World Wide Web
https://doi.org/10.1007/s11280-018-0595-9

A novel temporal and topic-aware recommender model

Dandan Song[1] · **Zhifan Li**[1] · **Mingming Jiang**[1] ·
Lifei Qin[1] · **Lejian Liao**[1]

Received: 27 February 2018 / Revised: 25 April 2018 / Accepted: 24 May 2018

Abstract Individuals' interests and concerning topics are generally changing over time, with strong impact on their behaviors in social media. Accordingly, designing an intelligent recommender system which can adapt with the temporal characters of both factors becomes a significant research task. Namely both of temporal user interests and topics are important factors for improving the performance of recommender systems. In this paper, we suppose that users' current interests and topics are transferred from the previous time step with a Markov property. Based on this idea, we focus on designing a novel dynamic recommender model based on collective factorization, named *Temporal and Topic-Aware Recommender Model* (TTARM), which can express the transition process of both user interests and relevant topics in fine granularity. It is a hybrid recommender model which joint *Collaborative Filtering* (CF) and *Content-based* recommender method, thus can produce promising recommendations about both existing and newly published items. Experimental results on two

This article belongs to the Topical Collection: *Special Issue on Big Data Management and Intelligent Analytics*
Guest Editors: Junping Du, Panos Kalnis, Wenling Li, and Shuo Shang

✉ Dandan Song
sdd@bit.edu.cn

Zhifan Li
lizhifan@bit.edu.cn

Mingming Jiang
jiangmingming@cmbchina.com

Lifei Qin
qinlifei@bit.edu.cn

Lejian Liao
liaolj@bit.edu.cn

[1] Beijing Engineering Research Center of High Volume Language Information Processing & Cloud Computing Applications, Beijing Key Laboratory of Intelligent Information Technology, School of Computer Science & Technology, Beijing Institute of Technology, Beijing 100081, China

Published online: 02 August 2018

real life data sets from CiteULike and MovieLens, demonstrate the effectiveness of our proposed model.

Keywords Recommender system · Collaborative filtering · Matrix factorization

1 Introduction

Nowadays recommender systems are playing an extremely important role for people to find attractive items more accurately and efficiently. E-commerce or social websites such as Amazon, MovieLens or Delicious collect historical ratings or comments of items to design their own recommender systems. Making good recommendations to users is crucial for achieving better use experience, promoting products, and enhancing business values.

Recently, approaches [1, 13, 19, 24] based on *Collaborative Filtering* (CF) [8] have achieved big success in practice. However, the fact is user interests and relevant topics are changing over time, which is illustrated in Figure 1. For user i, he has three different states (i.e. $U_{i,t-1}, U_{i,t}, U_{i,t+1}$) at time step $t-1, t, t+1$, and at time step $t-1$, he was interested in *data mining*, *information extraction and retrieval*, then he focused on *topic model* and *natural language processing* area, finally he turned to *recommender system* and *social networks*. Meantime, he paid attention to *data mining* all the time. Therefore, learning user's interests accurately contributes to the performance of recommender systems. As this fact, those static CF methods usually can not track these variations and adjust to propose appropriate suggestions. Considering this drawback, many research strategies such as [6, 14, 45] have been undertaken to introduce time feature into their methods. However, these previous works infer user interests by decaying weights of instances according to time, or analyzing their historical behaviors throughout the life span. However, they do not describe the

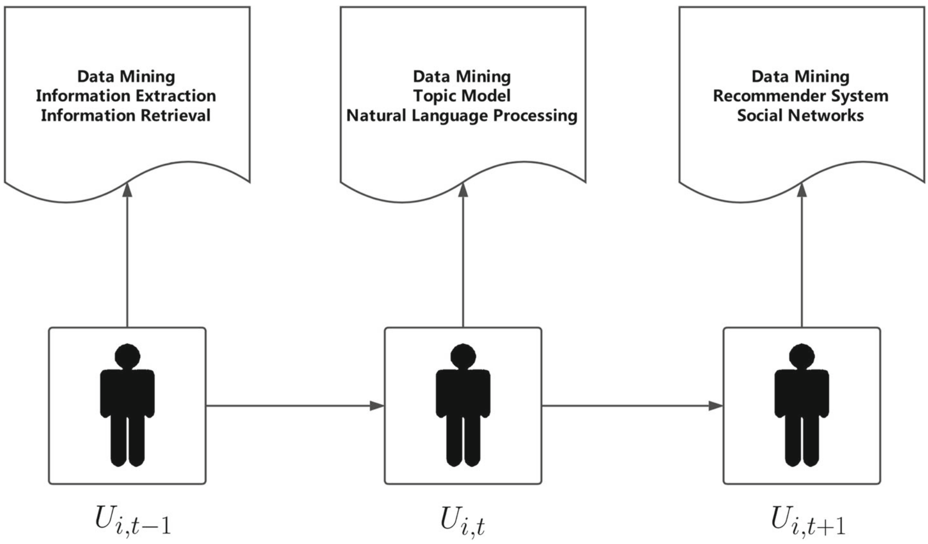


Figure 1 User interests drift over time

interests transition process in fine granularity. Therefore, delicately modeling temporal transitions of user interests is of significant importance for recommender systems to provide better personalized recommendation meeting individual user's needs [39].

