BEEP: a Bayesian perspective Early stage Event Prediction model for online social networks

ANONYMOUS

***Abstract*—In recent years, predicting future hot events in online social networks is becoming increasingly meaningful in marketing, advertisement, and recommendation systems to support companies' strategy making. Currently, most prediction models require long-term observations over the event or depend a lot on other features which are expensive to extract. However, at the early stage of an event, the temporal features of hot events and non-hot events are not distinctive yet. Besides, given the small amount of available data, high noise and complex network structure, those state-of-art models are unable to give an accurate prediction at the very early stage of an event. Hence, we propose two Bayesian perspective models to handle this dilemma. We first mathematically define the hot event prediction problem and introduce the general early stage event prediction framework, then modeled the five selected features into several continuous distributions, and present two Semi-Naive Bayes Classifier based prediction models, BEEP and SimBEEP, which is the simplified version of BEEP. Extensive experiments on real dataset have demonstrated that our model significantly outperforms the baseline methods.**

I. INTRODUCTION

Through the years, the event prediction problem, which refers to predicting whether an observed event will become hot in the future, has continuously drawn scholars' attention. Popular events are meaningful to many services and applications, such as marketing, advertisement and recommendation systems, which can assist companies' strategy making and boost their profit.

Recently, the online social networks, such as Twitter, Weibo, and YouTube, have gradually become an indispensable part of people's daily life. They provide people with a quick access to information, communication, entertainment and study. Twitter and Weibo, more importantly, have established a brand new platform for people to exchange information, share their opinions and express their emotions through 140 words short messages (for Chinese, 70 words) by posting the tweet, retweeting or commenting on others' tweets. All of these properties have made online social networks a perfect platform for researchers to study the problem of event prediction. More specifically, online social networks have the following characteristics.

- **Good Timeliness**: they are always the first place for an event or topic to appear [1].
- **Fast Transmission Rate**: the hot event will spread over the social network at an extremely high speed, which leads to the explosive growth of the popularity of an event or topic.
- **Low Dissemination Cost**: there are no restrictions on user's behavior and retweet only needs several clicks to accomplish, which also contributes to the fast transmission rate of the information.

These characteristics have further supported the research over event prediction.

Although scholars have proposed many powerful models for event prediction problem in online social networks, all of these models have the same limitation: unable to predict the popularity of an event at the very early stage. It is an intuitive idea to make the prediction as early as possible. This is very reasonable since the earlier the possible hot information is recommended to the users, the higher profit the company may gain. However, this is also a challenging task even in online social networks due to the following reasons. First, the frequency at which the events discussed by users always changes irregularly in online social networks. Many classic forecasting models based on time series analysis need a complete observation over a full evolution period of the time series, and they will not be able to handle such data with a high noise. Second, the structure of the online social network, which is actually a heterogeneous network [2], [3], [4], is complex and dynamic. Extracting features in such complex networks may pose high time complexity. However, the prediction problem requires high timeliness, which can possibly be affected by the feature extraction process. Third and most important, at the early stage when an event just emerges, the amount of the available information is highly limited and the temporal features, such as the sum of the discussion frequencies is indistinctive between hot events and non-hot events. Temporal features will no longer be able to support the prediction problem at the very early stage.

In order to overcome the above-mentioned challenges, the proposed model should have the following properties:

- **Effectiveness**: The prediction model should be able to handle the prediction problem accurately under the high noise with the limited amount of the observed data at the very early stage.
- **Timeliness**: The computational complexity of the prediction model should be controlled within an acceptable threshold to adapt to the requirement of huge and complex networks.

In this paper, jointly considering the three challenges, we propose an efficient and effective Bayesian perspective early stage event prediction model for online social networks to satisfy the above two properties. We formulate the event prediction problem as a classification problem and build a Bayesian perspective Early stage Event Prediction model, in

abbreviation, BEEP. We first study the distributions of the 5 selected features, including 3 temporal features and 2 assistive structural features and fit them into proper random variables. It is because only temporal features are not able to support the prediction problem at the early stage and network structure is also another indicator for the information dissemination process. As different variables may have dependencies between each other, we then utilize Semi-Naive Bayes Classifier to model the feature dependencies and further support the event prediction. Semi-Naive Bayes Classifier is intrinsically suitable to handle the prediction problem in online social networks. By modeling the features into random variables, we can successfully handle the high noise in social networks. Besides, as Semi-Naive Bayes Classifier is a generative model with simple network structure, its training and prediction process can be relatively fast. Most importantly, having combined the above 5 features, our model can successfully output the prediction result effectively and timely.

Our main contribution is to establish a powerful event prediction model for online social networks to handle the high noise and the data deficiency problem at the early stage of an event. It shows how to handle the feature selection, modeling and prediction even when the data is extremely deficient at the very early stage. We conducted extensive experiments to demonstrate the effectiveness of our model.

The rest of the paper is organized as follows. In Section II, we will briefly review the related works. In Section III, we formally formulate the early stage event prediction problem. Next, our Bayesian perspective prediction model will be introduced in Section V and extensive experiment results will follow in Section VI.

II. Related Works

Through the years, social networks have always attracted the scholars' attention, especially the prediction of the social network contents, including predicting the popularity of a single tweet, of an event and of a topic.

One of the pioneers is Suh et al [5]. They jointly considered the content and user profile of a microblog, then designed a model based on PCA and Generalized Linear Model (GLM) to find what may influence the repost of a microblog. Later scholars also adopted the method of combining feature and general machine learning techniques to handle the problem of prediction in social networks.

Later, various feature-based methods have been developed. Some features are directly related to the content of the text. Chenliang Li et al. [6] developed an event detection model taking the sentiment analysis into consideration. In [7], Naveed el al. proposed a model to predict the retweet behavior using Natural Language Processing techniques. In addition, some other researchers have also studied the relationship between the sentiment and the popularity of a tweet [8]. They indicated that negative sentiment has higher relation with the popularity of the tweets. There are many other content related feature based methods, such as using the emotional divergence [9], tweet length [10], number of hashtags [5] and number of mentions [11].

Some other feature based methods adopt other kinds of features related to the users. [12] considered the number of repost of the author in the four weeks before the current time as an indicator of popularity. [13] used the number of follower and friends of the author as another feature, since with higher number of audience will lead to higher possibility a tweet will be reposted. Besides directly consider the author, [5], [13] considered the property of the reposter. They took the mean value of the number of the reposts of the reposters in a given time period and the mean value of their friends and follower into account.

