

Predicting Trends in Social Networks via Dynamic Activeness Model

Shuyang Lin[†] Xiangnan Kong[†] Philip S. Yu^{†*}

[†]Department of Computer Science
University of Illinois at Chicago
Illinois, USA

^{*}Computer Science Department
King Abdulaziz University
Jeddah, Saudi Arabia

{slin38,xkong4,psyu}@uic.edu

ABSTRACT

With the effect of word-of-the-mouth, trends in social networks are now playing a significant role in shaping people's lives. Predicting dynamic trends is an important problem with many useful applications. There are three dynamic characteristics of a trend that should be captured by a trend model: intensity, coverage and duration. However, existing approaches on the information diffusion are not capable of capturing these three characteristics. In this paper, we study the problem of predicting dynamic trends in social networks. We first define related concepts to quantify the dynamic characteristics of trends in social networks, and formalize the problem of trend prediction. We then propose a Dynamic Activeness (DA) model based on the novel concept of activeness, and design a trend prediction algorithm using the DA model. Due to the use of stacking principle, we are able to make the prediction algorithm very efficient. We examine the prediction algorithm on a number of real social network datasets, and show that it is more accurate than state-of-the-art approaches.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Application—*Data Mining*

General Terms

Algorithms, Experimentation

Keywords

Information Diffusion, Social Influence

1. INTRODUCTION

Online social networks have become increasingly important for interpersonal communication and information sharing. Trends in online social networks now have large impacts

on people's lives. Trends are represented by sequences of actions that are taken by users in a social network. According to the type of the social network, an action can be posting a blog or sharing a webpage about a certain topic, or joining an online activity.

Predicting the dynamic behavior of trends is an interesting problem with wide applications. Some examples of such applications are as follows:

1. Online video providers may want to predict how many times a video will be played by users in the next month, so that they can decide the bandwidth needed for the server.
2. Disease control facilities may want to predict how many people will suffer from a contagion in the following week, so that they can be prepared for an outbreak.
3. Manufacturers may want to predict how long an existing product will continue to be popular, so that they can decide the most suitable time for the debut of a new model.

The three applications above require the prediction of trends from three different perspectives. The first example considers the **intensity** of a trend, which is the volume of actions during a fixed length of time. The second one focuses on the **coverage** of a trend, which is the number of people taking the given action during a fixed length of time. The third one considers the **duration** of a trend, which is the time span that the intensity or coverage is above a given threshold.

To better explain these three perspectives (intensity, coverage and duration), we show in Figure 1 a toy example of a trend on a social network which contains three users. Table (b) shows the intensity, coverage and duration that aggregated from actions listed in Table (a). For example, at 2008, v_1 and v_2 take 3 and 2 actions, respectively, while v_3 taking no action, so the coverage (i.e. the number of people taking actions) is 2, and the intensity (i.e., the total number of actions taken) is 5. Though the coverage and intensity are correlated with each other, they are not interchangeable in the sense that the corresponding time series are neither similar nor synchronized. In this example, the maximum value of coverage is reached at year 2009, while the maximum value of intensity is reached at year 2008. Duration reflects how long the trend lasts. If we set the threshold to 0, duration of the trend will be 4 years, from 2007 to 2010.

Based on our observation, each of the three perspectives is useful for many real applications. A trend model should

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WOODSTOCK '97 El Paso, Texas USA

Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...\$15.00.

	2007	2008	2009	2010	2011
v_1	1	3	2	1	0
v_2	0	2	1	0	0
v_3	0	0	1	1	0

(a) Number of actions of the trend

	2007	2008	2009	2010	2011
Coverage	1	2	3	2	0
Intensity	1	5	4	2	0
Duration	2007 - 2010				

(b) Coverage, intensity, and duration of the trend

Figure 1: An example of trend in a social network

be able to characterize trends from all of these three perspectives.

Though the actions of social network users have been studied in the context of information diffusion models (e.g. the independent cascade (IC) model) [9, 7, 13, 12, 5, 4, 8, 10], the existing information diffusion models are not suitable for modeling dynamic trends for three main reasons: **First**, most of these models assume that the diffusion processes take place in discretized time and the propagation of information between two nodes always takes one unit of time, which does not reflect the real dynamic of trends as time unfolds. Therefore, they cannot reflect the dynamic nature of intensity and coverage, or the duration of trends. **Second**, information diffusion models focus on the visible path of propagation, and they usually assume that the propagation can only occur between a pair of nodes that are directly linked to each other, while the trend model should focus on predicting the aggregate characteristics of trends, and the path of propagation is not important for the prediction. Besides, because of the existence of homophily [2, 19], the propagation through direct links may not be good enough to explain trends in social networks. The model of trends should be more flexible with regard to the propagation mechanism. **Third**, information diffusion models focus on the prediction on the individual user level, but not on the trend level. As a result, they allow the probability of influence to be different between each pair of users, but assume that the probability remains the same for all the trends. This makes them not suitable for predicting trends based on different properties of trends.

In this paper, we formally define the three dynamic characteristics of a trend (intensity, coverage and duration), and the problem of trend prediction. We introduce a novel concept of activeness, which reflects a user’s interest toward the given trend at a given point of time. The dynamic nature of activeness enables us to model the dynamic characteristics of trends. Due to the introduction of activeness, a more flexible propagation mechanism is made possible, so that the correlation as well as influence through direct links can be captured in our model. We propose a Dynamic Activeness (DA) model based on the concept of activeness. Each component of the model is built on observations on real trends, and the model is capable of capturing all the three dynamic characteristics of trends. We design a trend prediction algorithm based on the DA model. For each trend, the parameters of the model are learned specifically from the history data of that trend. The learned model can then be used to predict the dynamic characteristics of the trend in the future. For

the efficiency consideration, we identify the stacking principle and utilize it to transform the effect of the sequence of actions to the sum of the individual effects by each single action in isolation. This makes the prediction based on the DA model have a similar computational complexity to the IC model. We show the performance of the DA model on real trends in social networks.

2. PRELIMINARIES

2.1 Notations and Definitions

Let $G = (V, E)$ be a social network, in which $V = \{v_1, \dots, v_n\}$ is the set of nodes and $E \subseteq V \times V$ is the set of edges. We consider the network to be static, since the evolution of networks is much slower than that of the trends. During the time span of a given trend, the change of the network is negligible.

A trend on the social network G is defined as follows:

DEFINITION 1. Trend A trend $S = [(v_1, t_1), \dots, (v_m, t_m)]$ on the social network G is a chronological sequence that consists of a given type of actions in G , where $t_i \leq t_j (\forall 0 \leq i < j \leq m)$ and $v_i \in V (\forall i \in \{1, \dots, m\})$. An element (v_i, t_i) in S corresponds to an action of that type taken by the node v_i at time t_i .

