Data-driven Outbreak Detection in Social Networks
Group No: 26

Jiayuan Ma
jiayuanm@stanford.edu

Xincheng Zhang
xinchen2@stanford.edu

Pearl Tan
pearltan@stanford.edu

ABSTRACT

In social networks, influence such as information, virus, innovations spreads from node to node through the edges of the networks. Understanding this diffusion process is the key to many important real-world applications, for example marketing strategies, virus and pollution control. The problem of outbreak detection and influence maximization ask about selecting a set of "important" nodes (sensor locations, influential bloggers) in a network, from which influence may outbreak or we can detect influence outbursts. These two problems can be modeled as selecting nodes in order to maximize some measures defined over all the sets of nodes. These measures are usually chosen in terms of the number of influenced nodes, the time passed from outbreak until detection and etc.

Most traditional studies [6][8] focus on how to maximize the defined measures effectively and efficiently. Research efforts have been made to cast the original problem into the submodular function maximization where greedy hill climbing algorithms can deliver 63% approximation of the optimal.

However, what the previous research failed to capture is that in real-world scenarios multiple influences spread through the network simultaneously, and these spread of influences tend to correlate with content of information being spread. Furthermore, nodes may have specific preferences towards specific contents, so that nodes that are chosen to be the "outbreakers" for one type of information cascades may not be the "outbreakers" for information cascades of another content type.

This project addresses how to incorporate content into outbreak detection, which we refer to "data-driven outbreak detection". We propose an adapted model that allows for modeling outbreak detection with content information, and evaluate our model on real world collaboration network and Twitter network. Moreover, we will use the proposed model and machine learning techniques to train classifiers that predict whether a specific piece of content information will cause an outbreak in the network. Finally, we discuss the pros and cons of incorporating content information into the analysis of information outbreak.

1. INTRODUCTION

This is the information age. Everyday everyone in the world is receiving tons of information on an unparalleled scale, and no one could afford to consume and accept all the incoming information. However, every one wants to be kept most up to date, in order not to be left behind in this race on the information highway. One simple way to resolve this dilemma is to keep one's eyes on some trusted influential nodes in the information diffusion networks, so that one can keep himself/herself updated with the information from these sources. The general problem of detecting outbreaks in networks asks how to select a set of nodes to detect some dynamic diffusion/spreading process over a network.

Many real-world problems can be modeled under this setting. For example, a graduate student in Physics has just started doing research on high energy physics theory. Since there are too many papers on various aspects of this area, one can never read all of them in a short period of time. If the student aims at having a rough understanding of the entire community, the first step he/she would like to take is to survey the area by reading some influential and fundamental papers. Therefore, selecting papers to read is naturally an instance of outbreak detection problem on the paper citation network.

In the domain of weblogs (such as Twitter), bloggers publish (tweet) posts and use hyper-links (retweet) to refer other bloggers' posts and contents on the web. Under this circumstances, we want to select a set of blogs to read so that we can keep ourselves updated with most of the stories (whether it can be linked or direct posts) that propagate over the blogger's network. This is also an example of a outbreak detection.

Previous researches focuses on modeling the information cascades directly on the network structure (see section 1.1), such that the higher-degree nodes are more likely to participate in a cascade, and the lower-degree nodes make less contribution to the cascade. However, in reality, content also matters, the likelihood of nodes participating in a cascade is relevant to the type of information they receives. In the blogger's example, an individual blogger may be a professional sports fan who only cares about sports stories/news. In this case, he/she may prefer to reading and spreading (re-tweeting) sports news. And these people are only likely to participate in sports-related information cascades, and if these information cascades finally become an outbreak we just need to consider these nodes.

This project presents data-driven outbreak detection, which studies how to incorporate content-related information into normal outbreak detection. We propose an adapted model that allows for modeling outbreak detection with content information, and evaluate our model on real world collaboration network and Twitter network. Furthermore, because we take into consideration the content information flowing on the network, we can use some standardized machine learning techniques to predict if a given specific piece of content information will cause an information outbreak. We will also show some quantitative results on this outbreak prediction task as well.

This paper is organized as follows. In section 1.1, we give the definitions of basic diffusion models where the rest of paper depends upon. In section 2, we review the previous relevant research. In

section 3, we give the formulation of our problems. We provide our approach to solving the problem in section 4. We report our empirical results in section 5 and wrap up with section 6 on what we plan to do in future.

1.1 Basic Diffusion Models

In Kempe et al. [6]'s highly-cited paper, two basic information diffusion models have been generalized: Independent Cascades (IC) model and Linear Threshold (LT) model. Both of them are the most widely-studied diffusion models.

Given a graph $G = (V, E, w)$, where V is a set of n nodes, $E \subseteq V \times V$ is a set of m directed edges, and $w : V \times V \longrightarrow [0, 1]$ is a weight function such that $w_{u,v} = 0$ if and only if $(u, v) \notin E$. We start the diffusion process with an initial set of active nodes S_0, and the process cascades in discrete steps $i = 0, 1, 2, \ldots$. Let S_i denote the set of vertices activated at step i. The whole process stops at t when $S_t = \phi$. IC and LT only differ in how every individual node is activated, and we will explain their differences.

