Cost-effective online trending topic detection and popularity prediction in microblogging

Miao, Zhongchen; Chen, Kai; Fang, Yi; He, Jianhua; Zhou, Yi; Zhang, Wenjun

Published in:
ACM Transactions on Information Systems

DOI:
10.1145/3001833

Publication date:
2017

Document Version
Peer reviewed version

Cost-effective Online Trending Topic Detection and Popularity Prediction in Microblogging

Zhongchen Miao, Shanghai Jiao Tong University
Kai Chen, Shanghai Jiao Tong University
Yi Fang, Santa Clara University
Jianhua He, Aston University
Yi Zhou, Shanghai Jiao Tong University
Wenjun Zhang, Shanghai Jiao Tong University
Hongyuan Zha, Georgia Institute of Technology

Identifying topic trends on microblogging services such as Twitter and estimating those topics' future popularity have great academic and business value, especially when the operations can be done in real time. For any third party, however, capturing and processing such huge volumes of real-time data in microblogs are almost infeasible tasks, as there always exist API request limits, monitoring and computing budgets, as well as timeliness requirements. To deal with these challenges, we propose a cost-effective system framework with algorithms that can automatically select a subset of representative users in microblogging networks in offline, under given cost constraints. Then the proposed system can online monitor and utilize only these selected users' real-time microposts to detect the overall trending topics and predict their future popularity among the whole microblogging network. Therefore, our proposed system framework is practical for real-time usage as it avoids the high cost in capturing and processing full real-time data, while not compromising detection and prediction performance under given cost constraints. Experiments with real microblogs dataset show that by tracking only 500 users out of 0.6 million users and processing no more than 30,000 microposts daily, about 92% trending topics could be detected and predicted by the proposed system, and on average more than 10 hours earlier than they appear in official trends list.

CCS Concepts: •**Information systems** → **Data stream mining; Document topic models; Retrieval efficiency;** *Content analysis and feature selection; Information extraction;* Social networks;

Additional Key Words and Phrases: Topic detection, prediction, microblogging, cost

ACM Reference Format:
Zhongchen Miao, Kai Chen, Yi Fang, Jianhua He, Yi Zhou, Wenjun Zhang, and Hongyuan Zha. 2016. Cost-effective Online Trending Topic Detection and Popularity Prediction in Microblogging. *ACM Trans. Inf. Syst.* 0, 0, Article 00 (September 2016), 35 pages.
DOI: 0000001.0000001

This work was supported in part by the National Key Research and Development Program of China (2016YFB1001003), National Natural Science Foundation of China (61521062, 61527804), Shanghai Science and Technology Committees of Scientific Research Project (Grant No. 14XD1402100, 15JC1401700), and the 111 Program (B07022).
Author's addresses: Zhongchen Miao and Kai Chen (corresponding author) and Yi Zhou and Wenjun Zhang, Department of Electronic Engineering, Shanghai Jiao Tong University, {miaozhongchen, kchen, zy_21th, zhangwenjun}@sjtu.edu.cn; Yi Fang, Department of Computer Engineering, Santa Clara University, yfang@scu.edu; Jianhua He, School of Engineering and Applied Science, j.he7@aston.ac.uk; Hongyuan Zha, School of Computational Science and Engineering, Georgia Institute of Technology, zha@cc.gatech.edu.

DOI: 0000001.0000001

1. INTRODUCTION

Nowadays people's daily life across the world is closely tied to online social networks. Microblogging service (e.g. Twitter[1] and Weibo[2]), being one of the representative online social network services, provides a more convenient approach for everybody around the world to read news, deliver messages and exchange opinions than traditional media such as TV or newspaper. So huge quantities of users tend to post microposts in microblogging services and talk about the things they just witness, the news they just hear or the ideas they just think. In our paper, the topic of a micropost refers to a group of keywords (such as the name of items, news headline or thesis of ideas) in its content. And those semantically related microposts that talk about the same items, news and thoughts within a given time window are the set of microposts of that topic.

Commonly, microblogging social networks are filled with a large number of varied topics all the time. However if one topic is suddenly mentioned or discussed by an unusual amount of microposts within a relatively short time period, that topic is becoming trending in microblogging services. As microblogging are becoming the earliest and fastest sources of information, these trending topics shown on social networks are often referred to the breaking news or events that have or will have societal impacts in our real lives, such as first-hand report of natural or man-made disasters, leak of an excellent product, unexpected sports winners, and very controversial remarks/opinions.

Because of this, identifying trending topics in microblogging networks is receiving increasing interest among academic researchers as well as industries. Moreover, it will produce even more scientific, social and commercial values if the trending topics can be detected in real-time and those topics' future popularity can be predicted at early stages. For example, the early awareness of first-hand report of disasters can give rescuers more priceless time to reach the incident site and help more victims. Taking another example, higher predicted popularity and longer lifetime of "leaks of a specific product" trending topic can infer good sign of the product's future reputation and sales, so businessmen can be prepared to increase inventory and production.

In fact, microblogging service providers themselves such as Twitter and Weibo are publishing their official trending topic lists regularly. However unfortunately, these official lists are commonly delayed in publishing, small in size (Top-10 only), and not fully customized by individual user's preferences. They also do not contain topics' future popularity prediction function at all, so it is hard to tell how long a trending topic will last. More critically, there are concerns that some trending topics will never appear in these official lists that are subjected to the service provider's commercial considerations, or even government censorship policy [Chen et al. 2013a]. If we rely only on the official trends lists, some topics would most likely be missed or delayed by us. Therefore, business companies, organizations or even individuals are in bad need of a reliable online real-time trending topics detection and prediction system on microblogging services and other social networks, which can produce impartial, accurate and even customized results from a third party perspective.

Traditionally, online trending topics detection and prediction systems for microblogging comprises three major steps: 1) retrieving microposts and related information from microblogging website as much as possible; 2) detecting trends from the obtained microblog dataset; 3) predicting the detected topics' future popularity using the obtained microblog dataset. In this way, the performance in steps 2 and 3 largely depends on the quantity and quality of dataset retrieved in step 1. If at some time periods the sampled data source is biased [Morstatter et al. 2013; Morstatter et al. 2014], extra

[1]http://twitter.com, the top microblogging service worldwide.
[2]http://weibo.com, the top microblogging service in China.

large-scaled dataset at those time periods will be needed in order to remove the bias and get representative topic detection result for all times. However for any third party analyzers, the tremendous number of users in microblogging services and the ever-growing volumes of microposts pose significant challenges to the capturing and processing of such big scale dataset *in real time*. Although we are in an era of cloud-based services, it is still very challenging for any third party analyzers to acquire the full real-time data stream of the whole microblogging network in time, as microblogging services companies are heavily limiting and narrowing the API requests rate per account/IP to prevent large dataset collection[3]. Moreover, to get online detection results in real-time, it requires large resource budget on network bandwidth, IP address, storage, CPU and RAM in cloud-based services for collecting and processing these large scaled data in time. As a result, in practical usage the cost to obtain the fresh data and to detect and predict trending topics in real time should be seriously considered, and how to make a full use of the limited budget becomes a very important problem.

To deal with this difficulty, in this paper we propose a cost-effective detection and prediction framework for trending topics in microblogging services. The core notion of the framework is to select a small subset of representative users among the whole microblog users in offline based on historical data. Then in online the system will continuously track this small-sized subset of representative users, and utilize their real-time microposts to detect the trending topics and predict these topics' future popularity. Therefore, our proposed system can run under limited resources, which sharply reduces data retrieval and computation cost and not compromise on performance.

The idea of selecting a subset of users in a microblogging network for trending topics detection and prediction is somewhat similar to the question of putting alerting sensors in a city electricity or water monitoring networks that are analyzed by [Leskovec et al. 2007], in which any single point power failure in the electricity monitoring network can be covered by a nearby alerting sensor. For microblogs, a topic can be viewed to be covered by a user if he posts[4] or re-posts[5] a micropost that is related to that topic. Thus, both problems aim to decide where to put the "sensors" in the network, given constraints on monitoring cost. However, electricity or water monitoring is a single-coverage outbreak detection system, which means any abnormal signal detected by one sensor should be reported as an issue. In contrast, when a new topic appears in the microblogging network and it is covered by one or a few users simultaneously, it should not be treated as a trending topic until a certain coverage degree is reached indicating the topic is really trendy among the whole network. Therefore, the placement of such "sensors", i.e., selecting a proper subset of representative users in a microblogging network is a multi-coverage problem. And the selected users should be both effective in detecting trending topics and predicting those topic's future popularity.

It is worth pointing out that the representative users can be selected offline and fixed for a period of time in online usage. After some time the set of selected users can be updated by running user selection algorithms again using the newly collected training data. However as more microblog data is needed to run the user selection algorithm, in real-world usage the updating frequency need not be too high, or there will be no advantage in saving the data retrieving and processing cost.

The contributions of this paper can be summarized as follows:

(1) We treat online trending topics detection in microblogs as a multi-coverage problem: How to select a subset of users from all users in microblogging networks first,

[3]See Twitter API Rate Limits as an example, https://dev.twitter.com/rest/public/rate-limits
[4]Post a micropost is also called "Tweet" in Twitter
[5]Re-post a micropost is similar to "Forward" action in Email. It is also called "ReTweet" in Twitter

so that trending topics in the whole network-wide can be detected by monitoring only this subset of users and utilizing their posted/re-posted microposts. In this way, the real-time monitoring and computation costs can be greatly reduced.

(2) We formulate the subset user selection problem as mixed-integer optimization problem with cost constraints, topic coverage and prediction requirements. The topic coverage requirements can be customized for different individual topics or even different categories of topics, which enables the system to be more sensitive on high priority topics that users are more interested in.

(3) We integrate trending topics detection and their future popularity prediction into a single system. We propose efficient subset user selection algorithms for the optimization task by taking into account both detection and prediction accuracy. The experimental results show that the proposed algorithms outperform the state-of-art algorithms.

(4) We collect nearly 1.6 million real-world microposts in Weibo as the testbed, and evaluate performance of the proposed system and algorithms from several dimensions. The real-time testing evaluations show that using only 500 out of 0.6 million users in the dataset, our proposed system can detect 92% of the trending topics that are published in Weibo official trends. Besides, it can also detect and predict the topics much earlier than they are published in the official trends. We also release our source code and the collected dataset to public[6].

2. RELATED WORKS

In the work by [Allan 2002; Fung et al. 2007; Makkonen et al. 2004], a topic is defined as a coherent set of semantically related terms or documents that express a single argument. In this paper we follow the similar definitions, so microposts of one topic are those semantically related microposts/reposts that talk about the same items or news within a given time window. With fast development of online services in recent years, detection and analysis of topics over microblogging services and other websites with user generated contents are receiving more and more research interests.

One aspect of the research works focuses on emerging topic discovery in online contents, such as real-time earthquakes detecting over Twitter [Sakaki et al. 2010], "SigniTrend" emerging topics early detection with hashed significance thresholds [Schubert et al. 2014], real-time emergent topic detection in blogs [Alvanaki et al. 2012], "TwitterMonitor" trend detection system that treats bursting keywords as entry points [Mathioudakis and Koudas 2010], and two-level clustering methods [Petkos et al. 2014] that improves document-pivot algorithms in detection.

Some research papers track and analyse topics in longer time period. Memes are identified on a daily basis by [Leskovec et al. 2009], and their temporal variation is discussed by [Yang and Leskovec 2011]. The work by [Cataldi et al. 2010] also uses life cycle models of key words to detect emerging topics. Event evolutions are mined with short-text streams by [Huang et al. 2015].

Besides detecting emerging topics in social networks, some other works propose algorithms and techniques for analysing topic patterns and predicting trends online [Han et al. 2013]. The works by [Myers and Leskovec 2014; Naaman et al. 2011] discuss factors that affect topic trends and the bursty dynamics in Twitter, and hashtags in microposts are utilized by [Tsur and Rappoport 2012] for predicting topic propagation. Regression and classification algorithms are used by [Asur et al. 2011; Bandari et al. 2012] to predict news popularity in social media, temporal patterns evolution and state transition based topic popularity prediction methods are discussed by [Ahmed et al. 2013], and Gradient Boosted Decision Tree model for microposts show counts

[6]The dataset and source code is available at https://github.com/zcmiao/Topic

is proposed by [Kupavskii et al. 2013]. There are also other purposes of topic analysis in social networks. For example, the event classification approach with utilization of spatio-temporal information carried by microposts is proposed by [Lee et al. 2011], activity networks are used to identify interesting social events by [Rozenshtein et al. 2014], and events trends are modeled with cascades of Poisson processing by [Simma and Jordan 2010].

From all the above works, we see that various topic detection and analysis systems with different purposes, structures and algorithms are developed for social networks. However the above reported systems need to process all data streams and extract features from it to accomplish these tasks. This will generate very heavy communication and computation loads, which requires large time and resource costs, hence its performance is restricted in real-time operations.

Our proposed online microblogging trending topics detection and popularity prediction system differs from the above reported systems in that our system tracks only a very small number of microblog users that are pre-selected by our algorithms, and utilizes their real-time microposts to accomplish real-time detection and prediction tasks for trending topics in the whole microblogging network. One of the main contributions in this paper is how to select most representative subset of users that are vital in both detection and prediction. The concept is somewhat similar to influence maximization problem, which is to acquire maximum users or events cover under limited cost in social networks that first proposed by [Domingos and Richardson 2001], and further discussed by [Estevez et al. 2007; Narayanam and Narahari 2011; Pal and Counts 2011; Weng et al. 2010]. The influence maximization problem is formulated as an optimization task by [Kempe et al. 2003] and is proved to be NP-hard, then a greedy algorithm is proposed to solve it approximately. Sub-modular property of nodes selection problem in networks is found by [Leskovec et al. 2007], and faster greedy algorithms were developed by [Chen et al. 2009]. In our preliminary works [Chen et al. 2013b; Miao et al. 2015; Yang et al. 2014], the idea of selecting subset users for single tasks such as topic detection or topic prediction in microblogs are proposed. Some other greedy-based algorithm to get top-K influential users in social networks are proposed by [Du et al. 2013; Wang et al. 2010], and an algorithm is proposed by [Gomez-Rodriguez et al. 2012] to infer website influence in blogs. In addition, topic-specific influence and backbone structures in networks are studied by [Bi et al. 2014; Bogdanov et al. 2013].

In this paper, we extend the cost-effective framework and propose an integrated system for both trending topics detection as well as topic future popularity prediction in microblogs. Hence, subset users selection algorithm for joint detection and prediction are developed, and extensive experiments are carried out to evaluate joint multi-coverage and prediction performance under cost constraints.

3. OVERALL FRAMEWORK OF THE SYSTEM

In this section, we introduce the framework of our proposed system, and then explain each module in detail. The overall system structure is shown as Fig. 1, comprising the following 5 function modules:

— Module I: Training data retrieval;
— Module II: Subset microblog user selection;
— Module III: Real-time online data retrieval;
— Module IV: Online trending topic detection;
— Module V: Online trending topic popularity prediction.

In general, Module I and II run in offline, and they are mainly used for selecting representative users. Module I is used to obtain historical training dataset including microposts, microposts' propagation (re-posting) links and user profiles from microblog-

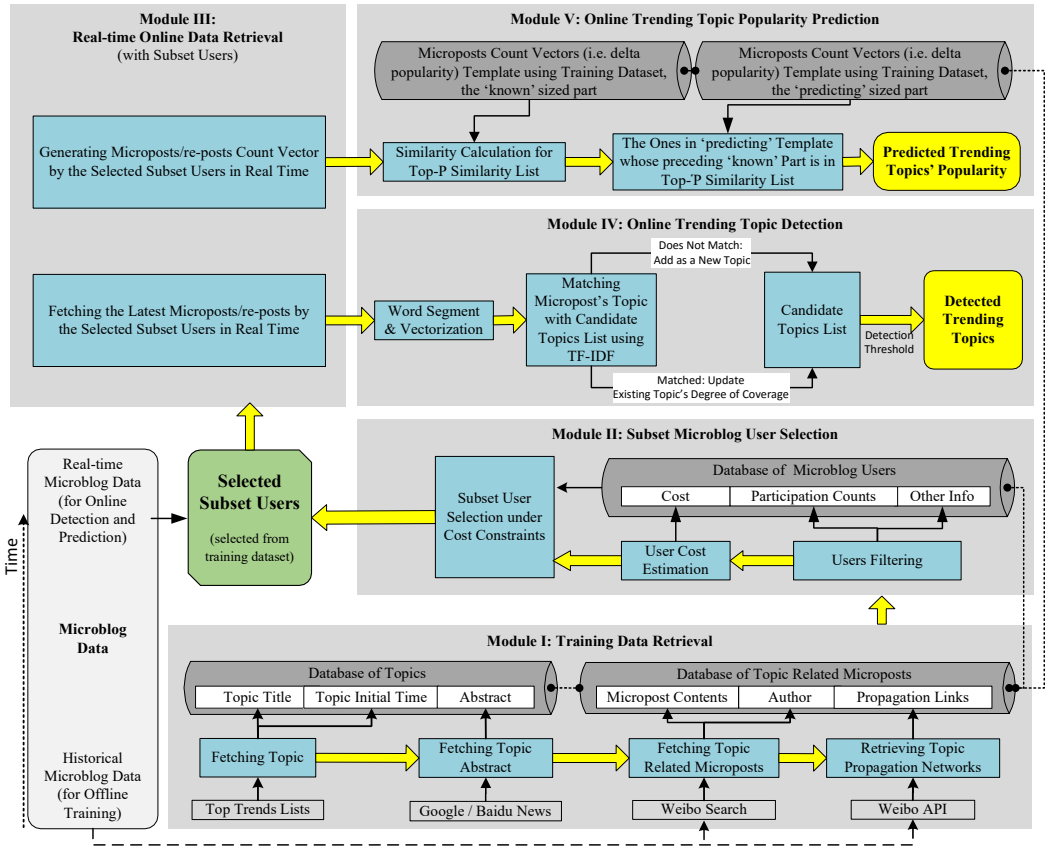

Fig. 1. Overall framework of our microblog trending topics detection and prediction system. Subset users are selected by Module I and II using training dataset, and the real-time microposts by these selected subset users are used for online detection and prediction in Module III, IV and V.

ging websites. The "ground truth" trending topics are also collected in this module. Module II plays a role in selecting subset users from training dataset, and they should be optimally selected according to cost constraints and other configurable settings.