For simplicity of notation, we denote the time sequence of actions in trend S as $T(S) = [t_1, \dots, t_m]$, and denote the subsequence of S that consists of all the actions taken by node v as $S^v = [(v, t_{v,1}), \dots, (v, t_{v,m_v})]$, where $(v, t_{v,i}) \in S (\forall 1 \leq i \leq m_v)$. We also use S_t to denote the prefix of S which contains all the actions taken before the time point t , i.e., $S_t = [(v_i, t_i) : (v_i, t_i) \in S, t_i \leq t]$.

Based on the above definition of a trend, we define intensity, coverage, and duration of a trend as follows:

DEFINITION 2. Intensity The Intensity of a trend S on a time interval $I = [t_{min}, t_{max})$ is the number of actions in S that are taken during I . Formally, $Intensity(S, I) = |\{(v_i, t_i) : t_i \in I \wedge (v_i, t_i) \in S\}|$.

DEFINITION 3. Coverage The Coverage of a trend S on a time interval $I = [t_{min}, t_{max})$ is the number of nodes in V that take at least one action during I . Formally, $Coverage(S, I) = |\{v_i : (v_i, t_i) \in S \wedge t_i \in I\}|$.

DEFINITION 4. Duration Let $\mathcal{I} = \{I_1, \dots, I_s\}$ be a set of intervals, where $I_i = [t_{min}^i, t_{max}^i)$. Given a threshold θ , the duration of S on \mathcal{I} is the number of consecutive intervals in \mathcal{I} that the intensity (coverage) is above θ . Formally, $Duration_{cov}(S, \mathcal{I}, \theta) = \max(j - i + 1), 1 \leq i \leq j \leq s$, s.t. $\forall k, i \leq k \leq j$, $Coverage(S, I_k) > \theta$ and $Duration_{int}(S, \mathcal{I}, \theta) = \max(j - i + 1), 1 \leq i \leq j \leq s$, s.t. $\forall k, i \leq k \leq j$, $Intensity(S, I_k) > \theta$.

The intensity quantifies the overall activeness of a trend within a social network. The coverage quantifies how broad a trend has impacts in a social network, i.e., the number of nodes involved within a time interval. The larger the coverage value of a trend is, the more nodes of the network are affected/involved in the trend. The duration quantifies how long a trend lasts within the social network.

For the duration, we usually want I_1, \dots, I_s to be consecutively connected intervals with equal length, i.e., $t_{max}^i =$

t_{min}^{i+1} ($\forall i \in \{1, \dots, s-1\}$) and $t_{max}^i - t_{min}^i = t_{max}^j - t_{min}^j$ ($\forall i, j \in \{1, \dots, s\}$). We like to point out that, given the intensity and coverage, the duration of trend can be defined in many different ways. We define it as the largest number of consecutive intervals above the threshold because this definition is most straightforward. By carefully setting the threshold θ , the definition will accord with the intuitive understanding of the word “duration”.

The prediction problem of trends is defined as follows:

DEFINITION 5. Trend Prediction Problem Given S_{t_*} , the prefix of sequence S before time t_* , the problem of trend prediction is to predict the intensity, coverage and duration of trend S after time t_* .

Typically, we solve the prediction problem on a set of consecutively connected equal-length intervals $\mathcal{I} = \{I_1, \dots, I_s\}$, where $I_i = [t_{min}^i, t_{max}^i)$ and $t_{min}^1 = t_*$. The problem is to predict $Coverage(S, I_i)$ and $Intensity(S, I_i)$ for each $I_i \in \mathcal{I}$, and $Duration_{cov}(S, \mathcal{I}, \theta_S)$ or $Duration_{int}(S, \mathcal{I}, \theta_S)$ for a given θ_S .

2.2 Datasets

We take our observation and evaluation on two social networks: DBLP co-author network and Twitter user network.

DBLP co-author network: In this network, the nodes correspond to the authors and the edges correspond to the co-authorship. The dataset contained 934,672 nodes and 8,850,502 edges. The trend data are extracted from the DBLP data by detecting terms in the titles of publications. In each trend, an action (v_i, t_i) corresponds to a publication of the author v_i at time t_i that contains the given term in the title. For publications with multiple authors, there is an action for each of the authors.

Twitter user network: In the network, the nodes correspond to the users and the edges correspond to the who-follows-whom relationships. The edges are directed from the users that are being followed to the followers. We randomly crawl a sub-network of Twitter network, which contains 40,906 nodes and 7,829,834 edges. The trends are defined by hashtags in tweets. An action (v_i, t_i) in a trend corresponds to a tweet of user v_i at time t_i that contains the given hashtag.

3. DYNAMIC ACTIVENESS (DA) MODEL

3.1 Concept of Activeness

The DA model for trend in social network is based on the novel concept of *activeness*. For each trend, each node in the social network has an activeness function associated with it. The two main aspects of the concept are:

- Activeness of a node is defined as its interest toward the given trend. It is a function of time. As time goes by, activeness may increase as a result of information diffusion or social influence, or decrease as the node loses interest to the given trend.
- Activeness decides the frequency of actions taken by the node. The higher the activeness of a node is, the more actions it is likely to make in a unit time. In this sense, we can also define activeness as the “action rate” of a user.

Since activeness is dynamic in nature, we are able to design the DA model based on it, so that the model can capture the three dynamic characteristics of trends. Besides, by using activeness in the model, we are also able to design

a more flexible information propagation mechanism, as we will show in Section 3.3.1.

3.2 Framework of DA Model

As shown in Figure 2, the DA model contains three elements: activeness propagation, decay of activeness and action generating process. Each of them is based on observations on real trends. Actions and activeness are connected to each other in the DA model. On the one hand, actions trigger activeness propagations in the social network. Activeness propagation, together with the decay of activeness, decides the activeness of each user at each point of time. On the other hand, actions are generated by the action generating process which takes the activeness as input.

As shown in Figure 2, the prediction algorithm contains two phases. In the learning phase, parameters of the DA model are learned from the observed part of trends. In the prediction phase, the DA model predicts the actions in the future. Intensity, coverage and duration of trends can be predicted by aggregating the predicted future actions.