Another important factor for information dissemination in social networks is the structure of the network, in other words, the structure of the friendship network. Meeyoung Cha and et al. [14] indicated that whether a user will be influenced depends more on the diversity of his friendship network. More specifically, the higher number of the connected components of a user's friendship network will increase the possibility a user will be influenced. Similarly, Wenzheng Xu et al. [15] proposed a method to find Top-K influential users under a structural diversity model, which actually depends on the diversity of the connected component in a friendship network. [16], [17] also suggested that using structural features can significantly improve the prediction accuracy, but evidences were given [18], [17] that structural features are not as effective as temporal features. Still, structural features are widely used in prediction tasks. [19] made use of the reciprocity of the network, number of the connected components, maximum size of the connected component, average authority of authors and link density in the network to establish their prediction model for tweet popularity; [20] used the clustering coefficient network; [21] used the authorities of the authors of the original post; [16] used the number of the connected components in the network; [22] used the number of edges from early adopters to the entire graph as a strong indicator of the dissemination possibility at the early stage.

In addition to those types of methods, researchers have shown that temporal feature based methods actually have the best performance [23], [24], [17]. Szabo et al. [25] also suggested that temporal features alone can reliably predict the future popularity. [20] used the time between the ith repost and the $(i-1)$th repost to support prediction; mean time between reposts for the first half (rounded down) of the reposts is used in [24], and mean time between reposts for the last half (rounded up) of the reposts, coeffcients of polynomial curve fitting and symbolic sequences were used in [23]; sum of the frequencies of the first 9 time window(8 hours for each) a topic is discussed by users, average rate of change and standard deviation were used in [26] to predict whether and when a topic will become hot in social networks.

The majority of the previous works aim to predict the popularity from a fixed corpus during a long time period, but they are unable to handle the timeliness of predicting the popularity of the emerging events. Qi Dang et al. [27] proposed

TABLE I: Notations

E_i	the ith event observed
$\mathbf{E}$	the collection of events
$\mathbf{E}_h$	the collection of hot events
$\mathbf{E}_n$	the collection of non-hot events
$f_i^{(j)}$	the frequency event E_i was discussed at in time window j
S_i	the sum of the frequencies of E_i
α	hot event threshold
X_i	feature vector of event E_i
S_i	temporal feature of event E_i
V_i	velocity feature of event E_i
A_i	acceleration feature of event E_i
T_i	tweet chain feature of event E_i
C_i	community feature of event E_i
H_1	hot event
H_0	non-hot event
Z_i	joint distribution of V_i and A_i

a Dynamic Bayesian Network based model to detect events at the very early stage in micro-blogging networks. Here, they indicated that the temporal features are still indistinctive at the very early stage when an event just emerges, and concentration shoul be placed on other kinds of features. Hence, in their model, structural features and user features are taken into consideration, and they modeled them into a Dynamic Bayesian Network to predict when the event will emerge. However, since they discretize the variables and ignored the temporal features, which could be the most critical feature in the prediction problem, the performance of their model is doubtable.

In order to predict the whether an observed event will become popular at the very early stage, we jointly consider the diverse features and the timeliness requirement of the task and proposed our prediction model, which will be introduced in the following parts of this paper.

III. Problem Statement

A. Problem Formulation

In this section, we will introduce the formal definitions of the event prediction problem in online social networks.

First, we consider the definition of an event. Similar to many existing works, we define an event as a collection of keywords, namely $E_i = \{k_1, k_2, \ldots, k_i\}$. In online social networks, we denote the collection of the events as $\mathbf{E} = \{E_i | E_i \text{ is an detected event}\}$. We can further seperate $\mathbf{E}$ into two sets:

- $\mathbf{E}_h$: the collection of the hot events
- $\mathbf{E}_n$: the collection of the normal event

where the hot events can be defined as:

Definition 1 (Hot Event)**.** *Let $S_i = \sum_{j=0}^{T} f_i^{(j)}$, where T is the total number of time windows and $f_i^{(j)}$ is the frequency at which event E_i is discussed during time window j. Given the hot event threshold α, we say event E_i is a hot event if $S_i > \alpha$*

We want to explain the frequency more specifically. Normally, when denoting the popularity of an item in online social networks, including events, topics, or even a single specific tweet, we refer to the sum of the tweets involved in this item. We could have multiple tweets talking about the same event posted on the social networks. However, one thing has to be emphasized is that a tweet could also contribute to multiple events. It is rather intuitive that one person could talk about several topics in one message. Moreover, in online social networks, some events may be so hot that the number of tweets involved would be much greater than other events, which can be hard for a statistical model to fit. Hence, in our model, we assume every tweet would contribute to multiple events, and we choose the frequency an event is discussed in the given time window to represent the popularity of an event rather than the general sum.

For event E_i, what we actually get is the time series data $\{f_i^{(i_0)}, f_i^{(i_0+1)}, \ldots, f_i^m\}$ since not every event emerges at the initial time window. If $\sum_{j=i_0}^{m} f_i^{(j)} < \alpha$, what we expect is to predict whether it would become hot in the near future based on its features, which could be denoted as a vector $X_i = \{x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\}$. Various features can be used here, such as the sum of the frequencies, the standard error and the growth rate.

We then formally define our hot event prediction problem.

Definition 2 (Hot Event Prediction)**.** *With the observed time series data $\{f_i^{(i_0)}, f_i^{(i_0+1)}, \ldots, f_i^{(m)}\}$ and the feature vector, $X_i = \{x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\}$ of an event i with $\sum_{j=j_0}^{m} f_i^{(j)} < \alpha$, we expect to testify whether $\exists(t^* > m)(\sum_{j=j_0}^{t^*} f_i^{(j)} \geq \alpha)$. If the proposition is true, then we say this event is a potential hot event.*

Although according to the definition above, people may easily refer to the time series methods which are frequently used and produced many influential works over the years, we want to note that it is totally not enough to handle the prediction problem at the very early stage when an event just emerges. Different from the state-of-art works, we aim to forecast whether an event would become hot in the near future. However, at that special time, it could be hard to distinguish the temporal features of a hot event to a non-hot event. Thus, we pay special attention to the other structural features to further support the prediction.

IV. Model Features

A. Feature Selection

Feature selection can be considered as one of the most important parts of the model since it will directly influence the performance of the prediction model.

There are four categories of the normally used features for the popularity prediction problem in online social networks, and we will first give a brief introduction to them in order.

- **Content Features**: As the the name suggests, content features refer to the textual information retrieved from the original messages in online social networks. Some content features can be directly derived from the text, such

as the number of the hash-tags, URLs or mentions [5]; other more complex features, such as sentiment, can be extracted using NLP (Natural Language Processing) techniques [6], [7].