On another hand, user topic feature is also a crucial factor that influences the performance of recommender systems. In the past decade, with the advent of user-driven social medias that allow users to store resources (contents, bookmarks, comments, and others) and associate them with personalized words, an army of topic-based models [16, 27, 40] have been proposed. They implement probability-based methods such as pLSA [10] and LDA [2] to extract latent topics from available contents in the user or object space and then produce recommendations [21, 52] or search results [53–55, 57, 59]. These methods achieve big success in solving the cold-start problem generally suffered by CF-based methods and enhancing performance of recommender systems. Therefore, it will be of imaginable significance to analyze topics in fine-grained time steps and import the temporal topic information into our temporal modeling of user interests for recommendation.

In order to take both above observational factors (*temporal user interests* and *user topics*) into account, we propose a novel *Temporal and Topic-Aware Recommender Model*, namely TTARM, to model the transition process of user interests and topics over time.

The main contributions of our work are summarized as follows.

- Supposing that an individual user's current interests and topics are shifted from the previous time step, we first propose a novel *Temporal Recommender Model*, namely TRM, based on *Joint Past-Present Decomposition* and *Collaborative Filtering* with Markov property to learn temporal trend of user interests.
- By defining dynamic topic similarity between users and items over time and incorporating it into TRM, we design a *Temporal and Topic-Aware Recommender Model* (TTARM) with both temporal interests and topic information.
- We systematically conduct extensive experiments on two large real datasets from CiteULike and MovieLens to evaluate the performance of our proposed model. And experimental results demonstrate that our model consistently outperforms other competitive methods.

The remaining of this paper is organized as follows. We review related methods in Section 2. Then introduce the details of our proposed recommender model and its learning algorithms in Section 3. We demonstrate the performance our model with a series of experiments and discuss the results in Section 4. Finally, we make a conclusion and look into the future work in Section 5. A preliminary version of this work has been published in BigComp 2018 [29]. Compared to the conference version, we refine the design process of our model, extend our experiments and give equation derivations details in Appendix.

2 Related work

In this section, we review quite a few existing relevant research works, which focus on temporal and topic-involved recommendation respectively.

2.1 Temporal recommendation

Many collaborative filtering based recommender algorithms, which incorporate temporal feature, have been proposed. Typically, several methods [5, 46] based on tensor factorization are developed to take time information into account. In [46], a Bayesian probabilistic tensor factorization model (BPTF) was proposed, ratings are represented as triples – (user, item, time), then these triples are organized into a three-dimensional tensor, finally the tensor is decomposed and ratings are predicted using the inner product of the latent factor vector. Their experimental results demonstrated that BPTF performs better than other static recommender methods on both Netflix and MovieLens datasets because they introduce time feature into the recommender model. While Xiang et al. [44] argued that the time dimension is a local effect and should not be compared cross all users arbitrarily in a recommender system. Therefore, they proposed a Session-based Temporal Graph (STG) which incorporates temporal information to model long-term and short-term preferences simultaneously. There is another time-based model called timeSVD++ [14], which is the state-of-the-art temporal model for the Netflix. timeSVD++ can track the time changing behavior throughout the life span of the data by incorporating latent temporal components into SVD++ [13], a famous collaborative filtering approach implemented by latent factor model. In addition, [7, 9, 20, 30] also took the time information into account in their models respectively. However, all the above methods do not depict the evolving process of user interests over time.

Recently, in [51] Zhang et al. assumed that user preferences evolve gradually, user's current preference depends on his preference at previous time step. Based on this assumption, they proposed temporal probabilistic matrix factorization (TMF) and its fully Bayesian treatment model (BTMF), by incorporating a transition matrix into the conventional matrix factorization methods. With the analogous consideration, Li et al. [18] defined the transition of user interests in a way to let the user feature in previous time step be the Dirichlet prior of that in the current time step. While we don't make any distribution assumptions of the data, thus our method is more adaptive to different kinds of situations. Additionally, they predict rating scores of the items by users, different with our goal which is to make rankings and recommend the top ranked items to users. And we model user interests' drift over time by introducing user previous preferences into current time step through *collective factorization*.

2.2 Topic-involved recommendation

A slice of collaborative filtering approaches incorporating topic features of users and items arose to improve the cold start and rating sparse problem which generally exist in CF-based recommender systems [12, 26, 31–38]. For instance, [42] proposed an algorithm for recommending scientific articles by combining the merits of traditional collaborative filtering and probabilistic topic model. Moreover, [50] proposed a location-content-aware topic model called LCARS for recommendation by learning the interest of each user and the local preference of each city by capturing item co-occurrence patterns and exploiting item contents. [48] proposed a probabilistic generative model, called social influenced selection (SIS), to model user preferences in terms of a number of latent topics and to correlate the items with users through these latent topics.

However, topics are generally evolving over time, which are not considered in those static models. To address this issue, [49] proposed a temporal context-aware mixture model (TCAM) to model users' rating behaviors by taking into account user-oriented topics (intrinsic interests) and time-oriented topics (general public's interests). Although this model incorporates the public's temporal interests and topics, the historical behaviors are analyzed throughout the life span, whose intrinsic interests transition process is not described in fine granularity. Conversely, we suppose users' current interests and topics are shifted from the previous time step, and with delicately modeling the temporal transitions of user interests with a Markov property, the performance of recommender systems is to be improved.

There are also some research conducted on spatial recommendations, which are interesting and meaningful [3, 25, 47, 56, 58]. But they are out of the discussion scope of this paper.