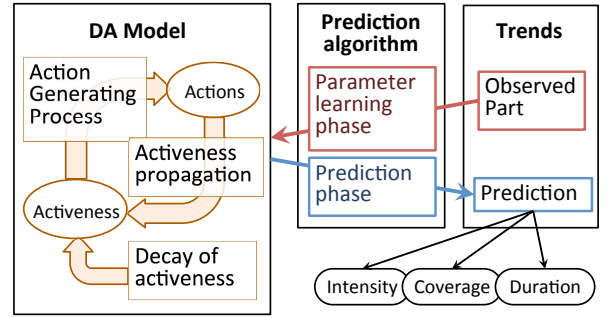


Figure 2: Block Diagram of the DA Model

3.3 Activeness Modeling

For each trend S , let $r_v(t)$ be the activeness or action rate of node v at time t . In this section, we discuss the modeling of $r_v(t)$. As shown in the left most box in Figure 2, the model of activeness contains two parts: activeness propagation and activeness decay.

3.3.1 Activeness Propagation

Intuitively, as a result of social influence or homophily [19, 3], the activeness of nodes are correlated with each other. The closer two nodes are, the larger the correlation is. Thus, when a node takes an action in a trend, we can expect that nodes in proximity to it have larger activeness for the trend than other nodes in the social network.

Our observation on the real trends supports this intuition. Figure 3(a) shows the curves of activeness for four trends in DBLP network, i.e., trends about “boosting”, “privacy” etc. Other trends have similar curves. For each trend, we plot the average activeness (i.e. the average number of actions per unit time) for nodes with different shortest path distances to nodes with previous actions. As we can see from the figure, as the distance increases, the activeness of nodes exponentially decreases. An important remark is that the information diffusion along direct links is not enough for explaining trends. Because if we explain trends in that way, all the nodes except for those that are directly linked to the

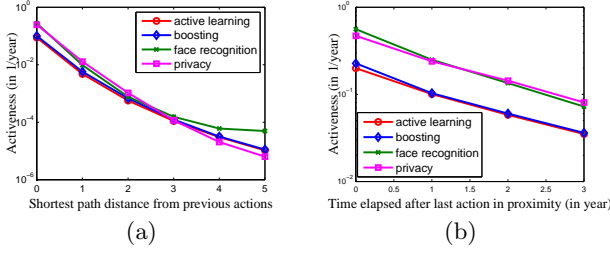


Figure 3: (a) Activeness by different distances to nodes with previous actions. (b) Activeness by the time elapsed since last action in proximity

nodes with previous actions should have the same activeness. It is also interesting to point out the exponential decreases also fits for action rates of nodes with 0-hop distance (i.e. nodes themselves have previous actions).

Based on this observation, we use the previous actions of a trend to model the activeness of nodes. Let $prox(u, v)$ be the proximity measurement from u to v . When an action is taken by node u , the increase of activeness of v is proportional to $prox(u, v)$, i.e., if u takes an action at time t_a , for every node v in the network we have:

$$\lim_{t \rightarrow t_a +} r_v(t) = \lim_{t \rightarrow t_a -} r_v(t) + \alpha \cdot prox(u, v) \quad (1)$$

where $\lim_{t \rightarrow t_a +} r_v(t)$ and $\lim_{t \rightarrow t_a -} r_v(t)$ are the activeness of node v after and before the jump at time t_a , and α is the propagation ratio that depends on the trend.

We study two different measurements for proximity. The first one uses the shortest path distance from the source node to destination node. As we observed in the real trend data, proximity is defined as an exponential decreasing function of shortest distance, i.e., $prox(u, v) = exp(-b \cdot dist(u, v))$, where $b \in \mathcal{R}^+$ and $dist(u, v)$ is the length of shortest path from u to v . The second proximity measure is based on random walk, which is described as rooted PageRank in [14]. To measure the proximity of nodes from a given node u , the random walk is started at node u . At each step, it has a probability of p to return to node u , and $1 - p$ probability to move to neighbor nodes. The proximity of node v from node u is defined as the stationary probability for v . In both of the measurements, if v is not reachable from u , we have $prox(u, v) = 0$.

While information diffusion models define the influence between nodes along the edges, the “propagation” of activeness captures a more general sense of correlation between activeness of nodes, rather than the process of information diffusion. Besides, it is different from information diffusion model in how it is parameterized. The parameter α of the activeness propagation depends only on the trend, but not on the node that takes the action. To the contrary, the information diffusion models usually have diffusion probabilities with individual edges as parameters, and the diffusion probabilities are constant for all the trends. We parameterize the activeness propagation in a different way for two reasons: First, it is simply not practical to make the parameters depend on trends and edges at the same time, since there will be not enough actions to be used for the learning of each parameter. Second, the main purpose of the proposed trend model is to predict the aggregate characteristics of different

trends, so it is more meaningful to bind the parameter with the trends instead of the edges.

3.3.2 Decay of activeness

Intuitively, if a user is not exposed to any new information or influence from a certain trend, nor does he create any new content that belongs to that trend, the user’s interest to that trend will gradually decay. In other words, the activeness of a node spontaneously decays if there is no new action taken by nodes in proximity to it. This spontaneous decay is observed from real trends.

Figure 3(b) shows the average activeness for nodes as time progresses since last action in proximity. Let $R(v, k) = \{u \in V | sp(u, v) = k\}$ represent the set of nodes that are k hops away from v , where $sp(u, v)$ is the shortest path distance from u to v . The X-axis of the figure is the time elapsed since last action taken by nodes in $R(v, 1)$. The Y-axis of the figure is the average activeness. As shown in the figure, activeness decreases when the node is not exposed to a new action. Decrease of activeness can roughly be regarded as exponential. (We only show the case of $k = 1$ and the curves for four trends in this figure. But actually we have done the same test for different k values and for different trends, and the other curves are similar.)

According to the observation, we introduce an exponential decrease to the activeness model. For each interval $[t_0, t_1]$ when $r_v(t)$ is not increased by the activeness propagation, $r_v(t) = r_v(t_0)e^{-(t-t_0)/\tau}$ for any $t \in [t_0, t_1]$. τ is a rate of the activeness decay, For similar reasons as to the parameter α , τ depends on the trend, but not depends on the node or the edge.

3.3.3 Summary of Activeness Model

Combining the two parts discussed in Section 3.3.1 and Section 3.3.2, $r_v(t)$, the activeness of node v at time t , is given by:

$$r_v(t) = \alpha \sum_{(v_i, t_i) \in S_t} (prox(v_i, v) e^{-(t-t_i)/\tau}) + r_v(t_0) e^{-(t-t_0)/\tau} \quad (2)$$

where t_0 is the start time of the trend. We set $r_v(t_0)$ to a small value ϵ . $r_v(t)$ is discontinuous at time points when there is a new action taken by nodes in $R(v)$. In each interval between those discontinuous points, $r_v(t)$ is subject to an exponential decay.