1. **Independent Cascades** (IC): If a node u first become active in step i, it is given a *single* chance to activate each of its inactive neighbor v, with the probability of success being $w_{u,v}$. Each successfully-activated node v will become active in step $i + 1$. Notice that this process is *unrepeatable*: we cannot make any further attempts on the same edge.

2. **Linear Threshold** (LT): An LT influence graph further assumes that $\sum_{v \in V} w_{u,v} \leq 1$ for every u. The dynamics of LT proceed as follows.

   Each node u has a threshold θ_u which is uniformly distributed in the interval $[0, 1]$, which models the uncertainty of individual's conversion threshold.

   The node u is activated when the weighted sum of its activated neighbors v is no less than the threshold θ_u. Mathematically, node u will become active if

$$\sum_{v \in \cup_{0 \leq j \leq i-1} S_j} w_{u,v} \geq \theta_u \tag{1}$$

Both the two models have, to some extent, simulated the way how information spreads. The IC model treats the spread of information independently and probabilistically, where information spreads through edges independently and with a given probability. The LT model focuses more on collaborative and threshold side of influence propagation, where sufficient number of neighboring influence will help spread the information. Notably, both of these two models are progressive, which means that nodes can switch from being inactive to being active, but do not switch in the other direction. In our project, we also assume that the information spread is progressive.

2. PRIOR WORK

Kempe et al. [6] generalized the IC and the LT model from the previous research, and formulated influence maximization into a NP-hard optimization problem. The authors further utilized an analysis framework based on submodular functions, and proposed an efficient greedy algorithm that approximates the optimal solution with a provable bound. Leskovec et al. [8] studied the problem of outbreak detection, generalized influence maximization into one form of outbreak detection under a specific measure and demonstrated that objective functions in outbreak detection are submodular functions, The authors combined the original and the normalized greedy algorithm to get an algorithm which effectively approximates the optimal solution. Furthermore, the authors proposed the CELF algorithm, which exploits submodularity to speed up the computation by 700 times. In our project, we give an adapted version of CELF and use it for scalable outbreak detection on real-world networks. Chen et al. [2] focused on improving the time efficiency of [6] and [8] in very large networks. Their contributions are two-fold: (i) improvement of greedy algorithms in IC models by pre-generating the graph (ii) proposal of novel degree discount heuristics that is cheap to compute (in milliseconds) and outperforms traditional degree and centrality heuristics. One of the first influence maximization algorithms specifically designed for the LT model was proposed in Chen et al. [3]. The authors proved NP-hardness of IM under the LT model in general graphs, proposed a linear-time algorithm for DAGs, and applied the proposed linear-time algorithm to local DAG structures in general graphs. Goyal et al. [5] proposed a framework for modeling the LT model weight parameter using regression analysis on the information propagation time. Their experimental result shows this time variant model significantly outperforms previous models. Myers et al. [10]'s work on information contagion models interactions between multiple information contagions using statistical methods. The similarity between their work and our project is that we both group information content under high-level topics/clusters so that computation is more tractable and interactions modeled are meaningful. In particular, the authors of [10] employed an supervised approach for content modeling (e.g. human labeling and pre-assumed dictionary). Conversely, our project used unsupervised clustering approaches from the NLP research community which do not suggest any prior knowledge. Haewoon et al.'s paper [7] provided a thorough quantitative analysis on the Twitter blogsphere, and generously released the entire Twitter social graph data on their project page. In our project, we are using a part of data they provided.

3. PROBLEM FORMULATION

Given a social network structure $G = (V, E)$, we want to select a subset $\mathcal{A} \in \mathcal{P}(V)$ of nodes in the network such that $\mathcal{A}$ is solution for

$$\begin{aligned} \max_{\mathcal{A} \subseteq V} : \quad & R(\mathcal{A}) \\ \text{subject to} : \quad & c(\mathcal{A}) \leq B \end{aligned} \tag{2}$$

where R is a submodular set function defined on $R : \mathcal{P}(V) \longrightarrow \mathbb{R}$ according to some measure (in terms of the number of nodes affected or the reduction of time spent on detecting outbreaks) and $c(x)$ is a non-negative cost function defined on each vertex $c : V \longrightarrow \mathbb{R}$. $c(\mathcal{A})$ is further defined as $c(\mathcal{A}) = \sum_{s \in \mathcal{A}} c(s)$, and B is a budget we can spend for selecting the nodes. In this project, we use the number of seed nodes as the definition of $c(\cdot)$ such that $c(\mathcal{A}) = |\mathcal{A}|$.

The above is the original outbreak detection formulation, and we incorporate content information $\mathcal{C}$ into the formulation below. Given a social network structure $G = (V, E)$ where each node $v \in V$ is attached with content information $c_v \in \mathcal{C}$.[1] The probability of information will propagate from node u to node v through edge uv is given by

$$p_{u,v} = Sim(c_u, c_v) \tag{3}$$

where Sim is defined as a function $Sim : \mathcal{C} \times \mathcal{C} \longrightarrow [0, 1]$ which measures the similarity of content information c_u and c_v corresponding to node u and v, and normalizes into a valid probability

[1] In section 4, we will see that we characterize the content space $\mathcal{C}$ using n-dimensional vector space $\mathbb{R}^n$.