- **User Features**: User features are the characteristics of the users that have participated in the diffusion of the specific information (about a topic, event, or simply an item). They are extracted either from the user's historical activities status or the user's profile, such as her follower count, the messeages she posts and the number of users she mentioned.
- **Structural Features**: Structural features are extracted from the users' friendship network. Normally, PageRank Algorithm [28] is used to get those structural features of the network, including discovering the influential users and identifying connected components of the networks [29], [21], [30].
- **Temporal Features**: Temporal features are concerned with the repost time and count with a specific object in the online social networks (event, topic or simply a specific item). They can be further represented as the mean or standard deviation within a specific time period. Most of the popularity prediction problems in the online social network have utilized the temporal features.

Many experiment results have suggested that temporal features are the most effective type of the features for prediction problem in online social networks [18], [24], [31]. Szabo et al. [25] have also indicated that temporal features along can also perform the prediction reliably. However, in our scenario, the early stage event prediction problem for online social networks, temporal features alone are not sufficient.

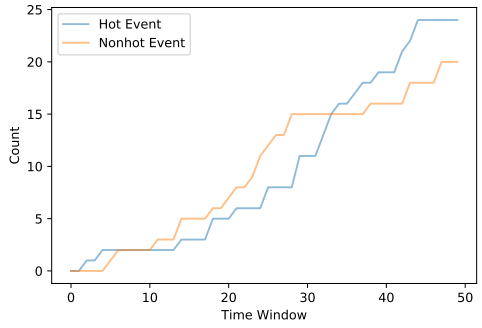


Fig. 1: Hot Event V.S. Non-Hot Event at Early Stage

Fig. 1 illustrates the evolvement of two time series within the first 50 time windows; one of them is a hot Event after a period of time, while another is not. However, we can find in Fig. 1, the temporal trend of the non-hot event appears also likely to be a hot event at an early stage. Actually, at the very early stage, especially when earlier than the first 50 time windows, we can hardly extract distinguishing temporal features for the hot event and the non-hot event, and the prediction accuracy would be greatly damaged due to this reason.

Actually, structural features are also shown to be effective. Some researches [14], [15] indicate that the diversity of the network structure contributes more to the information diffusion process in online social networks.

Another point of the feature selection is that the time complexity of the feature extraction process should be controlled within an acceptable threshold. When it comes to the structural features, some of them may be too hard to extract, since they always involve heavy calculations with the graph structure, especially the identifying of the influential users. However, our goal does not allow us to bear such heavy calculations. With the growing of the graph, the computational cost will increase geometrically, and it is necessary to make a trade off between the accuracy and the time complexity.

Hence, in our model, we consider the choice of 5 simple features that both utilized the temporal and structural characteristic of an event in online social networks within a specific slices of time windows. More specifically, in our experiment, we divide a day into several time windows with equal size and observe the evolotuion of the event for 8 continuous time windows. Then we extract the following 5 features correspondingly from the observed data of event E_i. Here, to keep the notation clean, we set the start time window of the event as 1.

- **Sum**: the sum of the discuss frequencies of the event in the observed 8 time windows. We can write it down as $S_i = \sum_{j=1}^{8} f_i^{(j)}$.
- **Velocity**: the velocity is actually the average rate of change of the sum. Here the velocity have two attributes: the first one is the velocity of the first 4 time windows, and the second one is the velocity of the following 4 time windows. In other words, $V_i = \left(\frac{\sum_{j=1}^{4} f_i^{(j)}}{4}, \frac{\sum_{j=5}^{8} f_i^{(j)}}{4} \right)$.
- **Acceleration**: the acceleration is the average rage of change of the velocity, namely $A_i = \frac{V_i^{(2)} - V_i^{(1)}}{4}$, where $V_i^{(j)}$ denote the j^{th} attibute of V_i.
- **Tweet Chain**: this feature is the maximum length of the retweet chain of all the tweets related with this event. Sometimes, a tweet will be retweeted by people since its content is attractive or its poster is very influential in the online social network. It can approximately represent the existence of the influential user which can significantly enhance the dissemination of the information. We can extract this feature by monitoring the retweet status of the network.
- **Community**: this feature refers to the number of the communities in the friendship network. Friendship network is built from the follow relationships among the users, and in the friendship network of an emerging event, the number of the communities will greatly enrich the way the information spread. Here, we refer to some previous works to extract this feature, and will introduce it in the following part of this paper.

Jointly considering the above 5 features, we can build our early stage event prediction model which will be introduced in the latter part.

B. Feature Extraction

As mentioned above, we select 5 simple features in order to control the time complexity within an acceptable threshold, and the next step is to extract them correspondingly. More specifically, since the extraction of other three features is quite simple, we focus on the extraction of the **Tweet Chain** and the **Community**.

The extraction of the **Tweet Chain** is very easy. We track the status of every tweet, including its status ID and its original status ID (the status ID of the tweet it retweets). Every retweet will increase the retweet chain of the original tweet by 1, and that's how a tweet chain is formed. After the observations over 8 time windows, we can find the max length of the tweet chain, and get this feature.

Compared with other features, the extraction of **Community** is relatively complex. Over the years, community detection in divergent networks have continuously drawn reserachers attention [32], [33], [34]. Community detection algorithms often pose high time complexity, especially with the development of the computational ability of the computer. However, since in our scenario, we hope to extract the feature as fast as possible to satisfy the timeliness requirement of the real-time prediction, we would rather sacrifice the accuracy in order to get low computational complexity. Hence, we choose the Louvain Method [35] to exploit the community detection, and calculate the number of the returned communities.

C. Feature Modeling

According to the histograms, we find that the features have the following characteristics conditioned on whether the event is hot or not when assumes they are continuous to variables:

- The distribution of the **Sum** feature seems best to fit the Beta Distribution.
- The distribution of the **Velocity** features seem best fit the Gamma Distribution.
- The distribution of the **Acceleration** feature seems best to fit the Gaussian Distribution.
- The distribution of the **Tweet Chain** feature seems best to fit the Gamma Distribution.
- The distribution of the **Community** feature seems best to fit the Gamma distribution, too.

Although we have concluded the seemingly distributions for the variables here, in the following parts of the thesis, we have to consider the joint distributions of the variables in order to capture the dependencies between the variables.

V. BEEP: Bayesian perspective Early statge Event Prediction model

In this section, we present our Bayesian Network based event prediction model.

A. BEEP: Network Structures

We first present the network structure of our prediction model, BEEP (Bayesian network based Early stage Event Prediction model).

BEEP is formulated as a Semi-Naive Bayes Classifier [36], which is a kind of Bayesian Network Classifier as well as an improved Naive Bayes Classifier with a looser constriant. In Semi-Naive Bayes Classifier, the conditionally independent assumption of the variables is loosened: there can be edges, in other words, the dependencies, among some of the variables.