3.4 Action Generating Process

In this section, we discuss the generating process for actions. As we mentioned in Section 3.1, the activeness of a node serves as the action rate in the generating process. With the assumption that time points of actions are conditionally independent, given the activeness function, the generating process for the sequence of time points is a non-homogeneous Poisson process. This assumption keeps the model simple. We will first show that this assumption is reasonable for the action generating process, and then discuss the detail of the generating process under this assumption.

The verification is based on the fact that inter-action time of a homogeneous Poisson process, i.e. a Poisson process with constant action rate, follows exponential distribution. The sequence for each node usually contains only a few actions in a trend, which are not sufficient for the analysis. Instead, we use the global time sequence of all the actions in

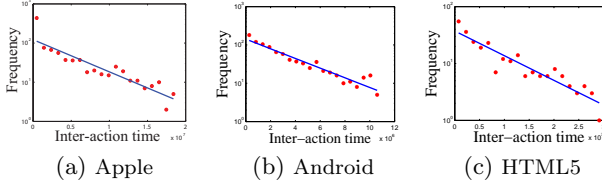


Figure 4: Distribution of inter-action time for three real trends. Other trends have similar curves.

a trend. As a property of Poisson processes, the sum of two Poisson processes is also a Poisson process. For our case, if each individual action sequence is generated by a Poisson process, the global sequence is also generated by a Poisson process. The action rate function of this sequence is the sum of all nodes' action functions, i.e., $r(t) = \sum_{v \in V} r_v(t)$. Since $r(t)$ reflects the global activeness of all nodes, it will not change too much in a short term. Therefore, the distribution of the inter-action time should roughly be an exponential distribution.

We plot the inter-action time distribution for three trends in the Twitter network in Figure 4. For each of the trends, we show the distribution of inter-action time during a week period when the trend is popular in the Twitter network. We divide the inter-action time into 20 bins according to the length, and plot the frequency of each bin. As showed by these figures, the inter-action time fits exponential distributions quite well. Thus, we conclude that the independent assumption is reasonable.

Under the conditional independent assumption, we derive the process of generating actions as follows. For a non-homogeneous Poisson process, the number of actions taken by this node in any time interval $[t', t)$ follows a Poisson distribution:

$$P[|S_t^v| - |S_{t'}^v| = k] = \frac{e^{-\int_{t'}^t r_v(t) dt} (\int_{t'}^t r_v(t) dt)^k}{k!}$$

where $|S_t^v|$ is the number of actions taken by node v before time point t .

Suppose we are now at the time point t' and want to generate the next time point after t' . By taking derivative with respect to t , we can get the probability density function for the waiting time until next action:

$$f_{v,t'}(t) = r_v(t) \cdot \exp\left(-\int_{t'}^t r_v(t) dt\right) \quad (3)$$

We then can generate the next time point in the sequence by drawing from the distribution of the waiting time.

4. PREDICTION ALGORITHM

In this section, we show prediction algorithm based on the proposed DA model. The algorithm contains two phases: the parameter learning phase and the prediction phase. In the first phase, parameters for each trend are learned from the part of the trend before t_* by maximum likelihood estimation. In the second phase, we use the learned model to predict the future trend sequence after t_* .

4.1 Parameter Learning Phase

For each trend, two parameters in the DA model need to be learned: the proportionality factor α in activeness propagation and mean lifetime τ of activeness decay. We use maximum likelihood estimation for the parameter learning.

The likelihood function is given by:

$$L(\alpha, \tau) = \prod_{v \in V} f(T(S_{t_*}^v), |S_{t_*}^v|; \alpha, \tau) \quad (4)$$

where $T(S_{t_*}^v)$ is the time sequence for node v 's actions before time t_* , and $|S_{t_*}^v|$ is the number of actions taken by v before time t_* . $f(\cdot)$ is the joint probability density function of the time sequence $T(S_{t_*}^v)$ and $|S_{t_*}^v|$.

As we show in Appendix A, by taking the partial derivative of $\log L(\alpha, \tau)$ with respect to α , we get the estimate values for α :

$$\hat{\alpha} = \frac{|S_{t_*}|}{\sum_{v \in V} H_v(t_*, \tau)} \quad (5)$$

Fixing α to $\hat{\alpha}$, we get

$$\begin{aligned} \hat{\tau} = \operatorname{argmax} [& |S_{t_*}| \log\left(\frac{|S_{t_*}|}{\sum_{v \in V} H_v(t_*, \tau)}\right) - |S_{t_*}| \\ & + \sum_{(v_i, t_i) \in S_{t_*}} \log(h_{v_i}(t_i, \tau))] \end{aligned} \quad (6)$$

where $h_v(t, \tau)$ and $H_v(t, \tau)$ are introduced for simplicity's sake:

$$h_v(t, \tau) = \sum_{(v_i, t_i) \in S_t} (\operatorname{prox}(v_i, v) e^{-(t-t_i)/\tau}) \quad (7)$$

and

$$H_v(t, \tau) = \tau \sum_{(v_i, t_i) \in S_t} (\operatorname{prox}(v_i, v) (1 - e^{-(t-t_i)/\tau})) \quad (8)$$

Due to the complexity of $H(t, \tau)$ and $h(t, \tau)$ with regards to τ , it is not possible to obtain a closed-form solution for the maximum-likelihood estimate of τ . However, since τ is the only variable here, we can use any line search algorithm to find the $\hat{\tau}$, and techniques such as simulated annealing can be adopted to avoid falling into local optimal value.

4.2 Prediction Phase

After we learn the parameters τ and α , we can generate the prediction of the action sequence after t_* . To do this, we keep track of the next action of each user in a list in the time order. Every time we pull the earliest action from the list and add it to prediction, then we update the list to capture the future actions that will be triggered by this action. The procedure is as follows:

1. Calculate $r_v(t_*)$, the activeness at time t_* for each node v in the network, using Equation 2.
2. Generate a next action for each node in the network from the pdf in Equation 3. Sort the actions in the ascending order of time, store them in a list L .
3. While L is not empty, pull the first action (v_i, t_i) from it.
 - i if $t_i > t_{end}$, jump to step 4.
 - ii Add (v_i, t_i) to S' , the predicted sequence.
 - iii For all the nodes that are reachable from v_i , update their activeness $r_v(t_i)$ by Equation 2.
 - iv Update the next action time using the pdf in Equation 3, and sort the list L again.

4. Calculate **intensity**, **coverage**, **duration** using the predicted sequence S' .

t_{end} in Step 3-i is the end point of the last time interval on which we want to predict the trend. Notice that the sequence generating process is a random process. We may repeat Step 3 several times to get the average value of aggregates. If implemented in a naive manner, the calculation of $r_v(\cdot)$ can be very time-consuming. We will describe an efficient implementation in Section 4.3.