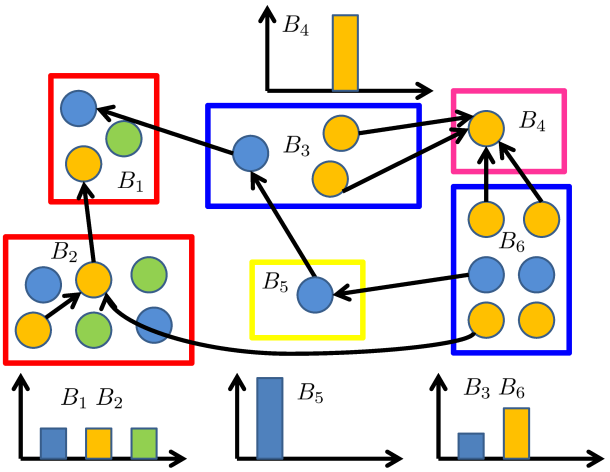

Figure 1: An illustration of blogger network. Each blogger is denoted by a rectangle, and each post tweeted/posted by bloggers are represented by colored circles. All the circles enclosed by the blogger rectangle are the tweets/posts published/referred by this blogger. The links between circles denotes the reference relationships. For example, A circle is pointed by B circle means that post A gets referenced by post B. The color information on each post circle denotes the type of content/topic the post belongs to. As we can observe from this diagram, post reference are more likely to happen between posts with the same content (circles with the same color in this case). The color of the bounding rectangles in this picture means the latent topic distribution for each individual bloggers. The topic distribution of each bloggers are also displayed in this diagram. Notice that in this example bloggers have the same topic distribution are more likely to exchange information with each other.

value. As we can see from the definition, the more similar contents c_u and c_v are, the more likely information will spread between u and v.

The problem boils down to finding a reasonable representation of c_u for each $u \in V$ quantitatively. In our project, we use the content history of a node for quantization. In the case of collaboration network, the content history of a node is all the papers the author node has published before. In the case of Twitter network, the content history of a node is all the posts the node has published/retweeted beforehand. The rationale of this formulation is based on the assumption that nodes that share a history are more likely to propagate similar types of information. An illustration is given in Figure 1.

Comparison to previous approaches: In the previous researches, the probability of $p_{u,v}$ is only relevant to the degree information of node u and v. For example, in independent cascade models, the whole graph $G = (V, E)$ is parameterized by a single uniform probability p to each edge of the graph (usually from 1% to 10%) in separate trials. In this case, if nodes u and v have $c_{u,v}$ parallel edges, u has a total probability of $1 - (1 - p)^{c_{u,v}}$ of activating v if the information reaches u. In linear threshold models, each edge from u to v is assigned probability as $p_{u,v} = 1/d_v$ where d_v is the degree of node v. Unlike these traditional approaches, we tend to bias the assignment of $p_{u,v}$ based on the content correlation of node u and v. In our project, $p_{u,v}$'s are no longer simply determined by the network structure, and they are related to the proximity in the content space.

4. MODEL AND ALGORITHM

In this section, we describe our approach of incorporating content information in the task of outbreak detection. Section 4.1 introduces two unsupervised approaches of modeling content space, which are used to aggregate information content into different high-level topics. Section 4.2 describes a speeded-up version of CELF, which is used in our project as the infrastructure of outbreak detection. Section 4.3 gives several approaches to define $Sim(\cdot)$ function in section 3, and in section 4.4 we describe our approach of predicting the outbreak of a given specific piece of information.

4.1 Content Modeling

In our project, we used two popular unsupervised techniques in natural language processing communities to model the property of information content that propagates over the network.

4.1.1 Latent Semantic Analysis

For the papers in collaboration network, we use Latent Semantic Analysis (LSA) [4] to carry out content modeling on the this network data. Each node in this network represents an author in the research community, and the content attached to this node is the paper information the author published in the past. To simplify the process, we only use the title of the published papers as the content information in our project. LSA is a very intuitive and powerful method of retrieving textual materials depend on a lexical match between words in users' requests and those in or assigned to database objects. It is assumed that there is some underlying "latent" semantic structure in word usage data that is partially obscured by the variability of word choice. Using statistical techniques, we can estimate this latent structure and get rid of the obscuring "noise" (e.g. diversities of word choices).