3 Method

In this section, we first propose problem formulation for recommendation in Section 3.1 and then our *Temporal Recommender Model* (TRM) inspired by Joint Past-Present Decomposition Model (JPP) in Section 3.2. In Section 3.3, we employ Topic Model to learn dynamic topic similarity between users and items, and then introduce this temporal topic feature into TRM and derived our proposed model – *Temporal and Topic-Aware Recommender Model* (TTARM). The learning algorithm and prediction are given in Sections 3.4 and 3.5.

3.1 Problem formulation of recommendation

Given a set of users U, and a set of items D. Let R be the matrix that contains all the ratings that the users have assigned to the items, whose size is $N_u \times N_d$ (where $N_u = |U|$ and $N_d = |D|$).

The task of recommendation could be taken as predicting the missing ratings in R, which can be considered as filling in the blanks such that the values would be consistent with the existing ratings in the matrix. And the intuition behind using matrix factorization to solve this problem is that there should be some latent features that determine how a user rates an item.

Assume that there are N_f latent features, then the traditional matrix factorization task is then to find two matrices P (whose size is $N_u \times N_f$) and Q (whose size is $N_f \times N_d$) such that their product approximates R:

$$R \approx P \times Q \tag{1}$$

3.2 Temporal recommender model

Joint Past-Present Decomposition Model, proposed by [41], is a time-based *collective factorization* [28] algorithm for topic discovery and monitoring of evolving input streams. Inspired by this model and [15], we proposed a novel *Temporal Recommender Model*, which suppose that users' current interests are transferred from the previous time step with a Markov property, then express the transition process of user interests in fine granularity.

Assume a collection of user-item ratings arrives continuously in batches. Each batch is represented by a data matrix $R^{(t)}$ of size $N_u^{(t)} \times N_i^{(t)}$, where $N_u^{(t)}$ is the number of users and $N_i^{(t)}$ is the number of items at time step t.

Analogous to the *Joint Past-Present decomposition model*, we derive the present decomposition at time t:

$$R^{(t)} \approx P^{(t)} Q^{(t)} \tag{2}$$

where $P^{(t)}$ has a size of $N_u^{(t)} \times N_f$ and $Q^{(t)}$ has a size of $N_f \times N_i^{(t)}$, with N_f represents the number of latent factors. Obviously, $P^{(t)}$ measures the extent of interests that users have on the corresponding factors, while $Q^{(t)}$ measures the extent to which items possess those factors. Usually, N_f is much smaller than $N_i^{(t)}$.

However, user interests vary over time, so we assume that users' *Present* interests (i.e. $P^{(t)}$) transmit from the previous interests (i.e. $P^{(t-1)}$). Although the observation data is dynamic, we assume that user interests evolve smoothly during one time period, and the current interests depend on the interests that appear in the previous time-slot, not on the sequence of interests that preceded it. Therefore, it has a Markov property, and correspondingly we derive the *Past* decomposition of $R^{(t)}$ at time t:

$$R^{(t)} \approx S^{(t)} P^{(t-1)} Q^{(t)} \tag{3}$$

with $P^{(t-1)}$ given. $S^{(t)}$ is a interest-transition matrix trying to capture how much the current users' interests distribution ($P^{(t)}$) linearly transmits from the previous one ($P^{(t-1)}$).

Accordingly, for each time step t, given $R^{(t)}$ and $P^{(t-1)}$, joint the above two decompositions, we derive:

$$\begin{cases} R^{(t)} \approx P^{(t)} Q^{(t)} \\ R^{(t)} \approx S^{(t)} P^{(t-1)} Q^{(t)} \end{cases} \tag{4}$$

This model is a combination of *Collaborative Filtering* and *Joint Past-Present* decomposition model, and we call it *Temporal Recommender Model*, namely TRM.

3.3 Topic-aware enhancement

The dynamic topic similarity between users and items is another crucial factor for users' rating behaviors. By incorporating this property into the TRM (4), we obtain:

$$\begin{cases} R^{(t)} \approx (1-\eta) P^{(t)} Q^{(t)} + \eta C^{(t)} \\ R^{(t)} \approx (1-\eta) S^{(t)} P^{(t-1)} Q^{(t)} + \eta C^{(t)} \end{cases} \tag{5}$$

where parameter $\eta \in [0, 1]$ is used to balance the basic rating score and topic similarity factor $C^{(t)}$. And $C^{(t)}_{ui}$ (an element of matrix $C^{(t)}$) denotes the topic similarity extent between user u and item i at time step t, is defined below:

$$\begin{aligned} rClC^{(t)}_{ui} &= \frac{W^{(t)}_u \cdot W^{(t)}_i}{|W^{(t)}_u| \cdot |W^{(t)}_i|} \\ &= \frac{\left(\frac{1}{\sum_{j \in D^{(t)}_u} R^{(t)}_{uj}} \sum_{j \in D^{(t)}_u} (R^{(t)}_{uj} W^{(t)}_j)\right) \cdot W^{(t)}_i}{\left|\frac{1}{\sum_{j \in D^{(t)}_u} R^{(t)}_{uj}} \sum_{j \in D^{(t)}_u} (R^{(t)}_{uj} W^{(t)}_j)\right| \cdot |W^{(t)}_i|} \end{aligned} \tag{6}$$

where $|.|$ denotes norm of vector, $D^{(t)}_u$ is the set of items that user u rated at time step t, and $R^{(t)}_{uj}$ is the rating user u gived to item j at time step t. Thus we can define the topic distribute of user u by calculating the average topic distribution of all items which user u rated at time step t, refers to $\frac{1}{\sum_{j \in D^{(t)}_u} R^{(t)}_{uj}} \sum_{j \in D^{(t)}_u} (R^{(t)}_{uj} W^{(t)}_j)$ in (6), where $R^{(t)}_{uj}$ serves as the weight of items' topic influence on user u. And $W^{(t)}_j$ is the topic distribution of item j at time step t, which can be obtained by applying Topic Model, like pLSA [10] and LDA [2] , on items' content.

