A Temporal Context-Aware Model for User Behavior Modeling in Social Media Systems

Hongzhi Yin[†] Bin Cui[†] Ling Chen[‡] Zhiting Hu[†] Zi Huang[§]
[†]Key Lab of High Confidence Software Technologies (MOE), School of EECS, Peking University
[‡]QCIS, University of Technology, Sydney
[§]School of Information Technology Electrical Engineering, University of Queensland
[†]{bestzhi, bin.cui, huzhiting}@pku.edu.cn [‡]ling.chen@uts.edu.au [§]huang@itee.uq.edu.au

ABSTRACT

Social media provides valuable resources to analyze user behaviors and capture user preferences. This paper focuses on analyzing user behaviors in social media systems and designing a latent class statistical mixture model, named *temporal context-aware mixture model* (TCAM), to account for the intentions and preferences behind user behaviors. Based on the observation that the behaviors of a user in social media systems are generally influenced by *intrinsic interest* as well as the *temporal context* (e.g., the public's attention at that time), TCAM simultaneously models the topics related to users' intrinsic interests and the topics related to temporal context and then combines the influences from the two factors to model user behaviors in a unified way. To further improve the performance of TCAM, an item-weighting scheme is proposed to enable TCAM to favor items that better represent topics related to user interests and topics related to temporal context, respectively. Based on TCAM, we design an efficient query processing technique to support fast online recommendation for large social media data. Extensive experiments have been conducted to evaluate the performance of TCAM on four real-world datasets crawled from different social media sites. The experimental results demonstrate the superiority of the TCAM models, compared with the state-of-the-art competitor methods, by modeling user behaviors more precisely and making more effective and efficient recommendations.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering; H.2.8 [**Database Applications**]: Data mining

Keywords

User Behavior Modeling; Temporal Recommender system; Probabilistic generative model; Social Media Mining

1. INTRODUCTION

With the rising popularity of social media, a better understanding of users' rating behaviors[1] is of great importance for the design of many applications, such as personalized recommendation, information filtering, behavioral targeting and computational advertising. Research efforts [21, 24] have been undertaken to model users' interests to help them find interesting items by analyzing their historical behaviors. However, existing work [21, 24, 27] simply assumes that users prefer items based on their intrinsic interests, which may not be accurate in many social application scenarios. For example, when choosing a book to read or a movie to watch, the users are likely to prefer books/movies that interest them. In contrast, when selecting news to read or users to follow in a social network (e.g., Twitter), it is most likely that users will be attracted respectively by breaking news or famous users who are followed by the general public [30, 18, 5]. Therefore, users' rating behaviors on items may not necessarily indicate users' intrinsic interests. New models are desired to better account for user behaviors in social medias to learn user preferences more precisely.

After investigating multiple social media systems, we observe that user rating behaviors are generally influenced by two factors: *the intrinsic interest of the user* and *the attention of the general public*. While the user's intrinsic interest is relatively stable, the attention of the general public changes from time to time; for example, the hot topics on a microblogging site evolve over time. Hence, in our work, we refer to the attention of the public during a particular time period as *temporal context*.

The two factors have different degrees of influence on user rating behaviors for different types of social media platforms as a result of the different characteristics (e.g., life cycles and updating rates) of various types of social media items. For instance, news is a type of time-sensitive item with a short life cycle − few people want to read outdated news; while the life cycle of movies is relatively longer, with many classic old movies being highly ranked in the popularity list. For time-sensitive social media items, users are more easily influenced by the temporal context, whereas they tend to make decisions based on their intrinsic interests when choosing less time-sensitive items such as books and movies.

To model user rating behaviors in social media systems, therefore, it is critical to identify users' intrinsic interests as well as the temporal context (i.e., the attention of the general public during a particular time period). Moreover, it is essential to model the influence degrees of the two factors in different social media systems.

To this end, we propose a temporal context-aware mixture model (TCAM) to mimic user rating behaviors in a process of decision making. As shown in Figure 1, TCAM is a latent class statistical mixture model that simultaneously models the topics [13, 3] related to users' intrinsic interests and the topics related to temporal context, and then combines the influences from the user interest and the temporal context to model user behaviors in a unified manner. Specifically, the model discovers (1) users' personal interest distri-

[1] We use the term "rating behavior" to denote general user actions on items in social media systems, such as rating and viewing.

SIGMOD/PODS'14, June 22–27, 2014, Snowbird, UT, USA.
Copyright 2014 ACM 978-1-4503-2376-5/14/06 ...$15.00.
http://dx.doi.org/10.1145/2588555.2593685.

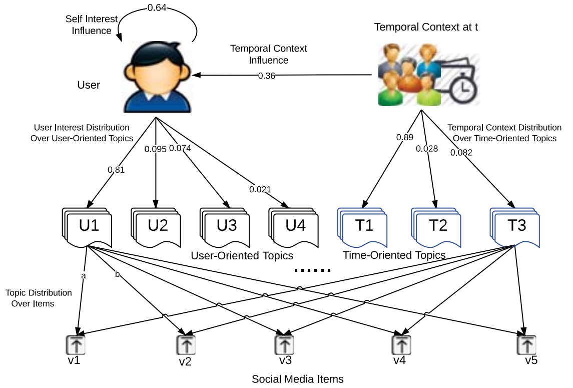


Figure 1: An Example of TCAM Model

bution over a set of latent topics; (2) the temporal context distribution over a set of latent topics; (3) an item generative distribution for each latent topic; and (4) the mixing weights that represent the influence probabilities of users' personal interest and the temporal context. It is worth mentioning that the set of latent topics used to model user interest is different from the topics used to model the temporal context. The former are called *user-oriented topics* and the latter are referred to as *time-oriented topics*.

The generative process of user rating behaviors in TCAM is briefly illustrated as follows. Suppose a user u selects an item v in a time interval t. TCAM first tosses a coin, based on the influence probabilities of the two factors, to decide whether this behavior results from the influence of the user's personal interest or the influence of the temporal context. If it results from the influence of the user's personal interest, TCAM chooses a *user-oriented topic* for u based on the user's intrinsic interest (with a certain probability). The selected topic in turn generates an item v following on from the topic's item generative distribution. Otherwise, if the influence from the temporal context is sampled, TCAM chooses a *time-oriented topic* according to the general public's interest during t, which in turn generates an item v.

Similar to traditional topic models where popular words in a document corpus are usually ranked high in each topic [5, 4], popular social media items tend to be estimated as having high generation probability by TCAM, which impairs the quality of the discovered user-oriented topics and time-oriented topics. User-oriented topics are supposed to capture user intrinsic interests, but a popular item favored by many users conveys less information about a user's intrinsic interest than an item favored by few users (i.e., a salient item) [32]. Similarly, a popular item constantly favored by users cannot well represent a time-oriented topic because the public's attentions change over time. Hence, to improve the performance of TCAM, we devise an item-weighting scheme to promote the importance of salient items and bursty items, which enhances the quality of the underlying topics detected by TCAM.

To demonstrate the applicability of our proposed TCAM model, we investigate how TCAM can be deployed to facilitate temporal recommendation in social media systems, i.e., making recommendations based on not only user interests but also temporal context. To speed up the process of online recommendation for large social media data, we design an efficient query-processing technique by extending the Threshold Algorithm (TA) [10]. Briefly, we precompute K sorted lists of items according to the K latent topics learned by TCAM. In each list, items are sorted based on their generative probabilities with respect to the corresponding topic. At query time, we access items from the K sorted lists and compute top-k items by extending the TA algorithm. The algorithm has the nice property of terminating early without scanning all items. Specifically, it terminates when the ranking score of the k-th item in the result list is higher than the threshold score. This TA-based scheme allows us to efficiently return the top-k recommendations by examining the minimum number of items.

The main contributions of our work are summarized as follows.

- We design a novel temporal context-aware mixture model (TCAM) to model user rating behaviors in social media systems which considers the influences of both user interest and temporal context in item selection process.
- We distinguish between user-oriented topics and time-oriented topics, which enables the precise identification of user personal interest and the temporal context.
- We propose an item-weighting scheme to enhance the performance of TCAM by exploiting the frequency distribution and temporal distribution of social items.
- To enhance the performance of TCAM in temporal recommendation for large social media data, we design an efficient query-processing technique.
- We conduct extensive experiments to evaluate the performance of the proposed models based on four sets of real-life data from different social media systems. The experimental results demonstrate the superiority of TCAM over existing approaches.

The remainder of the paper is organized as follows. Section 2 reviews the existing work related to our research. Section 3 details the temporal context-aware mixture model (TCAM). We deploy TCAM model to facilitate temporal recommendation in Section 4. We carry out extensive experiments and report the experimental results in Section 5 and conclude the paper in Section 6.

2. RELATED WORK

In this section, we review related research from the following two areas: topic modeling and temporal recommendation techniques.

Topic Model. Topic models provide a useful means to discover topic structures from large document collections. While traditional topic models, such as LDA [3] and PLSA [13], do not address the temporal information in a document corpus, a number of temporal topic models have been proposed to consider topic evolution over time. Mei and Zhai [20, 26] studied mining evolutionary topics from texts by comparing topics modeled in consecutive time intervals. Wang and McCallum [25] designed the TOT model that treats time stamps of documents as an observed continuous variable generated by topics. This model is designed to capture temporal features with beta distribution, confining each topic into a narrow time distribution. Some other models, such as Dynamic Topic Model (DTM) [2], MAP-PLSA [12] and Online LDA [1], are also proposed to study topic changes over time.