We filter out a list of stop words (like "a", "the", "of" etc.) from the titles of those papers and treat the rest words in the titles as terms. From above, we know we are just using the titles of those papers as the content information, and we will just build corpus from each document (title) and its terms. Before we actually using the Latent Semantic Analysis, we did the term frequency times inverse document frequency (*tf-idf*) weighting on corpus to lower the influence of common words appearing in every documents. The *term frequency* $tf(t, d)$ of a term t with regard to document d is simply defined as the raw frequency of t in document d. The *inverse document frequency* is obtained by dividing the total number of documents by the number of documents containing the term t, and then taking the logarithm of that quotient. Therefore, the idf of term t with regard to the set of all the documents D is defined as

$$idf(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{4}$$

One important step of LSA is to perform singular vector composition (SVD) on the term-document matrix X, where X_{td} is given by $tf(t, d) \times idf(t, D)$. By using SVD, we have

$$X = U\Sigma V^T \tag{5}$$

By only retaining the k largest singular values and their corresponding columns in the U and V matrices, we have a low-rank approximation of the matrix X

$$X \approx \hat{X} = U_k \Sigma_k V_k^T \tag{6}$$

By doing this, we are mapping features in high-dimensional space into a much lower-dimension space (k in this example) while preserving as much content information as possible. When projecting

to the lower dimension space, we can represent each title with vectors in $\mathbb{R}^k$ space. In practice, we choose the lower dimension k to be 10 for the balance of computational feasibility and information preserving criterion. After acquiring the content vector, we employed spectral clustering on those vectors to label the latent topic for each paper title. Therefore, we quantize each node (author) with a distribution on a latent topic space which we mined from the content information.

4.1.2 Latent Dirichlet Allocation

Though LSA is powerful and informative, but it requires much effort and computing resources to run and experiment with. When very large and noisy data come in, we need a more robust model for modeling the content information. Latent Dirichlet Allocation (LDA) [1] is such a probabilistic model for uncovering the underlying semantic structure of a document collection based on hierarchical Bayesian analysis. The original idea of LDA is to model documents as if they arise from multiple topics, where each topic is defined to be a distribution over a fixed vocabulary of terms. Let us take Twitter network as an example. In Twitter network where each node represents a Twitter user, the content information attached to each node is the tweets (microblogs) the user published or retweeted before. The tweets may have different latent topics, such as sports, finance, politics and entertainment. We want to model Twitter users and their microblogs as a distribution over the latent topics.

LDA can be naturally applied as an enhancement of LSA, and it assigns a document with a distribution of multiple topics, and emphasize the topic which is more relevant to the document itself. The basic technique of LDA is linearly scanning through each word in the document, and initializes by randomly assigning some topics to each word. Afterwards, LDA goes through a iterative improvement process (like Expectation Maximization) until convergence: for each word, it computes the probability of topic given document and word given topic. Then we reassign each word to a new topic based on the maximum likelihood estimation from the previous iteration. Finally, this process converges and each document is assigned a distribution over the latent topic space.

For Twitter data, we follow the same step as in the collaboration network. First, we filter out stop words and http hyperlinks[2]. Secondly, we build corpus using only term frequency of the documents. We utilize a parallel version of LDA implementation from PLDA package [9] to speed up the computation latency.

4.2 Speeded-up CELF

Following the idea in [2], we implemented a further-optimized version of CELF[8] (Cost Effective Lazy Forward) algorithm. In outbreak detection, the main operation when deciding whether or not to add a node to the "outbreaker" set is to retrieve a random cascade emanating from that node. For IC models, we toss the biased random coins and remove all edges not for propagation from G to obtain a new graph G' in advance. With this approach, the random cascade from a node v is simply the set of nodes that are reachable from v in graph G', denoted as $R'_G(\{v\})$. This speed up can be achieved with a linear scan of the graph G' by DFS where we can obtain $R'_G(\{v\})$ for all vertices $v \in V$. In the following rounds, we use the CELF optimization to select remaining seeds. The reason why we can use a priority queue for lazy forwarding is due to the "diminishing effect" of the submodular function. For all sensor placements $\mathcal{A} \subseteq \mathcal{B} \subseteq V$ and node $s \in V \backslash \mathcal{B}$, we have

$$R(\mathcal{A} \cup \{s\}) - R(\mathcal{A}) \geq R(\mathcal{B} \cup \{s\}) - R(\mathcal{B}) \tag{7}$$

Therefore, the $R(\mathcal{S})$ computed in the previous iteration can be used to bound $R(\mathcal{S} \cup \{u\})$ in the current iteration. For more details of this algorithm, we refer the readers to the pseudocode in Algorithm 1. The time complexity of the speeded-up CELF algorithm is $O(|E||V|)$.

[2]In this project, we regard the hyperlinks in the tweets as irrelevant information.

Algorithm 1 Speeded-up CELF(G, k, p)

```
// R is the number of simulation iterations.
Initialize S = φ, R = 10000.
// Build graphs in advance and initialize R heaps.
for i = 1 to R do
    Compute G'_i by removing each edge from G with probability 1 − p.
    Run Depth-first Search on G'_i and compute R_{G'_i}({v}) for all v ∈ V.
    H_i = MakeHeap(v, |R_{G'_i}({v})|) for all v ∈ V.
    for every v ∈ V do
        cur_v^i ← false
    end for
end for
// Lazy forward
for i = 1 to k do
    Set s_v = 0 for all v ∈ V/S.
    for iter = 1 to R do
        while true do
            v ← DeleteMin(H_i)
            if cur_v^i then
                break
            else
                DecreaseKey(H_i, v, |R_{G'_i}({v}) ∩ complement(R_{G'_i}(S))|)
                cur_v^i ← true
            end if
        end while
        s_v += |R_{G'_i}({v}) ∩ complement(R_{G'_i}(S))|
    end for
    S = S ∪ argmax_{v∈V/S}{s_v}
end for
Output S.
```

4.3 Content Integration

After using content modeling techniques in 4.1, for each node v, we have $c_v \in \mathbb{R}^n$. We instantiate the definitions of mappings in section 3 to show how we integrate content correlation in our project.