4.3 Efficient Implementation

As a property of the DA model, the effect of the sequence of actions can be considered as the sum of the individual effects by each single action. We call it **stacking principle**. We can reduce the complexity of prediction algorithm based on the stacking principle. The general idea is to transform a complicated formula to a summation over the sequence of actions, which is easier to calculate. The principle can be applied to both the parameter learning phase and prediction phase. We are going to show the details of the implementation in Sections 4.3.1 and 4.3.2, and discuss the time complexity in Section 4.3.3.

4.3.1 Speedup of Parameter Learning Phase

For the parameter learning phase, the main complexity of computation comes from the calculation of $\sum_{v \in V} H_v(t_*, \tau)$ which is in both Equations 5 and 6. It will be very inefficient if we calculate the summation over all the nodes in the network in every step of the search algorithm. To reduce the complexity of calculation, we use the following equation:

$$\begin{aligned} \sum_{v \in V} H_v(t_*, \tau) &= \sum_{v \in V} \left[\tau \sum_{(v_i, t_i) \in S_{t_*}} (prox(v_i, v)(1 - e^{-(t_* - t_i)/\tau})) \right] \\ &= \tau \sum_{(v_i, t_i) \in S_{t_*}} \left[(1 - e^{-(t_* - t_i)/\tau}) \sum_{v \in V} prox(v_i, v) \right] \end{aligned}$$

The summation over all the nodes $\sum_{v \in V} prox(v_i, v)$ in the last equation does not depend on τ . It implies that for each v_i that $(v_i, t_i) \in S_{t_*}$, we can calculate the summation $\sum_{v \in V} prox(v_i, v)$ once and store it. Then in each step we only need to take a summation over all the actions in the trend sequence S_{t_*} . Notice that the number of actions in S_{t_*} is usually much smaller compared to the total number of nodes in the graph. Therefore, it is much more efficient to be calculated than the original form.

4.3.2 Efficient Implementation of Prediction Phase

For the prediction phase, the action rate function $r_v(t)$ in Equation 2 is a summation over all the actions in the network before time t . According to the property of Poisson processes, we can generate an action sequence for each action (v_i, t_i) before time t with the action rate function $\alpha \cdot prox(v_i, v)e^{-(t - t_i)/\tau}$, and then adding all these sequences together to get the action sequence for v . In other words, instead of calculating $r_v(t)$ for node v repeatedly, we can use the follow procedure to get exactly the same result: (1) For the action (v_i, t_i) , generate the sequence of actions that is caused by this single action for every node reachable from v_i , (2) and then merge this sequence of actions with the current list of action to generate the new list. This stacking principle greatly simplifies the prediction algorithm.

4.3.3 Time complexity

Due to the use of stacking principle, we can reduce the time complexity of the algorithm. Let the values of α and τ be n_p -digit precision, and m_{prox} be the maximum number of nodes in the proximity of a node, $|S|$ be the total number of actions in the sequence S . In the learning phase, calculating the intermediate result takes $O(m_{prox} \cdot |S|)$. The total number of steps of line search is $O(n_p)$, and each step of line search takes $O(|S|)$ to calculate the new value of objective function. Thus, the time complexity of learning phase is $O(n_p \cdot |S| + m_{prox} \cdot |S|)$. In the prediction phase, for each action in the S , the algorithm takes $O(|S|)$ time to generating the prediction sequence. The time complexity of prediction phase is $O(m_{prox} \cdot |S|)$. Thus, the total complexity is $O(n_p \cdot |S| + m_{prox} \cdot |S|)$. Usually, n_p is much smaller than m_{prox} , we can think the time complexity as $O(m_{prox} \cdot |S|)$.

The time complexity of a naive algorithm that does not make use of the stacking principle is $O(n_p \cdot m \cdot |S| + m \cdot |S|^2)$. It is much slower than our algorithm. The time complexity of prediction model based on the IC model is $O(m_{degree} \cdot |S|)$, where m_{degree} is the maximum number of nodes that are directly linked to a node. It is different from the time complexity of our algorithm in m_{degree} . That is because the IC model only considers the influence between nodes that are directly linked to each other. Our algorithm has similar time complexity with the IC model, though the IC model is a much more simplistic model.

5. EXPERIMENT

5.1 Algorithms and Performance Measures

As we mentioned in Section 3.3, for the DA model, we use two different measurements for the proximity between two nodes in the network. For the shortest path measurement (DA-sp), we set the decay factor b to 10 in the experiment. For the random walk measurement (DA-rw), the restart probability p is set to 0.4.

We compare the DA model with three variants of the widely used IC model. All of the three variants assume that each action comes with a delay, so that they can be used to model dynamic trends:

1. **eExp** (Edge-dependent exponential delay model [17]) assumes that all the actions propagate through a certain edge are drawn independently from the same exponential distribution.
2. **tExp** (Trend-dependent exponential delay model [17]) assumes that delays for all the actions of a certain trend are drawn independently from the same exponential distribution.
3. **tEqu** (Trend-dependent equal-length delay model [12]) assumes that there is a fixed-length delay for all the actions of a certain trend.

Parameters of the three baselines model are learned by the algorithm proposed in [17]. For the intensity prediction, we extend the IC variants with a “multiple actions factor” to allow a node to perform actions more than once. These extended IC variants will be explained later in Section 5.4.

To evaluate coverage and intensity, we use two measures: the error ratio and the coefficient of variation.

Error ratio is used to evaluate the goodness of the prediction compared with the true value. The formula of error

DBLP		Twitter	
Concept drift	Boosting	Sarcasm	Twibbon
Kernel methods	Privacy	SocialMedia	Wikileaks
SQL	Face recognition	Geek	Awesome
Heterogeneous network	Decision tree	SoundCloud	Android
Active learning	Streams	HTML5	Apple

Table 1: Trends in the DBLP and Twitter dataset

ratio is given by:

$$\text{error ratio} = \frac{|\text{truth} - \text{prediction}|}{\text{truth}}$$

Since all of the tested algorithms are stochastic algorithms, we also evaluate the variance of the outputs. For every test, we run each algorithm for multiple times and estimate the coefficient of variation. The coefficient of variation is estimated by:

$$\hat{C}_v = \frac{s}{\bar{x}}$$

where s is the sample standard deviation, and \bar{x} is the sample mean.

5.2 Trend Data for Evaluation

As we described in Section 2.2, we use two social networks for the evaluation. For each network, we conduct experiments on multiple trends, as listed in Table 1.