The TimeUserLDA model proposed by Diao et al. [9] is designed for finding bursty topics from microblogs. Although this model assumes that user posting behaviors are influenced by both user interest and global topic trends, there is only one shared set of underlying topics in TimeUserLDA, i.e., there is no distinction between underlying user-oriented topics and time-oriented topics. Thus, the topics detected by their models look confusing and noisy since they conflate both user interest and temporal context. To improve the topic discovery process, Yin et al. [33] recently proposed a unified model to detect both stable and temporal topics simultaneously from social media data.

Recently, topic models have been applied to collaborative filtering. X. Jin et al. [14] proposed an approach based on LDA to discover the hidden semantic relationships among items for recommendation. In [6], a hard-constraint-based LDA method was used to deal with user-community data, in which each user is viewed as a document and the communities that this user joins are viewed as words in the document. In contrast, Y. Kang et al. [15] proposed a soft-constraint-based LDA method for community recommendations. We refer to these topic model-based CF methods as "traditional" recommendation techniques which simply assume that items rated by users represent their intrinsic interests and ignore the influence from other factors. Recently, Yin et al. [34] proposed a location-content-aware topic model to produce location-based recommendation. Xu et al. [30] and Mao et al. [31] assumed that user behaviors are influenced by both user interests and the behaviors of their social friends, and proposed mixture latent topic models to capture these factors. Again, these models make use of one set of shared topics to model two factors. The estimated topics are confusing and difficult to interpret, which causes the recommendation results to degenerate. Moreover, these models do not exploit the temporal information.

Temporal Recommendation. Many successful temporal collaborative filtering methods are based on latent factor models. For example, the Netflix award winning algorithm *timeSVD*++ [16] assumes that the latent features consist of components that evolve over time and a dedicated bias for each user at each specific time point. This model can effectively capture local changes to user preferences which the authors claim to be vital for improving performance. Xiong et al. proposed a Bayesian probabilistic tensor factorization model (BPTF) in [29]. BPTF represents users, items and time in a shared low-dimensional space, and predicts the rating score that a user u will assign to item v at time t using the inner product of their latent representations. Demonstrated by the experimental results on Netflix data, both BPTF and timeSVD++ perform well on the rating prediction task because they incorporate time effects into models. One main disadvantage with these models is that the learnt latent low-dimensional space is difficult to interpret. Recently, Liu et al. [19] addressed a new problem - online evolutionary collaborative filtering, which tracks user interests over time for the purpose of making timely recommendations. They extended the widely used neighborhood based algorithms by incorporating temporal information and developed an incremental algorithm for updating neighborhood similarities with new data. However, most of the existing temporal recommendation models [16, 29, 19] are designed for the task of rating prediction rather than top-k recommendation.

3. USER RATING BEHAVIOR MODELING

In this section, we first introduce relevant definitions and notations used throughout this paper. We then present the novel temporal context-aware mixture model for modeling user rating behaviors in social media systems.

3.1 Notations and Definitions

The notations used in this paper are summarized in Table 1.

Definition 1. **(User Rating)** A user rating is a triple $(u,\ t,\ v)$ that denotes a rating behavior (e.g., purchasing, clicking and tagging) made by user u on item v during time interval t.

Definition 2. **(User Document)** Given a user u, the user document, D_u, is a set of pairs $\{(v,\ t)\}$ representing the rating behaviors on items during different time intervals made by u.

Definition 3. **(Rating Cuboid)** A rating cuboid $\boldsymbol{C}$ is an $N \times T \times V$ cuboid, where N is the number of users, T is the number of time intervals and V is the number of items. A cell indexed by $(u,\ t,\ v)$ stores the rating score that user u assigned to item v during time interval t.

User actions on items, such as tagging, downloading, purchasing and clicking, can be represented as a user rating. Either explicit feedback or implicit feedback can be used to compute the value of rating score. For example, given a user u who frequently uses a tag v during time interval t, the usage frequency can be used as the rating score to reflect the user's preference on the tag during that time period.

Definition 4. **(Topic)** Given a collection of items $\boldsymbol{I} = \{v_i\}_{i=1}^{V}$, a topic z is represented by a topic model $\boldsymbol{\phi}_z$, which is a multinomial distribution over items $\boldsymbol{\phi}_z = \{P(v_i|\boldsymbol{\phi}_z)$ or $\phi_{zv_i}\}_{i=1}^{V}$.

To illustrate the semantic meaning of a topic, we choose top-k items that have the highest probability under the topic, as shown in Figure 2. In our work, we distinguish between **user-oriented topics** $\boldsymbol{\phi}_z$ and **time-oriented topics** $\boldsymbol{\phi}'_x$ although both of them are represented by a multinomial distribution over items. User-oriented topics are used to model user interest, which is assumed to be generally stable over time. In contrast, time-oriented topics are used to model the temporal context (i.e., the public's attention during a particular time), which has a clear temporal feature. For example, the popularity of the topics may increase or decrease over time and reach a peak during a certain period of time, as shown in Figure 2.

Definition 5. **(User Interest)** Given a user u, her/his intrinsic interest, denoted as $\boldsymbol{\theta}_u$, is a multinomial distribution over user-oriented topics.

Definition 6. **(Temporal Context)** Given a time interval t, the temporal context during t, denoted as $\boldsymbol{\theta}'_t$, is a multinomial distribution over time-oriented topics or items.

SYMBOL	DESCRIPTION
u, t, v	user u, time interval t, item v
N, T, V	number of users, time intervals, and items
M_u	number of items rated by user u
λ_u	the mixing weight specific to user u
K_1	number of user-oriented topics
θ_{uz}	probability that user-oriented topic z is chosen by user u
$\boldsymbol{\theta}_u$	intrinsic interest of user u denoted by $\boldsymbol{\theta}_u = \{\theta_{uz}\}_{z=1}^{K_1}$
$\boldsymbol{\phi}_z$	item proportions of user-oriented topic z, denoted by $\boldsymbol{\phi}_z = \{\phi_{zv}\}_{v=1}^{V}$
ϕ_{zv}	probability that item v is generated by user-oriented topic z
K_2	number of time-oriented topics
$\boldsymbol{\theta}'_t$	the temporal context during time interval t denoted by $\boldsymbol{\theta}'_t = \{\theta'_{tx}\}_{x=1}^{K_2}$
θ'_{tx}	probability that time-oriented topic x is generated by time interval t
$\boldsymbol{\phi}'_x$	item proportions of time-oriented topic x, denoted by $\boldsymbol{\phi}'_x = \{\phi'_{xv}\}_{v=1}^{V}$
ϕ'_{xv}	probability that item v is generated by time-oriented topic x

Table 1: Notations used in this model

3.2 Temporal Context-Aware Mixture Model

Given a rating cuboid $\boldsymbol{C}$ which stores users' rating histories, we aim to model user rating behaviors by exploiting the information captured in $\boldsymbol{C}$. Before presenting the devised model, we first describe an example to illustrate the motivation of our design.

As mentioned before, users' rating behaviors in social media systems are influenced by not only intrinsic interest but also the

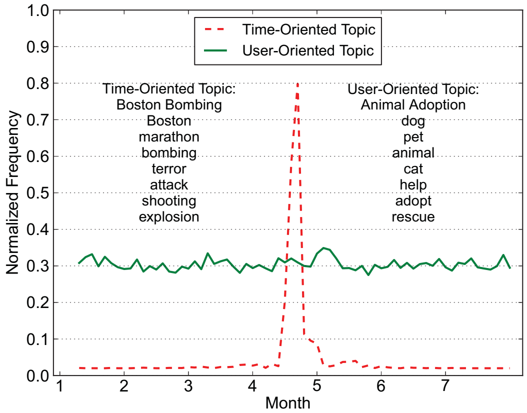


Figure 2: An Example of Two Types of Topics in Delicious

temporal context. It is crucial to distinguish between user-oriented topics and time-oriented topics, because the two have very different characteristics. For example, Figure 2 shows an example of a user-oriented topic and a time-oriented topic detected by TCAM model from Delicious. For demonstration, we present only the top eight tags that have the highest probability under each topic. We can easily tell the difference between the two topics from both their temporal distributions and the content descriptions. For the time-oriented topic, the items (i.e., tags) are related to a certain event (e.g.,"Boston Marathon bombings"). The popularity of the topic experiences a sharp increase during a particular time interval (e.g., in April 2013). For the user-oriented topic, the items are about the user's regular interest (e.g., "Pet Adoption"). The temporal distribution of the topic does not show any spike-like fluctuation. Hence, our TCAM models the user-oriented topics and the time-oriented topics simultaneously.

To consider the influence of the user intrinsic interest and the temporal context in a unified manner, TCAM computes the likelihood that a user u will rate an item v during a time interval t as follows.

$$P(v|u,t,\boldsymbol{\Psi}) = \lambda_u P(v|\boldsymbol{\theta}_u) + (1-\lambda_u)P(v|\boldsymbol{\theta}'_t) \tag{1}$$

where $\boldsymbol{\Psi}$ denotes the model parameter set, $P(v|\boldsymbol{\theta}_u)$ is the probability that item v is generated from u's intrinsic interest, denoted as $\boldsymbol{\theta}_u$, and $P(v|\boldsymbol{\theta}'_t)$ denotes the probability that item v is generated from the temporal context during time interval t, i.e., $\boldsymbol{\theta}'_t$. The parameter λ_u is the mixing weight which represents the influence probability of the user interest. That is, user u is influenced by personal interest $\boldsymbol{\theta}_u$ with probability λ_u, and is influenced by the temporal context $\boldsymbol{\theta}'_t$ with probability $1-\lambda_u$, for decision making. It is worth mentioning that TCAM holds personalized mixing weights for individual users, considering the differences between users in personalities (e.g., openness and agreeableness).