We define $Sim : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow [0, 1]$ based on the cosine distance $\cos(x, y) = \frac{x^T y}{\|x\|\|y\|}$ to capture the proximity of vector space. We use thresholding heuristics to define a saturated function is

$$Sim(c_u, c_v) = \begin{cases} 1 & \text{if } \cos(c_u, c_v) \geq \theta_{\text{high}} \\ 0 & \text{if } \cos(c_u, c_v) \leq \theta_{\text{low}} \\ \frac{1}{d_{u,v}} & \text{otherwise} \end{cases} \tag{8}$$

where θ_{high} and θ_{low} are some pre-defined thresholds. This kind of high-end and low-end clipping makes the connections between nodes that share very little content in common, and enforce the connections between nodes that have very similar content.

We further elaborate on how to get c_u for each node when we have modeled the content that is attached to the node u. In this project, we simply sum up all the content distribution and normalize it to a valid distribution.

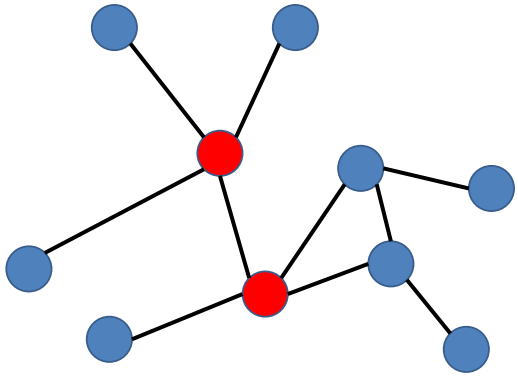

Figure 2: An example network. The node in red are influential "seed nodes" picked by the CELF algorithm.

4.4 Outbreak Prediction

With the content information being modeled, we can predict if a specific given tweet will blow up in the information spreading network. After running CELF in our network, we have chosen a set of "seed nodes" as our proposed sensor locations. Because information emanating from these "seed nodes" will propagate through the entire network very quickly, these nodes are the key to understanding the information diffusion process. After we finished modeling the content in section 4.1, we have obtained the latent topic distribution for each of the node in the graph. Therefore, for each of these "seed node"s we know exactly their content. Consider a scenario where one piece of specific information (e.g. a piece of tweet) come into the network, how likely is it going to trigger a cascade of information diffusions so that this information can reach a large portion of the nodes in the network? We try to answer this question by evaluating the likelihood of these "seed node"s accepting this piece of twitter information. Because we have modeled the tweets in the same latent topic space as the nodes in the network, we can measure the proximity between the content of tweets and the topic distribution of nodes in the graph. We can use these proximities (like anchor distances) to predict if this piece of tweet will be accepted by the influential seed nodes and later forwarded to other nodes in the network.

With the help of machine learning approaches, we use the distances between influential "seed nodes" and the topic distribution of tweets to train classifiers which predict if a specifically-given tweets will trigger a large cascade of impact (or outbreaks) in the information diffusion network.

5. EMPIRICAL RESULTS

In this section, we showcase some of the experimental results we obtained on two datasets we used in our project. We also summarize the major findings when running various experiments. Furthermore, we will show the prediction accuracies for outbreak prediction tasks.

5.1 Dataset

5.1.1 Collaboration network data

By using the citation network in KDDCup 2003, we wrote a python crawler to request the arxiv API and download all the titles, abstract and authors into our local database, which can be later served as the content information. We treat all the names with the same first letter of given name and same last name as the same author. As the paper[6] pointed out, this approach won't corrupt the data a lot. We show a plot of degree distribution for this collaboration network in Figure 3. There are $8,179$ nodes and $26,140$ edges in this network.