After selecting subset users offline, these users will be used in online modules, namely Module III, IV and V. The Module III will continuously monitor only the selected users in real-time and gather their fresh microposts/re-posts as data sources (these microposts/re-posts' further re-posting links are not gathered) for online trending topics detection in Module IV and prediction in Module V.

The above five modules work together to accomplish the overall online detection and prediction tasks under monitoring and computation cost constraints. In addition, the offline training can also be run periodically when newly collected training dataset is ready, so that the selected subset users can be updated for online operations. But please notice that the updating frequency needn't be too high in order to save the cost for building up new training dataset.

3.1. Training Data Retrieval

In this subsection, we explain the components of Module I, so the building process of training dataset including historical microblog data gathering and several pre-

processing procedures will be introduced. We use Weibo, the largest microblogging service in China, as data source in our experiment. While most microblog contents in Weibo are written in Chinese language, the proposed framework can be readily applied to other languages, with removal of some steps pertinent to the Chinese language such as Chinese word segmentation [Foo and Li 2004] during content vectorization.

3.1.1. Fetching Topics. Except for the microblogging service providers themselves, it is almost impossible to obtain dataset containing all microposts and corresponding topics over the whole microblogging network. Therefore, a microblog dataset should be collected according to background knowledge of specific problem definitions and targets. As our research is focused on trending topics, the first thing we need to know is what topics are indeed popular over Weibo microblogging network, so that we could pay more emphasis on gathering these trending topic related microposts.

Every 10 minutes we collected titles of the Top-10 Trends published officially by Weibo. To reduce potential risk of commercial and political bias from Weibo Official Trends, we also collected titles of Top-10 Trends provided by some popular search engine companies in China, namely Baidu[7], Sogou[8] and Soso[9]. Generally, the titles in all these Top Trends List are too short (commonly less than 20 Chinese characters) to describe the topic in detail. Therefore, we searched the titles in Google News[10] and Baidu News[11] to get more textual information and keywords about the topic. They form an "abstract" of a trending topic, which contains around 80–160 Chinese characters or about 15–30 phrases on average. We make sure that the publishing time of these abstracts is consistent with the fetch time of the corresponding topic title. In addition, Term Frequency Inverted Document Frequency (TF-IDF) vectors of the title and the corresponding abstract are also compared, ensuring they discuss the same topic. Topics are also combined if they have large similarity on titles/abstracts within a given time period. Afterwards, the keywords (especially nouns, names, places and verbs) from titles and abstracts can act as "descriptor" of trending topics that discriminate each topic clearly, so a topic will contain these keywords and the timestamp.

It is worth noting that the trending topics collected from these search engines may not completely eliminate the bias from Weibo Official Trends, due to the commercial considerations or government censorship policy in these sources themselves.

3.1.2. Fetching Topics Related Microposts and Their Propagation Networks. In a microblogging network, a user u_a can start a topic e' by posting a micropost, and the timestamp and keywords of the micropost content will also be the topic's timestamp and keywords. In fact, if that topic e' matches an existing topic e (i.e. e is started some times earlier by another user u_b) in keywords and in the same time period, user u_a is actually joining the existing topic e unconsciously, even though u_a and u_b are in disjoint social relation/following networks (i.e. they don't know/follow each other). These posted (or to say non-reposted) microposts posted by u_a and u_b are then viewed as different *initializing microposts* of the same topic e, and there will be no topic e'.

Besides that, microblog user u_c can also join topic e by re-posting one of e's existing microposts (including re-posts), thus he also spreads the topic to his followers by his *re-posting micropost*. The re-posting action is actually one of the most effective ways on microblogs that attracts users' interests.

[7]http://top.baidu.com
[8]http://top.sogou.com
[9]http://top.soso.com, currently unavailable.
[10]http://news.google.com/news?ned=cn
[11]http://news.baidu.com

We use different strategies in fetching these two kinds of microposts for the same topic e obtained in Section 3.1.1:

— In order to retrieve *initializing microposts* that are related to a specific topic e, keywords of the topic are used as query strings in Weibo Search API. In the returned results, a micropost is marked as related to that topic e if it meets all of the following rule: 1) it is not a re-posting micropost; 2) it matches TF-IDF of that topic e's keywords; 3) it is posted within a reasonable time window of topic e's timestamp; and 4) it has re-posts count larger than a threshold (e.g. 5) to speed up retrieval efficiency.
— For every *initializing microposts* fetched above, we recursively retrieved its full re-posting networks (including its re-posts and re-re-posts, etc.) using Weibo API. All *re-posting microposts* in a re-posting network discussing the same topic during a time period belong to that topic e.

Every author (microblog user) of topic e's initializing and re-posting microposts is participating in e, and they can be regarded as nodes of topic e's propagation network.

3.1.3. Topic Filtering. The trending topics fetched in Section 3.1.1 are crawled from the top trends lists provided by both Weibo and search engines. However, topic trends provided by search engines come from various kinds of information sources such as portal websites, blogs, forums and microblogs. Therefore, if the number of participants/nodes in a topic's full propagation network is less than a threshold (e.g. 750), this topic seems to be not popular in microblogging services and will not be used. On the contrary, the rest topics with participants count bigger than the threshold are indeed trendy, and thus we regard these topics as "ground-truth" trending topics in our dataset.

Although we have tried our best to avoid the bias in the "ground-truth" trending topics and training dataset, the bias might still not be completely eliminated. So please note there is a chance that the final detection results may also reflect such bias.

3.2. Subset Microblog User Selection

Module II is to select a suitable subset of representative users among all users in the whole trending topics propagation networks in offline, whose real-time posted/re-posted microposts can then be used to detect trending topics as well as to predict their popularity online, in a cost effective way. The whole user selection procedures comprises the following steps.

3.2.1. User Filtering. In microblogging networks, there are many inactive users and even spam users that should be excluded from selection, since efficiency is one major concern in our system. As this paper is not focused on identifying spam users, we firstly apply some filtering rules on the domains of users. These filter rules remove the users who are highly inactive (far less than the average posting/re-posting frequency), apparently not influential (very low on followers count), or with spam-like behavior (such as repeatedly re-posting the same topic/micropost or putting many irrelevant keywords together into a single micropost). Filtering these users could reduce computation loads in later steps, and the final system accuracy will not likely to be affected as these users are not likely to be selected anyway according to the strategies of all the user selection algorithms mentioned in Section 6.2.

3.2.2. User Cost Estimation. When a user is selected as a representative user into the subset, the proposed system will keep monitoring and retrieving his/her microposts continuously, and then his microposts will be the input of real-time topic detection and prediction modules. The cost for monitoring and retrieving such real-time data is related to the user's posting/re-posting frequency, as the number of API calls that fetches the microposts content is limited during each time window (e.g. per 15 minutes

in Twitter) by the microblogging service provides. So the cost will arise if more API requests are needed to collect each selected user's data continuously. Besides that, the computational cost such as CPU and RAM needed for online detection and prediction algorithms are also related to input data scale, or to say the selected user's posting frequency. Therefore, to quantitatively measure the system's monitoring and computational cost spent on each selected user, we define *user cost* as the average number of microposts that posted/re-posted by him/her per day during a long period of time. User cost will be taken into account during user selection for the sake of efficiency and system overall cost constraints.

Technically, we assume that user cost would not change much during a long period of time, thus *user cost* is estimated according to the time difference between the first and the last micropost among his/her latest 100 microposts (including re-posts). For example, if it takes a user 8 days to post his latest 100 microposts, his cost is estimated as 100/8=12.5. The number 100 here is enough as we have tested the numbers much bigger than 100 and found no more than 10% difference on estimated costs. Moreover, due to API limitations on the max number of one user's microposts that can be retrieved using one API call as well as the API rates limit per time window, to estimate user cost by retrieving latest microposts information much more than 100 will cause extra consumptions on API resources that are not worthwhile or affordable.

3.2.3. Subset User Selection. This is one of the core procedures in our system. An optimal subset of users are selected by minimizing detection and prediction loss while satisfying the system constraints. The formal problem definitions and solutions will be explained in detail in Section 4 and 5.

3.3. Real-time Online Data Retrieval

In this module, previously selected subset users are monitored continuously online. The microposts that are posted/re-posted by these users within the latest time slot are periodically collected by our system using Weibo API, and thus selected users' microposts are gathered as real-time online dataset to be used in detection and prediction. It is notable that the further re-posting links or networks of these subset users' microposts/re-posts is not needed for the following real-time detection and prediction[12].

3.4. Online Trending Topic Detection

Real-time microposts by the selected users that are collected in the previous module are fed into this module as the input dataset for trends detection. Generally, we can use almost any text mining and trend identification methods with these data. Nevertheless, many research works focus on extracting features using a *huge* amount of data, and this is not suitable here since the input microposts data of this module is already downsized. Therefore, in order to meet the intention of our cost effective framework and demonstrate the power of proposed subset user selection algorithms, we just apply a simple content matching based single pass clustering algorithm [Papka and Allan 1998; Yang et al. 1998] in this online detection module.

The online trending topic detection steps are outlined as follows, while the mathematical definition will be stated later in Section 4.2.

(1) Microposts that posted or re-posted by the subset users within the latest time slot are fetched periodically using Module III;

[12]With the intention of evaluating the detection and prediction performance of our system, we still retrieved these microposts propagation links as ground truth. For the same reason, the "ground truth" trending topics of the real-time microblog dataset are also collected using similar methods mentioned in Module I.

(2) Word segmentation[13], stop-words filtering and text vectorization are applied to micropost contents;
(3) Each micropost is compared with the topic list that has been specified in the latest N_h (a configurable threshold) time slots using TF-IDF:
   — If a micropost is matched with an existing topic with high similarity, mark the micropost to be related with that topic;
   — Otherwise, a new topic is created and added to the topic list whose timestamp and keywords are based on that micropost's timestamp and its content keywords.
(4) Update detection coverage of all the topics. If one topic's detection coverage goes beyond a predefined threshold, it is regarded to be detected as a trending topic.

It is worth pointing out that this detection module can be updated with more advanced text mining or any other types of detection methods that are compatible with our framework in accomplishing online trending topics detection task.

3.5. Online Trending Topic Popularity Prediction

After a trending topic is detected, our system can predict its future popularity. In this paper we define a topic's *popularity* at a given time point as the total number of microposts and re-posts of that topic since the topic begun to that time point. Similar to the considerations in choosing detection methods, we again propose a simple algorithm in terms of topic popularity prediction, whose formal definition and detailed method will be explained in Section 4.3 and 5.3. The basic idea is to calculate weighted average over template vectors as prediction results: At first we calculate similarity between "known" part of a detected trending topic' delta popularity vector (its size is τ) among selected users and each τ sized part of the template vector taken from training dataset. Then "predicting" part of trending topic's delta popularity vector among all users can be predicted by weighted majority voting of succeeding part of the top-$\mathcal{P}$ most similar templates' "known" part and other factors.

It is also worth noting that this prediction module can be updated with any other prediction algorithms that are compatible with our framework that uses subset users' microposts to predict the topic's future popularity among all users.

4. PROBLEM STATEMENT FOR SUBSET USER SELECTION

4.1. Basic Settings

Given a set of trending topics $\mathcal{E}$, the users who have posted/re-posted microposts for at least one trending topic in $\mathcal{E}$ can be seen as the nodes of topic set $\mathcal{E}$'s propagation network $\mathcal{G}$. Let $\mathcal{V}$ denote the whole nodes set in the network, the goal is to select a suitable subset of representative nodes $\mathcal{S}$ from $\mathcal{V}$ ($\mathcal{S} \subseteq \mathcal{V}$), so that trending topics $\mathcal{E}$ among users $\mathcal{V}$ can still be detected and their future popularity can be predicted using only the microposts from $\mathcal{S}$.

There are two basic but necessary constraints when selecting subset nodes $\mathcal{S}$: the maximum number of nodes (K) in the subset and the maximum total cost of all nodes (M) in the subset. The purpose of constraints K and M is to keep the real monitoring, data retrieving and processing cost within budgets when solving practical problems.

Denoting m_v as node v's cost (defined in Section 3.2.2), the above two constraints can be represented by Eq. 1.

$$|\mathcal{S}| \leq K, \quad \sum_{v \in \mathcal{S}} m_v \leq M \tag{1}$$

[13]Chinese word segmentation system ICTCLAS is used, available at http://ictclas.nlpir.org

4.2. Loss Function for Detection

This subsection formulates the loss function of trending topic detection in microblogs by selected subset users $\mathcal{S}$.

A node v ($v \in \mathcal{V}$) is regarded as a participant of a topic e ($e \in \mathcal{E}$), by posting or re-posting topic e related microposts within a given time period T_M since topic e was initiated by its earliest micropost[14]. And topic e is viewed to be covered for one time by node v if v participates in topic e. If node v participates in e for multiple times, topic e is still viewed to be covered once by v. So binary variable $a_{v,e}$ is used to indicate this status, where $a_{v,e} = 1$ if and only if v participates in topic e for at least once. Otherwise, the value of $a_{v,e}$ is 0.

As mentioned in Section 1, selecting subset users for trending topic detection is a multi-coverage "sensor placement" problem in microblog propagation network. Therefore, we define a concept called *Degree of Coverage* (*DoC*), denoted as $D_e(\mathcal{S})$, to measure the degree that a topic e has been covered by a subset of users $\mathcal{S}$ ($\mathcal{S} \subseteq \mathcal{V}$). In the simplest form, $D_e(\mathcal{S})$ can be calculated by e's participants count in $\mathcal{S}$, shown in Eq. 2.

$$D_e(\mathcal{S}) = \sum_{v \in \mathcal{S}} a_{v,e} \tag{2}$$

Given a threshold X_e, topic e is said to be multi-covered (or to say detected as a trending topic) by user set $\mathcal{S}$, if and only if $D_e(\mathcal{S}) \geq X_e$. This detection threshold can be set accordingly for different training datasets and different cost constraints. Furthermore, the threshold for each topic X_e ($e \in \mathcal{E}$) or the threshold for topics in different categories can be customized according to system user's preferences. For example, one topic containing a specific keyword can be set to have smaller detection threshold than the other topics, so it is easier for this topic to be covered as less users are needed.

The loss function for detecting trending topics $\mathcal{E}$ using subset $\mathcal{S}$ is shown as Eq. 3. The value of function $\mathbb{1}(x)$ is equal to 1 if x is logical True, and it is equal to 0 if x is False. So there is no loss for a topic if its *DoC* reaches the detection threshold.

$$\mathcal{L}_{detect}(\mathcal{E}, \mathcal{S}) = \sum_{e \in \mathcal{E}} \mathcal{L}_{detect}(e, \mathcal{S}) = \sum_{e \in \mathcal{E}} \mathbb{1}\big(X_e > D_e(\mathcal{S})\big) \tag{3}$$

4.3. Loss Function for Prediction

Besides identifying e as a trending topic with subset user $\mathcal{S}$, we also would like to predict e's future popularity among all users $\mathcal{V}$, using only the existing observed microblog data from subset user $\mathcal{S}$. With the predicted future popularity among all users, analyzers can understand the importance of the topic in advance, as well as how long will this trending topic last.