For **DBLP** dataset, we test 10 trends of hot keywords in the areas of data mining and machine learning. For each trend, we use the trend sequence before year 2005 (2005 excluded) as the training sequence, and the sequence from year 2005 to 2009 (2009 included) as the test sequence. We take each year as a time interval, on which we consider intensity and coverage.

For **Twitter** dataset, we test the algorithms on trends of 10 most popular hashtags in the last four months of year 2010. For each of the trends, we use the trend sequence from the 40th week to 47th week as training sequence, and the sequence from the 48th week to 52nd week as the test sequence. We take each week as a time interval, on which we consider intensity and coverage.

5.3 Coverage

In Figure 5, we plot the average error ratio for 5 consecutive time intervals after the end point of training sequence. As shown in the figures, error ratios for all the algorithms tend to increase as time progresses. For the DBLP dataset, the error ratio of tEqu deteriorates very quickly, while for the Twitter dataset, tExp is the worst one. In both figures, DA-rw and DA-sp have lower error ratios than the baselines for most of the ranges of X-axes, while eExp is in the middle. The difference between the error ratio of DA-rw and the error ratio of DA-sp is not large, which shows that the DA model is not sensitive to the different measurements of proximity, as long as the measurements are reasonable.

In Figure 6, we show the coefficient of variation for the coverage prediction. For the DBLP dataset, DA-rw and DA-sp have a lower coefficient of variation than the baselines, which means that the predictions got by the DA model are more stable. For the Twitter dataset, the coefficient of variation of tEqu is similar to the DA model. tExp has a strange decreasing curve, and its coefficient of variation is the lowest in the last two weeks of prediction. Considering the large error ratio of tExp in later weeks in Figure 5(b),

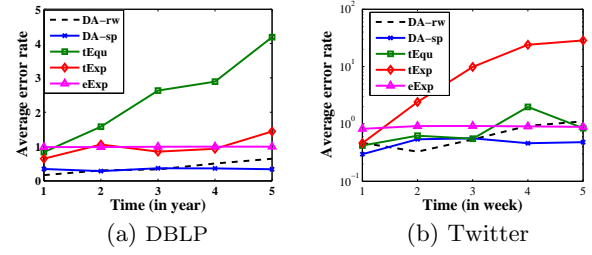


Figure 5: Error Ratio for Coverage Prediction

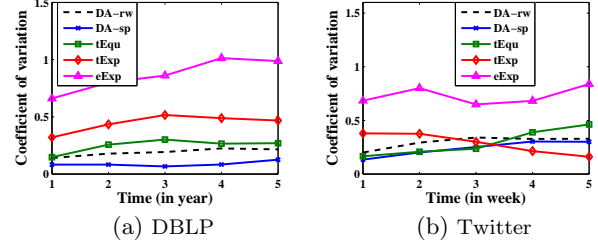


Figure 6: Coefficient of Variation for Coverage Prediction

its low coefficient of variation is most likely to be caused by a large mean value of its prediction. Thus, it is not comparable with the other three algorithms. Among all the algorithms, eExp have highest coefficient of variation, hence it is less stable than all the other algorithms. On both of the datasets, DA-sp has a slightly lower coefficient of variation than DA-rw, which mean that the shortest path distance measurement makes the result more stable than the random walk measurement.

5.4 Intensity

For the evaluation of intensity prediction, we use tEqu-mult, tExp-mult and eExp-mult instead of tEqu, tExp, and eExp as baselines. It is because the tEqu, tExp and eExp, like the standard IC model, do not allow multiple actions being performed by the same node. To construct better baselines, we try to capture the relationship between the coverage and intensity. Figure 7 shows our observation on the relationship.

For each dataset, we calculate coverage and intensity for all the intervals in the training time, and plot each pair of coverage and intensity in this Figure 7. In both figures, the values of coverage are illustrated on the X-axes, and the values of intensity are illustrated on the Y-axes. As shown in Figure 7, for the DBLP trends, the proportional function $y = 1.1215x$ fits the relationship of intensity and coverage quite well, while for the Twitter trends, the $y = 1.2969x$ fits the relationship.

Based on this observation, we add a multiple action factor to the three baselines and get three new baselines. Figures 8(a) and 8(b) show the improvement of tEqu-mult from tEqu on the DBLP and Twitter datasets respectively. For each trend, we show the predictions of intensity made by tEqu-mult and tEqu for in the first prediction interval (year 2005 for the DBLP trends and the 48nd week for the Twitter trends) as well as the true value. As illustrated in the figure, tEqu tends to make a prediction lower than the true value

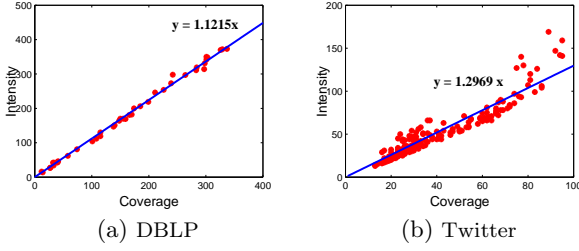


Figure 7: Relationship between Intensity and Coverage

because it does not allow multiple actions taken by each node. tEqu-mult makes a better prediction by adding a multiple factor.

Figure 9 shows the error ratios for intensity prediction. As shown in the figures, for all the algorithms, error ratios tend to increase as time progresses. Among the four algorithms, tEqu-mult has the highest error ratio for the DBLP dataset, while tExp-mult is the worst one for the Twitter dataset. DA-sp performs well on the DBLP dataset, but has a high error rate on the Twitter dataset. That may due to the relatively smaller graph diameter of the Twitter dataset. DA-rw has lower error ratios than other algorithms for most of the ranges of the X-axes for both datasets.

In Figure 10, we show the coefficient of variation for the intensity prediction. Notice that the result is similar to the coefficient of variation for coverage prediction in Figure 6. It is because the result of tExp-mult is proportional to the result of tExp, their coefficients of variation should be the same. Therefore, the curves for tExp-mult in Figure 10 are exactly the same as the curves for tExp in Figure 6. So are the curves for tEqu-mult in Figure 10 and curves for tEqu in Figure 6. For the DBLP dataset, our algorithms have lower coefficient of variation than the baselines. For the Twitter dataset, the coefficient of variation of tExp-mult drops as time progresses, as a result of its large and inaccurate prediction for intensity, while the curves for DA-rw, DA-sp and tEqu-mult are similar. For both of the datasets, the prediction of eExp is less stable than the other four algorithms.