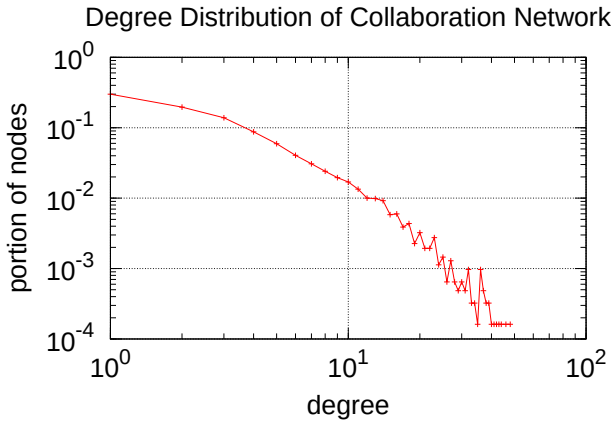

Figure 3: Degree distribution of collaboration network dataset

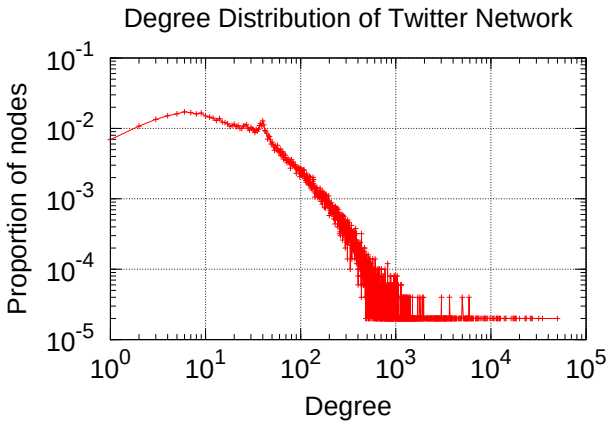

Figure 4: Degree distribution of Twitter dataset

5.1.2 Twitter network

We are using the Twitter tweets data on the SNAP website and the Twitter social graph data from [7]. Instead of using the whole Twitter social graph, which is prohibitive for computational reasons, we downsample a subset of the original network to conduct our algorithm for experiments and parameter tuning. The sampling process is simply a 1-step breadth first search (BFS) starting from a list of celebrity nodes provided in [7]. We use the subgraph induced by all the traversed node in BFS. There are $50,011$ nodes and $3,954,516$ edges in this network, and the corresponding degree distribution is in Figure 4.

5.2 Results on the Collaboration Network

In this section, we are showing results on the collaboration net-

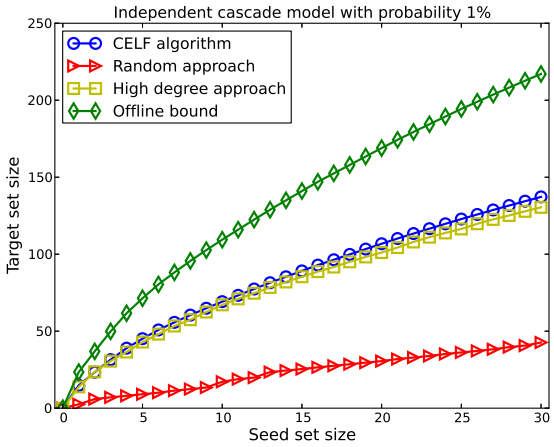

Figure 5: Independent cascades model with probability 1% on the collaboration network

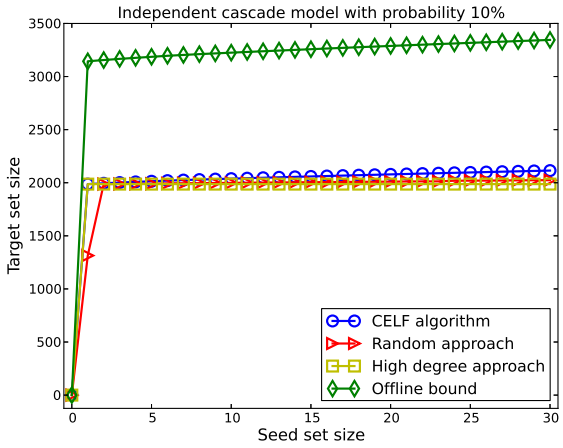

Figure 6: Independent cascades model with probability 10% on the collaboration network

works. Figure 5 and 6 are the results of running our implementation of speeded-up CELF algorithm on the collaboration network. Here we use the Information Cascade model with probability $p = 0.01$ and 0.1 respectively. We run the simulation 1000 times to report the results, and we compare the results with the strategies of selecting random nodes (Random Approach) and high degree nodes (High Degree Approach). We also compared the curve of the result with the theoretical off-line upper-bound curve (Offline Bound), which is simply $|R(S)|/(1 - 1/e)$. As can be shown in the plots, our results are very similar to the findings in [6]. In Figure 7, we show the topic clustering results on the collaboration network on high energy physics theory. The topic distribution is quite well-balanced through all the papers we crawled from the arxiv. In Table 1, we also demonstrate some interesting clustering results on the paper titles. The key words part are extracted manually from the titles that are aggregated under the same topic label.

We give a plot of topic similarity score distribution in Figure 8, so that we can choose θ_{low} and θ_{high} according to this distribution.