Fig. 5 illustrated the network structure of BEEP. In our model, node S is for the **Sum** feature, and similarly, V, A, T and C are for the **Velocity**, **Acceleration**, **Tweet Chain** and **Community** features. S, T and C is conditionally independent with each other, while A is also dependent on V. This is an intuitive dependency since the acceleration is determined by the velocity, and this is what makes our model the Semi-Naive Bayes Classifier. The reason for introducing this dependency is that even with the same acceleration, the difference of the velocity could lead to different possibility. For example, two event may have the same acceleration, but the one with a higher velocity may have a higher possibility to become a hot event in the future.

Here, the double lined circle, V is called the composite node. The composite node is the composition of a set of traditional nodes with similar distributions. The purpose of introducing composite nodes is to reduce the learning time and spatial complexity of the model [37], [38]. Hence, we actually have $V = [V_1, V_2]$.

The network is very similar to traditional Semi-Naive Bayes Classifier, except that BEEP network contains one composite node. The joint distribution of the network is given by

$$\begin{aligned} &P(H, S, V, A, T, C) \\ &= P(H)P(S|H)P(V|H, S)P(A|H, V)P(T|H)P(C|H) \end{aligned} \tag{1}$$

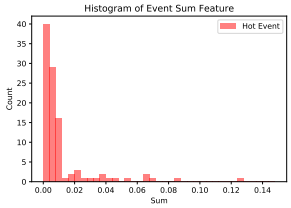
Besides, although temporal features, especially the **Sum** feature is a strong evidence for the hot event prediction problem, it can be still useless sometimes since in the very early stage of an event, the **Sum** feature of the hot event and the non-hot event is not that distinguishing. We also developed another simplified version of BEEP, namely SimBEEP.

As shown in Fig. 6, SimBEEP discarts the **Sum** feature, only the other 4 features remains. In later experiment part, we will show that under some special circumstances, SimBEEP can also output a satisfiable result. The joint distribution of the SimBEEP network is as follows

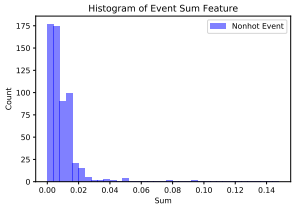
$$P(H, V, A, T, C) = P(H)P(V|H)P(A|H, V)P(T|H)P(C|H) \tag{2}$$

B. BEEP: Model Learning

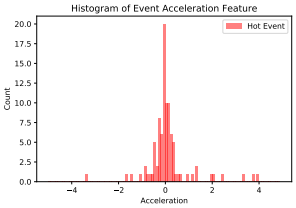
We then go into the details of the learning algorithm for BEEP. As a full generative model, the learning process of



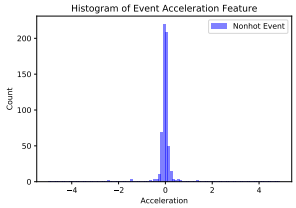
(a) Hot Event Sum



(b) Non-Hot Event Sum

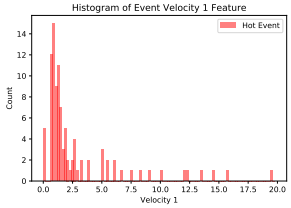


(c) Hot Event Acceleration

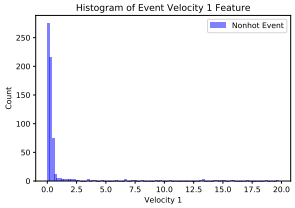


(d) Non-Hot Event Acceleration

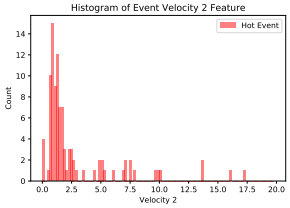
Fig. 2: Histograms of Sum and Acceleration Feature



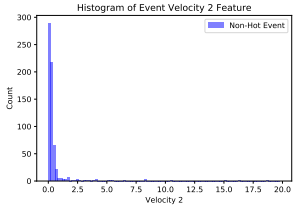
(a) Hot Event Velocity 1



(b) Non-Hot Event Velocity 1

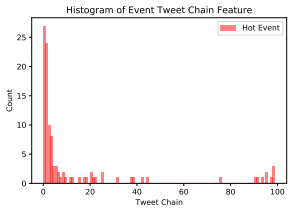


(c) Hot Event Velocity 2

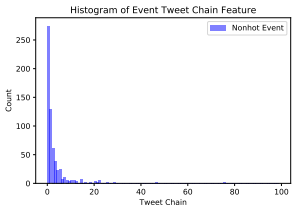


(d) Non-Hot Event Velocity 2

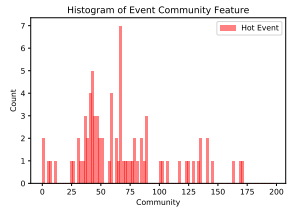
Fig. 3: Histograms of Velocity Feature



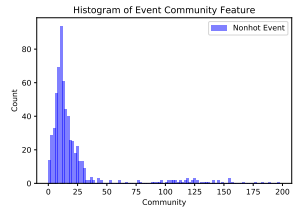
(a) Hot Event Tweet Chain



(b) Non-Hot Event Tweet Chain



(c) Hot Event Community



(d) Non-Hot Event Community

Fig. 4: Histograms of Tweet Chain and Community Feature

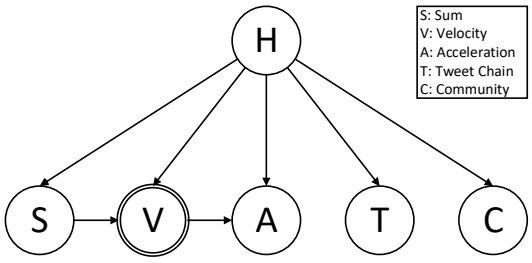


Fig. 5: Network Structure of BEEP

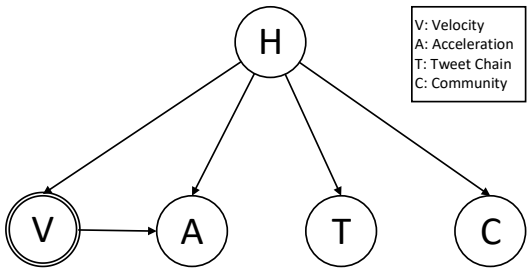


Fig. 6: Network Structure of SimBEEP

BEEP aims to maximize the likelihood of the joint distribution of the network. More specifically, we aim to maximize **Equation** 1. To do this, given the observations of the past result, including the temporal and feature evolution of the observed events, we aim to maximize the likelihood of their joint distribution.