We call (5) *Temporal and Topic-Aware Recommender Model*, namely TTARM. Obviously, it leads to *Temporal Recommender Model* when $\eta = 0$ and pure topic-oriented recommender model when $\eta = 1$. The learning methods of TRM and TTARM are given in Section 3.4.

3.4 Derived algorithm

In order to learn *Temporal and Topic-Aware Recommender Model*, specific loss function $\mathcal{L}(R^{(t)}; P^{(t)}; Q^{(t)}; S^{(t)}; P^{(t-1)})$ for (5) needs to be specified. Consulting the work in [41], the following loss function is defined:

$$\begin{aligned} rCl\mathcal{L} = & \underset{S^{(t)},P^{(t)},Q^{(t)}}{\arg\min} \|R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\|_F^2 \\ & + \|R^{(t)} - [(1-\eta)S^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}]\|_F^2 \\ & + \alpha\|P^{(t)}\|_1 + \beta\|Q^{(t)}\|_1 + \gamma\|S^{(t)}\|_1 \\ & + \lambda\|S^{(t)} - I\|_F^2 \\ & \text{subject to } P^{(t)} \geq 0,\ Q^{(t)} \geq 0,\ S^{(t)} \geq 0 \end{aligned} \tag{7}$$

where $\|.\|_F$ represents the Frobenius norm and $\|.\|_1$ stands for the L1 norm. The temporal regularization $\lambda\|S^{(t)} - I\|_F^2$ controls how much we want to bias the decomposition towards $P^{(t-1)}$. Thus the λ parameter $\in (0, \infty)$ balances present and past information; it quantifies the extent to which the model is past (i.e. $\lambda \to \infty$) or present oriented (i.e. $\lambda \to 0$).

Our goal is to minimize the loss function in (7), but it is not convex for all parameters $P^{(t)}$, $Q^{(t)}$, $S^{(t)}$ simultaneously. Learning from [17, 41], a local minimum for the objective function could be reached using multiplicative updates.

First, considering the Karush-Kuhn-Tucker (KKT) first-order conditions applied to our problem, we derive:

$$\begin{cases} P^{(t)} \odot \nabla_{P^{(t)}}\mathcal{L} = 0, & P^{(t)} \geq 0, \quad \nabla_{P^{(t)}}\mathcal{L} \geq 0 \\ Q^{(t)} \odot \nabla_{Q^{(t)}}\mathcal{L} = 0, & Q^{(t)} \geq 0, \quad \nabla_{Q^{(t)}}\mathcal{L} \geq 0 \\ S^{(t)} \odot \nabla_{S^{(t)}}\mathcal{L} = 0, & S^{(t)} \geq 0, \quad \nabla_{S^{(t)}}\mathcal{L} \geq 0 \end{cases} \tag{8}$$

where $\odot$ is the element-wise product.

According to the loss function in (7), the gradients for each parameter are derived respectively:

$$\begin{aligned} lCl\nabla_{P^{(t)}}\mathcal{L} = & 2P^{(t)}[(1-\eta)^2 Q^{(t)}{Q^{(t)}}^T + \alpha I] \\ & - 2[(1-\eta)R^{(t)} - \eta(1-\eta)C^{(t)}]{Q^{(t)}}^T \end{aligned} \tag{9}$$

$$\begin{aligned} \nabla_{Q^{(t)}}\mathcal{L} = & 2(1-\eta){P^{(t)}}^T[\eta C^{(t)} + (1-\eta)P^{(t)}Q^{(t)}] \\ & + 2(1-\eta){P^{(t-1)}}^T{S^{(t)}}^T \\ & \cdot [\eta C^{(t)} + (1-\eta)S^{(t)}P^{(t-1)}Q^{(t)}] \\ & - 2(1-\eta)({P^{(t)}}^T + {P^{(t-1)}}^T{S^{(t)}}^T)R^{(t)} \\ & + 2\beta Q^{(t)} \end{aligned} \tag{10}$$

$$\begin{aligned} \nabla_{S^{(t)}}\mathcal{L} = & 2(1-\eta)[(1-\eta)S^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}] \\ & \cdot {Q^{(t)}}^T{P^{(t-1)}}^T + 2(\lambda + \gamma)S^{(t)} \\ & - 2[(1-\eta)R^{(t)}{Q^{(t)}}^T{P^{(t-1)}}^T + \lambda I] \end{aligned} \tag{11}$$