In this paper, the popularity of a topic is measured by its total micropost (including re-posts) count at a given time point since the topic begun. For convenience, we segment a topic's whole lifetime T_M from when it is initiated till it ends into discrete time slots. These time slots can be indexed as $\{T_s^{(1)}, ..., T_s^{(i)}, ...\}$, and the time points right between every time slot are denoted as $\{t_1, ..., t_i, ...|t_0 = 0\}$. Comparing with a topic's whole lifetime window T_M, each time slot length L_s should be set relatively small[15].

We use a time-series $\{y_e(1, \mathcal{V}), y_e(2, \mathcal{V}),...\}$ to represent topic e's microposts (including re-posts) count that are posted/re-posted by users in $\mathcal{V}$ during each time slot $T_s^{(1)}$, $T_s^{(2)}$ ... Thus, denoting the counting time-series among all users $\mathcal{V}$ since the first time slot $T_s^{(1)}$ till $T_s^{(\tau)}$ as $\vec{y}_e([1, \tau], \mathcal{V})$, popularity of topic e at time point t_τ (i.e. right after time slot

[14] T_M is uniformly set to 3 days in our experiment settings.
[15] Length of a time slot L_s is set to 6 minutes in our experiment settings.

$T_s^{(\tau)}$) can be calculated by summing up its elements using Eq. 4. As micropost count are always non-negative, the sum can also be denoted as L1 norm of $\vec{y}$.

$$\mathrm{pop}_e(t_\tau, \mathcal{V}) = \sum_{i=1}^{\tau} y_e(i, \mathcal{V}) = \|\vec{y}_e([1,\tau], \mathcal{V})\|_1 \tag{4}$$

Following the above definitions and design philosophy of our system, in real-time online prediction the actual microposts we observe are the ones posted or re-posted by subset user $\mathcal{S}$ from beginning till t_τ, and the observed counting time-series known to us can be denoted as $\vec{y}_e([1,\tau], \mathcal{S})$. [16]

We denote a prediction function Ψ in Eq. 5 that can predict the future microposts counting time-series $\vec{\hat{y}}_e([\tau+1,\kappa], \mathcal{V})$ among the whole users $\mathcal{V}$ from time slot $T_s^{(\tau+1)}$ till $T_s^{(\kappa)}$, using input time-series $\vec{y}_e([1,\tau], \mathcal{S})$. The value of κ indicates the longest time that can be predicted by function Ψ. Then the topic's future popularity at time point t_κ can be predicted using Eq. 6, by summing the known counts till t_τ (the first term) as well as the predicted micropost counts from $t_{\tau+1}$ till t_κ (the second term) at each time slot.

$$\Psi\big(\kappa, \vec{y}_e([1,\tau], \mathcal{S})\big) = \hat{\vec{y}}_e([\tau+1,\kappa], \mathcal{V}) \tag{5}$$

$$\begin{aligned} \widehat{\mathrm{pop}}_e(t_\kappa, \mathcal{V}|\Psi, \tau, \mathcal{S}) &= \|\vec{y}_e([1,\tau], \mathcal{V})\|_1 + \|\hat{\vec{y}}_e([\tau+1,\kappa], \mathcal{V})\|_1 \\ &= \mathrm{pop}_e(t_\tau, \mathcal{V}) + \big\|\Psi\big(\kappa, \vec{y}_e([1,\tau], \mathcal{S})\big)\big\|_1 \end{aligned} \tag{6}$$

Having all the definitions above, the loss of popularity prediction on trending topics $\mathcal{E}$ by a subset user $\mathcal{S}$ and a prediction function Ψ can be defined as the absolute popularity prediction error at time point t_κ ($\kappa > \tau$), shown in Eq. 7.

$$\begin{aligned} \mathcal{L}_{predict}(\mathcal{E}, \mathcal{S}) &= \sum_{e\in\mathcal{E}} \mathcal{L}_{predict}(e, \mathcal{S}) \\ &= \sum_{e\in\mathcal{E}} |\widehat{\mathrm{pop}}_e(t_\kappa, \mathcal{V}|\Psi, \tau, \mathcal{S}) - \mathrm{pop}_e(t_\kappa, \mathcal{V})| \end{aligned} \tag{7}$$

Substituting the predicted popularity term in Eq. 7 by Eq. 5 and Eq. 6, the loss can then be calculated by the sum of absolute micropost count prediction error in each time slot from time point $t_{\tau+1}$ till t_κ. The deduction is demonstrated in Eq. 8.

$$\begin{aligned} \mathcal{L}_{predict}(\mathcal{E}, \mathcal{S}) &= \sum_{e\in\mathcal{E}} |\widehat{\mathrm{pop}}_e(t_\kappa, \mathcal{V}|\Psi, \tau, \mathcal{S}) - \mathrm{pop}_e(t_\kappa, \mathcal{V})| \\ &= \sum_{e\in\mathcal{E}} \Big|\widehat{\mathrm{pop}}_e(t_\kappa, \mathcal{V}|\Psi, \tau, \mathcal{S}) - \Big(\mathrm{pop}_e(t_\tau, \mathcal{V}) + \|\vec{y}_e([\tau+1,\kappa], \mathcal{V})\|_1\Big)\Big| \\ &= \sum_{e\in\mathcal{E}} \Big|\|\hat{\vec{y}}_e([\tau+1,\kappa], \mathcal{V})\|_1 - \|\vec{y}_e([\tau+1,\kappa], \mathcal{V})\|_1\Big| \\ &= \sum_{e\in\mathcal{E}} \big|\|\Psi\big(\kappa, \vec{y}_e([1,\tau], \mathcal{S})\big)\|_1 - \|\vec{y}_e([\tau+1,\kappa], \mathcal{V})\|_1\big| \end{aligned} \tag{8}$$

[16] We would like to give an example for better illustration: Suppose there are 10 users in $\mathcal{V}$ for a topic e. The first half of them each posted one micropost at the first time slot, and the other half of them each posted one micropost during the second time slot. Then the time-series $\vec{y}_e([1:2], \mathcal{V})$ will be (5,5), and topic e's popularity $\mathrm{pop}_e(t_2, \mathcal{V})$ at time point t_2 is 5+5=10. If 2 users in the first half are selected as subset users $\mathcal{S}$, then the $\vec{y}_e([1:2], \mathcal{S})$ observed by the system will be (2,0), and $\mathrm{pop}_e(t_2, \mathcal{S})$ is 2.

It should be pointed out that time point of prediction function Ψ's output is $t_{\tau+1}$ till $t\kappa$ given the input from t_1 till t_τ. If the input of Ψ are more recent observations such as $\vec{y}_e([1+k,\tau+k],\mathcal{S})$ $(k>0)$, it can then produce prediction results $\vec{\hat{y}}_e([\tau+1+k,\kappa+k],\mathcal{V})$ at further time points. In this way, the prediction results at any future time points can be recursively predicted.

4.4. Combined Objective Function

Based on the above loss functions, we formulate an optimization task for selecting a subset of nodes $\mathcal{S}$ from the whole node set $\mathcal{V}$ in network $\mathcal{G}$ under resource constraints. In the optimization, argument is $\mathcal{S}$, and target objective function is minimizing both detection loss $\mathcal{L}_{detect}(e,\mathcal{S})$ and prediction loss $\mathcal{L}_{predict}(e,\mathcal{S})$ for all topics $e\in\mathcal{E}$.

Let b_v be a binary variable where $b_v=1$ indicates node $v\in\mathcal{V}$ is selected as one of the subset users and $b_v=0$ otherwise. Overall optimization objective function can then be represented in Eq. 9, by mixing up Eq. 1, Eq. 3 and Eq. 7. In the equation, λ is a coefficient that indicates the weight of prediction loss when selecting subset users $\mathcal{S}$. When $\lambda=0$, the prediction loss will not be considered during user selection. The effect of λ in experiments will be discussed in Section 6.4.3.

$$\begin{aligned}\underset{\mathcal{S}}{\operatorname{argmin}}\Big(\sum_{e\in\mathcal{E}}\mathcal{L}_{detect}(e,\mathcal{S})+\lambda\cdot\sum_{e\in\mathcal{E}}\mathcal{L}_{predict}(e,\mathcal{S})\Big)\\ s.t.\quad\sum_{v\in\mathcal{V}}m_v b_v\le M\\ \sum_{v\in\mathcal{V}}b_v\le K\\ \lambda\ge 0\\ \mathcal{S}=\{v|b_v=1,v\in\mathcal{V}\}\end{aligned}\tag{9}$$

5. EFFICIENT ALGORITHMS

Generally speaking, the original problem formulated in Section 4 is mixed-integer programming (MIP) [Bertacco 2006], and we propose efficient algorithms to find a feasible solution that satisfies all constraints. For our joint detection and prediction system, we define a "reward" function $R(\Lambda,\Theta)$, which maps a subset Λ ($\Lambda\subseteq\mathcal{V}$) of nodes and a subset of topics Θ ($\Theta\subseteq\mathcal{E}$) into a real number. The value of this number shows the current detection and prediction "reward" on the topics set Θ using selected user subset Λ. Therefore, different ways of selecting subset users will lead to different sets of detected trending topics, and thus the rewards are different. We define the total joint reward in Eq. 10, in which detection and prediction rewards are the derived and opposite of loss function $\mathcal{L}_{detect}$ and $\mathcal{L}_{predict}$, respectively.

$$\begin{aligned}R(\Lambda,\Theta)&=R_{detect}(\Lambda,\Theta)+\lambda\cdot R_{predict}(\Lambda,\Theta)\\ R_{detect}(\Lambda,\Theta)&=\sum_{e\in\Theta}\mathbb{1}(D_e(\Lambda)\ge X_e)\\ R_{predict}(\Lambda,\Theta)&=-\sum_{e\in\Theta}\big|\|\Psi\big(\kappa,\vec{y}_e([1,\tau],\Lambda)\big)\|_1-\|\vec{y}_e([\tau+1,\kappa],\mathcal{V})\|_1\big|\end{aligned}\tag{10}$$

With the help of function R, various ways of utilizing reward values can be developed, i.e. different heuristic strategies in selecting subset users. In following Section 5.1, we first introduce a straightforward user selection algorithm *SWC*, then in Section

5.2 a more effective user selection method *JNT* is proposed. Section 5.3 describes the popularity prediction algorithm and the prediction reward calculation in detail.

5.1. Algorithm SWC

In single coverage problems where the objective is maximizing node placement coverage with nodes having equal or unequal cost, a widely used heuristic is the greedy algorithm described in [Leskovec et al. 2007]. In that paper, the node with maximized ratio of reward to cost is chosen iteratively in each round of selection. Based on the idea of maximizing ratio in that greedy algorithm, we adapted it to be compatible for solving subset user selection problems with multiple coverage requirements. This algorithm runs in a Stage-Wise Covering manner, thus called algorithm *SWC*.

At first, the algorithm is initiated with an empty set of selected nodes $\mathcal{S} = \varnothing$ and an empty set of topics $\mathcal{E}_c = \varnothing$ that includes the topics with $DoC \geq X$ (i.e. the trending topics that are multi-covered by $\mathcal{S}$). Then the multi-covering problem can split into looping single-covering stages. During each single coverage stage, every uncovered topic e ($e \in \mathcal{E} \setminus \mathcal{E}_c$) needs only to be covered once. In subsequent single coverage stages, topic e still needs to be covered one time in each stage until its overall *DoC* reaches X_e and is moved into $\mathcal{E}_c$. In total, there will be at most $\max(X_e), e \in \mathcal{E}$ single-coverage stages.

More specifically, at the initiation step of the i^{th} single coverage stage, $\mathcal{E}_c^{(i)}$ (denoting the topics that has been single-covered in i^{th} stage) is set to be empty; the detection threshold $X_e^{(i)}$ of each not yet multi-covered topic e's ($e \in \mathcal{E} \setminus \mathcal{E}_c$) is set to 1 in this stage, and the threshold $X^{(i)}$ of the rest topics in $\mathcal{E}_c \cup \mathcal{E}_c^{(i)}$ are set to $+\infty$ indicating the reward for these topics are not considered. The optimization target of this stage is to find a subset of nodes that can single-cover topics set $\mathcal{E} \setminus \mathcal{E}_c$.

For each single coverage stage, users are iteratively selected in rounds. In each user selection round, marginal detection reward/cost ratio of each user v ($v \in \mathcal{V} \setminus \mathcal{S}$) is calculated with Eq. 11. A user $v_{\max}$ with the largest marginal detection reward per unit cost is then selected and added to subset $\mathcal{S}$. Afterwards, the topics covered by user $v_{\max}$ are added to $\mathcal{E}_c^{(i)}$, and marginal detection reward/cost ratio is recalculated using Eq. 11 again. Then in next round another user with the largest marginal reward/cost ratio is selected. In this way, users can be iteratively selected for each singe coverage stage. Each single coverage stage stops when all topics are single covered (i.e. $\mathcal{E}_c^{(i)} = \mathcal{E} \setminus \mathcal{E}_c$), or when the overall cost constraints are reached. At the end of i^{th} stage, $\mathcal{E}_c$ is updated. If the overall cost constraints are not reached, the $i + 1^{th}$ stage will then begin. In case there are more than one user maximizing Eq. 11, we can select the user that participates in more topics to break the tie.

$$v_{\max} = \underset{v \in \mathcal{V} \setminus \mathcal{S}}{\operatorname{argmax}} \frac{R_{detect}\big(\mathcal{S}^{(i)} \cup \{v\}, \mathcal{E} \setminus (\mathcal{E}_c \cup \mathcal{E}_c^{(i)})\big) - R_{detect}\big(\mathcal{S}^{(i)}, \mathcal{E} \setminus (\mathcal{E}_c \cup \mathcal{E}_c^{(i)})\big)}{m_v} \tag{11}$$

After running all the stages in algorithm *SWC*, a subset of users $\mathcal{S}$ are finally selected, and then those real-time microposts of $\mathcal{S}$ will be retrieved and used in subsequent real-time detection and prediction procedures. Pseudo code of the whole algorithm *SWC* is listed in Algorithm 1.

5.2. Algorithm JNT

In user selection algorithm *SWC*, target of topic coverage is set to 1 per single covering stage. It is not efficient as it needs many loops when overall detection threshold X is large. As a matter of fact, when solving multi-coverage problems, it is more efficient to cover a topic more than once by different users during one user selection iteration.

ALGORITHM 1: Algorithm *SWC* for Subset User Selection

Require: Full nodes set $\mathcal{V}$, nodes cost $m_v(v \in \mathcal{V})$, trending topics $\mathcal{E}$, constraints M and K
Ensure: A set of optimal selected nodes $\mathcal{S} \subseteq \mathcal{V}$

$\mathcal{S} \leftarrow \varnothing$, $\mathcal{E}_c \leftarrow \varnothing$, $M_{curr} \leftarrow 0$, $i \leftarrow 1$
while $|\mathcal{S}| < K$ **and** $M_{curr} < M$ **and** $i \leq \max\{X_e | e \in \mathcal{E}\}$ **do**
  $\mathcal{E}_c^{(i)} \leftarrow \varnothing$, $\mathcal{S}^{(i)} \leftarrow \varnothing$, $X_e^{(i)} = 1(e \in \mathcal{E} \setminus \mathcal{E}_c)$, $X_e^{(i)} = +\infty(e \in \mathcal{E}_c)$
  while $\mathcal{E}_c^{(i)} \neq \mathcal{E} \setminus \mathcal{E}_c$ **do**
    Calculate current reward $R_{detect}\big(\mathcal{S}^{(i)}, \mathcal{E} \setminus (\mathcal{E}_c \cup \mathcal{E}_c^{(i)})\big)$ by Eq. 10
    Find a node $v_{\max} \in \mathcal{V} \setminus \mathcal{S}$ with max reward/cost ratio by Eq. 11
    if $M_{curr} + m_{v_{\max}} \leq M$ **and** $|\mathcal{S}| + 1 \leq K$ **then**
      $\mathcal{E}_c^{(i)} \leftarrow \mathcal{E}_c^{(i)} \cup \{e | a_{v_{\max},e} = 1, e \in \mathcal{E} \setminus \mathcal{E}_c\}$
      $\mathcal{S}^{(i)} \leftarrow \mathcal{S}^{(i)} \cup \{v_{\max}\}$, $\mathcal{S} \leftarrow \mathcal{S} \cup \{v_{\max}\}$
      $M_{curr} \leftarrow M_{curr} + m_{v_{\max}}$
    else
      Abort user selection, **return** $\mathcal{S}$
    end if
  end while
  $\mathcal{E}_c \leftarrow \mathcal{E}_c \cup \{e | D_e(\mathcal{S}) \geq X_e, e \in \mathcal{E}_c^{(i)}\}$, $i \leftarrow i + 1$
end while
return $\mathcal{S}$

Additionally, algorithm *SWC* does not take the selected user's prediction performance into consideration at all, which might not be appropriate for the joint task. Therefore, we propose an efficient algorithm to solve the multi-coverage problem that takes into account the JoiNT detection and prediction accuracy of selected users. Thus we name it algorithm *JNT*, and it contains three major improvements compared with *SWC*.