First, we set the parameters for the network. As mentioned in the previous section, we assume the features S, V_1, V_2, A, T, C follow Beta Distribution, Gamma Distribution, Gamma Distribution, Gaussian Distribution, Gamma Distribution and Gamma Distribution accordingly. To keep the notation clean, we denote the feature vector as $X = \langle S, V_1, V_2, A, T, C \rangle$. Next we assume that H_1 denotes the event is hot, and H_0 denotes the event is not hot, and we can further have the following distributions: $p(S|H_1) = Beta(a_s^1, b_s^1)$, $p(V_1|H_1) = Gamma(c_{v_1}^1, d_{v_1}^1)$, $p(V_2|H_1) = Gamma(c_{v_2}^1, d_{v_2}^1)$, $p(A|H_1) = \mathcal{N}(\mu_a^1, \sigma_a^1)$, $p(T|H_1) = Gamma(c_t^1, d_t^1)$ and $p(C|H_1) = Gamma(c_c^1, d_c^1)$ for H_1; $p(S|H_0) = Beta(a_s^0, b_s^0)$, $p(V_1|H_0) = Gamma(c_{v_1}^0, d_{v_1}^0)$,

$p(V_2|H_0) = Gamma(c^0_{v_2}, d^0_{v_2})$, $p(A|H_0) = \mathcal{N}(\mu^0_a, \sigma^0_a)$, $p(T|H_0) = Gamma(c^0_t, d^0_t)$ and $p(C|H_0) = Gamma(c^0_c, d^0_c)$ for H_0.

In addition, to calculate $p(V|S, H_1)$ and $p(V|S, H_0)$, we apply the Bayesian Rules. Take $p(V|S, H_1)$ as an example.

$$p(V|S, H_1) = \frac{p(V_1, V_2, S|H_1)}{p(S|H_1)} \tag{3}$$

Similarly, to calculate $p(A|V, H_1)$ and $p(A|V, H_0)$, we apply the Bayesian Rules again. Take $p(A|V, H_1)$ as an example.

$$p(A|V, H_1) = \frac{p(V_1, V_2, A|H)}{p(V_1, V_2|H_1)} \tag{4}$$

Since V_1 and V_2 are conditionally independent with each other, we further have

$$\begin{aligned} & p(A|V, H_1) \\ = & \frac{p(V_1, V_2, A|H)}{p(V_1, V_2|H_1)} \\ = & \frac{p(V_1, V_2, A|H_1)}{p(V_1|H_1)p(V_2|H_1)} \end{aligned} \tag{5}$$

To calculate $p(V|S, H_1)$ and $p(A|V, H_1)$, we have to get $p(S, V_1, V_2|H_1)$ and $p(V_1, V_2, A|H_1)$. These multivariate distributions captures the inner relationships among variables. To do this, we make following assumptions.

For $p(S, V_1, V_2|H_1)$, we generalize V_1 and V_2 and model it using a Dirichlet Distribution. More specifically, let $V_1' = \frac{T}{T+1}V_1$, $V_2' = \frac{T}{T+1}V_2$ and $S' = \frac{T}{T+1}S$, where T is the observation time window, which in our settings, equals to 8. By doing so, we may have

$$\begin{aligned} & S' + V_1' + V_2' \\ = & \frac{T}{T+1}S + \frac{T}{T+1}V_1 + \frac{T}{T+1}V_2 \\ = & \frac{T}{T+1}S + \frac{T}{T+1}(V_1 + V_2) \\ = & \frac{T}{T+1}S + \frac{T}{T+1} * \frac{S}{T} \\ = & S \in [0, 1] \end{aligned} \tag{6}$$

which satisfies the constriant of the Dirichlet Distribution. Hence, we can model S', V_1' and V_2' into a Dirichlet Distribution. Let $\Delta = [S', V_1', V_2']$, we have $\Delta \sim Dir(\delta)$, where δ is a parameter vector with 3 dimensions.

For $p(V_1, V_2, A|H_1)$, we assume that $p(V_1, V_2, A|H_1)$ will follow a Multivariate Gaussian Distribution. It is actually an approximation to the combinations of Gamma Distributions and Gaussian Distributions. More Specifically, Let $Z = [V_1, V_2, A]$, we have $Z \sim \mathcal{N}(\mu^1_z, \Sigma^1_z)$, where μ is the mean vector and $\mathbf{\Sigma}$ is the covariance matrix of this Multivariate Gaussian Distribution Z.

Now, for a feature vector $X_i = \langle S^i, V^i, A^i, T^i, C^i \rangle$ from class H_1, we have

$$\begin{aligned} & p(X_i, H_1) \\ = & p(H_1)p(S^i|H_1)p(V^i|S^i, H_1)p(A^i|H_1, V)p(T^i|H_1)p(C^i|H_1) \\ = & p(H_1)p(S^i, V^i|H_1)\frac{p(A^i, V^i|H_1)}{p(V^i|H_1)}p(T^i|H_1)p(C^i|H_1) \\ = & \beta Dir(S^i, V_1^i, V_2^i|\delta^1)\mathcal{N}(A^i, V_1^i, V_2^i|\mu^1_z, \Sigma^1_z) \\ & /(Gamma(V_1^i|c^1_{v_1}, d^1_{v_1})Gamma(V_2^i|c^1_{v_2}, d^1_{v_1})) \\ & \cdot Gamma(T^i|c^1_t, d^1_t)Gamma(C^i|c^1_c, d^1_c) \end{aligned} \tag{7}$$

For class H_0, it is similar, so we will not list it in detail.

To keep the notation clean, let $\omega_1 = \langle \delta^1, \mu^1_z, \Sigma^1_z, c^1_{v_1}, d^1_{v_1}, c^1_{v_2}, d^1_{v_2}, c^1_t, d^1_t, c^1_c, d^1_c \rangle$ and $\omega_0 = \langle \delta^0, \mu^0_z, \Sigma^0_z, c^0_{v_1}, d^0_{v_1}, c^0_{v_2}, d^0_{v_2}, c^0_t, d^0_t, c^0_c, d^0_c \rangle$ and

$$t_i = \begin{cases} 1, & \text{if } E_i \in \mathbf{E}_h \\ 0, & \text{if } E_i \in \mathbf{E}_n \end{cases} \tag{8}$$

Now we can train the parameters on training set D where $|D| = k$ with the likelihood function

$$\begin{aligned} & p(D|\beta, \omega_0, \omega_1) \\ = & \prod_{i=1}^{k} [\beta Dir(S^i, V_1^i, V_2^i|\delta^1)\mathcal{N}(A^i, V_1^i, V_2^i|\mu^1_z, \Sigma^1_z) \\ & /(Gamma(V_1^i|c^1_{v_1}, d^1_{v_1})Gamma(V_2^i|c^1_{v_2}, d^1_{v_1})) \\ & \cdot Gamma(T^i|c^1_t, d^1_t)Gamma(C^i|c^1_c, d^1_c)]^{t_i} \\ & \cdot [(1-\beta)Dir(S^i, V_1^i, V_2^i|\delta^0)\mathcal{N}(A^i, V_1^i, V_2^i|\mu^0_z, \Sigma^0_z) \\ & /(Gamma(V_1^i|c^0_{v_1}, d^0_{v_1})Gamma(V_2^i|c^0_{v_2}, d^0_{v_1})) \\ & \cdot Gamma(T^i|c^0_t, d^0_t)Gamma(C^i|c^0_c, d^0_c)]^{1-t_i} \end{aligned} \tag{9}$$