The user interest component $\boldsymbol{\theta}_u$ is modeled by a multinomial distribution over user-oriented topics, and each item is generated from a user-oriented topic z. Thus, $P(v|\boldsymbol{\theta}_u)$ is computed as follows.

$$P(v|\boldsymbol{\theta_u}) = \sum_{z=1}^{K_1} P(v|\boldsymbol{\phi}_z)P(z|\boldsymbol{\theta}_u) \tag{2}$$

As for the temporal context component $\boldsymbol{\theta}'_t$, we design two alternative methods to model it, which leads to two variations of TCAM: Item-based TCAM (ITCAM) and Topic-based TCAM (TTCAM). For the former, we model each temporal context $\boldsymbol{\theta}'_t$ as a time-oriented topic which is a multinomial distribution over all items during time interval t. For the latter, we model each temporal context $\boldsymbol{\theta}'_t$ as a multinomial distribution over latent time-oriented topics. As a running example in Figure 1, the user is influenced by personal interest and the temporal context with probabilities 0.64 and 0.36, respectively. Four user-oriented topics and three time-oriented topics are also shown respectively, where the weights representing the user's interest distribution over the user-oriented topics as well as the temporal context distribution over the time-oriented topics are labeled in the corresponding edges. We can see that user-oriented topic U1 dominates the user's interest, and time-oriented topic T1 attracts most attentions from the general public at time t. The probabilities of topic generating items are also labeled in the corresponding edges. For example, the weight b on the edge linking topic U1 and item v2 represents the probability of U1 generating item v2.

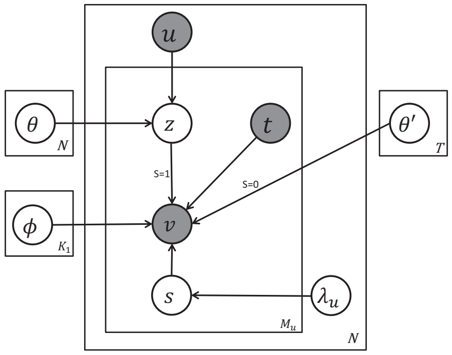


Figure 3: The Graphical Representation of ITCAM

Below, we will describe the details of the two variations of TCAM model and compare their performance in Section 5.

3.2.1 *Item-based TCAM*

Figure 3 illustrates the generative process of ITCAM with a graph model. The structure of ITCAM is similar to the PLSA model, but ITCAM has additional machinery to handle the mixing weight λ_u. In particular, a latent random variable s, associated with each item, is adopted as a switch to determine whether the item is generated according to the temporal context $\boldsymbol{\theta}'_t$ or the user's interest $\boldsymbol{\theta}_u$. s is sampled from a user-specific Bernoulli distribution with the mean λ_u. N indicates the number of users; K_1 is the number of user-oriented topics; T is the number of time intervals and M_u is the number of items rated by u. Figure 3 shows that each temporal context $\boldsymbol{\theta}'_t$ is modeled as a multinomial distribution over items. The generative process of ITCAM is summarized as follows.

For each item v rated by u during time interval t:

1. Sample s from $Bernoulli(\lambda_u)$
2. If $s = 1$
   (a) Sample topic z from $Multinomial(\boldsymbol{\theta}_u)$
   (b) Sample item v from $Multinomial(\boldsymbol{\phi}_z)$
3. Else
   (a) Sample item v from $Multinomial(\boldsymbol{\theta}'_t)$

Model Inference. Given a rating cuboid $\boldsymbol{C}$, the learning procedure of our model is to estimate the unknown model parameter set $\boldsymbol{\Psi} = \{\boldsymbol{\theta},\ \boldsymbol{\phi},\ \boldsymbol{\theta}',\ \boldsymbol{\lambda}\}$. The log likelihood is derived as follows:

$$L(\boldsymbol{\Psi}|\boldsymbol{C}) = \sum_{u=1}^{N}\sum_{t=1}^{T}\sum_{v=1}^{V} C[u,t,v]\log P(v|u,t,\boldsymbol{\Psi}) \tag{3}$$

where $P(v|u,t,\boldsymbol{\Psi})$ is defined in Equation (1).

The goal of parameter estimation is to maximize the log likelihood in Equation (3). As this equation cannot be solved directly by applying Maximum Likelihood Estimation (MLE), we apply an EM approach instead. In an expectation (E) step of the EM approach, we introduce $P(s|u,t,v;\widehat{\boldsymbol{\Psi}})$ which is the posterior probability of choosing personal interest $\boldsymbol{\theta}_u$ (i.e., $s=1$) or temporal context $\boldsymbol{\theta}'_t$ (i.e., $s=0$) respectively given user rating behavior (u,t,v) and the current estimations of the parameters $\widehat{\boldsymbol{\Psi}}$. In a maximization (M) step, parameters are updated by maximizing the expected complete data log-likelihood $Q(\boldsymbol{\Psi}|\widehat{\boldsymbol{\Psi}})$ based on the posterior probability computed in the E-step.

In the **E-step**, $P(s|u,t,v;\widehat{\boldsymbol{\Psi}})$ is updated according to Bayes formulas as follows,

$$P(s|u,t,v;\widehat{\boldsymbol{\Psi}}) = \frac{s\lambda_u P(v|\boldsymbol{\theta}_u) + (1-s)(1-\lambda_u)P(v|\boldsymbol{\theta}'_t)}{\lambda_u P(v|\boldsymbol{\theta}_u) + (1-\lambda_u)P(v|\boldsymbol{\theta_t}')} \tag{4}$$

where $P(v|\boldsymbol{\theta}_u)$ is defined as in Equation (2). To obtain the updated parameters $P(z|\boldsymbol{\theta}_u)$ and $P(v|\boldsymbol{\phi}_z)$, another posterior probability $P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}})$ is computed as follows:

$$P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}}) = \frac{P(v|\boldsymbol{\phi}_z)P(z|\boldsymbol{\theta}_u)}{\sum_{z'=1}^{K_1} P(v|\boldsymbol{\phi}_{z'})P(z'|\boldsymbol{\theta}_u)} \tag{5}$$

Based on $P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}})$ and $P(s=1|u,t,v;\widehat{\boldsymbol{\Psi}})$, we introduce the notation $P(z|u,t,v;\widehat{\boldsymbol{\Psi}})$ as follows,

$$P(z|u,t,v;\widehat{\boldsymbol{\Psi}}) = P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}})P(s=1|u,t,v;\widehat{\boldsymbol{\Psi}}) \tag{6}$$

With simple derivations [13], we can obtain the expectation of complete data log-likelihood for ITCAM:

$$\begin{aligned} Q(\boldsymbol{\Psi}|\widehat{\boldsymbol{\Psi}}) = &\sum_{u=1}^{N}\sum_{v=1}^{V}\sum_{t=1}^{T} C[u,t,v]\{P(s=1|u,t,v;\widehat{\boldsymbol{\Psi}}) \\ &\sum_{z=1}^{K_1} P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}})\log[\lambda_u P(v|\boldsymbol{\phi}_z)P(z|\boldsymbol{\theta}_u)] \\ &+ P(s=0|u,t,v;\widehat{\boldsymbol{\Psi}})\log[(1-\lambda_u)P(v|\boldsymbol{\theta}'_t)]\} \end{aligned} \tag{7}$$

In the **M-step**, we find the estimation $\boldsymbol{\Psi}$ that maximizes the expectation of the complete data log-likelihood $Q(\boldsymbol{\Psi}|\widehat{\boldsymbol{\Psi}})$ with the constraints $\sum_{v=1}^{V} P(v|\boldsymbol{\phi}_z) = 1$, $\sum_{v=1}^{V} P(v|\boldsymbol{\theta}'_t) = 1$ and $\sum_{z=1}^{K_1}$ $P(z|\boldsymbol{\theta}_u) = 1$ using the following updating formulas.

$$P(z|\boldsymbol{\theta}_u) = \frac{\sum_{v=1}^{V}\sum_{t=1}^{T} C[u,t,v]P(z|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{z'=1}^{K_1}\sum_{v=1}^{V}\sum_{t=1}^{T} C[u,t,v]P(z'|u,t,v;\widehat{\boldsymbol{\Psi}})} \tag{8}$$

$$P(v|\boldsymbol{\phi}_z) = \frac{\sum_{t=1}^{T}\sum_{u=1}^{N} C[u,t,v]P(z|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{v'=1}^{V}\sum_{t=1}^{T}\sum_{u=1}^{N} C[u,t,v']P(z|u,t,v';\widehat{\boldsymbol{\Psi}})} \tag{9}$$

$$P(v|\boldsymbol{\theta}'_t) = \frac{\sum_{u=1}^{N} C[u,t,v]P(s=0|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{v'=1}^{V}\sum_{u=1}^{N} C[u,t,v']P(s=0|u,t,v';\widehat{\boldsymbol{\Psi}})} \tag{10}$$