The first improvement in algorithm *JNT* is that dynamic detection reward is used for different topics in user selection, based on the gap between each topic's current *DoC* and its detection threshold X. In the original reward function Eq. 10, subset users' detection reward for each topic is binary valued depending on whether its current *DoC* reaches detection threshold X or not, which is fine in single-coverage situations. However, in multi-coverage problems the reward should be measured more precisely as the gap between a topic's current *DoC* and X could be quite different among various topics and users. For example, suppose the detection threshold X is 10 in a user selection process, the current *DoC* of trending topic e_1 and e_2 are 2 and 8 respectively, user u_1 can cover e_1 once while u_2 can cover e_2 once, and one of them are to be selected. In this situation, other things being equal, topic e_1 is more urgent to be covered than e_2, because e_1 needs 8 more coverages to reach threshold X while e_2 needs only 2. Thus, the reward for covering e_1 by u_1 should be higher than covering e_2 by u_2 so that u_1 can be selected. However the binary valued detection reward can not handle this case as threshold X is not reached and the reward for u_1 and u_2 would be both 0.

Therefore, to improve the overall topic coverage of $\mathcal{S}$, a dynamic reward function is defined according to the difference between a topic's current *DoC* and its threshold X. If topic e's current *DoC* hasn't reached detection threshold yet, it is urgent to be covered by the selected user, so the reward for covering e can be defined to be proportional (linear) to the difference between X and it *DoC* (i.e. X_e-$D_e(\mathcal{S})$). Consequently, topics with lower coverage degree are prioritized to have higher reward, thus they are stimulated to be covered by the selected users in subsequent user selections. In contrast, when topic e's *DoC* has reached X, it is not urgent to be covered any more, so its reward is set to be inversely proportional to its *DoC* to discourage further covers on this topic in subsequent subset user selection operations.

Denoting Λ as the set of already selected subset users, the dynamic reward $r_e(\Lambda)$ for topic e is denoted by Eq. 12, based on the above settings. In Eq. 12, reward is commonly no larger than 1, and α ($0 \leq \alpha \leq 1$) is a configurable number to control sensitivity level in dynamic reward calculation. α is set to 0.01 in our experiments, and more about it is discussed in Section 6.4.4.

$$r_e(\Lambda) = \begin{cases} \frac{(1-\alpha)[X_e - D_e(\Lambda)]}{X_e}, & D_e(\Lambda) < X_e \\ \frac{\alpha}{D_e(\Lambda)}, & D_e(\Lambda) \geq X_e \end{cases}. \tag{12}$$

Hence, the definition of detection reward for algorithm ***JNT*** should be updated accordingly, shown in Eq. 13. Theoretically speaking, using dynamic reward in user selection can be helpful in improving overall recall rate of trending topics detection.

$$R^{JNT}_{detect}(\Lambda, \Theta) = \sum_{e \in \Theta} r_e(\Lambda) \tag{13}$$

The second improvement in algorithm ***JNT*** is that we apply a dynamic user cost boundary in user selection, so the users whose cost is beyond boundary are excluded from selection. Afterwards, the users having maximum reward and within cost boundary are selected iteratively as subset users. At the beginning of each iteration, the cost boundary is dynamically updated according to current system spare cost and current subset users' size. Comparing to the strategy that just selecting the user with highest marginal reward/cost ratio in algorithm ***SWC***, the aforementioned operation is a more flexible user selection strategy that can make full usage of the remaining available cost budget, especially when the total cost constraints is not so tight.

Concretely, when there are K_l available nodes (maximum is K) and M_l available microposts monitoring and processing cost (maximum is M), the cost boundary $M_b(K_l, M_l)$ is proposed by us in Eq. 14. In the equation, γ is a configurable value to control boundary size. The cost boundary will always be bigger than the current average available cost per user (M_l/K_l), so the users with better coverages but relatively larger cost are allowed to be selected; and the boundary will be no larger than the current total available cost M_l in order to meet the system cost constraints. γ is set to 0.7 in our experiments, and is discussed in Section 6.4.4.

$$M_b(K_l, M_l) = \min(\frac{M_l}{K_l^{\gamma}} + \frac{M_l}{K_l}, M_l) \tag{14}$$

Thirdly, in algorithm ***JNT*** we consider the selected users' prediction reward during user selection. For algorithm ***SWC***, users who have the best marginal detection reward per unit cost are selected. However the fact that a user is doing well in detection does not necessarily mean that he will also be the best choice in prediction. For example, the detection result will not be affected if most of the selected users prefer to attend trending topics relatively later than the other users, but their microposts might be too late to be used as prediction input and the prediction result may not be so ideal.

Therefore, prediction reward is added into the total reward function R^{JNT} for algorithm ***JNT*** shown in Eq. 15. In the equation, coefficient λ controls the prediction reward weight[17]. By default λ is bigger than 0 in ***JNT***, so both the detection and prediction will be taken into account when selecting users, and it is discussed in Section 6.4.3.

[17]If the detection and prediction reward are not normalized, the weight coefficient should be denoted as $\lambda_s \cdot \lambda$, where λ_s is the scale factor. In this paper, we assume λ_s=1.

ALGORITHM 2: Algorithm *JNT* for Subset User Selection

Require: Full nodes set $\mathcal{V}$, nodes cost $m_v(v \in \mathcal{V})$, trending topics $\mathcal{E}$, constraints M and K
Ensure: A set of optimal selected nodes $\mathcal{S} \subseteq \mathcal{V}$
$\mathcal{S} \leftarrow \varnothing$, $M_l \leftarrow M$, $K_l \leftarrow K$
while $K_l > 0$ **and** $M_l > 0$ **do**
  Calculate cost boundary $M_b(K_l, M_l)$ by Eq. 14
  Calculate current joint reward $R^{JNT}(\mathcal{S}, \mathcal{E})$ by Eq. 15
  Find a node $v_{\max} \in \mathcal{V} \setminus \mathcal{S}$ with max joint reward increment by Eq. 16
  if $m_{v_{\max}} \leq M_l$ **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{v_{\max}\}$, $K_l \leftarrow K_l - 1$, $M_l \leftarrow M_l - m_{v_{\max}}$
  else
    Abort user selection, **return** $\mathcal{S}$
  end if
end while
return $\mathcal{S}$

ALGORITHM 3: Prediction Algorithm with Selected Subset Users

Require: Training dataset Ω_{tra}, user subset $\mathcal{S}$, observed $YG_e^{\mathcal{S}}$ with length τ
Ensure: Predicted time-series $YU_e^{\mathcal{V}}$ and $YU_e^{\mathcal{S}}$ from $t_{\tau+1}$ till t_κ
for all topic's full counting time-series in Ω_{tra} **do**
  Generate template vectors $\boldsymbol{PG}^{\mathcal{V}}$ and $\boldsymbol{PU}^{\mathcal{V}}$ using sliding window
  Generate template vectors $\boldsymbol{PG}^{\mathcal{S}}$ and $\boldsymbol{PU}^{\mathcal{S}}$ using sliding window
end for
for all template vector $PG_j^{\mathcal{S}} \in \boldsymbol{PG}^{\mathcal{S}}$ **do**
  Calculate similarity between $PG_j^{\mathcal{S}}$ and $YG_e^{\mathcal{S}}$
end for
Get index i_ρ ($\rho \in [1, \mathcal{P}]$) of Top-$\mathcal{P}$ most similar template vector with $YG_e^{\mathcal{S}}$ in $\boldsymbol{PG}^{\mathcal{S}}$
Predict $YU_e^{\mathcal{V}}$ with Eq. 17, 18 and 19
Predict $YU_e^{\mathcal{S}}$ with Eq. 20

$$R^{JNT}(\Lambda, \Theta) = R^{JNT}_{detect}(\Lambda, \Theta) + \lambda \cdot R_{predict}(\Lambda, \Theta) \tag{15}$$

Combining the above three improvements and modifications together, algorithm ***JNT*** selects the subset users in an iterative manner: In each selecting iteration, cost boundary M_b is updated based on current available budget, then reward of each user in $\mathcal{V} \setminus \mathcal{S}$ whose cost is within current cost boundary is calculated, and user $v_{\max}$ with maximized total reward among them is selected using Eq. 16. After adding $v_{\max}$ into $\mathcal{S}$ and updating K_l and M_l, cost boundary M_b is re-calculated, and then next user selection iteration begins. The user selection process will stop when any cost constraints is met. The full procedures of algorithm ***JNT*** is summarized in Algorithm 2.

$$v_{\max} = \underset{v \in \mathcal{V} \setminus \mathcal{S}, m_v \leq M_b(K_l, M_l)}{\operatorname{argmax}} \Big(R^{JNT}(\mathcal{S} \cup \{v\}, \mathcal{E}) - R^{JNT}(\mathcal{S}, \mathcal{E}) \Big) \tag{16}$$

5.3. Prediction Algorithm

In the above two subsections, we introduced different user selection algorithms and corresponding reward calculation methods. In this subsection, we explain the algorithm of utilizing selected subset users' microposts for predicting topic's future popularity among whole users. After that, selected users' prediction reward as well as topic popularity prediction result can be calculated.

The intention and definition of prediction function Ψ is already listed in Eq. 5. To mathematically describe the detailed algorithm for the prediction function, we follow the setting introduced in Section 4.3: When a subset of users $\mathcal{S}$ is selected, what we can observe by monitoring their microposts are the beginning *known part* of each topic e's microposts/re-posts counting time-series $\vec{y}_e([1,\tau],\mathcal{S})$ till time point t_τ. The prediction target is the succeeding unknown part $\vec{y}_e([\tau+1,\kappa],\mathcal{V})$ among all users $\mathcal{V}$, from time point $t_{\tau+1}$ to t_κ. That is to say, we will use current subset users' microposts of a topic to predict the topic's future popularity among all users. For simplicity, the aforementioned known part and predicting part are denoted as $YG_e^{\mathcal{S}}$ and $YU_e^{\mathcal{V}}$, respectively.

5.3.1. Template Vectors. In our prediction algorithm, there is an assumption that if the first part of two time-series vectors are high in similarity, their succeeding part within a small time period will also likely to be similar, especially when the two vectors represent the same group of users' posting behavior on trending topics. Therefore, besides the observed *known* vectors $YG_e^{\mathcal{S}}$, additional *template* time-series vectors that can reflect subset users $\mathcal{S}$'s posting/re-posting counts on historical trending topics are needed. Each template vector consists of two parts: the τ sized known part used for similarity calculation; and the succeeding $\kappa-\tau$ sized *predicting* part used for prediction.

For a given selected user subset $\mathcal{S}$, a set of template vectors $\boldsymbol{P}^{\mathcal{S}}$ can be extracted from the training dataset. In training dataset, each trending topic's full counting time-series has $L_M = \lfloor T_M/L_s \rfloor$ (Max life-time of a topic / length of a time slot) time slots in total, and it is commonly much bigger than κ. Thus, we use a sliding window with size=κ and step=1 to extract every κ sized template vector from each trending topic's full counting time-series in training dataset. After that, each template vector $P_j^{\mathcal{S}} \in \boldsymbol{P}^{\mathcal{S}}$ is segmented into known and succeeding predicting part as $\langle PG_j^{\mathcal{S}}, PU_j^{\mathcal{S}}\rangle$, with size $\langle \tau, \kappa-\tau\rangle$.

Concretely, denoting $\vec{y}_A([1,L_M],\mathcal{S})$ as topic A's full counting time-series among users set $\mathcal{S}$ in training dataset, the first extracted template will be $PG_1^{\mathcal{S}} = \vec{y}_A([1,\tau],\mathcal{S})$, $PU_1^{\mathcal{S}} = \vec{y}_A([\tau+1,\kappa],\mathcal{S})$. Then, window will slide 1 step and the second extracted template will be $PG_2^{\mathcal{S}} = \vec{y}_A([2,\tau+1],\mathcal{S})$, $PU_2^{\mathcal{S}} = \vec{y}_A([\tau+2,\kappa+1],\mathcal{S})$, and so on. The extraction window will gradually slide for $L_M-\kappa$ times till the other side of the window reaches the last time slot of $\vec{y}_A$, and thus $L_M-\kappa+1$ templates are extracted. Afterwards, extraction of the next trending topic B's full counting time-series $\vec{y}_B([1,L_M],\mathcal{S})$ begins, and so on. In the end, the set $\boldsymbol{P}^{\mathcal{S}}$ will include all the template vectors extracted from all trending topics in training dataset.

Using similar operations, template vectors set $\boldsymbol{P}^{\mathcal{V}}$ that contains the microposts counting time-series among whole user set $\mathcal{V}$ is also built. Moreover, if $\boldsymbol{P}^{\mathcal{V}}$ and $\boldsymbol{P}^{\mathcal{S}}$ are extracted from training dataset exactly in the same order, they have synchronized indexes on topics and time points as users set $\mathcal{S} \subseteq \mathcal{V}$.

5.3.2. Similarity Calculation and Popularity Prediction. After building up template vectors with offline training dataset, it is time to predict $YU_e^{\mathcal{V}}$ for topic e. At first similarity between $YG_e^{\mathcal{S}}$ and every template $PG_j^{\mathcal{S}} \in \boldsymbol{PG}^{\mathcal{S}}$ are calculated using Pearson's correlation coefficient. If all the elements in $YG_e^{\mathcal{S}}$ or $PG_j^{\mathcal{S}}$ are in the same value, the Pearson's coefficient cannot be calculated. In this case, we use cosine correlation coefficient instead to represent the similarity. The template whose similarity with $YG_e^{\mathcal{S}}$ is less than a threshold (0.5 in our experiments) will be skipped in following steps.

Denoting i_ρ ($\rho \in [1,\mathcal{P}]$) as indexes of top-$\mathcal{P}$ most similar template vectors $PG_{i_\rho}^{\mathcal{S}}$ to $YG_e^{\mathcal{S}}$, then those top-$\mathcal{P}$ template vectors' succeeding part $PU_{i_\rho}^{\mathcal{S}}$ and their same-indexed template vectors $PU_{i_\rho}^{\mathcal{V}}$ among all users $\mathcal{V}$ can be used to predict the time-series $YU_e^{\mathcal{V}}$ among all users $\mathcal{V}$. The prediction calculation is done by weighted average of the tem-

plate vectors $PU^{\mathcal{V}}_{i_\rho}$ ($\rho \in [1, \mathcal{P}]$) shown in Eq. 17, in which w_{i_ρ} is the weight of i_ρth template $PU^{\mathcal{V}}_{i_\rho}$ and is measured by the similarity between $YU^{\mathcal{V}}_e$ and $PU^{\mathcal{V}}_{i_\rho}$. Also, η_{i_ρ} in Eq. 17 is a scale coefficient that measures the scale ratio between template vector $PU^{\mathcal{V}}_{i_\rho}$ and predicting target $YU^{\mathcal{V}}_e$, so the scale of predicted result will not be affected by the scale of template vectors, which are commonly different.