By substituting the corresponding gradients in (8), the following update rules are obtained:

$$P^{(t)} = P^{(t)} \odot \frac{[(1-\eta)R^{(t)} - \eta(1-\eta)C^{(t)}]{Q^{(t)}}^T}{P^{(t)}[(1-\eta)^2 q^{(t)}{Q^{(t)}}^T + \alpha I]} \tag{12}$$

$$\begin{cases} Y &= (1-\eta)\{{P^{(t)}}^T[\eta C^{(t)} + (1-\eta)P^{(t)}Q^{(t)}] \\ &\quad + {P^{(t-1)}}^T {S^{(t)}}^T[\eta C^{(t)} \\ &\quad + (1-\eta)S^{(t)}P^{(t-1)}Q^{(t)}]\} + \beta Q^{(t)} \\ Q^{(t)} &= Q^{(t)} \odot \dfrac{(1-\eta)({P^{(t)}}^T + {P^{(t-1)}}^T {S^{(t)}}^T)R^{(t)}}{Y} \end{cases} \tag{13}$$

$$\begin{cases} Z &= (1-\eta)[(1-\eta)S^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}] \\ &\quad \cdot {Q^{(t)}}^T {P^{(t-1)}}^T + (\lambda+\gamma)S^{(t)} \\ S^{(t)} &= S^{(t)} \odot \dfrac{(1-\eta)R^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T + \lambda I}{Z} \end{cases} \tag{14}$$

And the update (12, (13), and (14) lead to Algorithm 1, which learns the *Temporal and Topic-Aware Recommender Model* (TTARM). Obviously, by setting the parameter $\eta = 0$ in Algorithm 1, it degenerates to the learning algorithm for *Temporal Recommender Model* (TRM), which doesn't take users' topics variation into account.

Algorithm 1 Learning Algorithm of TTARM

Input: $R^{(t)}, C^{(t)}, P^{(t-1)}, \lambda, \alpha, \beta, \gamma, \eta, \epsilon$
Output: $P^{(t)}, Q^{(t)}, S^{(t)}$

$P^{(t)}, Q^{(t)}, S^{(t)}$ ←random initialized non-negative;

$\delta' \leftarrow maxInt, \delta \leftarrow \frac{\delta'}{2}$;

while $abs(\delta' - \delta) \geq \epsilon$ **do**

$$P^{(t)} = P^{(t)} \odot \frac{[(1-\eta)R^{(t)} - \eta(1-\eta)C^{(t)}]{Q^{(t)}}^T}{P^{(t)}[(1-\eta)^2 q^{(t)}{Q^{(t)}}^T + \alpha I]}$$

$$\begin{aligned} Y &= (1-\eta)\{{P^{(t)}}^T[\eta C^{(t)} + (1-\eta)P^{(t)}Q^{(t)}] \\ &\quad + {P^{(t-1)}}^T {S^{(t)}}^T[\eta C^{(t)} \\ &\quad + (1-\eta)S^{(t)}P^{(t-1)}Q^{(t)}]\} + \beta Q^{(t)} \\ Q^{(t)} &= Q^{(t)} \odot \frac{(1-\eta)({P^{(t)}}^T + {P^{(t-1)}}^T {S^{(t)}}^T)R^{(t)}}{Y} \end{aligned}$$

$$\begin{aligned} Z &= (1-\eta)[(1-\eta)S^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}] \\ &\quad \cdot {Q^{(t)}}^T {P^{(t-1)}}^T + (\lambda+\gamma)S^{(t)} \\ S^{(t)} &= S^{(t)} \odot \frac{(1-\eta)R^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T + \lambda I}{Z} \end{aligned}$$

$\delta' \leftarrow \delta$;
$\delta \leftarrow \mathcal{L}(R^{(t)}, P^{(t)}, Q^{(t)}, S^{(t)}, P^{(t-1)})$;

end

return $P^{(t)}, Q^{(t)}, S^{(t)}$;

3.5 Rating prediction

We propose *Temporal Recommender Model* and *Temporal and Topic-Aware Recommender Model* in Sections 3.2 and 3.3, and they are learned in Section 3.4, and TTARM's rating prediction equation is inferred as follows:

$$\hat{R}_{ui}^{(t)} = (1-\eta)P_u^{(t)}Q_i^{(t)} + \eta C_{ui}^{(t)} \tag{15}$$

where $P_u^{(t)}$ and $Q_i^{(t)}$ are the weight distribution vectors on latent factors for user u and item i respectively, which are learned by Algorithm 1.

In (15), by setting $\eta = 0$, we get the following rating prediction equation for TRM:

$$\hat{R}_{ui}^{(t)} = P_u^{(t)}Q_i^{(t)} \tag{16}$$

4 Experiments

In this section, to demonstrate the performance of our proposed recommender model, extensive experiments are conducted and results are analyzed. At first, we introduce the datasets, evaluation metrics and experimental settings, then present the main findings.

4.1 Datasets

We evaluate our method and comparative methods on two large real life datasets — **CiteULike**[1] and **MovieLens**[2].

– **CiteULike**. This dataset is collected from CiteULike, which is a Web service allows users to save and share citations to academic papers. And the related articles' abstracts are provided by [42]. The time span is from November 4th in 2004 to April 16th in 2014. After merging duplicate articles, empty articles, users with library's size fewer than 10, articles with number of "like" users fewer than 10 are removed. The statistics of preprocessed experimental dataset are listed in Table 1.

  For each user-article pair, we generate a tetrad — (user_id, article_id, score, time), where the score is always 1, for we suppose user likes the article when he collects it in his library. Therefore, the rating matrix at time t is composed through the definition below:

$$R_{ij}^{(t)} = \begin{cases} 1 & \text{if article j is in user i's library at time t} \\ 0 & \text{other} \end{cases} \tag{17}$$

– **MovieLens**. Ratings in this dataset are collected from MovieLens website, which can make personalized movie recommendations to users and every user can give a rating to all movies. There are 2113 users, 9801 items and 824600 ratings(varying from 1 to 5) from Nov.1997 to Dec.2008 in this dataset. It's very sparse, therefore similar to CiteULike dataset we remove items which were tagged by less than 10 users and users who tagged less than 10 items. The resulting dataset's statistics are listed in Table 1.