With an initial random guess of $\boldsymbol{\Psi}$, we alternately apply the E-step and M-step until a termination condition is met. To adapt to different users, we estimate the parameter λ_u in M-step, instead of picking a fixed λ value for all users. This personalized treatment can automatically adapt the model parameter estimation to various users. Specifically, λ_u is estimated as follows.

$$\lambda_u = \frac{\sum_{t=1}^{T}\sum_{v=1}^{V} C[u,t,v]P(s=1|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{t=1}^{T}\sum_{v=1}^{V}\sum_{s=0}^{1} C[u,t,v]P(s|u,t,v;\widehat{\boldsymbol{\Psi}})} \tag{11}$$

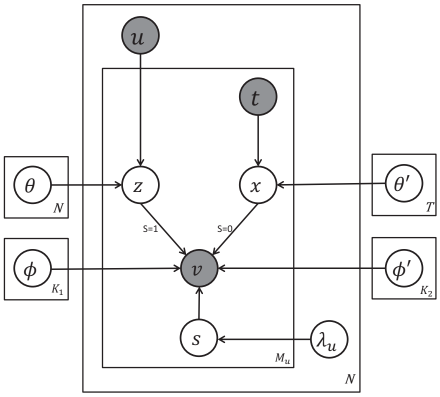


Figure 4: The Graphical Representation of TTCAM

3.2.2 *Topic-based TCAM*

Figure 4 illustrates the generative process of TTCAM with a graph model. In particular, the temporal context $\boldsymbol{\theta}'_t$ is modeled as a multinomial distribution over a set of time-oriented topics instead of a set of items. Thus, the computation of $P(v|\boldsymbol{\theta}'_t)$ is reformulated as in Equation (12).

$$P(v|\boldsymbol{\theta}'_t) = \sum_{x=1}^{K_2} P(v|\boldsymbol{\phi}'_x)P(x|\boldsymbol{\theta}'_t) \tag{12}$$

The generative process is summarized as follows. For each item v rated by u during time interval t:

1. Sample s from $Bernoulli(\lambda_u)$
2. If $s=1$
   - (a) Sample topic z from $Multinomial(\boldsymbol{\theta}_u)$
   - (b) Sample item v from $Multinomial(\boldsymbol{\phi}_z)$
3. Else
   - (a) Sample topic x from $Multinomial(\boldsymbol{\theta}'_t)$
   - (b) Sample item v from $Multinomial(\boldsymbol{\phi}'_x)$

Model Inference. To estimate model parameters in TTCAM, we apply an EM approach similarly.

In the **E-step**, similar to the parameter estimation in ITCAM, $P(s|u,t,v;\widehat{\boldsymbol{\Psi}})$ and $P(z|s=1,u,t,v;\widehat{\boldsymbol{\Psi}})$ are updated according to Equations (4, 5) where $P(v|\boldsymbol{\theta}'_t)$ is defined as in Equation (12) and $P(v|\boldsymbol{\theta}_u)$ is defined as in Equation (2). To obtain the updated parameters $P(x|\boldsymbol{\theta}'_t)$ and $P(v|\boldsymbol{\phi}'_x)$, we introduce the posterior probability $P(x|s=0,u,t,v;\widehat{\boldsymbol{\Psi}})$, defined as follows:

$$P(x|s=0,u,t,v;\widehat{\boldsymbol{\Psi}}) = \frac{P(v|\boldsymbol{\phi}'_x)P(x|\boldsymbol{\theta}'_t)}{\sum_{x'=1}^{K_2} P(v|\boldsymbol{\phi}'_{x'})P(x'|\boldsymbol{\theta}'_t)} \tag{13}$$

Based on $P(x|s=0,u,t,v;\widehat{\boldsymbol{\Psi}})$ and $P(s=0|u,t,v;\widehat{\boldsymbol{\Psi}})$, we introduce the notation $P(x|u,t,v;\widehat{\boldsymbol{\Psi}})$ as follows.

$$P(x|u,t,v;\widehat{\boldsymbol{\Psi}}) = P(x|s=0,u,t,v;\widehat{\boldsymbol{\Psi}})P(s=0|u,t,v;\widehat{\boldsymbol{\Psi}}) \tag{14}$$

In the **M-step**, the model parameters $P(z|\boldsymbol{\theta}_u)$, $P(v|\boldsymbol{\phi}_z)$, and λ_u are estimated according to Equations (8, 9, 11). $P(x|\boldsymbol{\theta}'_t)$ and $P(v|\boldsymbol{\phi}'_x)$ are inferred using the following formulas.

$$P(x|\boldsymbol{\theta}'_t) = \frac{\sum_{v=1}^{V}\sum_{u=1}^{N} C[u,t,v]P(x|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{x'=1}^{K_2}\sum_{v=1}^{V}\sum_{u=1}^{N} C[u,t,v]P(x'|u,t,v;\widehat{\boldsymbol{\Psi}})} \tag{15}$$

$$P(v|\boldsymbol{\phi}'_x) = \frac{\sum_{t=1}^{T}\sum_{u=1}^{N} C[u,t,v]P(x|u,t,v;\widehat{\boldsymbol{\Psi}})}{\sum_{v'=1}^{V}\sum_{t=1}^{T}\sum_{u=1}^{N} C[u,t,v']P(x|u,t,v';\widehat{\boldsymbol{\Psi}})} \tag{16}$$

3.2.3 Discussion about Parameter setting

In our model, we still have four hyper-parameters to tune manually, including the number of user-oriented topics K_1, the number of time-oriented topics K_2, the number of time intervals T and the number of EM iterations. K_1 and K_2 are the desired numbers of user-oriented topics and time-oriented topics respectively, which need to be tuned empirically. T is the number of time intervals used in our model for generating time-oriented topics, which provides users with the flexibility to adjust the granularity/length of time interval. The larger T is, the more fine-grained time intervals are. Regarding the number of EM iterations, we observe that convergence can be achieved in a few iterations (e.g., 50) because the model inference procedure using the EM approach is fast. It is worth mentioning that EM algorithms can be easily expressed in MapReduce [8, 28], so the inference procedure of TCAM can be naturally decomposed for parallel processing, which is scalable to large-scale datasets.

3.3 Enhancement of TCAM

In this section, we first introduce the challenge, arising from popular items, encountered by TCAM, and then propose an item weighting scheme to improve our proposed models.

Similar to traditional topic models, both ITCAM and TTCAM assume that all items are equally important in computing generation probabilities. As a result, popular items with more ratings tend to be estimated with high generation probability and ranked in top positions in each topic, which impairs the quality of both the user-oriented topics and the time-oriented topics.

For user-oriented topics, popular items are not good indicators of user intrinsic interests. A popular item rated by many users conveys less information about a user's interest than an item rated by few users. For time-oriented topics, it is expected that items representing the public's attention at a given time should be ranked high, such as items with bursty temporal distributions, since bursts of items are generally triggered by breaking news or events that attract the public's attention. Unfortunately, bursty items are most likely to be overwhelmed by long-standing popular items. Figure 5 shows the temporal frequency of the top six tags of a sample time-oriented topic discovered from Delicious. It can be observed that the topic concerns swine flu. The temporal distributions of three bursty tags, "flu", "mexico" and "swineflu", experience sharp spikes. Although the trends of the three tags do not always synchronize, they each go through a drastic increase and reach a peak in July 2009. The bursts in these curves are triggered by a real-world event, i.e., the swine flu outbreak in Mexico. The other three tags, "news", "health" and "death", maintain high frequency throughout the year. However, they convey little information about the event. Although they are relevant to the event, they are related to many other topics as well. Hence, it is desirable to rank bursty items higher than popular item when representing time-oriented topics.

To address the challenge posed by the popular items, we propose an item-weighting scheme to reduce the importance of popular items while promoting weights for salient (i.e., infrequent) and bursty items in computing generation probability. From the viewpoint of information theory [7], the entropy of an item v is defined as follows:

$$E(v) = -\sum_{u} P(u|v) \log P(u|v)$$

Suppose that the item v is preferred by users with equal probability $P(u|v) = \frac{1}{N(v)}$, the maximum entropy is,

$$E(v) = \log N(v)$$

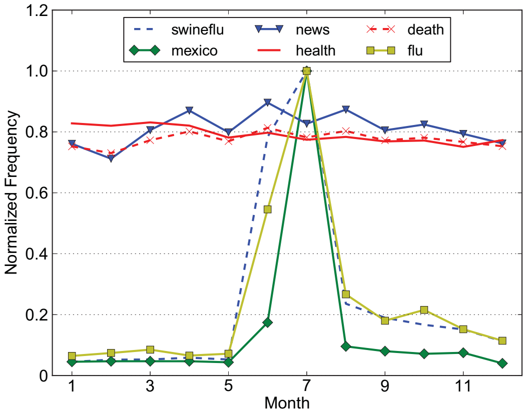


Figure 5: An Example of Bursty Tags and Popular Tags.

Generally, the entropy of an item tends to be proportional to its frequency/popularity $N(v)$. Hence, in the following analysis, we use the maximum entropy to approximate the exact entropy to simplify the calculation.

To allow salient items to be ranked higher in user-oriented topics, the weights of items should be inversely proportional to the entropy, as discussed above. Hence, we propose a concept called *inverse user frequency* to measure the ability of items to represent salient information. Let N be the total number of users in the entire data set; the *inverse user frequency* for the item v is defined as follows:

$$iuf(v) = \log N - \log N(v) = \log \frac{N}{N(v)} \tag{17}$$

which is similar to the inverse document frequency for a term in text mining.