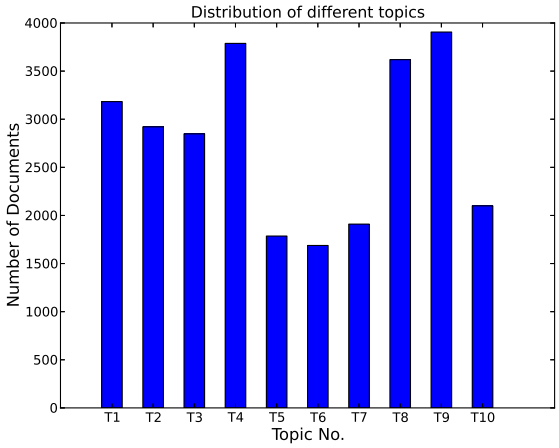

Figure 7: Distribution of 10 different topics after content modeling

Keywords	**Paper Titles**
Gravity	The Shape of Gravity Gravity in the Randall-Sundrum Brane World
String	Four Dimensional String Triality Enhanced Gauge Symmetry in String Theory
Geometry	Matter from Toric Geometry Compactification, Geometry and Duality: N=2
Symmetry	Second-Quantized Mirror Symmetry Mirror Symmetry is T-Duality

Table 1: Examples of topic modeling

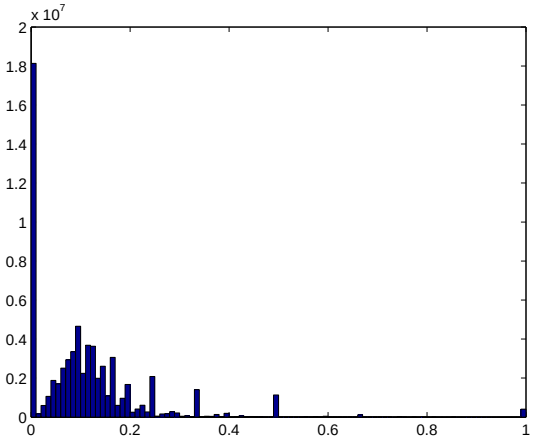

Figure 8: Topic similarity score distribution of collaboration network dataset

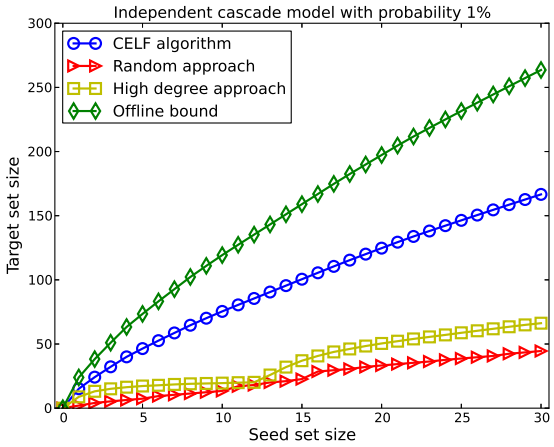

Figure 9: Outbreak detection after incorporating content

In Figure 9, we give the result after adding content information into outbreak detection. Interestingly, we observe that compared to Figure 5, the outbreak detection is more efficient in a sense that we have more activated nodes with the same size of seed nodes.

5.3 Results on the Twitter Network

We preprocessed the tweets data from the SNAP website with LSA/LDA to assign each documents (tweets) a topic distribution. After that, we built user profile base on the document they authored. We chose the social sensors (users) based on the CELF algorithm, but with an enhanced-by-topic-similarity weighted probability graph and a plain linear threshold model graph. We re-fetched tweets which are retweeted by tweetmeme account, which is outbreak as positive example and manually select some tweets which are not trending as negative example. Finally, we built a discriminative classifier by using those sensors as feature vector. We trained the classifier and tested if the classifier can predict which tweets are more likely to outbreak. In Figure 10 and 11, we plot the results for running outbreak detection on the Twitter network using independent cascade model and linear threshold model. We enhance the edge probability of the graph given as input to CELF algorithm. We calculate the intersections between users under one topic and we reward that overlap. The user pairs without interest similarities will have a default edge probability 0.01 if they have a follow relation. For each topic, we generate a different weighted graph and feed it as input to CELF algorithm. We noticed that we actually choose different set of users for different topics since each topic will enhance the edge probability of different user pairs. The plot of this set of results can be found in Figure 12. We then train an SVM classifier based on the distance of user profile to a tweets to see if we can distinguish those tweets with higher probability to outbreak. We achieve an approximately 70% successful rate and we didnt observe a big improvement when we add content information to choose the different set of sensors. The results are summarized in Table 3. An example of topic distribution for different users is shown in Figure 13. Furthermore, we demonstrate a comparison of Tweets Analysis using LDA topic modeling in Table 4 and 5.

6. FUTURE WORK

In future, we are going to run experiments on the larger dataset, which is much harder to tackle in our current framework. Also,