[1] http://www.citeulike.org/faq/data.adp

[2] http://www.grouplens.org/node/12

Table 1 Statistics of the CiteULike dataset

Dataset	CiteULike	MovieLens
#Rating	109,052	814,589
#User	3386	2113
#Item	7695	6624
#Avg. user-tag	32.20	385.5
#Avg. item-tagged	14.17	122.97
Sparse rate	99.58%	94.18%
Timespan	2004.11-2014.4	1997.11 – 2008.12

4.2 Evaluation metrics

For each user u, we predict his ratings at items which he has not rated before the current time step t, then sort these ratings descendingly, and recommend the top-k items to him. If a recommended item is liked by the user u at time step t according to the test set, we call it a "hit" item, otherwise it's a "miss" item [49]. In order to evaluate the experimental results more legitimately, for each user u we define three well-known metrics as follows:

$$Recall@k = \frac{N(hits)}{N(items)}$$

where $N(hits)$ is the number of "hit" items in the top-k recommended items, $N(items)$ is the number of all items in the test set of user u. Obviously, a high recall with lower k indicates a better recommender system.

Concerning that Recall can not reflect the position importance of "hit" items in the ranked list, we also use $NDCG@k$, a widely used metric in information retrieval, which 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 [49].

The metrics defined above are all user-oriented. Therefore, metric for the entire recommender system can be summarized using the average metric value of all users, defined below:

$$Metric@k = \frac{\sum_{i=1}^{N} M_i@k}{N}$$

where N is the number of users, $M_i@k$ is the metric value for user i at position k, and metric refers to *Recall*, *NDCG*.

4.3 Comparative methods

In order to analyze the performance of our proposed models, we design the comparison experiments between the following methods, including **TRM** (in Section 3.2) and **TTARM** (in Section 3.3).

- **BPMF**. A fully Bayesian treatment of the Probabilistic Matrix Factorization (**PMF** [22]) model in which model capacity is controlled automatically by integrating over all model parameters and hyperparameters [24].
- **timeSVD++**. A temporal recommender model which extends the **SVD++** [13] by introducing a time-variant bias for each user and item at every individual time step.
- **WALS**. A simple extension for Alternating Least Squares (ALS) where each user/item pair has an additional weight, which is a tensor algorithm since besides of the rating it also maintains a weight for each rating [11].
- **TensorALS**. A temporal recommender algorithm based on tensor factorization and alternating least squares, which considers time step as the third dimension [5].
- **BTMF**. A temporal bayesian probabilistic matrix factorization model (BTMF) [51], which incorporates a transition matrix into the conventional matrix factorization methods.
- **TRM**. The *Temporal Recommender Model* proposed in Section 3.2 of this paper without topic feature, learning by Algorithm 1 with parameter $\eta = 0$.
- **TTARM**. The *Temporal and Topic-Aware Recommender Model* proposed in Section 3.3 of this paper, also learnt by Algorithm 1, while the parameter $\eta \in (0, 1)$ implies a combination of temporal
  user interests and topics.

4.4 Experimental setup

For each dataset, we create 10 time steps by splitting ratings yearly, merging extra rating to closest time step, and the distributions of ratings are shown in Figure 2. Obviously, the CiteULike dataset is more sparse than the MovieLens dataset, and at time step 10, the two datasets both have very few ratings. TTARM can only run on CiteULike dataset since MovieLens dataset doesn't have items' content information and TTARM is a content-based