To take into account the bursty information of items, we propose to compute the *bursty degree* of an item v using the following equation:

$$B(v,t) = \frac{N_t(v)}{N_t} \frac{N}{N(v)} \tag{18}$$

where $N_t(v)$ represents the popularity of item v during time interval t, i.e., the number of users who rate item v during time interval t, N_t is the number of active users during time interval t, $N(v)$ is the overall popularity of v across all time intervals, and N is the total number of users in the data set.

Combining the inverse user frequency and the bursty degree of items, we assign weight to the item v as follows.

$$w(v,t) = iuf(v) \times B(v,t) \tag{19}$$

Integrating the weights of items defined in Equation (19), we obtain the weighted user-time-item cuboid $\bar{\boldsymbol{C}}$ from the original $\boldsymbol{C}$ as follows:

$$\bar{C}[u,t,v] = C[u,t,v]w(v,t) \tag{20}$$

which can be used in both variations of the TCAM model.

4. TEMPORAL RECOMMENDATION

The conventional top-k recommendation task can be stated as follows: given a user, the recommender system should recommend a small number, say k, of items from all the available items. Note that the conventional top-k recommendation task does not consider the temporal information. However, in reality, user rating behaviors, influenced by both user interests and the temporal context, are dynamic. For example, user u rating item v in time interval

t does not mean that u still favors v in time interval $t+1$. Besides, each item has its own lifespan, especially for time-sensitive items such as news. It is undesirable to recommend outdated news. Hence, an ideal recommender system is expected to have the ability to recommend the right item v to user u in the right time interval t, rather than in other time intervals. In this paper, we propose the task of temporal top-k recommendation as follows: given a query $q = (u,\ t)$, i.e., a querying user u with a time interval t, the recommender model recommends k items which match u's interests and the temporal context at t.

Below, we will present how to deploy TCAM to facilitate temporal recommendations.

4.1 Computation of Ranking Score

Once we have inferred model parameters of TCAM, such as user interest $\boldsymbol{\theta}$, temporal context $\boldsymbol{\theta'}$, user-oriented topics $\boldsymbol{\phi}$, time-oriented topics $\boldsymbol{\phi'}$ and mixing weights $\boldsymbol{\lambda}$, given a query $q = (u,\ t)$, a ranking score $S(u,t,v)$ for each item v can be computed according to Equation (22), and then the top-k items with highest ranking score will be returned. Specifically, when receiving a query $q = (u,\ t)$, a new multinomial distribution for the query, $\boldsymbol{\vartheta}_q$, is first constructed by combining $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}'_t$. More specifically, we expand the user interest and temporal context spaces to be of the same dimension. For example, if there are K_1 user-oriented topics and K_2 time-oriented topics, the expanded topic space will have $K = K_1 + K_2$ topics. The expanded user interest distribution is defined as $\widetilde{\boldsymbol{\theta}}_u =< \boldsymbol{\theta}_u, 0, \cdots, 0 >$, where we set 0 on the time-oriented topics. Similarly, we define the expanded temporal context distribution to be $\widetilde{\boldsymbol{\theta}}'_t =< 0, \cdots, 0, \boldsymbol{\theta}'_t >$. The new distribution is defined as $\boldsymbol{\vartheta}_q = \lambda_u \widetilde{\boldsymbol{\theta}}_u + (1-\lambda_u)\widetilde{\boldsymbol{\theta}}'_t$. Correspondingly, we renumber the time-oriented topic x and change its range from $[1, \cdots, K_2]$ to $[K_1+1, \cdots, K]$. Then, we use $\varphi_{\widetilde{z}v}$ to denote the weight of item v on dimension $\widetilde{z}$ that corresponds to user-oriented topic z or time-oriented topic x, which depends on the value of $\widetilde{z}$, as shown in Equation (21).

$$\varphi_{\widetilde{z}v} = \begin{cases} \phi_{zv} & \widetilde{z} \leq K1 \\ \phi'_{xv} & \widetilde{z} > K1 \end{cases} \tag{21}$$

$$S(u,t,v) = \sum_{\widetilde{z}=1}^{K} \vartheta_{q\widetilde{z}} \varphi_{\widetilde{z}v} \tag{22}$$

4.2 Fast Top-k Recommendation

The straightforward method of generating the top-k items needs to compute the ranking scores for all items according to Equation (22), which is computationally inefficient, especially when the number of items becomes large. To speed up the online process of producing recommendations, we extend the Threshold-based Algorithm (TA) [10], which is capable of finding the top-k results by examining the minimum number of items.

We first pre-compute ordered lists of items, where each list corresponds to a latent topic learned by TCAM model. For example, given K topics (K_1 user-oriented topics plus K_2 time-oriented topics), we will compute K lists of sorted items, $L_{\widetilde{z}}$, $\widetilde{z} \in \{1, 2, ..., K\}$, where items in each list $L_{\widetilde{z}}$ are sorted according to $\varphi_{\widetilde{z}v}$ as defined in Equation (21). Given a query $q = (u,\ t)$, we run Algorithm 1 to compute the top-k items from the K sorted lists and return them in the priority list L. As shown in Algorithm 1, we first maintain a priority list PL for the K lists where the priority of a list $L_{\widetilde{z}}$ is the ranking score (i.e., $S(u,t,v)$) of the first item v in $L_{\widetilde{z}}$ (Lines 2-6). In each iteration, we select the most promising item (i.e., the first item) from the list that has the highest priority in PL and add it to the result list L (Lines 9-16). When the size of L is no less than k, we will examine the k-th item in the result list L. If the ranking score of the k-th item is higher than the *threshold score* (i.e., S_{Ta}), the algorithm terminates early without checking any subsequent items (Lines 18-21). Otherwise, the k-th item v' in L is replaced by the current item v if v's ranking score is higher than that of v' (Lines 22-25). In the end of each iteration, we update the priority of current list as well as the threshold score (lines 28-33).

Algorithm 1: Threshold-based algorithm

```
   Input: A query $q = (u,\ t)$; inferred model parameters $\boldsymbol{\theta}_u$, $\boldsymbol{\theta}'_t$ and
          $\lambda_u$; ranked lists $(L_1, ..., L_K)$;
   Output: List $L$ with all the $k$ highest ranked items;
 1 Initialize priority lists $PQ$, $L$ and the threshold score $S_{Ta}$;
 2 for $\widetilde{z} = 1$ to $K$ do
 3     $v = L_{\widetilde{z}}.getfirst()$;
 4     Compute $S(u,t,v)$ according to Equation (23);
 5     $PQ.insert(\widetilde{z}, S(u,t,v))$;
 6 end
 7 Compute $S_{Ta}$ according to Equation (24);
 8 while true do
 9     $nextListToCheck = PQ.getfirst()$;
10     $PQ.removefirst()$;
11     $v = L_{nextListToCheck}.getfirst()$;
12     $L_{nextListToCheck}.removefirst()$;
13     if $v \notin L$ then
14         if $L.size() < k$ then
15             $L.insert(v, S(u,t,v))$;
16         end
17         else
18             $v' = L.get(k)$;
19             if $S(u,t,v') > S_{Ta}$ then
20                 break;
21             end
22             if $S(u,t,v') < S(u,t,v)$ then
23                 $L.remove(k)$;
24                 $L.insert(v, S(u,t,v))$;
25             end
26         end
27     end
28     if $L_{nextListToCheck}.hasMore()$ then
29         $v = L_{nextListToCheck}.getfirst()$;
30         Compute $S(u,t,v)$ according to Equation (23);
31         $PQ.insert(nextListToCheck, S(u,t,v))$;
32         Compute $S_{Ta}$ according to Equation (24);
33     end
34     else
35         break;
36     end
37 end
```

Equation (23) illustrates the computation of the threshold score, which is obtained by aggregating the maximum $\varphi_{\widetilde{z}v}$ represented by the first item in each list $L_{\widetilde{z}}$ (i.e., $\max_{v \in L_{\widetilde{z}}} \varphi_{\widetilde{z}v}$). Consequently, it is the maximum possible ranking score that can be achieved by remaining unexamined items. Hence, if the ranking score of the k-th item in the result list L is higher than the threshold score, L can be returned immediately because no remaining item will have a higher ranking score than the k-th item.

$$S_{Ta} = \sum_{\widetilde{z}=1}^{K} \vartheta_{q\widetilde{z}} \max_{v \in L_{\widetilde{z}}} \varphi_{\widetilde{z}v} \tag{23}$$

5. EXPERIMENTS

In this section, we experimentally evaluate the performance of our proposed TCAM models.

5.1 Data Sets

Our experiments are conducted on four real data sets: Digg, MovieLens, Douban Movie and Delicious. The basic statistics of the four data sets are shown in Table 2.

- **Digg:** Digg is a popular social news aggregator, which allows users to vote news stories up or down, called *digging* or

	Digg	MovieLens
# of users	139,409	71,567
# of items	3,553	10,681
# of ratings	3,018,197	10,000,054
time span(year)	2009-2010	1998-2009
	Douban Movie	Delicious
# of users	50,885	201,663
# of items	69,908	2,828,304
# of ratings	52,604,958	36,966,661
time span(year)	2005-2010	2008-2009

Table 2: Basic statistics of the four data sets

burying, respectively. The Digg data set used in our experiment is Digg2009 [17], a publicly available data set containing 3,018,197 votes on 3553 popular stories cast by 139,409 distinct users. Although this data set contains only the IDs of news stories (the titles and the contents of stories are excluded), it is sufficient to evaluate the effectiveness of user behavior modeling in our work.