5.5 Duration

We test both coverage-based duration and intensity-based duration. To calculate the duration, we use the coverage and intensity at the last observed interval (year 2004 for the DBLP dataset, the 47th week for the Twitter dataset) as thresholds. Since the length of the test time is limited, coverage or intensity for about half of the trends never drops below the thresholds. We make each algorithm predict whether the duration covers all the 5 prediction intervals, and report the accuracy of this prediction.

Table 2 shows the accuracy of the duration prediction. In the table, “C-D” and “I-D” are short for “Coverage-based Duration” and “Intensity-based Duration”. As shown in the table, DA-rw makes the best predictions of accuracy in three of the four cases. The accuracy of DA-sp and tExp is similar expect for the last case. tEqu has the lowest accuracy on the DBLP dataset, while eExp has lowest accuracy on Twitter dataset.

5.6 Case Study

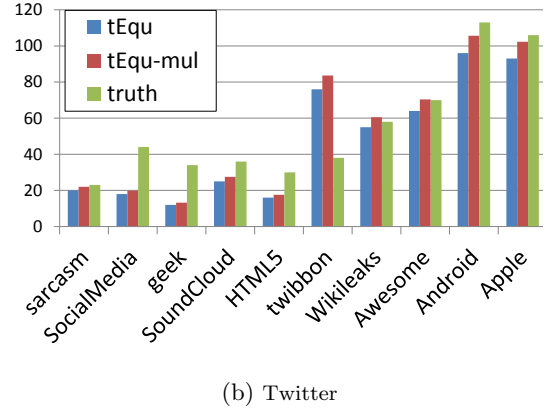
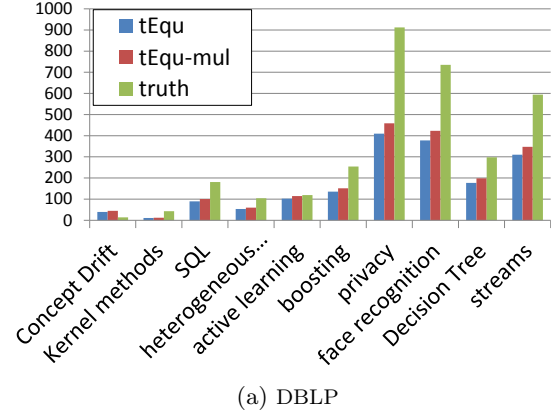


Figure 8: Comparison of tEqu and tEqu-mult

	DBLP		Twitter	
	C-D	I-D	C-D	I-D
DA-rw	0.9	0.9	0.8	0.5
DA-sp	0.8	0.9	0.6	0.5
tEqu(-mult)	0.6	0.5	0.4	0.5
tExp(-mult)	0.8	0.9	0.6	0.7
eExp(-mult)	0.7	0.7	0.3	0.4

Table 2: Accuracy of Duration Prediction

We select two trends for the case study: the trend for “android” in the Twitter network and the trend for “concept drift” in the DBLP network.

In Figure 11(a), we show the predicted and true coverage of the trend “android” in the last five weeks of 2011. The coverages predicted by DA-rw and DA-sp are close to the true value. Besides, only these two curves descend gently as the curve of true coverage. The coverage predicted by tExp quickly becomes much larger than the true value, while the coverage predicted by eExp is always very small. The coverage predicted by eExp fluctuates dramatically. If we choose $\theta = 86$ (the true coverage value in week 47), the coverage-based duration predicted by DA-rw, DA-sp will be 1 week, the same as the true value, while the duration predicted by three baselines will all be 0 week, since the predicted value falls below θ in the week 48.

In Figure 11(b), we show the predicted and true intensity of the trend “concept drift” in the years 2005-2009. All the curves keep increasing during the five intervals. But the

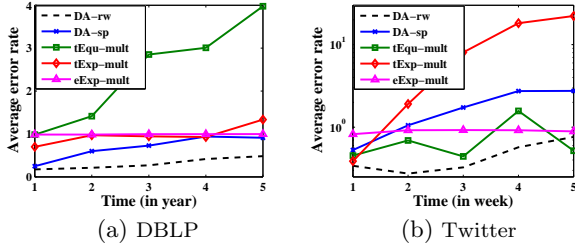


Figure 9: Error Ratio for Intensity Prediction

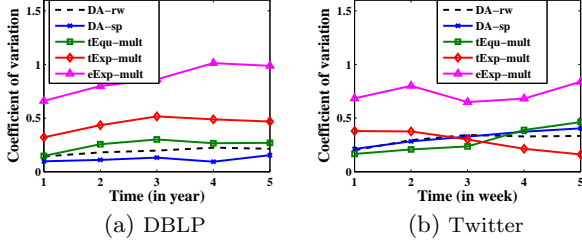


Figure 10: Coefficient of Variation for Intensity Prediction

intensity predicted by DA-rw and DA-sp is much closer to the true value than the value predicted by the baselines.

5.7 Variations of Parameters

Figure 12 shows the distributions of the parameters, mean life time τ and proportionality factor α , for DBLP data set. As shown in the figure, each of the parameters has a large variation over trends. For example, most of the trends have τ (mean lifetime of activeness decay) around 4 years, but some trends have very small τ which is less than 1 year. The large variations show that the cascade processes for trends are different from each other in nature. As a result, it is necessary to learn the model for each individual trend.

5.8 Summary and Discussion

We summarize the result as follows: For the prediction of coverage, both DA-rw and DA-sp have better performance than all the three baselines. For the prediction of intensity and duration, DA-rw works better than the baselines on both datasets. while DA-sp makes better predictions than baselines on the DBLP dataset. For all the evaluations, DA-rw has comparable or better results than DA-sp.

The two datasets we used are very different in many aspects. For example, the time granularity of the DBLP dataset is one year, while in the Twitter dataset the time granularity is one millisecond. The DA model can make accurate predictions on both of dataset. It shows that the DA model is practical for various applications.

According different applications, it may not always be necessary to make predictions on all of the three measures. As shown in Figure 7, the linear relationship between intensity and coverage is very significant in the DBLP dataset, comparing to that of the Twitter dataset. That is because the intensity of trends in the Twitter dataset can easily be spammed by a small fraction of users, but it is much harder to publish a paper than posting a tweet. Therefore, for applications on the DBLP network, the measures of intensity

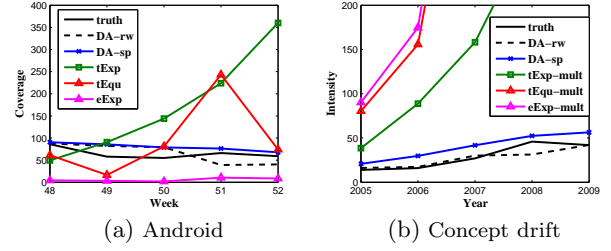


Figure 11: Case study on two trends in DBLP and Twitter dataset

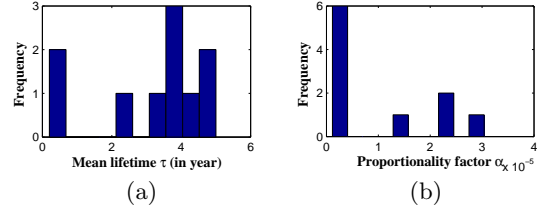


Figure 12: Distribution of Parameters

and coverage are roughly interchangeable. For the DBLP dataset, we may only need to predict one of them. Nevertheless, each of the three measures is useful for some applications, and it is desirable to propose a model that can capture all of them at the same time.