According to the assumption that similarity of two vectors would not change much within short time period, as well as the fact that subset users will be selected according to their representativeness among all users, we can estimate the similarity between $YU^{\mathcal{V}}_e$ and $PU^{\mathcal{V}}_{i_\rho}$ by using the previously calculated similarity between $YG^{\mathcal{S}}_e$ and $PG^{\mathcal{S}}_{i_\rho}$, shown in Eq. 18. Besides that, as $YG^{\mathcal{S}}$ and $\boldsymbol{PG}^{\mathcal{S}}$ are observed by the same group of users, the scale ratio of them can also be utilized to estimate the scale ratio η of their succeeding parts among all users $\mathcal{V}$. This estimation is stated in Eq. 19.

$$\widehat{YU}^{\mathcal{V}}_e = \frac{\sum\limits_{\rho\in[1,\mathcal{P}]} w_{i_\rho} \cdot \eta_{i_\rho} \cdot PU^{\mathcal{V}}_{i_\rho}}{\sum\limits_{\rho\in[1,\mathcal{P}]} w_{i_\rho}} \tag{17}$$

$$w_{i_\rho} = \mathrm{sim}(YU^{\mathcal{V}}_e, PU^{\mathcal{V}}_{i_\rho}) \approx \mathrm{sim}(YG^{\mathcal{V}}_e, PG^{\mathcal{V}}_{i_\rho}) \approx \mathrm{sim}(YG^{\mathcal{S}}_e, PG^{\mathcal{S}}_{i_\rho}) \tag{18}$$

$$\eta_{i_\rho} = \eta(YU^{\mathcal{V}}_e, PU^{\mathcal{V}}_{i_\rho}) \approx \eta(YG^{\mathcal{V}}_e, PG^{\mathcal{V}}_{i_\rho}) \approx \eta(YG^{\mathcal{S}}_e, PG^{\mathcal{S}}_{i_\rho}) \approx \frac{\sum YG^{\mathcal{S}}_e}{\sum PG^{\mathcal{S}}_{i_\rho}} \tag{19}$$

Additionally, the counting time-series $YU^{\mathcal{S}}_e$ among subset users can also be predicted similarly using Eq. 20.

$$\begin{aligned} \widehat{YU}^{\mathcal{S}}_e &= \frac{\sum\limits_{\rho\in[1,\mathcal{P}]} w_{i_\rho} \cdot \eta_{i_\rho} \cdot PU^{\mathcal{S}}_{i_\rho}}{\sum\limits_{\rho\in[1,\mathcal{P}]} w_{i_\rho}} \\ w_{i_\rho} &= \mathrm{sim}(YU^{\mathcal{S}}_e, PU^{\mathcal{S}}_{i_\rho}) \approx \mathrm{sim}(YG^{\mathcal{S}}_e, PG^{\mathcal{S}}_{i_\rho}) \\ \eta_{i_\rho} &= \eta(YU^{\mathcal{S}}_e, PU^{\mathcal{S}}_{i_\rho}) \approx \eta(YG^{\mathcal{S}}_e, PG^{\mathcal{S}}_{i_\rho}) \approx \frac{\sum YG^{\mathcal{S}}_e}{\sum PG^{\mathcal{S}}_{i_\rho}} \end{aligned} \tag{20}$$

The overall prediction algorithm is summarized in Algorithm 3. Using this algorithm, $\{\hat{y}_e(\tau+1, \mathcal{V})$, $\hat{y}_e(\tau+2, \mathcal{V})$, ..., $\hat{y}_e(\kappa, \mathcal{V})\}$ can be predicted, so the overall predicted popularity $\widehat{pop}_e(t_k, \mathcal{V})$ at any time point t_k, $k \in [\tau+1, \kappa]$ can be calculated with Eq. 6. For offline training (user selection) process, prediction reward $R_{predict}$ using the given selected subset users $\mathcal{S}$ can also be calculated with Eq. 10. Additionally, time-series further than $\hat{y}_e(\kappa, \mathcal{V})$ can also be recursively estimated by inputting newer $YG^{\mathcal{S}}_e$ (either observed or previously predicted) at more recent time points into the prediction system.

It is worth noting that during the whole prediction process, we never need or use $YG^{\mathcal{V}}$, i.e. the known part of trending topics' microposts count among *all* users.

Table I. Statistics of Training and Testing Dataset

Dataset	Time Period	Trending Topics	Microposts	Users
Ω_{tra}	10th Sept. – 25th Sept., 2012	75	753,486	585,640
Ω_{test}	26th Sept. – 10th Oct., 2012	93	840,572	634,840

6. EXPERIMENTS

6.1. Data Collections

We use Weibo as the microblog data source in our experiments. Weibo is the dominant microblogging service provider in China, which has more than 222 million monthly active users and 100 million daily active users[18] in September 2015.

Based on the procedures described in Section 3.1, we crawled the titles and abstracts of "ground truth" trending topics. We used the mentioned methods to retrieve microblog data in the period of September 10, 2012 to October 10, 2012. Meanwhile, we collected each topic's initializing microposts and their full reposting network in Weibo, where each topic was tracked for 3 days since it started. In total there are 168 trending topics in the dataset, and it contains 1,594,058 microposts/re-posts and 1,104,960 nodes (distinct users). We then split the topics and corresponding microposts into 2 disjoint parts for different purposes, whose statistics are listed in Table I.

— The first part contains the topics that initiated at the first 15 days, denoting as $\mathcal{E}_{tra}$. These topics' corresponding microposts are treated as training dataset Ω_{tra}. Given cost constraints K and M, the operations of Module I and II in our framework (see Section 3) and proposed efficient algorithms are applied to Ω_{tra}, thus the subset users $\mathcal{S}$ will be selected from all users set $\mathcal{V}_{tra}$ that participated in $\mathcal{E}_{tra}$.
— The rest of the trending topics $\mathcal{E}_{test}$ initiated during the last 15 days are used for testing, and all the microposts of topics $\mathcal{E}_{test}$ are regarded as full testing dataset Ω_{test} to simulate the real-time microposts exist in Weibo network. For our system, only the microposts/re-posts $\Omega_{test}^{\mathcal{S}} \subseteq \Omega_{test}$ that are generated by the selected subset users $\mathcal{S} \subseteq \mathcal{V}_{tra}$ will be used in real-time testing. The rest of the dataset $\Omega_{test} \setminus \Omega_{test}^{\mathcal{S}}$ is kept untouched during online detection and prediction, and it is only used as ground-truth in the final prediction performance evaluation.

It is worth noting that in real online environment, it is almost impossible for any third party analyzers to crawl and collect all of the newly generated microposts Ω_{test} *in real-time* because of the fact that microblog service providers are limiting API usage, as well as the high expense that will incur in gathering and processing the full-sized fresh data to fulfill the time requirements. However in our system, the needed testing dataset $\Omega_{test}^{\mathcal{S}}$ is small in size that can be easily picked by Module III of our system with small amount of Weibo API requests. And then the small-sized testing dataset can be used to conduct the detection and prediction tasks described in Module IV and V of our system framework.

To illustrate the basic characteristics of microblog dataset, some statistical analysis on training dataset Ω_{tra} are carried out, and it can be helpful in deciding some threshold configurations in data pre-processing and user selection algorithms. Distributions of microblog users' followers count is shown in Table II. From the table we can see that only 1.6% users have more than 5,000 followers, and nearly 93% users have less than 1,000 followers. In terms of per user's participation counts for trending topics in the training dataset, Table III shows that only 2.53% of users were observed in participating ≥ 4 different trending topics.

[18] Weibo Official Reports: http://ir.weibo.com/phoenix.zhtml?c=253076&p=irol-newsArticle&ID=2113781

Table II. Statistics of Followers for Microblog Users in Ω_{tra}

Followers No.	>2M	500k–2M	50k–500k	5k–50k	1k–5k	<1k
User Amount	17	338	1,580	7,363	32,536	the rest

Table III. Statistics of Trending Topic Participation Counts per Microblog User in Ω_{tra}

Participated Topics Count	1	2	3	4–10	≥10
User Amount	487,776	65,033	18,032	14,057	742
Percentage	83.29%	11.10%	3.08%	2.40%	0.13%

From the above statistics we can observe the long-tail phenomenon in microblogging network. Therefore, before running subset user selection algorithms (including the proposed algorithm *SWC*, *JNT* and all the other baseline user selection algorithms that are introduced in the next section) that compares all users in training dataset, we use pre-processing filtering mentioned in Section 3.2.1 to remove the inactive users and spam-like behaviour users. Thus the user selection process can be more efficient.

6.2. Evaluation Criteria

In this section, we will first introduce the methods that are used to compare with the proposed algorithms, and then the criteria for performance evaluation are explained.

According to statistical analysis in the previous section, there are two straightforward strategies for subset user selection problem. One strategy is iteratively picking the user that has the largest followers in the training dataset, which can be denoted as algorithm *FM*; The other is called algorithm *ECM* that iteratively picks a user who has the highest topic participation count. Besides that, we also use PageRank [Brin and Page 2012] as another baseline method *PR* in selecting subset users among all users in training dataset. In algorithm *PR*, the users that involved in topics $\mathcal{E}_{tra}$ and the re-posting actions between those users are treated as nodes and edges of a directed multi-graph. Then the nodes with highest PageRank values in the graph are selected as subset users. User selection operations in the above 3 methods will stop once the system cost constraints are reached.

In terms of system training parameter configurations (including cost constraints settings), for each offline user selection algorithm *FM*, *ECM*, *PR*, *SWC* and *JNT* we run 4 sets of parameters I through IV, listed in Table IV. Constraints of maximal microposts monitoring and processing cost M and maximal selected subset users size K are applied to all of the 5 user selection algorithms. Detection threshold X are applied when running algorithms *SWC* and *JNT*, and the same threshold is used for each topic. In other words, under identical cost constraints and parameters, our experiments will use the training dataset Ω_{tra} to select 5 different subsets of users $\mathcal{S}$ from $\mathcal{V}_{tra}$ using algorithms *FM*, *ECM*, *PR*, *SWC* and *JNT*, respectively. After selecting a subset users $\mathcal{S}$ by each algorithm in offline, real-time detection and prediction performances on topics $\mathcal{E}_{test}$ are evaluated using the corresponding real-time testing dataset $\Omega^{\mathcal{S}}_{test}$.

In general, the value of training parameter X can be set according to subset users size K as well as the desired quality of the selected users, since it can be used to control the least desired average *DoC* per subset user (denoted as d) on $\mathcal{E}_{tra}$. For example in parameter set II, if we want to have averagely at least d=8 trending topics covered per subset user on $\mathcal{E}_{tra}$, the corresponding X can be estimated by $K * d/|\mathcal{E}_{tra}|$=200*8/75=21.33≈20. In experiments, the value of d should be set based on the dataset characteristics, especially the statistics of each user's participation counts on $\mathcal{E}_{tra}$ shown in Table III. If d is set too small (e.g. d <4), a huge number of users that have lower trending topic participation counts (see Table III) could be selected into the subset, thus the subset users' overall coverage on trending topics and the training quality will not be ideal; if d is too big (e.g. d ≥10), the amount of users fulfilling

Table IV. Parameters Configuration in Offline Training

Parameters Set	M	K	X
Set I	8,000	100	10
Set II	15,000	200	20
Set III	30,000	500	40
Set IV	50,000	800	60

the coverage requirement could be quite small (also see Table III), thus the proposed algorithm will be somewhat similar to the strategy used in algorithm ***ECM***, i.e. only selecting the users with largest participation counts.

In addition to the methods that use only subset users' microposts as data sources for real-time online trending topic detection, we also run experiment using a state-of-art detection method called "Two-level clustering" [Petkos et al. 2014] on Weibo testing dataset. The algorithm is a document-pivot algorithm and is denoted as ***TLC***. It scans the contents and other features of all microposts in the full dataset Ω_{test} and put them into different clusters, and then extracts the top-ranked topics from the clusters. Algorithm ***TLC*** has no training or user selection procedures (or to say all users are selected, i.e. $\mathcal{S}$=$\mathcal{V}$ and $\Omega_{test}^{\mathcal{S}} = \Omega_{test}$ in this case), so in order to detect real-time trending topics online with algorithm ***TLC***, the full micropost dataset by all users must be obtained and processed in real-time, which is quite expensive in online environment.

In online evaluation, a trending topic e is viewed as detected if its ***DoC*** reaches or exceeds an online detecting threshold $\tilde{X}_e$ using microposts posted by subset user $\mathcal{S}$. It is noted that this online detection threshold $\tilde{X}$ in real-time testing is generally not equal to the threshold X used in offline subset user selection, because the scales of the dataset Ω_{tra} and $\Omega_{test}^{\mathcal{S}}$ are quite different. $\tilde{X}$ could also be set differently for each topic according to user preferences on its content, so a topic with lower $\tilde{X}$ can be detected more easily in online usage. Generally speaking, during real-time trending topic detection process, any topic whose ***DoC*** reaches the threshold $\tilde{X}$ will be identified as a trending topic by our system, so some of the topics that are not in the "ground truth" topic list $\mathcal{E}_{test}$ may also be detected. Denoting $\hat{\mathcal{E}}_{test}$ as all the trending topics detected by $\Omega_{test}^{\mathcal{S}}$ with ***DoC*** $\geq \tilde{X}$, the recall and precision that quantitatively measure trending topics detection performance can be defined to benchmark the results.

Detection recall rate is calculated by Eq. 21, i.e. the ratio of unique correctly matched trending topics' size to the ground-truth trending topics' size. In the equation, function $\mathbb{1}(x)$ equals to 1 if x is logical True, or 0 otherwise.

$$\text{recall}(\Omega_{test}^{\mathcal{S}}) = \frac{\sum_{e \in \mathcal{E}_{test}} \mathbb{1}(e \in \hat{\mathcal{E}}_{test})}{|\mathcal{E}_{test}|} \tag{21}$$

Detection precision rate is calculated in the following Eq. 22, i.e. the ratio of the number of total correctly matched trending topics to the number of detected trending topics. So if several trending topics in $\hat{\mathcal{E}}_{test}$ matches to the same trending topic in $\mathcal{E}_{test}$, they will be counted multiple times in precision calculation.

$$\text{precision}(\Omega_{test}^{\mathcal{S}}) = \frac{\sum_{e \in \hat{\mathcal{E}}_{test}} \mathbb{1}(e \in \mathcal{E}_{test})}{|\hat{\mathcal{E}}_{test}|} \tag{22}$$

The above precision value can be denoted as precision@All (or P@All), as it is calculated based on all detected trending topics in $\hat{\mathcal{E}}_{test}$ with Eq. 22. Sometimes however, in

order to put emphasis on the most trending topics in microblogs, only the Top-N topics (ranked by their *DoC*) in $\hat{\mathcal{E}}_{test}$ will be regarded as the detected trending topics. In this case, precision@N (or P@N) is reported by treating only Top-N topics as $\hat{\mathcal{E}}_{test}$ in Eq. 22.

It should be pointed out that the above recall and precision are based on the "ground-truth" topics $\mathcal{E}_{test}$ gathered from Weibo and other search engines' Top-10 Trends. But actually there are always some microposts discussing topics other than $\mathcal{E}_{test}$ in the dataset $\Omega^{\mathcal{S}}_{test}$, especially in users' comments when re-posting. That is to say, due to the incompleteness of real ground-truth topic lists for $\Omega^{\mathcal{S}}_{test}$, recall metric might be more convincing than precision in this evaluation scenario. Therefore, both F1-score ($\beta = 1$) and F2-score ($\beta = 2$) are calculated with Eq. 23 to benchmark the detection performance, in which $\beta > 1$ means the F-score relies more on recall than precision.

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \tag{23}$$

In terms of prediction, the predicted trending topic's popularity are measured by the commonly used Root Mean Square Error (RMSE) in topic wise manner. Let's denote $\hat{y}_e(k, \mathcal{V})$ as the predicted microposts count of topic e during a future time slot $T_s^{(k)}$ among all users $\mathcal{V}$, which is predicted by using real-time microposts dataset $\Omega^{\mathcal{S}}_{test}$ from selected users $\mathcal{S}$. Prediction RMSE of this time slot for all topics can then be calculated by Eq. 24 on the topic $e \in \mathcal{E}_{test} \cap \hat{\mathcal{E}}_{test}$ that belongs to the ground-truth topics and is detected by our system.

$$\text{RMSE}(T_s^{(k)}, \mathcal{V}|\mathcal{S}) = \sqrt{\frac{\sum_e \Big(\hat{y}_e(k, \mathcal{V}) - y_e(k, \mathcal{V})\Big)^2}{|\mathcal{E}_{test} \cap \hat{\mathcal{E}}_{test}|}} \tag{24}$$

In addition, in order to compare prediction result during a larger period of time between time slot $T_s^{(k_a)}$ and $T_s^{(k_b)}$, the mean RMSE per time slot is commonly used in later experiments using Eq. 25.

$$\overline{\text{RMSE}}(T_s^{(k_a)}, T_s^{(k_b)}) = \frac{\sum_{k=k_a}^{k_b} \text{RMSE}(T_s^{(k)}, \mathcal{V}|\mathcal{S})}{|k_b - k_a + 1|} \tag{25}$$

6.3. Topic Coverage Evaluation on Training Dataset

Before evaluating system performance on real-time testing dataset, in this section we will first exhibit the selected subset users' multi-covering performance with different user selection algorithms on topics $\mathcal{E}_{tra}$ in training dataset. That is to say, right after selecting subset users $\mathcal{S}$ by each offline user selection algorithm, we exhibit the detection performance on trending topics set $\mathcal{E}_{tra}$ in training dataset using the corresponding subset microposts $\Omega^{\mathcal{S}}_{tra}$.