- **Douban Movie:** Douban[2] is the largest movie review website in China. In total, we crawled $50,885$ unique users and $69,908$ unique movies with $52,604,958$ movie ratings.

- **MovieLens:** MovieLens is a publicly available movie data set from the web-based recommender system MovieLens. The data set contains $10M$ ratings on a scale from 1 to 5 made by 71567 users on 10681 movies. We selected users who had rated at least 20 movies.

- **Delicious:** Delicious is a collaborative tagging system where users can upload and tag web pages. We collected 201,663 users and their tagging behaviors during the period Feb. 2008 - Dec. 2009. The data set contains 2,828,304 tags. Topics on technology and electronics account for about half of all web pages. Most of the other web pages are about breaking news with strong temporal features.

5.2 Comparisons

The temporal context-aware mixture model (TCAM) has two variants: **ITCAM** (Section 3.2.1) and **TTCAM** (Section 3.2.2). Both models can be enhanced by the item-weighting scheme, which leads to two optimized versions: *W-ITCAM* and *W-TTCAM*. We compare them with four categories of competitor approaches.

- **User-Topic Model (UT)**: We implemented a user-topic model following the previous works [21, 24]. This model is similar to the classic author-topic model (AT model) [23] which assumes that topics are generated according to user interests. The probabilistic formula of the user topic model is presented as follows, where θ_B is a background for smoothing.

$$P(v|u;\mathbf{\Psi}) = \lambda_B P(v|\boldsymbol{\theta}_B)+(1-\lambda_B)\sum_z P(z|\boldsymbol{\theta}_u)P(v|\boldsymbol{\phi}_z)$$

- **Time-Topic Model (TT)**: Following previous works [20, 26], we implemented a time-topic model. This model considers only the temporal information and ignores user interests. TT assumes that topics are generated by the temporal context, and that user behaviors are influenced by the temporal context. The probabilistic formula of the time-topic model is presented as follows.

$$P(v|t;\mathbf{\Psi}) = \lambda_B P(v|\boldsymbol{\theta}_B) + (1-\lambda_B)\sum_x P(x|\boldsymbol{\theta}'_t)P(v|\boldsymbol{\phi}'_x)$$

- **BPRMF**: This is a state-of-the-art matrix factorization model for item ranking optimized using BPR [22]. This model outperforms most of the existing recommender models in the task of top-k item recommendation. We used the BPRMF implementation provided by MyMediaLite, a free software recommender system library [11].

- **BPTF**: This is a state-of-the-art recommender model for rating prediction which uses a probabilistic tensor factorization technique by introducing additional factors for time [29]. This model outperforms most of the existing recommender models that consider time information.

5.3 Performance on Recommendation

In this section, we evaluate both the effectiveness of the suggested recommendations and the efficiency of generating top-k recommendations. Below, we first introduce the evaluation methods.

5.3.1 Evaluation Methods

Recommendation Effectiveness. For each user u, we randomly split her rated items during time interval t, $S_t(u)$, into 80% training items $S_t^{train}(u)$, and 20% test items $S_t^{test}(u)$. Given a querying user u in time interval t, if a recommended item is in the test set $S_t^{test}(u)$, it's a "hit" item, otherwise it's a "miss". A five-fold cross validation is employed to ensure the validity of the experimental results. To make the experimental results comparable, we use multiple well-known metrics to measure the ranked results. Similar to evaluations in information retrieval, we first use Precision@k to assess the quality of the top-k recommended items as follows:

$$Precision@k = \frac{\#hits}{k}$$

where $\#hits$ is the number of "hit" items in the top-k recommended items. We also consider NDCG, a widely used metric for a ranked list. NDCG@k is defined as:

$$NDCG@k = \frac{1}{IDCG} \times \sum_{i=1}^{k} \frac{2^{r_i}-1}{\log(i+1)}$$

where r_i is 1 if the item at position i is a "hit" item and 0 otherwise. IDCG is chosen for the purpose of normalization so that the perfect ranking has an NDCG value of 1. Considering that some users may have a large number of items in the test data while some have just a few, we also adopt the $F1$ score as our metric.

Recommendation Efficiency. The efficiency of the online recommendation mainly depends on 1) the number of items in the dataset and 2) the number of items recommended. Therefore, we test the efficiency of our proposed methods over these two factors.

5.3.2 Effectiveness of Temporal Recommendations

Figures 6 and 7 report the performance of the proposed models and other competitors in terms of Precision@k, NDCG@k and F1@k on Digg and MovieLens data sets, respectively. From the reported results, we observe that: (1) Our proposed models ITCAM, TTCAM, W-ITCAM and W-TTCAM, outperform other competitors such as UT, TT and BPRMF consistently on both data sets. This observation shows that recommendation accuracy, especially temporal recommendation accuracy, can be improved by considering both user intrinsic interests and the temporal context simultaneously. Note that, although BPTF performs almost as well as our proposed ITCAM model since it also exploits the temporal context information when recommending items, our W-ITCAM, TTCAM and W-TTCAM consistently outperform it. (2) W-ITCAM and W-TTCAM achieve higher temporal recommendation accuracy than

[2] http://douban.com

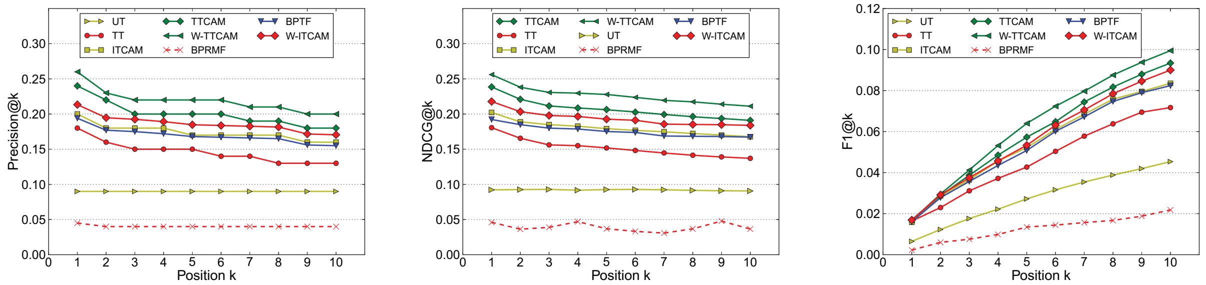


Figure 6: Temporal Accuracy on Digg Data Set

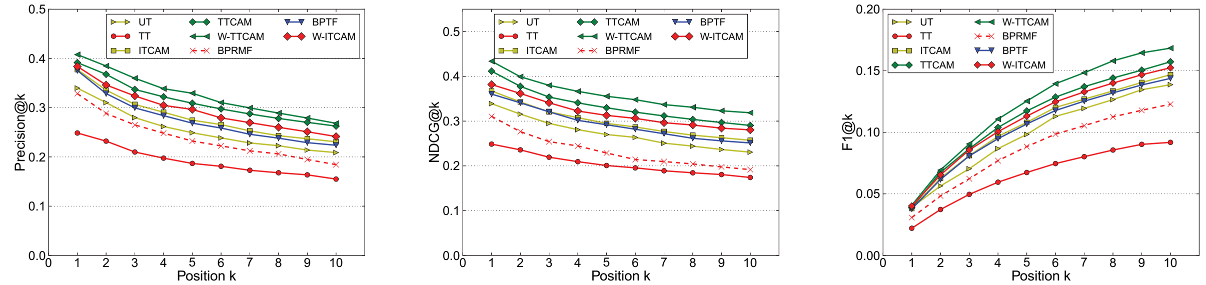


Figure 7: Temporal Accuracy on MovieLens Data Set

ITCAM and TTCAM, respectively, which shows the benefits brought by the item weighting scheme. Comparing TTCAM (W-TTCAM) with ITCAM (W-ITCAM), TTCAM (W-TTCAM) consistently performs better than ITCAM (W-ITCAM), verifying the advantage of modeling temporal context as a distribution over latent topics. (3) Comparing UT and TT on the two data sets, we find that UT performs better than TT on the MovieLens data set, while TT beats UT on the Digg data set. This is because news is a type of time-sensitive item while a movie is less time-sensitive. UT exploiting user interests works better in recommending less time-sensitive items while TT exploiting the temporal context is superior in recommending time-sensitive items.

There are three parameters in our models, namely, the length of time interval, the number of user-oriented topics (K1) and the number of time-oriented topics (K2). Tuning these model parameters is critical to the model performance. The experimental results presented above are obtained with the optimal parameter settings: (1) the optimal time intervals are one month for MovieLens and Douban Movie datasets, and three days for the Digg dataset, respectively; (2) the default values for K1 and K2 are 60 and 40 in TTCAM and W-TTCAM. We study the impacts of varying the three parameters below in details.

5.3.3 *Effect of the Length of Time Interval*

This experiment is to study the effect of the length of time interval. The length of time interval controls the time granularity of temporal recommendations. A larger length of time interval implies that the recommendation results will be less time-aware. To focus on the impact of the length of time interval, we only consider the models which utilize temporal influence. We report NDCG@5 on the Digg data set in Table 3. We also tune the length of time interval on other datasets and similar trends are observed, which are omitted due to space constraints. From the table, we have the following two observations.