6. RELATED WORK

Information diffusion processes in social networks have been intensively studied in [9, 7, 13, 12, 4, 8, 10]. Most of this work considers trends in social network as the results of information diffusion processes. These papers focus on predicting user-level behaviors rather than the aggregated measures of trends. They usually model trends on a discretized time, which makes them not practical for predicting dynamic properties of trends. Independent Cascade (IC) model and its variants are the most widely studied models for information diffusion processes [9, 12, 4]. Most of this work assumes that the models are given as an input and try to solve the influence maximization problem on the model. The work in [18, 17, 7, 6] focuses on learning the diffusion probabilities of the cascade models. The work in [17] defines a variant of the IC model that models the diffusion delay as a random variable with an exponential distribution, and provides an inference algorithm for the model. The work in [20] proposes a microscopic social influence model, in which influence is modeled as “heat” that flows through the network. It is conceptually resemble to the activeness in our model. But the approaches are fundamentally different, and their model focuses on the prediction on each individual node.

Some existing work argues that the information diffusion inside social networks is not the only explanation of trends. Homophily is also considered to be important for the modeling of trends [19, 3, 1], and the external influence from outside the network is studied in [15].

Recently, some work studies real-life trends in online communities. Most of this work focuses on analyzing observed trends rather than predicting future trends, and does not make use of the structure of the social network. The work

in [16] studies the dynamic of trends on the Twitter community. The work in [13] analyzes the cascade behaviors on blogspace, and the work in [11] analyzes trends on news website and blogs. The work in [3] provides a technique to identify popular trends, which utilizes the structural information of social networks.

7. CONCLUSIONS

This paper studies trends in social networks. We identify coverage, intensity and duration as the three characteristics of a trend. Though the phenomenon of trends has been widely observed and studied, none of the previous models can capture all the three important aspects of trends. We proposed a Dynamic Activeness model for trends based on the novel concept of node activeness. The experimental result shows that the proposed DA model can predict trends more accurately than information diffusion models.

8. REFERENCES

- [1] A. Anagnostopoulos, R. Kumar, and M. Mahdian. Influence and correlation in social networks. In *KDD*, 2008.
- [2] S. Aral, L. Muchnik, and A. Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *PNAS*, 106(51):21544–21549, 2009.
- [3] C. Budak, D. Agrawal, and A. El Abbadi. Structural trend analysis for online social networks. *Proc. VLDB Endow.*, 4(10):646–656, 2011.
- [4] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, 2010.
- [5] W. Chen and Y. Wang. Efficient influence maximization in social networks categories and subject descriptors. In *KDD*, 2009.
- [6] L. Dickens, I. Molloy, J. Lobo, P.-C. Cheng, and A. Russo. Learning stochastic models of information flow. In *ICDE*, 2012.
- [7] A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.
- [8] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW*, 2004.
- [9] D. Kempe and J. Kleinberg. Maximizing the spread of influence through a social network. In *KDD*, 2003.
- [10] G. Kossinets, J. Kleinberg, and D. Watts. The structure of information pathways in a social communication network. In *KDD*, 2008.
- [11] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, 2009.
- [12] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen, and N. Glance. Cost-effective outbreak detection in Networks. In *KDD*, 2007.
- [13] J. Leskovec, M. Mcglohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs patterns and a model. In *SDM*, 2007.
- [14] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.
- [15] S. Myers, C. Zhu, and J. Leskovec. Information Diffusion and external influence in networks. In *KDD*, 2012.
- [16] D. M. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *WWW*, 2011.
- [17] K. Saito, M. Kimura, K. Ohara, and H. Motoda. Learning continuous-time information diffusion model for social behavioral data analysis. In *ACML*, 2009.
- [18] K. Saito, R. Nakano, and M. Kimura. Prediction of information diffusion probabilities for independent cascade model. In *KES*, 2008.
- [19] C. R. Shalizi and A. C. Thomas. Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods and Research*, 40(2):211–239, 2011.
- [20] T. Wang, M. Srivatsa, D. Agrawal, and L. Liu. Microscopic social influence. In *SDM*, 2012.
- [21] M. Zhao and M. Xie. On maximum likelihood estimation for a general non-homogeneous Poisson process. *Scandinavian journal of statistics*, 23(4):597–607, 1996.

APPENDIX

A. MAXIMUM LIKELIHOOD ESTIMATION

The joint probability density function is given by [21]:

$$f(T(S_{t_*}^v), |S_{t_*}^v|; \alpha, \tau) = e^{-\int_0^{t_*} r_v(t) dt} \cdot \prod_{t_i \in T(S_{t_*}^v)} r_v(t_i) \quad (9)$$

For simplicity's sake, we introduce $H_v(\cdot)$ and $h_v(\cdot)$ as Equations 7 and 8. Take Equations 9, 7 and 8 into Equation 4, we get:

$$\begin{aligned} L(\alpha, \tau) &= \prod_{v \in V} (e^{-\alpha H_v(t_*, \tau)} \cdot \prod_{t_i \in T(S_{t_*}^v)} \alpha h_v(t_i, \tau)) \\ &= \alpha^{|S_{t_*}|} \prod_{v \in V} e^{-\alpha H_v(t_*, \tau)} \cdot \prod_{(v_i, t_i) \in S_{t_*}} h_{v_i}(t_i, \tau) \end{aligned}$$

The log-likelihood function is then given by:

$$\begin{aligned} \log L(\alpha, \tau) &= |S_{t_*}| \log \alpha - \alpha \sum_{v \in V} H_v(t_*, \tau) \\ &\quad + \sum_{(v_i, t_i) \in S_{t_*}} \log(h_{v_i}(t_i, \tau)) \end{aligned}$$

By taking the partial derivative of $\log L(\alpha, \tau)$, we get the maximum-likelihood estimation of α and τ . Notice that in the above inference we assume that the ϵ in Equation 2 is small enough and negligible.