During subset user selection process, algorithm *FM*, *ECM*, *PR* and *SWC* do not consider each user's prediction reward at all. In terms of *JNT*, it will consider prediction loss unless its coefficient λ is set to 0. So performances with some different λ values ranges from 0 to 1 are evaluated for algorithm *JNT* for more detailed comparison. Additionally, as algorithm *SWC* tends to select users with highest reward/cost ratio, its total cost of the selected users (denoted as $\text{Cost}_{\mathcal{S}}$) using parameters set I through IV is too low to be comparable with the other algorithms. Therefore, we run additional

Table V. Topic Coverage Comparison over Training Dataset with Selected Users

Set[19]	Alg.	$K_{\mathcal{S}}$	$Cost_{\mathcal{S}}$	Covered Topics Ratio				A-DoC	Run-Time
				$\tilde{X}=1$	$\tilde{X}=3$	$\tilde{X}=5$	$\tilde{X}=8$		
I	FM	100	3,631	50/75	27/75	15/75	11/75	3.2	0.5
	ECM	43	8,291	66/75	55/75	44/75	32/75	10.6	0.7
	PR	100	3,180	53/75	32/75	23/75	14/75	4.4	1.1
	SWC	100	32.9	74/75	20/75	8/75	2/75	2.5	32
	JNT (λ=0)	100	6,246	73/75	72/75	70/75	67/75	16.5	16
	JNT (λ=0.5)	100	4,925	73/75	73/75	71/75	62/75	15.5	233
	JNT (λ=1.0)	100	5,119	73/75	72/75	69/75	58/75	14.0	265
II	FM	200	13,392	54/75	36/75	25/75	19/75	5.8	0.4
	ECM	116	15,030	67/75	63/75	57/75	52/75	22.9	0.8
	PR	200	6,181	63/75	48/75	38/75	25/75	9.1	0.6
	SWC	200	104.8	74/75	74/75	26/75	11/75	5.3	63
	JNT (λ=0)	200	11,788	73/75	73/75	72/75	72/75	29.9	15
	JNT (λ=0.5)	200	12,140	73/75	73/75	72/75	70/75	29.5	534
	JNT (λ=1.0)	200	10,454	73/75	73/75	72/75	70/75	27.8	539
III	FM	500	29,752	67/75	53/75	45/75	37/75	14.9	0.4
	ECM	380	30,029	73/75	69/75	67/75	64/75	53.6	0.8
	PR	500	14,718	73/75	67/75	58/75	50/75	23.6	0.6
	SWC	500	456.4	74/75	74/75	74/75	74/75	12.9	152
	JNT (λ=0)	500	23,979	73/75	73/75	73/75	72/75	56.4	36
	JNT (λ=0.5)	500	23,358	73/75	73/75	73/75	72/75	56.0	1607
	JNT (λ=1.0)	500	22,891	73/75	73/75	72/75	72/75	54.9	1539
IV	FM	800	39,209	72/75	64/75	59/75	46/75	22.8	0.5
	ECM	800	49,200	74/75	70/75	67/75	66/75	88.4	0.8
	PR	800	25,491	73/75	70/75	66/75	61/75	38.8	0.6
	SWC	800	920.5	74/75	74/75	74/75	74/75	20.6	224
	SWC (K=1500)	1500	2,452	74/75	74/75	74/75	74/75	38.4	398
	SWC (K=2000)	2000	4,050	74/75	74/75	74/75	74/75	52.7	574
	SWC (K=3000)	3000	7,051	74/75	74/75	74/75	74/75	80.6	826
	JNT (λ=0)	800	34,288	74/75	74/75	73/75	72/75	79.1	116
	JNT (λ=0.5)	800	34,458	74/75	74/75	73/75	72/75	78.8	4311
	JNT (λ=1.0)	800	34,281	74/75	73/75	73/75	72/75	79.8	4248

experiments for *SWC* by enlarging the subset user size constraints K to 1500, 2000 and 3000 in parameter set IV, thus more users can be selected and included in $\mathcal{S}$. Algorithm *TLC* does not contain training or user selection process, so it is not included in the subset users' detection performance comparison over training dataset.

Table V shows the selected users performance on training dataset with different user selection algorithms, using the 4 sets of parameters mentioned in Table IV. In the table, column '$K_{\mathcal{S}}$' shows the size of subset users that are actually selected, and column 'Cost$_{\mathcal{S}}$' shows the total estimated cost $\sum_{v\in\mathcal{S}} m_v$ of these selected subset users. The values of these two columns are related to the system cost constraints K and M; Column 'Covered Topics' represents the proportion of topics in $\mathcal{E}_{tra}$ whose *Degree of Coverage* (*DoC*) reaches $\tilde{X}_e$ using $\Omega^{\mathcal{S}}_{tra}$, and the results with several different threshold $\tilde{X}_e$ values are reported; Column 'A-DoC' shows the actual average degree of coverage of topics in $\mathcal{E}_{tra}$, i.e. $\sum_{e\in\mathcal{E}_{tra}} D_e(\mathcal{S})/|\mathcal{E}_{tra}|$. Generally, higher average *DoC* means better overall trending topic coverage performance by the selected subset users $\mathcal{S}$; Column 'Run-Time' lists the running time of the offline user selection procedures in minutes[20].

Let's first focus on the comparison on topic coverage and selected user's cost. From Table V, it is easy to find out that algorithm *FM* has the lowest number of covered

[19]The parameters and constraints used in each set is listed in Table IV.

[20]We run the training experiments in a single virtual machine with 16 cores and 60G RAM from Google Computing Engine. Multiple cores are used when running algorithm *JNT*.

topics as well as the second lowest average *DoC* among the 5 user selection algorithms under the same parameter settings. It suggests that when some cost constraints are considered, following only the microblog users with largest followers is not a good strategy in covering more trending topics. This result may be a little bit beyond one's intuition, since in the real world we are more likely to follow the users with more followers, whom are often the celebrities or famous ones, to receive fresh news and information. In terms of *ECM*, as its philosophy is to select users who participate more trending topics, apparently it has higher average *DoC* and relatively large covered topic count than *FM*, *PR* and *SWC*. However, *ECM* is also the highest in $\text{Cost}_{\mathcal{S}}$ and lowest in $K_{\mathcal{S}}$ among all algorithms, which means that *ECM* is selecting the users with larger average cost per user. Therefore, it is probably not as cost-effective as other algorithms, especially when the total cost budget is tight and thus less users are selected. In contrast, per user cost for algorithm *SWC* is the lowest as it is designed to be, but its average *DoC* is also the lowest and its topic coverage is not as ideal as other algorithms when $K_{\mathcal{S}}$ is small. To improve topic coverage performance, *SWC* has to select twice or more users in amount than the other algorithms, which diverges our initial intention of small sized but representative $\mathcal{S}$. Moreover, continuously monitoring too many users will also cause extra consumptions on API requests that is strictly restricted by the microblogging company. For algorithm *PR*, its topic coverage performance is worse than *ECM* but better than *FM*, while per user cost is relatively fair. For algorithm *JNT* with $\lambda = 0$ or $\lambda > 0$, it outperforms all the other algorithms in topic coverage ratio and average *DoC* under identical constraints conditions, and its selected users' cost is moderately small. The covered topics and average *DoC* of *JNT* ($\lambda > 0$) is a little bit smaller than *JNT* ($\lambda = 0$), but the latter has slightly less $\text{Cost}_{\mathcal{S}}$. More discussions on λ value are covered in Sections 6.4.3.

Next is the discussion on the running time needed for selecting subset users. It can be found from column 'Run-Time' in Table V that algorithm *JNT* with $\lambda > 0$ runs slower than all the other algorithms. But in our opinion the longer training time in algorithm *JNT* ($\lambda > 0$) is acceptable for the following 3 reasons: 1) The whole user selection process is an offline running procedure, so the time requirement is much less urgent, and overall better detection performance is more preferred. Besides, it do take more time to run algorithm *JNT* ($\lambda > 0$) as it takes into account prediction reward while other algorithms do not; 2) During each user selection iteration, each user's reward on detection and prediction will be computed and then all the results are compared. Thus distributed processing techniques such as MapReduce [Dean and Ghemawat 2008] can be further applied to speed up the current training time; 3) Due to the rate limits on API usage by the microblog service providers, it is too expensive for third party users to collect all of the newly generated full training dataset within a short period of time, so in practice the training dataset itself would not be updated very frequently. As a consequence, it is not necessary to re-run offline user selection algorithms and to update the subset users very often if there is no new training dataset. Based on the above 3 reasons, we think it is fine to take longer time in running the user selection algorithm, so that better detection and prediction accuracy can be achieved.

Last in this subsection, the detection threshold $\tilde{X}$ is discussed. In Table V, It can be observed that trending topics' coverage changes with respect to detection threshold $\tilde{X}$. Generally speaking, detection threshold should be determined by the detectability of trending topics. For our system, this lies in finding a proper value of online detection threshold $\tilde{X}$, which is important in deciding whether a topic can be regarded as a trending topic, as well as evaluating recall and precision with Eq. 21 and Eq. 22 in the later real-time experiments with testing dataset. Thus empirically in the paper, we set $\tilde{X} = 3$ as default online detection threshold value based on the above evaluations over

training dataset, which could allow at least 95% ($\geq$72/75) topics in training dataset to be marked as trending topics, using ***JNT*** ($\lambda >= 0$) with the lowest cost constraints parameter set I. It should be pointed out that in later online evaluations, $\tilde{X} = 3$ is also globally used for all the 4 parameters set I through IV for comparison convenience across different constraints settings. However in practical usage, $\tilde{X}$ can be set larger than 3 accordingly when selected users' size is bigger.

6.4. Evaluation on Real-time Testing Dataset

After benchmarking topic coverage with selected subset users on training dataset, the proposed system performance is then evaluated on the real-time online testing dataset. With the subset users $\mathcal{S}$ that are selected from offline training dataset Ω_{tra} by different user selection algorithm, we use only their microposts in the testing dataset $\Omega_{test}^{\mathcal{S}} \subseteq \Omega_{test}$ to detect and predict the trending topics $\mathcal{E}_{test}$ online. For prediction, we set t_τ and t_κ to be 6 hours and 30 hours since each topic is initiated. This means we observe the first t_τ=6 hours of a topic's counting time-series using only the selected users microposts $\Omega_{test}^{\mathcal{S}}$, and then predict its popularity among all users $\mathcal{V}$ in the next $t_\kappa - t_\tau$=24 hours.

Table VI shows the real-time online testing performance results of ***FM***, ***ECM***, ***PR***, ***SWC*** and ***JNT*** using corresponding dataset $\Omega_{test}^{\mathcal{S}}$ by their selected subset users $\mathcal{S}$. In the table, column 'Recall' reports the recall rate using Eq. 21. In order to exhibit more detailed results, recall rates with various online detection threshold $\tilde{X}$ are reported. In column 'Precision', 'F1-Score' and 'F2-Score', as the purpose of using the same online detection threshold $\tilde{X}$ value for different parameters set in these evaluations is already explained at the end of previous subsection, the evaluation results of Eq. 22 and Eq. 23 with $\tilde{X} = 3$ are listed. In some cases, people may prefer to compare the precision rate based on the top-N detected trending topics in $\hat{\mathcal{E}}_{test}$. Thus the precision@N (N=30 & 50) as well as corresponding F1/F2 scores (N=50) are also listed in the performance comparison table. Column 'A-D' in Table VI is the average degree of coverages for topics in $\mathcal{E}_{test}$, and higher value of it means the trending topics are still likely to be detected even if threshold $\tilde{X}$ is set to be higher; Column 'T-G' shows the average time difference *in hours* that trending topics detected by our system *before* they are published by Weibo and Baidu/Sogou/Soso search engines Top Trend Lists, in which the posting time of the last micropost that makes topic e's *Degree of Coverage* reach detection threshold $\tilde{X}_e$ is regarded as the time point that topic e is detected as a trending topic by our system. In other words, column 'T-G' reflects the time gained by using our system than relying on the official trends list to get trending topics. Column 'RMSE' in the table reports the average popularity predicting RMSE per time-slot for the next 24 hours using Eq. 25. Column 'R-T' shows the total running time of both online detection and prediction procedures in minutes[21].

Besides evaluating online testing performance of the above 5 algorithms that uses a small subset of dataset $\Omega_{test}^{\mathcal{S}}$, we also evaluate detection performance of algorithm ***TLC*** that needs to use and process the full testing dataset Ω_{test} during online detection. That is to say, in order to detect trending topics with algorithm *TLC in real-time*, all users' microposts in the microblog website must be gathered quickly and continuously, so that all these microposts can then be processed for clustering and topic extraction in near real-time. In practical online environment, the size of microblog users and their newly generated microposts are extremely huge, thus the cost of collecting and processing such full-sized dataset in real-time is prohibitive. The detection performance of algorithm ***TLC*** using Ω_{test} and its running time (in minutes, without the data col-

[21] In all online testing experiments, a commodity computer with 2.0GHz CPU and 16G RAM is used.

lecting time) is listed in Table VII, in which the detected topics containing no less than $\tilde{X}$ microposts are treated as trending topics. As the size of input dataset Ω_{test} (shown in Table I) is much bigger than any Ω_{test}^{S}, detection performance with various online detection threshold $\tilde{X}$ much bigger than 3 are reported.

In the following subsections, performances by different algorithms with different parameter settings are compared and discussed in detail.

6.4.1. Discussions on Performance of Different Algorithms. Viewing the online performance of all the 5 user selection algorithms ***FM***, ***ECM***, ***PR***, ***SWC*** and ***JNT*** under each cost parameters Set I through IV as a whole in Table VI, the F-scores, average ***DoC*** and detection time gain apparently increase as value of cost constraints M and K increase from Set I to Set IV. This shows that system cost budget settings are indeed affecting online testing cost as well as the overall system online performance.

At first, we discuss the detection performance with algorithm ***TLC***, for which the system cost is not limited at all and all the testing dataset are used in online testing. According to the performance shown in Table VII, F-scores of algorithm ***TLC*** is indeed better than ***JNT*** and other algorithms (shown in Table VI) when ***TLC***'s $\tilde{X} < 300$. But please keep in mind that the former algorithm uses microposts from 0.6 million users and the latter uses only microposts from no more than 800 selected users. More importantly, the running time of online testing for ***TLC*** is also increasing as the size of its input dataset needed is apparently larger, and it takes more than 115 times (527.5 min v.s. 4.5 min) longer than any other algorithms with user selection mechanism. That is to say, although the detection performance of algorithm ***TLC*** is the best, it is not practically suitable to accomplish real-time online trending topics detection task by using the full dataset directly, let alone the cost and time needed to capture such full dataset in real time, especially for third-party analyzers who need to crawl the dataset on their own. As a conclusion, the *cost-effectiveness* of an algorithm should be considered seriously in online environment since there are always some kinds of cost constraints in practise.

Therefore, based on performance shown in Table VI and $\text{Cost}_{\mathcal{S}}$ shown in Table V, we draw a figure showing performance v.s. total cost of selected users ($\text{Cost}_{\mathcal{S}}$) in Fig. 2 to compare the cost-effectiveness of all the algorithms with user selection procedures. Algorithm ***TLC*** is excluded in this figure as it uses all users' real-time microposts and thus its total cost is too high to compare. In Fig. 2, x-axis is the total cost of selected users ($\text{Cost}_{\mathcal{S}}$) listed in Table V; y-axis of the sub-figures are F1-score, F2-score, average ***DoC*** (logarithmic scaled) and RMSE, respectively. The dash-dotted purple horizon line in the bottom left sub-figure indicates the detection threshold $\tilde{X}$, which is set to default value 3 for all topics in our experiments. All the other solid lines (representing ***JNT***) and dash lines (representing the other algorithms, including ***SWC*** with user size K bigger than 800) in the figure represent results using the microposts of the subset users that are selected by different user selection algorithms.