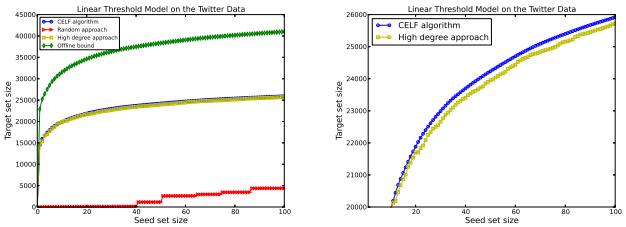

Figure 10: CELF results on the Twitter data with linear threshold model

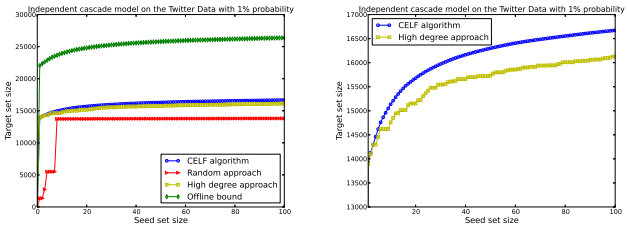

Figure 11: CELF results on the Twitter data with independent cascade model

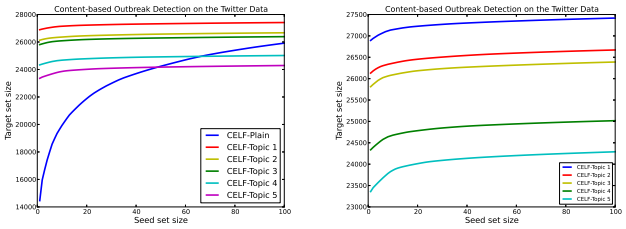

Figure 12: CELF results on the Twitter data with enhanced topic correlations

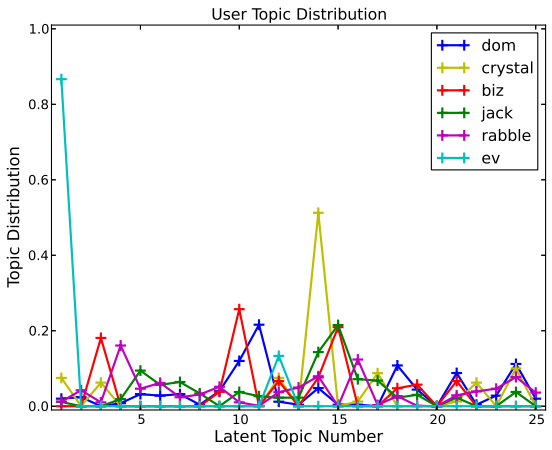

Figure 13: Topic distribution for different Twitter users

Table 2: Tweets Outbreak Prediction Using SVM

	Linear Kernel				RBF Kernel			
Seed Node Number	10	20	30	40	10	20	30	40
Plain CELF (Baseline)	51.8%	69.8%	72.2%	72.8%	64.5%	71.3%	73.1%	72.4%
Content-based CELF 1	63.6%	68.5%	69.0%	69.3%	66.9%	64.7%	70.6%	74.1%
Content-based CELF 2	59.4%	67.7%	71.3%	69.2%	62.7%	68.4%	71.6%	72.4%
Content-based CELF 3	53.8%	59.9%	70.9%	68.8%	62.9%	68.1%	72.6%	74.0%
Content-based CELF 4	52.7%	58.9%	70.4%	**74.4**%	57.5%	64.4%	71.0%	72.5%
Content-based CELF 5	52.9%	66.5%	71.3%	75.0%	54.8%	72.0%	74.0%	**75.1**%

Table 3: Accuracy of predicting the outbreak of individual tweet

Topics	**Example Tweets by Topic Integrated Sensors**
Politics	hhs is announcing a flu prevention psa contest hearing a lot about the citizen journalism site all voices reputation system revenue sharing noonan on palin thoughtful and needed republican rebuke of the antiintellectual section of her party
Entertainment	watched bigger stronger faster stars lost season fan made trailer Tim Armstrong's 100-Day Vision Quest Nearing End: Party in Dull
Journalism	Obama talks of stronger footing for relations adds tentatively I've been hearing a lot about the citizen journalism site All Voices: reputation system, revenue sharing Vanguards shame great post by one of best and most ethical journos

Table 5: Examples of topic modeling

Tweets by Plain Sensors
apple patching critical sms vulnerability in iphone os
first iphone gs jailbreak hits the web
people who rock on twitter
obama to seek new arms control deal in moscow
open forum by american express open a story of business engine optimization

Table 4: Examples of topic modeling

we'd like to investigate a more automatic way to choose the trending tweets. Since twitter data does not contain the retweet amount and comment amount information, it is hard to determine if a tweet is trended. However, by applying the LDA method, we can potentially identify which tweet is trending. We also want to explore a better way of sampling the network, while preserving the network structure. We only preserve the power distribution of the real world graph. Taking time factor as a consideration, running CELF with time reduction instead of the number of influenced nodes may also be good direction to move on. We can also choose the influential nodes by trying out other algorithms such as the PageRank, central betweeness and so on.

7. REFERENCES

[1] D. Blei and J. Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, pages 17–35, 2007.

[2] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208. ACM, 2009.

[3] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

[4] S. Dumais, G. Furnas, T. Landauer, S. Deerwester, S. Deerwester, et al. Latent semantic indexing. In *Proceedings of the Text Retrieval Conference*, 1995.

[5] A. Goyal, F. Bonchi, and L. Lakshmanan. Learning influence probabilities in social networks. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 241–250. ACM, 2010.

[6] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[7] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[8] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[9] Z. Liu, Y. Zhang, E. Y. Chang, and M. Sun. Plda+: Parallel latent dirichlet allocation with data placement and pipeline processing. *ACM Transactions on Intelligent Systems and Technology, special issue on Large Scale Machine Learning*, 2011.

[10] S. Myers and J. Leskovec. Clash of the contagions: Cooperation and competition in information diffusion.