thread Selection. Proceedings of the IEEE SOUTHEASTCON 2009: 404-409.

Moretti, E., 2011. Social Learning and Peer Effects in Consumption: Evidence from Movie Sales. Review of Economic Studies 78, 356-393.
Watts, D., 2004. The "New" Science of Networks. Annual Review of Sociology 30, 243-70.
Pang, B., Lee, L., Vaithyanathan, S., 2002. Thumbs up> sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP),* pages 79-86.
Zuckerman, A.S., 2005. The Social Logic of Politics: Personal networks as contexts for political behavior. Temple University Press., Philadelphia.