In the first place, we compare the recall and precision rate of different algorithms shown in Table VI as well as the F-scores shown in Fig. 2, for parameter set I through IV. Similar to the topic coverage performance in training dataset, in real-time evaluations algorithm ***FM*** and ***SWC*** have lower recall rate and average degree of coverages among all algorithms. The low average ***DoC*** suggests that their detection performance is too sensitive on cost constraints and online detection threshold. When cost constraint M decreases and their average ***DoC*** become lower than the detection threshold, their detection performance will be too low to be competitive. In contrast, average ***DoC*** of algorithm ***JNT*** ($\lambda \geq 0$) under various cost constraints are always beyond the detection threshold, so its recall is better all the time. The detection performance of algorithm ***PR*** is a little bit better than ***FM*** and ***SWC***, but worse than ***ECM*** and ***JNT***. For

Table VI. Real-Time Online Performance over Testing Dataset $\Omega^{\mathcal{S}}_{test}$ with Selected Users

Set	Alg[22]	Recall				Precision ($\tilde{X}$=3)			F1 ($\tilde{X}$=3)		F2 ($\tilde{X}$=3)		A-D	T-G[23]	RMSE	R-T[24]
		$\tilde{X}=1$	$\tilde{X}=3$	$\tilde{X}=5$	$\tilde{X}=8$	P@All	P@30	P@50	P@All	P@50	P@All	P@50				
I	FM	27/93	27/93	14/93	3/93	28/31	27/30	28/31	0.4394	0.4394	0.3359	0.3359	1.5	1.0	58.33	3.5
	ECM	66/93	66/93	61/93	42/93	88/108	27/30	44/50	0.7586	0.7857	0.7285	0.7383	9.7	12.6	58.18	3.7
	PR	23/93	23/93	15/93	5/93	27/32	26/30	27/32	0.3825	0.3825	0.2880	0.2880	1.7	1.0	57.67	3.2
	SWC	4/93	4/93	2/93	0/93	4/4	4/4	4/4	0.0825	0.0825	0.0532	0.0532	0.2	-0.4	60.71	3.2
	JNT (λ=0)	75/93	75/93	66/93	52/93	103/121	28/30	46/50	0.8282	0.8595	0.8150	0.8269	11.1	19.7	41.59	3.6
	JNT (λ=0.5)	71/93	71/93	61/93	37/93	93/112	27/30	41/50	0.7955	0.7907	0.7759	0.7741	8.3	15.7	40.90	3.6
	JNT (λ=1.0)	71/93	71/93	57/93	39/93	89/105	28/30	45/50	0.8033	0.8261	0.7789	0.7873	7.9	14.0	40.94	3.6
II	FM	33/93	33/93	22/93	9/93	36/41	26/30	36/41	0.5054	0.5054	0.4028	0.4028	2.7	1.5	57.08	3.6
	ECM	79/93	79/93	75/93	61/93	138/171	27/30	44/50	0.8277	0.8645	0.8406	0.8554	20.8	19.4	47.54	4.1
	PR	47/93	47/93	31/93	16/93	56/64	27/30	43/50	0.6407	0.6366	0.5520	0.5508	4.1	7.1	56.12	3.4
	SWC	14/93	14/93	5/93	1/93	15/16	15/16	15/16	0.2594	0.2594	0.1809	0.1809	0.8	0.8	45.55	3.3
	JNT (λ=0)	80/93	80/93	75/93	63/93	125/154	28/30	45/50	0.8352	0.8797	0.8501	0.8679	18.2	22.4	42.52	3.9
	JNT (λ=0.5)	83/93	83/93	76/93	61/93	128/159	29/30	44/50	0.8465	0.8862	0.8735	0.8900	18.0	23.3	42.94	3.8
	JNT (λ=1.0)	80/93	80/93	71/93	59/93	118/144	27/30	45/50	0.8393	0.8797	0.8517	0.8679	15.9	21.2	43.10	3.8
III	FM	62/93	62/93	42/93	31/93	77/89	27/30	44/50	0.7531	0.7586	0.6987	0.7006	7.8	9.4	40.57	3.7
	ECM	83/93	83/93	82/93	78/93	176/236	28/30	45/50	0.8125	0.8962	0.8587	0.8940	46.8	29.7	52.70	4.6
	PR	67/93	67/93	53/93	43/93	105/133	26/30	43/50	0.7534	0.7841	0.7333	0.7446	11.9	19.0	41.66	3.6
	SWC	36/93	36/93	17/93	8/93	42/44	29/30	42/44	0.5508	0.5508	0.4393	0.4393	2.4	0.6	42.68	3.4
	JNT (λ=0)	85/93	85/93	83/93	77/93	161/207	29/30	44/50	0.8404	0.8967	0.8831	0.9070	33.8	26.3	47.75	4.1
	JNT (λ=0.5)	85/93	85/93	84/93	78/93	162/209	28/30	45/50	0.8388	0.9069	0.8824	0.9111	33.5	27.9	46.79	4.1
	JNT (λ=1.0)	86/93	86/93	82/93	76/93	158/203	27/30	44/50	0.8452	0.9018	0.8912	0.9154	32.2	30.1	44.88	4.1
IV	FM	74/93	74/93	55/93	42/93	104/128	28/30	43/50	0.8040	0.8266	0.7990	0.8078	13.0	14.8	42.44	3.9
	ECM	87/93	87/93	85/93	84/93	198/266	28/30	45/50	0.8290	0.9174	0.8898	0.9282	76.4	39.0	65.85	5.0
	PR	75/93	75/93	66/93	55/93	135/174	28/30	45/50	0.7909	0.8507	0.8001	0.8236	20.2	25.2	41.07	3.9
	SWC	50/93	50/93	32/93	14/93	63/69	29/30	44/50	0.6768	0.6675	0.5858	0.5830	4.2	7.4	42.18	4.4
	SWC (K=1.5k)	64/93	64/93	47/93	30/93	82/98	29/30	44/50	0.7552	0.7724	0.7135	0.7195	7.9	9.3	41.31	3.7
	SWC (K=2k)	74/93	74/93	59/93	38/93	99/122	28/30	44/50	0.8035	0.8357	0.7988	0.8112	11.2	13.8	40.64	3.7
	SWC (K=3k)	80/93	80/93	68/93	53/93	121/154	28/30	44/50	0.8213	0.8700	0.8442	0.8641	18.3	17.4	98.51	3.9
	JNT (λ=0)	87/93	87/93	85/93	80/93	191/241	29/30	46/50	0.8581	0.9277	0.9029	0.9323	47.0	37.3	49.51	4.4
	JNT (λ=0.5)	86/93	86/93	83/93	78/93	192/241	28/30	45/50	0.8560	0.9122	0.8959	0.9197	47.0	33.2	43.53	4.4
	JNT (λ=1.0)	88/93	88/93	85/93	77/93	189/242	28/30	44/50	0.8557	0.9119	0.9078	0.9322	46.5	31.3	47.80	4.5

[22] Please also refer to the corresponding selected users' cost ($Cost_{\mathcal{S}}$) listed in Table V as well as Fig. 2 to compare the estimated online cost of each algorithm.

[23] The unit is in hours. It shows the time gained by using our system than relying on the official trends list to get trending topics.

[24] The unit is in minutes. It shows the total running time of both online detection and prediction procedures for algorithms with user selection mechanism, which is much faster than the running time of algorithm *TLC* listed in Table VII.

ACM Transactions on Information Systems, Vol. 0, No. 0, Article 00, Publication date: September 2016.

Table VII. Real-time Online Performance over Testing Dataset Ω_{test} using Algorithm *TLC*

Threshold	$\tilde{X} = 50$	$\tilde{X} = 100$	$\tilde{X} = 300$	$\tilde{X} = 500$	$\tilde{X} = 800$	$\tilde{X} = 1000$
Recall	90/93	88/93	83/93	78/93	64/93	59/93
Precision@All	999/1058	755/796	404/430	281/297	200/210	170/177
Precision@30	30/30	30/30	30/30	30/30	30/30	30/30
Precision@50	50/50	50/50	50/50	50/50	50/50	50/50
F1-Score (P@All)	0.9559	0.9474	0.9154	0.8892	0.7990	0.7641
F1-Score (P@50)	0.9836	0.9724	0.9432	0.9123	0.8153	0.7763
F2-Score (P@All)	0.9629	0.9467	0.9015	0.8582	0.7286	0.6806
F2-Score (P@50)	0.9740	0.9565	0.9121	0.8667	0.7339	0.6845
Average DoC	7,890	7,697	7,024	6,507	5,963	5,673
Running Time	527.5					

algorithm *SWC* with larger user size constraints $K > 800$, its detection performance can be comparable with *PR*, but its selected users size is several times bigger than all the other algorithms that diverges the intention for selecting small-sized but representative subset users, and its performance is still worse than *JNT*. In terms of algorithm *ECM*, it can be found from Fig. 2 that its F-scores are higher than *PR*, *SWC* and *FM*, but its detection performance is still worse than *JNT* in most cases while its cost is even larger. As *JNT* ($\lambda \geq 0$) has the highest F-scores under the same cost constraints, it is the most cost-effective one in detection performance among all algorithms.

Next, prediction performance is compared using column 'RMSE' in Table VI and bottom right sub-figure of Fig. 2. RMSE of *ECM* becomes higher than the others algorithms when its selected users size is getting larger, and the RMSE of algorithm *SWC* with larger user size constraints $K > 800$ is also very high. Due to the fact that algorithm *ECM* tends to select users with larger cost and the selected users' size of *SWC* with larger K is much bigger, the most reasonable explanation for their RMSE increment is that prediction accuracy are affected by the "noise" microposts in their selected users' microposts, since the two algorithms do not consider the users' prediction accuracy during subset user selection. In contrast, RMSE of *FM* and *PR* are higher when cost constraints are low, as in these cases they are short of valuable users' microposts for prediction. In other words, the prediction performance of the above 4 methods are too sensitive on cost constraints and thus not cost-effective. In general, RMSE of algorithm *JNT*($\lambda > 0$) is quite stable and relatively low among the 4 sets of parameters.

In light of the above, the proposed algorithm *JNT* has the overall best joint online detection and prediction performance over testing dataset within cost constraints.

6.4.2. Discussions on Early Detection and Prediction. In Table VI, column 'T-G' shows the average detection time advantage of our system in hours, which are always positive using proposed algorithm *JNT* ($\lambda \geq 0$). It means that our system can detect the trending topics much earlier than they appear in the official Trends Lists of Weibo and search engines. In our experiments the observation time t_τ needed for future popularity prediction is set to 6 hours since the trending topic is initiated and detected by our system. Removing the 6 hours from column 'T-G' in Table VI, the result is still decent, thus we can accomplish the joint tasks of trending topic detection and prediction several hours in advance of the official lists. This reveals another advantage of our proposed framework: it is a third party system that is very practical in both early trending topic detection and early prediction for real microblogging services, using a relatively small budget on cost.

6.4.3. Discussions on λ. During the user selection procedure for algorithm *JNT* with λ greater than 0, the reward of a user consists of both detection reward and prediction reward. During user selection, the system will consider more about selected user's

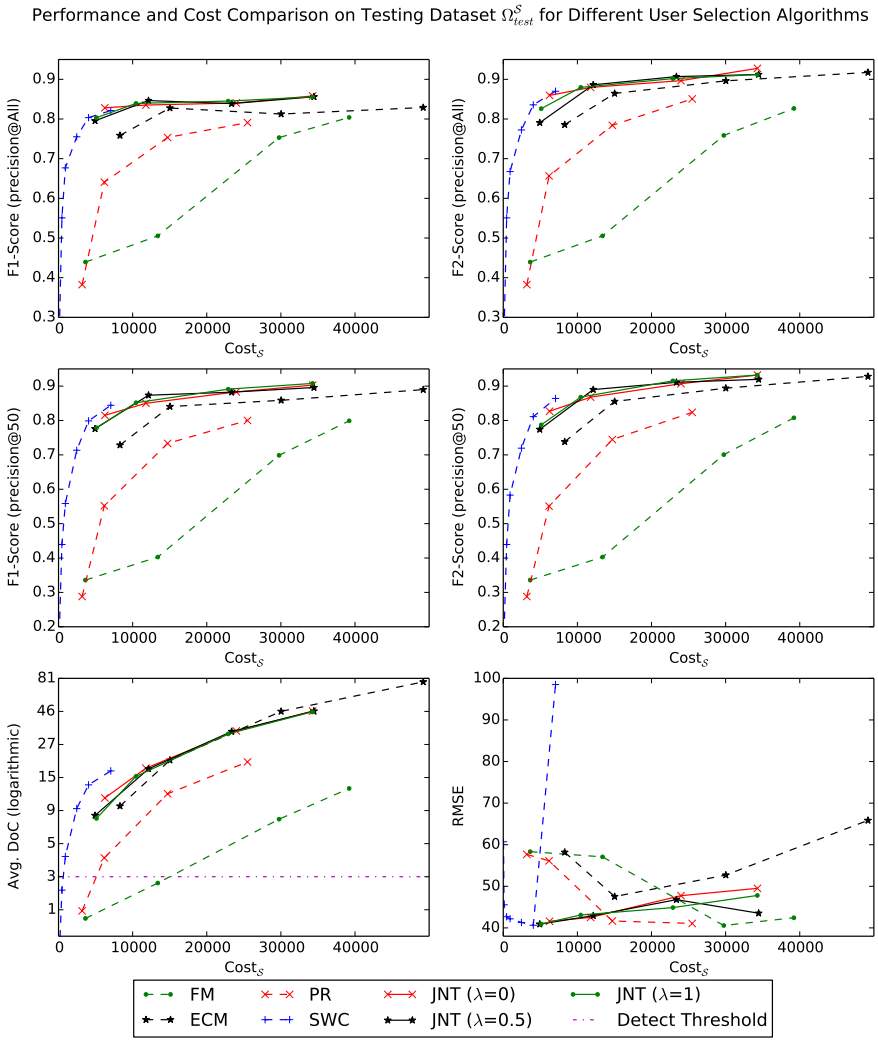

Fig. 2. Performance and cost comparison on testing dataset Ω^S_{test} with selected users. X-axis in the figure shows the total cost of selected users. Under the same cost constraints, the proposed algorithm *JNT* shows better performance in F-Scores, average degree of coverage, and lower RMSE than other algorithms.

contributions on prediction accuracy when value of λ increases; and the system will focus more on user's topic detection ability when λ drops.

To exhibit the effect of λ, we run additional experiments with various λ values. Taking experiments using parameter set III as an example, detection and prediction performance with different λ values are shown in Table VIII. The average RMSE per time slot within every hour for the next 24 hours are also shown in Fig. 3.

From Table VIII, it can be seen that recall rate drops a little bit as λ increases from 0. In the meanwhile corresponding prediction performance improves as expected, which can be observed in Fig. 3 and column 'RMSE' in Table VIII. Based on this trend, if λ is too high (e.g. $>$5), the detection performance, average *DoC* and 'T-G' (time gained)

Table VIII. Performance Comparison for Algorithm *JNT* with Different λ

Set	λ	Recall			Precision	F1 ($\tilde{X}$=3)	F2 ($\tilde{X}=3$)	A-D	T-G	RMSE
		$\tilde{X}=3$	$\tilde{X}=5$	$\tilde{X}=8$	P@All	P@All	P@All			
III	0	85/93	83/93	77/93	161/207	0.8404	0.8831	33.8	26.3	47.75
	0.1	86/93	84/93	77/93	162/209	0.8433	0.8904	33.2	25.8	48.21
	0.5	85/93	84/93	78/93	162/209	0.8388	0.8824	33.5	27.9	46.79
	1	86/93	82/93	76/93	158/203	0.8452	0.8912	32.2	30.1	44.88
	2	86/93	82/93	76/93	154/198	0.8449	0.8911	32.5	27.1	42.96
	5	84/93	82/93	74/93	141/175	0.8517	0.8819	27.8	25.1	43.87

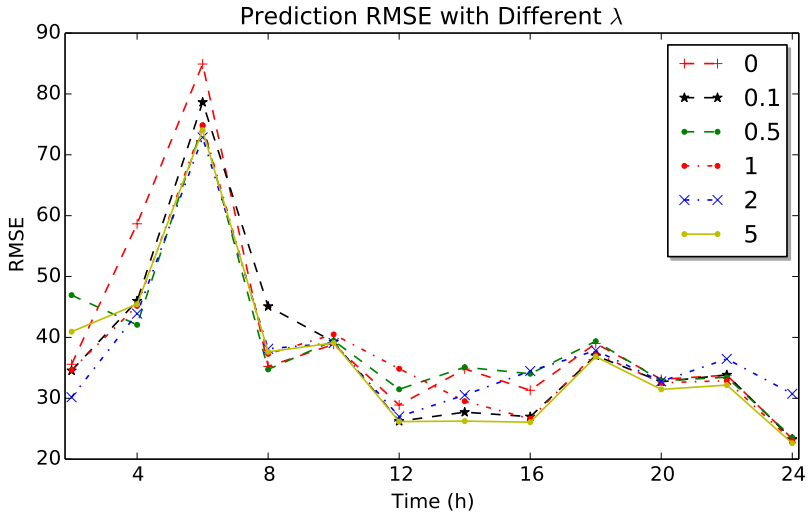

Fig. 3. Popularity prediction RMSE comparison for Algorithm *JNT* with different λ using parameter Set III. It shows the average RMSE per time slot within each hour for the next 24h since the prediction begins.

will drop a lot, thus the joint performance will not be ideal. Therefore, we should pay attention to the weight of prediction during user selection, so as to maintain good detection and prediction accuracy, as well as the timeliness to ensure the time gained is still enough to make early detection and prediction. In our datasets, it is desirable to set λ between 0.5 and 2.