- The first observation is that as the length of time interval increases, the NDCG values of all methods first increase, and then decrease. One possible reason for early increasing NDCG is that increasing the length makes the data in a time interval denser. Later on, NDCG decreases as the length of time interval gets larger, because increasing the length of time interval reduces the temporal influence. All methods achieve their best performance when the length of time interval is set to three days on this dataset, which could be a tradeoff of aforementioned two factors.
- Another important observation is that, for all lengths of time intervals, our proposed methods outperform other comparison methods. Among our proposed methods, W-TTCAM and W-ITCAM outperform TTCAM and ITCAM, respectively, showing the benefits brought by our proposed item-weighting technology.

5.3.4 *Effect of the Number of Topics*

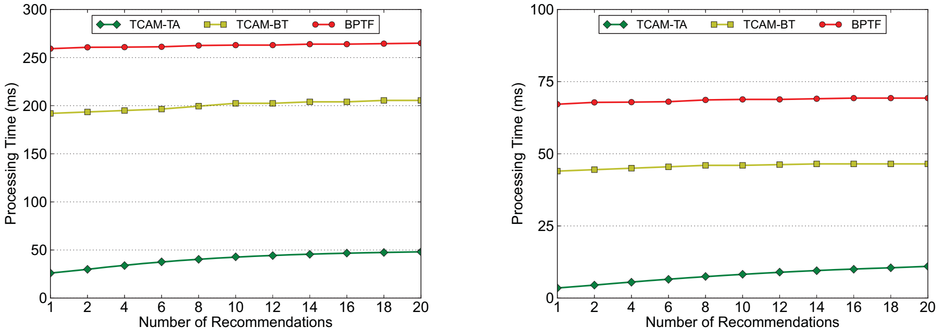
This experiment is to study the effect of number of user-oriented topics (K1) and number of time-oriented topics (K2) on Digg dataset. Figure 9 reports the results of varying K1 from 10 to 100 when fixing K2 to 20, 40, 60 and 80, respectively. For example, W-TTCAM-40 represents the model W-TTCAM with 40 time-oriented topics. From the figure, we observe that the performance of our proposed model W-TTCAM first increases with the increasing number of user-oriented topics (K1), and remains nearly stable when K1 is larger than 60. By comparing W-TTCAM-20, W-TTCAM-40, W-TTCAM-60 and W-TTCAM-80, we find that W-TTCAM-20 performs worst while other three almost perform equally well and their curves almost overlap. For this dataset, the performance of W-TTCAM does not change much when the number of time-oriented topics (K2) is larger than 40.

5.3.5 *Efficiency of Recommendations*

We next proceed to perform an efficiency comparison of different temporal recommendation algorithms on Douban Movie and Movielens datasets. It is worth mentioning that there is different number of movies in these two datasets, i.e., 69,908 and 10,681, respectively. All the recommendation algorithms are implemented

Length of time interval	TT	ITCAM	TTCAM	W-TTCAM	BPTF	W-ITCAM
1 day	0.1062	0.1254	0.1444	0.1594	0.1226	0.1287
2 days	0.1365	0.1612	0.1857	0.2050	0.1576	0.1655
3 days	**0.1517**	**0.1791**	**0.2063**	**0.2278**	**0.1751**	**0.1839**
4 days	0.1426	0.1684	0.1939	0.2141	0.1646	0.1729
5 days	0.1274	0.1505	0.1733	0.1913	0.1471	0.1545
6 days	0.1153	0.1361	0.1568	0.1731	0.1331	0.1397
7 days	0.1123	0.1325	0.1527	0.1685	0.1296	0.1361
8 days	0.1092	0.1290	0.1485	0.1640	0.1261	0.1324
9 days	0.1077	0.1272	0.1465	0.1617	0.1243	0.1306
10 days	0.1059	0.1253	0.1443	0.1589	0.1230	0.1283

Table 3: Performance of varying length of time interval on Digg dataset



(a) Douban Movie dataset

(b) Movielens dataset

Figure 8: Efficiency w.r.t Online Recommendations

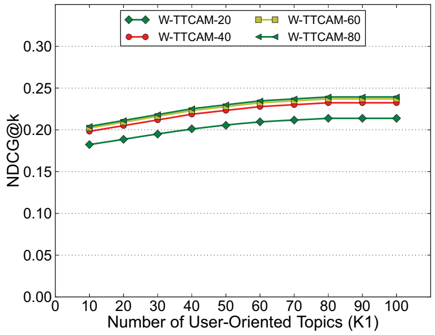


Figure 9: Performance of varying number of topics

in Java (JDK 1.6) and run on a Linux Server with 32G RAM. In this section, we first report their online recommendation time costs, and then present their offline training time.

Efficiency of Online Recommendations. In the efficiency study , we adopt two methods to utilize the knowledge learnt by TCAM to produce online recommendations. The first is called TCAM-TA and uses the proposed TA-based query processing technique in Section 4.2 to produce top-k recommendations. The second is called TCAM-BF and uses brute-force algorithm to produce top-k recommendations. In TCAM-BF, we scan all items within the dataset and compute their ranking scores, and then recommend k items with the highest scores. It should be noted that the state-of-the-art temporal recommender model BPTF also needs to scan all items to produce top-k recommendations because the ranking function in BPTF does not have the nice property of monotonicity which is required by TA algorithm [10].

Figures 8(a)-8(b) present the average online time costs of different methods with a varying number of recommendations for datasets Douban Movie and MovieLens respectively. For example, on average our proposed TCAM-TA can find top-10 items (that could most interest a user) from the Douban Movie dataset in 46ms, from the Movielens dataset in 9 ms. From the figures, we can observe that 1) TCAM-TA outperforms TCAM-BF and BPTF significantly in both datasets, justifying the benefits brought by the proposed TA-based query processing technique; 2) TCAM-BF is more efficient than BPTF in both datasets because BPTF computes the rating score that a user u will assign to item v at time t using the inner product of three vectors (i.e., user, item and time latent representations in a shared low-dimensional space) [29] while TCAM computes the ranking score using the inner product of two vectors (i.e., the query and item latent representations), as shown in Equation (22); 3) the time cost of TCAM-TA increases with the increasing number of recommendations because TCAM-TA needs to scan more items to find the k items with highest ranking scores, but TCAM-TA is still much more efficient than TCAM-BF and BPTF since the number of recommendations (i.e., k) is often constrained in a small range (e.g., $[1 \cdots 20]$); 4) the time cost (TC) of each algorithm in Douban Movie is more expensive than that in Movielens, showing that if a dataset has more items available, it requires more processing time to produce top-k recommendations.

Douban Movie			Movielens		
BPRMF	TCAM	BPTF	BPRMF	TCAM	BPTF
84.32	110.87	940.46	14.76	22.40	170.88

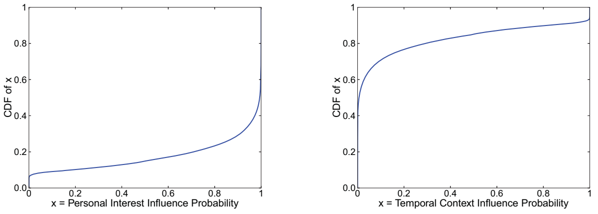
Table 4: Comparison on Model Training Time (Minute)

Efficiency of Offline Model Training. Table 4 shows the training time of three models on two datasets. From the table, we observe that our proposed TCAM model achieves comparable efficiency performance with BPRMF. As is expected, the training of BPTF is much slower than our TCAM and BPRMF. Note that although BPRMF achieves best efficiency performance, its recommendation accuracy value is much lower than our TCAM model because it does not exploit the temporal information, as is shown and analyzed in Section 5.3.2.

5.4 Temporal Context Influence Study

In this section, we study the influence degrees of users' personal interests and the temporal context on users' decision making. The user interest influence probability λ_u and the temporal context influence probability $1 - \lambda_u$ are learnt automatically in TCAM models. Due to space constraints, we focus on W-TTCAM. We are interested in how significantly the temporal context influences the user's decisions on different social media platforms.

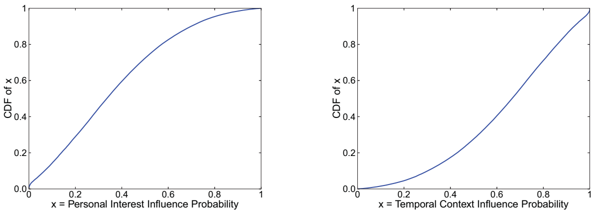
Since different people have different mixing weights, we plot the distributions of both the personal interest and the temporal context influence probabilities across all users. The results on the MovieLens data set are shown in Figure 10, where Figure 10 (a) plots the cumulative distribution of personal interest influence probabilities, and Figure 10(b) shows the temporal context influence probabilities. It can be observed that, in general, people's personal interest influence is significantly higher than the influence of the temporal context. For example, Figure 10(a) shows that the personal interest influence probability of more than 76% of users is higher than the 0.82. This observation indicates that most movies consumed by users are selected in accordance with their interests and tastes.



(a) Personal Interest Influence (b) Temporal Context Influence

Figure 10: Temporal Context Influence Result (MovieLens)

Figures 11(a) and 11(b) show respectively the personal interest influence probabilities and temporal context influence probabilities learnt from the Digg data. As shown in Figure 11(a), the personal interest influence probability is smaller than the temporal context influence probability. For example, the temporal context influence probability of more than 70% of users is higher than 0.5. The implication of this finding is that people are mainly influenced by the temporal context when choosing news to read. By comparing the analysis results obtained from the two datasets, we can observe that the temporal context influence on users' choice of news to read is much more significant than it is on the selection of movies to watch. This is probably because news is a time-sensitive item which is driven by offline social events, while movies are less time-sensitive.