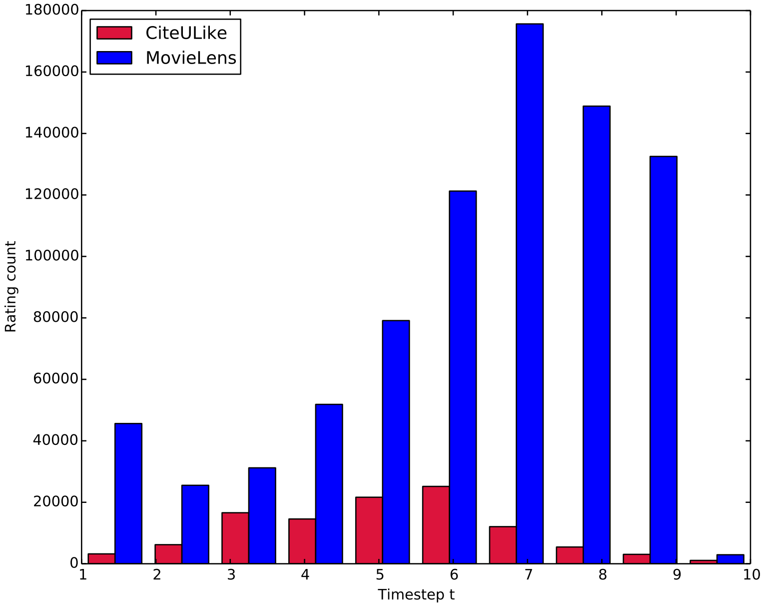


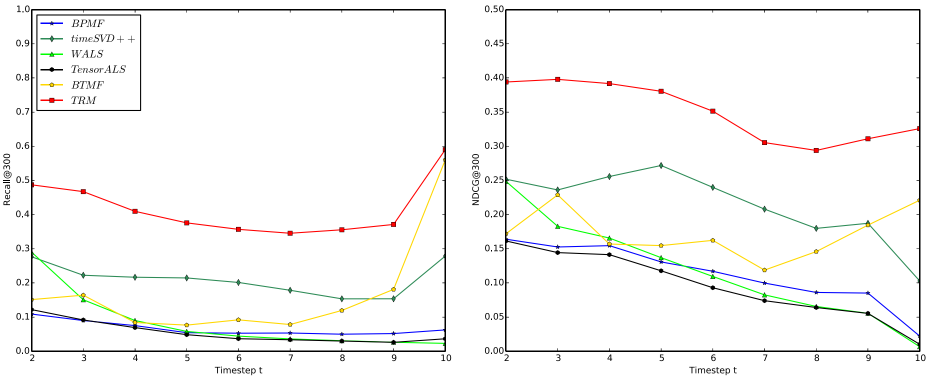
Figure 2 CiteULike and MovieLens datasets' rating distribution on 10 time steps

model. In addition, we set the number of latent factors D to be 20 for all latent factor models participate in competition. The parameter λ is set to be 10 for TRM and TTARM, and the topic balance parameter η is set to be 0.3 and topic number is set to be 50 for TTARM. Other parameters related to norm in loss function are set to be 0.05. According to Sections 3.2 and 3.3, both TRM and TTARM are launched by given the $P^{(t-1)}$ matrix. Therefore, in order to acquire a bootable $P^{(t-1)}$ we apply Non-negative matrix factorization (NMF) at the first time step, and launch all comparative models at the second time step. That is why we only show the performance for time steps 2 to 10.

In order to reduce the variance of all models' performance estimates, 5-fold cross-validation is adopted. Similar to the work in [42], the splitting method is that for each item, its related ratings at time step t is split into 5 folds. Thus for each time step t, we split the ratings appearing at that time into 5 folds, and iteratively consider each fold as the test set and the others as training set. After iteratively applying all the comparative models to the relevant 5-folds training set and test set, the mean performance in terms of the evaluation metrics is respectively calculated. In addition, for each experiment on models, we run 5 times with same parameters and calculate the mean value and standard error for the purpose of gaining more convincing experimental results. Top-k items are recommended, accordingly *Metric*@k is evaluated, namely *NDCG*@k, *Recall*@k. In practice, recall is very useful in recommender systems since it takes a global view on all items and a high recall indeed reflects user's adoption [51]. And NDCG is also very effective and significant for appraising the performance of Recommender Systems.

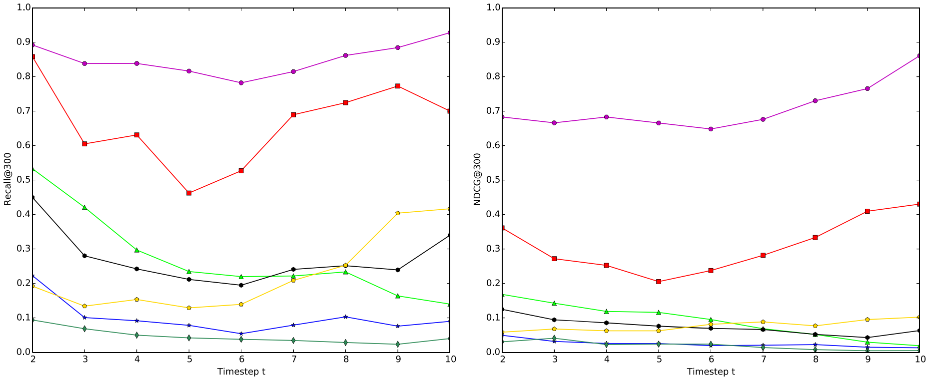
4.5 Experimental results

The experimental results of all comparative models on MovieLens and CiteULike dataset are shown in Figures 3 and 4 with Recall@300 and NDCG@300 from time step 2 to 10. It is obvious that both our TRM and TTARM outperform the other comparative recommender models (i.e. BPMF, timeSVD++, TensorALS, BTMF and WALS) consistently in the evaluation metrics. This comparison result is an indication of how temporal feature of user interests can remarkably contribute to the performance of recommender system. The fact that TTARM outperforms TRM at all time steps implies that incorporating the temporal topic similarity between users and articles into our model achieves great success. Moreover,