The goal of the parameter training is to get the parameters $\langle \beta, \omega_0, \omega_1 \rangle$ by solving the following problem

$$\max \log p(D|\beta, \omega_0, \omega_1) \tag{10}$$

More specifically, since the parameters of the log-likelihood of **Tweet Chain** and **Community** features are obvioursly independent with others, we can simply maximize them to maximize our objective function **Equation** 10. In other words, we simply fit them into random variables by solving the following two problems:

$$\max \log \prod_{i=1}^{k} Gamma(T^i|c^1_t, d^1_t)^{t_i} Gamma(T^i|c^0_t, d^0_t)^{1-t_i} \tag{11}$$

$$\max \log \prod_{i=1}^{k} Gamma(C^i|c^1_c, d^1_c)^{t_i} Gamma(C^i|c^0_c, d^0_c)^{1-t_i} \tag{12}$$

As for the rest part of the objective function, we observe that they all involve variable V_1 and V_2. In previous sections, we assume $p(V_1|H_1) \sim Gamma(c^1_{v_1}, d^1_{v_1})$ and $p(V_1|H_0) \sim Gamma(c^0_{v_1}, d^0_{v_1})$. Let $e^1_{v_1} = \mathbb{E}(V_1|H_1)$ and $e^0_{v_1} = \mathbb{E}(V_1|H_0)$, therefore we can further have $e^1_{v_1} = \frac{c_{v_1}}{d_{v_1}}$ and

$e^0_{v_1} = \frac{c_{v_0}}{d_{v_0}}$ according to the property of Gamma Distribution. Similarly, we can get $e^1_{v_2}$ and $e^0_{v_2}$. Here, we have assumed $\Delta \sim Dir(\delta) = Dir(\delta_s, \delta_{v_1}, \delta_{v_2})$, then we have

$$\begin{cases} (e^1_{v_1} - 1)\delta_{v_1} + e^1_{v_1}\delta_{v_2} + e^1_{v_1}\delta^1_s = 0 \\ (e^1_{v_2} - 1)\delta_{v_2} + e^1_{v_2}\delta_{v_1} + e^1_{v_2}\delta^1_s = 0 \end{cases} \tag{13}$$

By solving these equations, we can get

$$\begin{aligned} \delta^1_{v_1} &= \frac{e^1_{v_1}}{e^1_{v_1} + e^1_{v_2} - 1}\delta_s \\ \delta^1_{v_2} &= \frac{e^1_{v_2}}{e^1_{v_2} - e^1_{v_1} - 1}\delta_s \end{aligned} \tag{14}$$

Similarly, we can also get

$$\begin{aligned} \delta^0_{v_1} &= \frac{e^0_{v_1}}{e^1_{v_0} + e^0_{v_2} - 1}\delta_s \\ \delta^0_{v_2} &= \frac{e^0_{v_2}}{e^0_{v_2} - e^0_{v_1} - 1}\delta_s \end{aligned} \tag{15}$$

As for the Multivariate Gaussian Distribution, denoting $\mu^1_z = [\mu^1_{v_1}, \mu^1_{v_2}, \mu^1_a]$, we can easily have

$$\mu^1_{v_1} = e^1_{v_1} \text{ and } \mu^1_{v_2} = e^1_{v_2} \tag{16}$$

To keep the notation clean, let $\omega'_1 = \langle \delta^1, \mu^1_z, \Sigma^1_z, c^1_{v_1}, d^1_{v_1}, c^1_{v_2}, d^1_{v_2} \rangle$ and $\omega'_0 = \langle \delta^0, \mu^0_z, \Sigma^0_z, c^0_{v_1}, d^0_{v_1}, c^0_{v_2}, d^0_{v_2} \rangle$. Given the relationships shown by **Equation** 14, **Equation** 15 and **Equation** 16, the rest of the parameter learning problem can be shown as **Equation** 17.

$$\begin{aligned} & Q(\omega'_0, \omega'_1) \\ = \; & \arg\max \log(\beta \cdot (1-\beta)) \\ + \; & \sum_{i=1}^{k} t_i [\log Dir(S^i, V^i_1, V^i_2 | \delta^1) + \log \mathcal{N}(A^i, V^i_1, V^i_2 | \mu^1_z, \Sigma^1_z) \\ - \; & \log Gamma(V^i_1 | c^1_{v_1}, d^1_{v_1}) - \log Gamma(V^i_2 | c^1_{v_2}, d^1_{v_1})] \\ + \; & \sum_{i=1}^{k} (1 - t_i) [\log Dir(S^i, V^i_1, V^i_2 | \delta^0) \\ + \; & \log \mathcal{N}(A^i, V^i_1, V^i_2 | \mu^0_z, \Sigma^0_z) - \log Gamma(V^i_1 | c^0_{v_1}, d^0_{v_1}) \\ - \; & \log Gamma(V^i_2 | c^0_{v_2}, d^0_{v_1})] \end{aligned} \tag{17}$$

We use the Coordinate Ascent Algorithm to learn the optimized parameter set. Coordinate Ascent Algorithm is a derivative-free optimization algorithm, which does a line search along the coordinate direction at the current point in each iteration to find a local maximum of a function. More specifically, to keep the notation clean, let $\mathbf{y} = [y_1, y_2, \ldots, y_n]$ be the parameter set, in other words, $\mathbf{y} = [\omega'_0, \omega'_1]$ and $n = 10$. Then we propose our coordinate ascent learning algorithm in **Algorithm** 1.

In addition, β in the model can be calculated by $\beta = \frac{|E^D_h|}{|E^D_h| + |E^D_n|}$.

For SimBEEP, the learning process is very similar, so we will not go into the details of it.

Algorithm 1: Coordinate Learning Algorithm

```
Input: Training Dataset D;
Output: Estimated parameters y;
Initialize y with y^0 = [y_1^0, y_2^0, ..., y_n^0];
repeat
    foreach y_i^j ∈ y^j do
        y_i^{j+1} =
          arg max_{x∈R} Q(y_1^{j+1}, ..., y_{i-1}^{j+1}, x, y_{i+1}^j, ..., y_n^j);
        Replace y with
          [y_1^{j+1}, y_2^{j+1}, ..., y_i^{j+1}, y_{i+1}^j, ..., y_n^j];
until Converge;
Return y;
```

The whole process of parameter learning can be illustrated in **Algorithm** 2.