(a) Personal Interest Influence (b) Temporal Context Influence

Figure 11: Temporal Context Influence Result (Digg)

5.5 Analyzing Latent Topics

TT	TTCAM	W-TTCAM
latest	news	michaeljackson
headline	world	mj
news	death	moonwalk
investigative	jackson	death
michaeljackson	michaeljackson	investigative
event	headline	news

Table 5: Time-Oriented Topic "Michael Jackson" Detected by Different Models on Delicious

TTCAM	W-TTCAM
Forrest Gump (1994)	Transformers (2007)
Zodiac (2007)	Tau ming chong (2007)
Transformers (2007)	Zodiac (2007)
Roman Holiday (1953)	Becoming Jane (2007)
Tau ming chong (2007)	Kidnap (2007)
Edward Scissorhands (1990)	Secret (2007)
Kidnap (2007)	I Am Legend(2007)

Table 6: Time-Oriented Topic T2007 Detected by Different Models on Douban Movie

To analyze the topics detected by our TCAM models, we note that Table 5 and Table 6 respectively present two time-oriented topics detected by different models on the datasets of Delicious and Douban Movie. We can make the following observations from the tables: (1) On the Delicious dataset, the time-oriented topics of "michael jackson's death", detected by TT and TTCAM, rank popular tags with abstract semantics like "event", "headline", and "world" in the top positions because of their high frequencies. As a matter of fact, these tags are of little value in describing the event and users may not be interested in them. In contrast, our proposed W-TTCAM clearly promotes concrete words like "moonwalk" because of the tight co-burst relationship with the word "Michael Jackson", which shows that W-TTCAM can cluster correlated bursty tags into a time-oriented topic, enabling the time-oriented topic to be represented by tags that are more relevant to the real event. (2) On the Douban Movie data set, the time-oriented topic T2007, detected by TTCAM, ranks popular movies like "Forrest Gump", "Roman Holiday" and "Edward Scissorhands" in the top positions as a result of their high popularity. In contrast, the top movies in T2007 detected by W-TTCAM were all released in the year 2007, which shows that W-TTCAM can accurately cluster movies released around the same time into a time-oriented topic by promoting bursty movies such as "Becoming Jane" and "Secret". Based on the above observations, we conclude that the time-oriented topics can be better modeled with the item weighting scheme.

To examine whether the user-oriented topics and time-oriented topics detected by the TCAM models can really capture user interest and temporal context, we present both user-oriented and time-oriented topics detected by W-TTCAM on Douban Movie dataset in Table 7. From the table, we can easily tell the difference between the user-oriented and time-oriented topics on movies. Specifically, the user-oriented topics capture the genres of films by clustering similar-taste movies into a topic, while movies under each time-oriented topic share a similar release time and the popularity of the corresponding time-oriented topic also reaches its peak during that period. The movie types and release time which are respectively captured by the user-oriented and time-oriented topics are shown in the second row of the table.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a temporal context-aware mixture model (TCAM) to model user rating behaviors in social media systems, by taking into account two factors, user intrinsic interests as an internal factor, and temporal context as an external factor.

U1	U15	T2010	T2009
Same-Sex	Horror/Thriller	2010	2009
Shelter (2007)	Eden Lake (2008)	Salt (2010)	District 9 (2009)
Starcrossed (2005)	Dead Silence (2007)	Inception (2010)	The Boat That Rocked (2009)
Get Real (1999)	See prang (2008)	The Expendables (2010)	The Founding of A Republic (2009)
Maurice (1987)	The Hills Have Eyes (2006)	The Twilight Saga: Eclipse (2010)	Sophie's Revenge (2009)
For a Lost Soldier (1992)	Dawn of the Dead (2004)	Temple Grandin (2010)	Eternal Beloved (2009)
The Trip (2002)	Silent Hill (2006)	Sex and the City 2 (2010)	Empire of Silver (2009)
Lucky Blue (2007)	The Last House on the Left (2009)	Sherlock Season 1 (2010)	Overheard (2009)

Table 7: Comparison between User-Oriented and Time-Oriented Topics Detected on Douban Movie

We introduced two different types of topics to model user interests and temporal context, respectively. We designed two methods to model the temporal context, and an item-weighting scheme was developed to enhance the TCAM models by exploiting the frequency distribution and temporal distribution information of items. To demonstrate the applicability of TCAM, we deployed this model to facilitate temporal recommendation. We also designed an efficient query processing technique to speed up the process of online recommendation. We demonstrated the superiority of the proposed TCAM models on four popular social media datasets.

There are several directions for future research. First, we would like to explore enhancements to our models by exploiting the effect of user social network on user rating behaviors, e.g., to study how a user's friends affect her/his rating behaviors. Second, it would be an interesting future direction to consider time-evolving user interests which generally change over time. Third, since the user generated data in social media is very noisy, it would be interesting to incorporate a background distribution to filter the noise in social media data.

7. ACKNOWLEDGMENTS

This research is supported by the National Natural Science Foundation of China under Grant No. 61272155. This work is also supported, in part, by the Australian Research Council (ARC) Discovery Project under Grant No. DP140100545.

8. REFERENCES

[1] L. AlSumait, D. Barbara, and C. Domeniconi. On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *IEEE Conf. on Data Mining*, pages 993–1022, 2008.

[2] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML*, pages 113–120, 2006.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *Journal of Machine Learning Research*, pages 993–1022, 2003.

[4] Y. Cha, B. Bi, C.-C. Hsieh, and J. Cho. Incorporating popularity in topic models for social network analysis. In *SIGIR*, pages 223–232, 2013.

[5] Y. Cha and J. Cho. Social-network analysis using topic models. In *SIGIR*, pages 565–574, 2012.

[6] W. Y. Chen, J. C. Chu, J. Luan, H. Bai, Y. Wang, and E. Y. Chang. Collaborative filtering for orkut communities: discovery of user latent behavior. In *WWW*, pages 681–690, 2009.

[7] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.

[8] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: Scalable online collaborative filtering. In *WWW*, pages 271–280, 2007.

[9] Q. Diao, J. Jiang, F. Zhu, and E. P. Lim. Finding bursty topics from microblogs. In *ACL*, pages 536–544, 2012.

[10] R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. In *PODS*, pages 102–113, 2001.

[11] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt Thieme. Mymedialite: a free recommender system library. In *RecSys*, pages 305–308, 2011.

[12] A. Gohr and A. Hinneburg. Topic evlolution in a stream of documents. In *SIAM*, 2009.

[13] T. Hofmann. Probabilistic latent semantic analysis. In *UAI*, 1999.

[14] X. Jin, Y. Zhou, and B. Mobasher. A maximum entropy web recommendation system: combining collaborative and content features. In *KDD*, pages 612–617, 2005.

[15] Y. Kang and N. Yu. Soft-constraint based online lda for community recommendation. In *PCM*, pages 494–505, 2010.

[16] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.

[17] K. Lerman and R. Ghosh. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *ICWSM*, 2010.

[18] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *IUI*, pages 31–40, 2010.

[19] N. N. Liu, M. Zhao, E. Xiang, and Q. Yang. Online evolutionary collaborative filtering. In *RecSys*, pages 95–102, 2010.

[20] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: An exploration of temporal text mining. In *KDD*, pages 198–207, 2005.

[21] M. Michelson and S. A. Macskassy. Discovering users' topics of interest on twitter: a first look. In *AND*, pages 73–80, 2010.

[22] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[23] M. Rosen Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI*, pages 487–494, 2004.

[24] J. Stoyanovich, S. Amer Yahia, C. Marlow, and C. Yu. Leveraging tagging to model user interests in del.icio.us. In *AAAI*, 2008.

[25] X. Wang and A. McCallum. Topics over time: A non-markov continueous-time model of topical trends. In *KDD*, pages 424–433, 2006.

[26] X. Wang, C. Zhai, X. Hu, and R. Sproat. Mining correlated bursty topic patterns from coordinated text streams. In *KDD*, pages 784–793, 2007.

[27] Z. Wen and C. Y. Lin. On the quality of inferring interests from social neighbors. In *KDD*, pages 373–382, 2010.

[28] J. Wolfe, A. Haghighi, and D. Klein. Fully distributed em for very large datasets. In *ICML*, pages 1184–1191, 2008.

[29] L. Xiong, X. Chen, T. K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, pages 211–222, 2010.

[30] Z. Xu, Y. Zhang, Y. Wu, and Q. Yang. Modeling user posting behavior on social media. In *SIGIR*, pages 545–554, 2012.

[31] M. Ye, X. Liu, and W. C. Lee. Exploring social influence for recommendation: a generative model approach. In *SIGIR*, pages 671–680, 2012.

[32] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen. Challenging the long tail recommendation. *Proc. VLDB Endow.*, 5(9):896–907, 2012.

[33] H. Yin, B. Cui, H. Lu, Y. Huang, and J. Yao. A unified model for stable and temporal topic detection on social platform. In *ICDE*, pages 661–672, 2013.

[34] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen. Lcars: A location-content-aware recommender system. In *KDD*, pages 221–229, 2013.