6.4.4. Discussions on other coefficients in algorithm JNT. Besides using λ to indicate the weights of prediction performance in user selection, algorithm *JNT* also uses concept of *dynamic reward* and *dynamic cost boundary* to improve trending topic coverage that are explained in section 5.2. Here, we exhibit the impact of different α and γ values in Eq. 12 and Eq. 14 by experiments. The result comparison is shown in Fig. 4, in which parameters set II and λ=0.5 are used. The x-axis in the upper and lower sub-figure shows the varying α values and γ values, respectively. Y-axis are their corresponding F1-score, F2-score and RMSE with corresponding testing dataset Ω^S_{test}.

When α is smaller, the detection reward for covering topics with lower *DoC* will become a little bit larger, thus these topics are more urgent to be covered in user selection. In terms of γ, if its value is too small, the cost boundary will become quite large and users with quite big cost will be selected first, and it might not be so cost-effective. According to the results shown in Fig. 4, $\alpha = 0.01$ and $\gamma = 0.7$ are chosen as the default values in all experiments with algorithm *JNT*, as their F1 and F2 scores are the best and the RMSE are relatively small with these coefficient values.

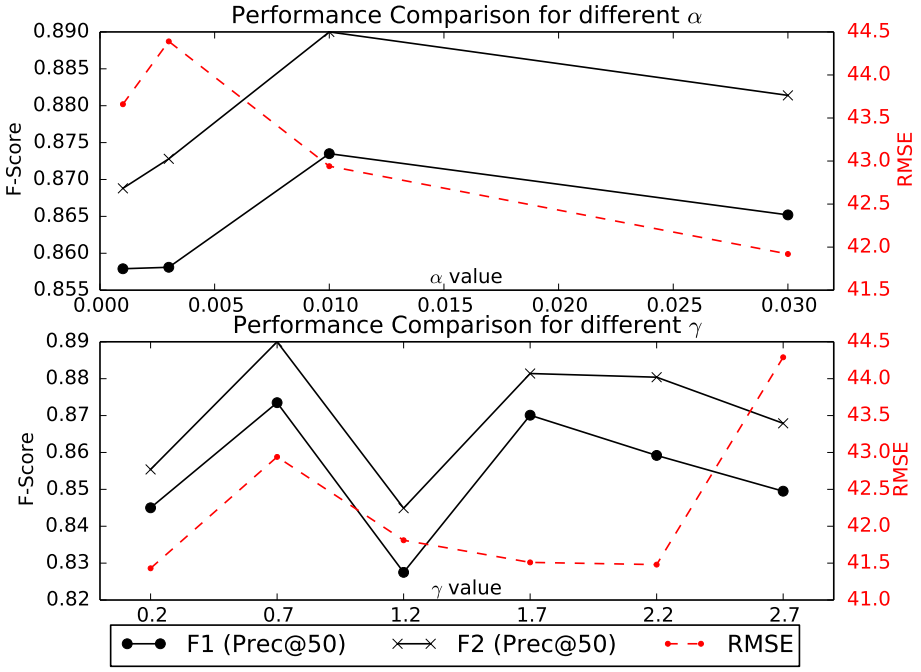

Fig. 4. Performance comparison for Algorithm *JNT* with different values of α and γ. The two coefficients α and γ are defined in Eq. 12 and Eq. 14. Parameter Set II and λ=0.5 are used in the comparison.

7. CONCLUSIONS AND FUTURE WORK

In this paper we present a cost-effective online trending topic detection and prediction system for microblogging services, from a third party perspective. The proposed system can run under strict resource constraints while not compromising on the performance in detection and prediction. In order to satisfy resource budget, online trending topic multi-coverage requirements as well as popularity prediction accuracy, we propose the notion of utilizing a subset of selected users to accomplish the task. We formulate the subset user selection problem as optimization tasks, and propose efficient algorithms to solve the problem.

To evaluate the online performance of joint detection and prediction system, we collect the experiment data from real microblogging service networks, and utilize them into offline dataset and real-time testing dataset that are used differently in our experiment settings. The performance comparison results prove that the proposed algorithm *JNT* outperforms the state-of-art algorithms in detection and prediction accuracy whiling being cost-effective. Experiments show that by tracking only 500 users out of 0.6 million microblog users and processing at most 30,000 microposts daily, about 92% of the trending topics among all users could be detected and then predicted by the proposed system. Moreover, the trending topics and their future popularity can be detected and predicted by our system much earlier than when they are published by official Trends List in microblogging services. As the proposed system is cost-effective, it is very practically applicable to real-world usage.

In future works, we plan to extend the system, algorithm and experiments on different categories of microposts, so users with different interests can be selected and utilized for topic analysis. Distributed computing technology can be applied to the user

selection algorithm to speed up the training. More factors in the dataset can also be used in the algorithms, for example the time factors that a user tends to participate in trending topics. In addition, new mechanism such as dynamically updating selected users according to overall performance or time factors is another interesting area.

ACKNOWLEDGMENTS

The authors would like to thank all colleagues and reviewers that contribute to this paper. The authors would also like to thank all the creators and maintainers of the tools we used in experiments.

REFERENCES

Mohamed Ahmed, Stella Spagna, Felipe Huici, and Saverio Niccolini. 2013. A Peek into the Future: Predicting the Evolution of Popularity in User Generated Content. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining (WSDM '13)*. ACM, New York, NY, USA, 607–616. DOI:http://dx.doi.org/10.1145/2433396.2433473

James Allan (Ed.). 2002. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers, Norwell, MA, USA. http://dl.acm.org/citation.cfm?id=772260

Foteini Alvanaki, Sebastian Michel, Krithi Ramamritham, and Gerhard Weikum. 2012. See What's enBlogue: Real-time Emergent Topic Identification in Social Media. In *Proceedings of the 15th International Conference on Extending Database Technology (EDBT '12)*. ACM, New York, NY, USA, 336–347. DOI:http://dx.doi.org/10.1145/2247596.2247636

Sitaram Asur, Bernardo A. Huberman, Gabor Szabo, and Chunyan Wang. 2011. Trends in Social Media: Persistence and Decay. *SSRN Electronic Journal* (Feb. 2011). DOI:http://dx.doi.org/10.2139/ssrn.1755748

Roja Bandari, Sitaram Asur, and Bernardo A. Huberman. 2012. The Pulse of News in Social Media: Forecasting Popularity. In *Proceedings of the Sixth International Conference on Weblogs and Social Media (ICWSM '12)*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM12/paper/view/4646

Livio Bertacco. 2006. *Exact and Heuristic Methods for Mixed Integer Linear Programs*. Ph.D. Dissertation. Ph. D. thesis, Università degli Studi di Padova.

Bin Bi, Yuanyuan Tian, Yannis Sismanis, Andrey Balmin, and Junghoo Cho. 2014. Scalable Topic-specific Influence Analysis on Microblogs. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. ACM, New York, NY, USA, 513–522. DOI:http://dx.doi.org/10.1145/2556195.2556229

Petko Bogdanov, Michael Busch, Jeff Moehlis, Ambuj K. Singh, and Boleslaw K. Szymanski. 2013. The Social Media Genome: Modeling Individual Topic-specific Behavior in Social Media. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM '13)*. ACM, New York, NY, USA, 236–242. DOI:http://dx.doi.org/10.1145/2492517.2492621

Sergey Brin and Lawrence Page. 2012. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer Networks* 56, 18 (2012), 3825–3833. DOI:http://dx.doi.org/10.1016/j.comnet.2012.10.007

Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. 2010. Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining (MDMKDD '10)*. ACM, New York, NY, USA, Article 4, 10 pages. DOI:http://dx.doi.org/10.1145/1814245.1814249

Kai Chen, Yi Zhou, Hongyuan Zha, Jianhua He, Pei Shen, and Xiaokang Yang. 2013b. Cost-effective Node Monitoring for Online Hot Eventdetection in Sina Weibo Microblogging. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13 Companion)*. ACM, New York, NY, USA, 107–108. DOI:http://dx.doi.org/10.1145/2487788.2487837

Le Chen, Chi Zhang, and Christo Wilson. 2013a. Tweeting Under Pressure: Analyzing Trending Topics and Evolving Word Choice on Sina Weibo. In *Proceedings of the First ACM Conference on Online Social Networks (COSN '13)*. ACM, New York, NY, USA, 89–100. DOI:http://dx.doi.org/10.1145/2512938.2512940

Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient Influence Maximization in Social Networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 199–208. DOI:http://dx.doi.org/10.1145/1557019.1557047

Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM* 51, 1 (Jan. 2008), 107–113. DOI:http://dx.doi.org/10.1145/1327452.1327492

Pedro Domingos and Matt Richardson. 2001. Mining the Network Value of Customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*. ACM, New York, NY, USA, 57–66. DOI:http://dx.doi.org/10.1145/502512.502525

Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuan Zha. 2013. Scalable Influence Estimation in Continuous-Time Diffusion Networks. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 3147–3155. http://papers.nips.cc/paper/4857-scalable-influence-estimation-in-continuous-time-diffusion-networks.pdf

Pablo A. Estevez, Pablo Vera, and Kazumi Saito. 2007. Selecting the Most Influential Nodes in Social Networks. In *2007 International Joint Conference on Neural Networks*. IEEE, 2397–2402. DOI:http://dx.doi.org/10.1109/IJCNN.2007.4371333

Schubert Foo and Hui Li. 2004. Chinese word segmentation and its effect on information retrieval. *Information Processing & Management* 40, 1 (Jan. 2004), 161–190. DOI:http://dx.doi.org/10.1016/S0306-4573(02)00079-1

Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Huan Liu, and Philip S. Yu. 2007. Time-dependent Event Hierarchy Construction. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. ACM, New York, NY, USA, 300–309. DOI:http://dx.doi.org/10.1145/1281192.1281227

Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. 2012. Inferring Networks of Diffusion and Influence. *ACM Trans. Knowl. Discov. Data* 5, 4, Article 21 (Feb. 2012), 37 pages. DOI:http://dx.doi.org/10.1145/2086737.2086741

Yi Han, Lei Deng, Binying Xu, Lumin Zhang, Bin Zhou, and Yan Jia. 2013. Predicting the Social Influence of Upcoming Contents in Large Social Networks. In *Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service (ICIMCS '13)*. ACM, New York, NY, USA, 17–22. DOI:http://dx.doi.org/10.1145/2499788.2499834

Guangyan Huang, Jing He, Yanchun Zhang, Wanlei Zhou, Hai Liu, Peng Zhang, Zhiming Ding, Yue You, and Jian Cao. 2015. Mining Streams of Short Text for Analysis of World-wide Event Evolutions. *World Wide Web* 18, 5 (2015), 1201–1217. DOI:http://dx.doi.org/10.1007/s11280-014-0293-1

David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, New York, NY, USA, 137–146. DOI:http://dx.doi.org/10.1145/956750.956769

Andrey Kupavskii, Alexey Umnov, Gleb Gusev, and Pavel Serdyukov. 2013. Predicting the Audience Size of a Tweet. In *Proceedings of the Seventh International Conference on Weblogs and Social Media (ICWSM '13)*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6077

Chung-Hong Lee, Hsin-Chang Yang, Tzan-Feng Chien, and Wei-Shiang Wen. 2011. A Novel Approach for Event Detection by Mining Spatio-temporal Information on Microblogs. In *2011 International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 254–259. DOI:http://dx.doi.org/10.1109/ASONAM.2011.74

Jure Leskovec, Lars Backstrom, and Jon Kleinberg. 2009. Meme-tracking and the Dynamics of the News Cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 497–506. DOI:http://dx.doi.org/10.1145/1557019.1557077

Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective Outbreak Detection in Networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. ACM, New York, NY, USA, 420–429. DOI:http://dx.doi.org/10.1145/1281192.1281239

Juha Makkonen, Helena Ahonen-Myka, and Marko Salmenkivi. 2004. Simple Semantics in Topic Detection and Tracking. *Information Retrieval* 7, 3-4 (sep 2004), 347–368. DOI:http://dx.doi.org/10.1023/B:INRT.0000011210.12953.86

Michael Mathioudakis and Nick Koudas. 2010. TwitterMonitor: Trend Detection over the Twitter Stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD '10)*. ACM, New York, NY, USA, 1155–1158. DOI:http://dx.doi.org/10.1145/1807167.1807306

Zhongchen Miao, Kai Chen, Yi Zhou, Hongyuan Zha, Jianhua He, Xiaokang Yang, and Wenjun Zhang. 2015. Online Trendy Topics Detection in Microblogs with Selective User Monitoring under Cost Constraints. In *2015 IEEE International Conference on Communications (ICC '15)*. 1194–1200. DOI:http://dx.doi.org/10.1109/ICC.2015.7248485

Fred Morstatter, Jürgen Pfeffer, and Huan Liu. 2014. When is It Biased?: Assessing the Representativeness of Twitter's Streaming API. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14 Companion)*. ACM, New York, NY, USA, 555–556. DOI:http://dx.doi.org/10.1145/2567948.2576952

Fred Morstatter, Jürgen Pfeffer, Huan Liu, and Kathleen M. Carley. 2013. Is the Sample Good Enough? Comparing Data from Twitter's Streaming API with Twitter's Firehose. In *Proceedings of the Seventh*

International Conference on Weblogs and Social Media (ICWSM '13). 400–408. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6071

Seth A. Myers and Jure Leskovec. 2014. The Bursty Dynamics of the Twitter Information Network. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. ACM, New York, NY, USA, 913–924. DOI: http://dx.doi.org/10.1145/2566486.2568043

Mor Naaman, Hila Becker, and Luis Gravano. 2011. Hip and Trendy: Characterizing Emerging Trends on Twitter. *Journal of the American Society for Information Science and Technology* 62, 5 (may 2011), 902–918. DOI: http://dx.doi.org/10.1002/asi.21489

Ramasuri Narayanam and Yadati Narahari. 2011. A Shapley Value-Based Approach to Discover Influential Nodes in Social Networks. *IEEE Transactions on Automation Science and Engineering* 8, 1 (jan 2011), 130–147. DOI: http://dx.doi.org/10.1109/TASE.2010.2052042

Aditya Pal and Scott Counts. 2011. Identifying Topical Authorities in Microblogs. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM '11)*. ACM, New York, NY, USA, 45–54. DOI: http://dx.doi.org/10.1145/1935826.1935843

R. Papka and J. Allan. 1998. *On-Line New Event Detection Using Single Pass Clustering*. Technical Report. Amherst, MA, USA.

Georgios Petkos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2014. Two-level Message Clustering for Topic Detection in Twitter. In *Proceedings of the SNOW 2014 Data Challenge*. 49–56. http://ceur-ws.org/Vol-1150/petkos.pdf

Polina Rozenshtein, Aris Anagnostopoulos, Aristides Gionis, and Nikolaj Tatti. 2014. Event Detection in Activity Networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 1176–1185. DOI: http://dx.doi.org/10.1145/2623330.2623674

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. ACM, New York, NY, USA, 851–860. DOI: http://dx.doi.org/10.1145/1772690.1772777

Erich Schubert, Michael Weiler, and Hans-Peter Kriegel. 2014. SigniTrend: Scalable Detection of Emerging Topics in Textual Streams by Hashed Significance Thresholds. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 871–880. DOI: http://dx.doi.org/10.1145/2623330.2623740

Aleksandr Simma and Michael I. Jordan. 2010. Modeling Events with Cascades of Poisson Processes. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI '10)*. AUAI Press, 546–555. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=2139&proceeding_id=26

Oren Tsur and Ari Rappoport. 2012. What's in a Hashtag?: Content Based Prediction of the Spread of Ideas in Microblogging Communities. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM '12)*. ACM, New York, NY, USA, 643–652. DOI: http://dx.doi.org/10.1145/2124295.2124320

Yu Wang, Gao Cong, Guojie Song, and Kunqing Xie. 2010. Community-based Greedy Algorithm for Mining top-K Influential Nodes in Mobile Social Networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 1039–1048. DOI: http://dx.doi.org/10.1145/1835804.1835935

Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. TwitterRank: Finding Topic-sensitive Influential Twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM '10)*. ACM, New York, NY, USA, 261–270. DOI: http://dx.doi.org/10.1145/1718487.1718520

Jaewon Yang and Jure Leskovec. 2011. Patterns of Temporal Variation in Online Media. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM '11)*. ACM, New York, NY, USA, 177–186. DOI: http://dx.doi.org/10.1145/1935826.1935863

Mengmeng Yang, Kai Chen, Zhongchen Miao, and Xiaokang Yang. 2014. Cost-Effective User Monitoring for Popularity Prediction of Online User-Generated Content. In *2014 IEEE International Conference on Data Mining Workshop*. 944–951. DOI: http://dx.doi.org/10.1109/ICDMW.2014.72

Yiming Yang, Tom Pierce, and Jaime Carbonell. 1998. A Study of Retrospective and On-line Event Detection. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. ACM, New York, NY, USA, 28–36. DOI: http://dx.doi.org/10.1145/290941.290953

Received January 2016; revised September 2016; accepted Semptember 2016