Algorithm 2: BEEP: Learning Algorithm

```
Input: Observed Event Set E with classified E_h and E_n;
       Observed tweets set S;
Output: Estimated parameters ω_0 and ω_1;
Step 1: Classify S into different event in E and further
        map them into different time windows;
Step 2: foreach E_i ∈ E do
    Extract the features, X = ⟨S, V, A, T, C⟩ accordingly;
Step 3: Maximizing Equation 10 with Equations 12
        and 18 by MLE
```

VI. Experiments

In this section, we will present our extensive experiment results over the performance of our model, BEEP and SimBEEP, against some traditional prediction models and some state-of-art models.

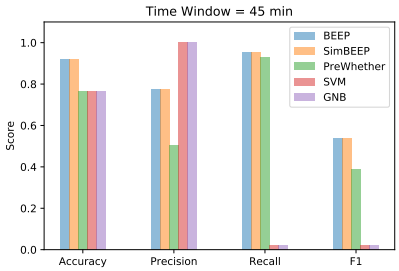
A. Dataset

The experiment dataset we use is a tweet dataset we crawled using *Tweepy* from 23rd, July, 2015 to 2nd, Dec, 2016. The dataset consists of 455253 tweets. Our dataset consists of two subset, *status* and *retweet*. *status* recorded the information of the tweet alone, and *retweet* recorded the retweet relationships of the tweets. For every item in the *status* dataset, there are following 5 attributes:

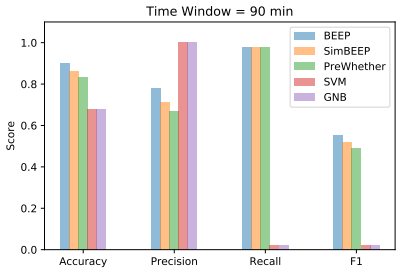
- *status_id*: The identifier of the tweet
- *created_at*: The post time of the tweet
- *text*: The content of the tweet
- *catch_time*: The time our crawler caught the tweet
- *user_id*: The ID of the user who first posted the tweet

As for the *retweet* dataset, there are following attributes:

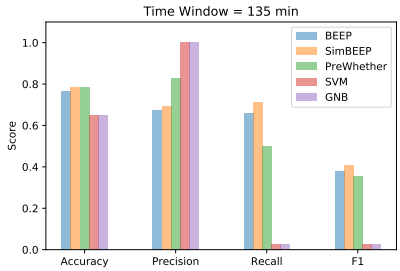
- *original_id*: The ID of the original tweet in the dataset
- *retweet_id*: The ID of the retweet status
- *retweet_user_id*: The ID of the user who posted this retweet



(a) Time Window = 45 min

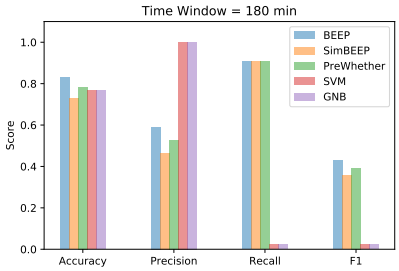


(b) Time Window = 90 min

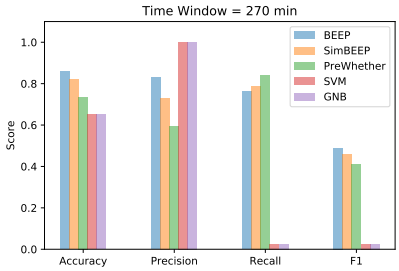


(c) Time Window = 135 min

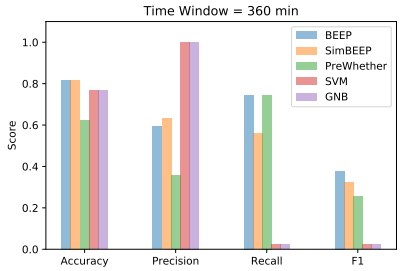
Fig. 7: Experiments Result with Different Time Windows Part 1



(a) Time Window = 180 min



(b) Time Window = 270 min



(c) Time Window = 360 min

Fig. 8: Experiments Result with Different Time Windows Part 2

For simplicity, we extract events based on hash-tags, and classify the tweets into different events. We then set threshold for the events, regarding events with a tweet count larger than 800 as hot events, and filtered the events with a tweet count smaller than 100. After these preprocessing to the data, we can get our refined dataset of 107 hot events and 608 non-hot events.

B. Baseline Methods

Many existing methods in time series modeling, recommendation systems and classification can be used to handle the problem of event prediction in online social networks. We compare our method with the following models of predicting whether an event will become hot at the very early stage.

- **Gaussian Naive Bayes:** Gaussian Naive Bayes Classifier is a well-known classifier which is a special kind of Naive Bayes. It assumes the variables is conditionally independent with each other and all follow the Gaussian Distribution. Gaussian Naive Bayes has many applications, the most famous one of which is the Naive Bayes Spam Filtering.
- **Support Vector Machine:** SVM is a famous supervised learning model invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. Through the years it has been well developed and has a large number of applications, such as text and hypertext categorization and image classification.
- **PreWhether:** PreWhether [26] is a model designed by Weiwei Liu et al. in 2015. It is also an probabilistic model to predict whether a topic would become hot in the future. It uses three features within 9 time windows(each equals to 8 hours) of a topic: sum, average speed of change and standard error, and model them using *Beta Distribution*, *Gaussian Distribution* and *Gamma Distribution*. Then it calculate the probability of the topic is hot and the topic is not hot given the feature vector, and output the result based on the comparison.

C. Experiment Results

In this subsection, we will introduce the experiment results of our model. More specifically, we compared the prediction performance of different models, including BEEP, SimBEEP, PreWhether, SVM and GNB. The metrics we use include *Accuracy*, *Precision*, *Recall* and *F1 Score*. The reason we choose these metrics is that the number of the non-hot events is substantially larger than the number of the hot events. Therefore, even if the model predict all test cases to be non-hot, it could still gain a high accuracy.

In our experiment, we tested the performance of the models with different time window length, in other words, with different observation time. The results can be shown as follows.

From Fig. 7 and Fig. 8, we can draw following conslusions:

- The accuracy is not reliable. For any model, the accuracy is very similar, since they have all produced enough

negative predictions which make up for a great part of the test set.

- The performance of SVM and GNB is very poor. Their precision is extremely high, both 100%, but recall is very low, which is because they have produced only a few number of positive predictions and they are all true positives. A better way to measure the performance is to refer to the F1 Score.
- The performances BEEP and SimBEEP are better than any other baseline methods. PreWhether can also present an acceptable prediction performance, but still worse than BEEP and SimBEEP.
- BEEP and SimBEEP act much better at the early stage prediction problem. In Fig. 7, we can see that the F1 Score of BEEP and SimBEEP is much higher than any other methods. When time window equals to 45 minutes, compared with PreWhether, BEEP and SimBEEP gained an improvement of 42.5%.
- SimBEEP acts similarly to BEEP when at the very early stage. In Fig. 7a, with the time window equals to 45 minutes, the performance of BEEP and SimBEEP is very close and in Fig. 7c, the performance of SimBEEP is even better than BEEP. This is because the temporal feature for the hot events and the non-hot events is not that distinctive.

VII. Conclusion

Predicting hot events in online social networks at an early stage is very meaningful for marketing, advertisement and recommendation systems.

We first mathematically formulate the early stage event prediction problem in online social networks, then give a brief introduction to Bayesian Networks and Bayesian Classifiers.

Next, we go into the details of our Bayesian perspective Early stage Event Prediction model, BEEP. We first discuss the available features for event prediction problem in online social networks, including content features, user features, structural features and temporal features, and focused on temporal features and structural features. Then we choose five features, including sum, velocity, acceleration, tweet chain and community, study their distributions and model them using Beta Distribution, Gaussian Distribution and Gamma Distribution. After that, combining them together, we present our model, BEEP and SimBEEP.

Substantial experiments on real dataset have proven that our model, BEEP and SimBEEP outperform the baseline methods and make much better predictions.

References

[1] F. Giummolè, S. Orlando, and G. Tolomei, "Trending topics on twitter improve the prediction of google hot queries," in *SocialCom*.

[2] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge University Press, 1994.

[3] R. A. Hanneman and M. Riddle, "Introduction to social network methods," 2005.

[4] A. Galeotti, S. Goyal, and J. Kamphorst, "Network formation with heterogeneous players," *Games and Economic Behavior*, 2006.

[5] B. Suh, L. Hong, P. Pirolli, and E. H. Chi, "Want to be retweeted? large scale analytics on factors impacting retweet in twitter network," in *SocialCom*, 2010.

[6] C. Li, A. Sun, and A. Datta, "Twevent: Segment-based event detection from tweets," in *CIKM*, 2012.

[7] N. Naveed, T. Gottron, J. Kunegis, and A. C. Alhadi, "Bad news travel fast: a content-based analysis of interestingness on twitter," in *WebSci*, 2011.

[8] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment in twitter events," *JASIST*, 2011.

[9] R. Pfitzner, A. Garas, and F. Schweitzer, "Emotional divergence influences information spreading in twitter," in *ICWSM*, 2012.

[10] S. Petrovic, M. Osborne, and V. Lavrenko, "RT to win! predicting message propagation in twitter," in *ICWSM*, 2011.

[11] Z. Luo, Y. Wang, X. Wu, W. Cai, and T. Chen, "On burst detection and prediction in retweeting sequence," in *PAKDD*, 2015.

[12] J. Yang and S. Counts, "Predicting the speed, scale, and range of information diffusion in twitter," in *ICWSM*, 2010.

[13] O. Tsur and A. Rappoport, "What's in a hashtag?: Content based prediction of the spread of ideas in microblogging communities," in *WSDM*, 2012.

[14] M. Cha, H. Haddadi, F. Benevenuto, and P. K. Gummadi, "Measuring user influence in twitter: The million follower fallacy," in *ICWSM*, 2010.

[15] W. Xu, W. Liang, X. Lin, and J. X. Yu, "Finding top-k influential users in social networks under the structural diversity model," *Info. Sci.*, 2016.

[16] D. M. Romero, C. Tan, and J. Ugander, "On the interplay between social and topical structure," in *ICWSM*, 2013.

[17] J. Cheng, L. A. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?" in *WWW*, 2014.

[18] S. Gao, J. Ma, and Z. Chen, "Effective and effortless features for popularity prediction in microblogging network," in *WWW*, 2014.

[19] X. Zhang, Z. Li, W. Chao, and J. Xia, "Popularity prediction of burst event in microblogging," in *WAIM*, 2014.

[20] L. Hong, O. Dan, and B. D. Davison, "Predicting popular messages in twitter," in *WWW*, 2011.

[21] Z. Ma, A. Sun, and G. Cong, "On predicting the popularity of newly emerging hashtags in twitter," *JASIST*, 2013.

[22] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts, "Everyone's an influencer: Quantifying influence on twitter," in *WSDM*, 2011.

[23] S. Kong, Q. Mei, L. Feng, F. Ye, and Z. Zhao, "Predicting bursts and popularity of hashtags in real-time," in *SIGIR*, 2014.

[24] B. Shulman, A. Sharma, and D. Cosley, "Predictability of popularity: Gaps between prediction and understanding," in *ICWSM*, 2016.

[25] G. Szabó and B. A. Huberman, "Predicting the popularity of online content," *Communications of the ACM*, 2010.

[26] W. Liu, Z. Deng, X. Gong, F. Jiang, and I. W. Tsang, "Effectively predicting whether and when a topic will become prevalent in a social network," in *AAAI*, 2015.

[27] Q. Dang, F. Gao, and Y. Zhou, "Early detection method for emerging topics based on dynamic bayesian networks in micro-blogging networks," *ESWA*, 2016.

[28] P. Lawrence, B. Sergey, M. Rajeev, and W. Terry, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[29] P. Cui, S. Jin, L. Yu, F. Wang, W. Zhu, and S. Yang, "Cascading outbreak prediction in networks: a data-driven approach," in *SIGKDD*, 2013.

[30] S. Wang, Z. Yan, X. Hu, P. S. Yu, and Z. Li, "Burst time prediction in cascades," in *AAAI*, 2015.

[31] A. Kupavskii, L. Ostroumova, A. Umnov, S. Usachev, P. Serdyukov, G. Gusev, and A. Kustarev, "Prediction of retweet cascade size over time," in *CIKM*, 2012.

[32] S. Fortunato, "Community detection in graphs," *CoRR*, 2009.

[33] H. Fani and E. Bagheri, "Community detection in social networks," *Encyclopedia with Semantic Computing and Robotic Intelligence*, 2017.

[34] F. D. Malliaros and M. Vazirgiannis, "Clustering and community detection in directed networks: A survey," *CoRR*, 2013.

[35] V. A. Traag, "Faster unfolding of communities: Speeding up the louvain algorithm," *CoRR*, 2015.

[36] F. Zheng and G. I. Webb, *Semi-Naive Bayesian Learning*, 2010.

[37] L. Jiang, H. Zhang, and Z. Cai, "A novel bayes model: Hidden naive bayes," *TKDE*, 2009.

[38] A. Fernández, M. Morales, C. Rodríguez, and A. Salmerón, "A system for relevance analysis of performance indicators in higher education using bayesian networks," *KAIS*, 2011.