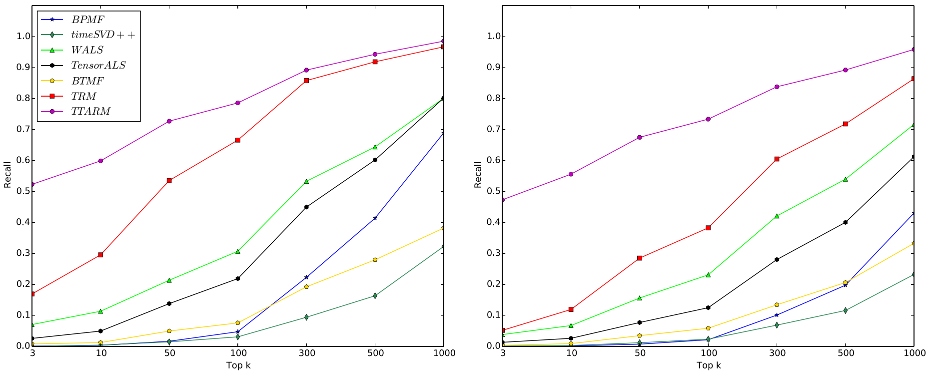
(a) Recall@300 (b) NDCG@300

Figure 3 Metric@300 on MovieLens dataset

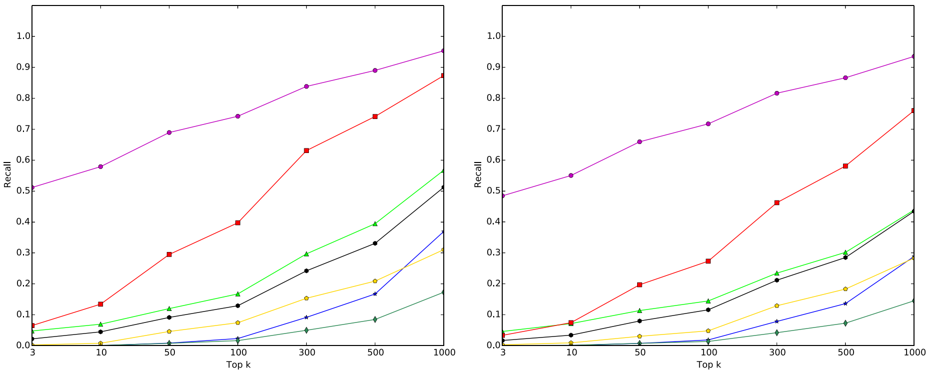


(a) Recall@300 (b) NDCG@300

Figure 4 Metric@300 on CiteULike dataset



(a) Time step 2 (b) Time step 3



(c) Time step 4 (d) Time step 5

Figure 5 Recall@k for all comparative models on CiteULike dataset at time step 2–5 and vary k from 3 to 1000. Higher values are better the introduction of similarity between topics benefits solving the cold start problem and the sparsity of dataset.

In Figure 4, TTARM and TRM's performance decreases at time steps 2–5 and rises after time step 6, while WALS, timeSVD++, BPMF's performance decreases almost across all time steps, BTMF and TensorALS's recall performance rises after time step 6. And in Figure 3, TRM decreases a little from time step 2 to 8 and rises afterward and BTMF has the similar trend across time steps. In addition, timeSVD++ beats other comparative methods at most time steps, and BTMF starts to show its performance after time step 7. Overall, our models (i.e., TRM and TTARM) show their steady and much better performance, which demonstrate the distinguishing ability of our methods in learning and predicting the variations of user interests and topics.

In details, Figures 5, 6, 7, and 8 show results at 2–5 time steps, whose experimental results are similar. In each figure, there are four subfigures respectively demonstrate models' performance at time step 2–5, and the x axis is the k in *Recall*@k and *NDCG*@k. From the results, we can find that for each method, the trend of its performance over time is analogous under the two metrics, which contributes to the authorities of experimental results. It is obvious that both our TRM and TTARM outperform the other recommender models (i.e. BPMF, timeSVD++, TensorALS, BTMF and WALS) consistently in terms of the two

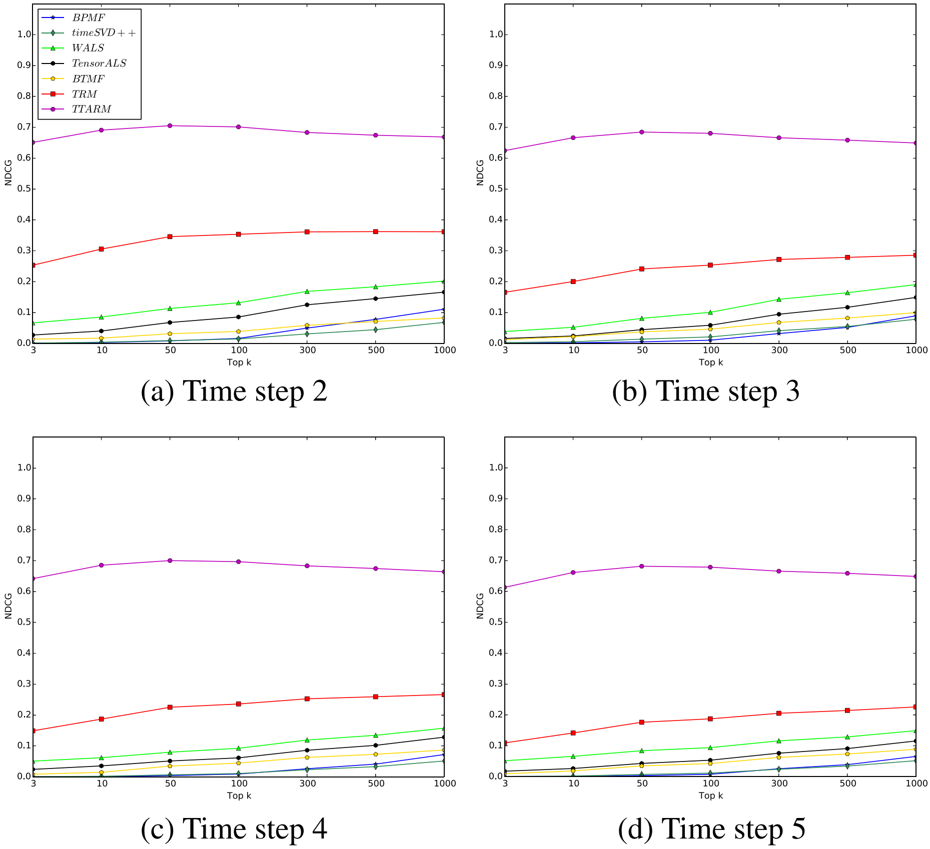
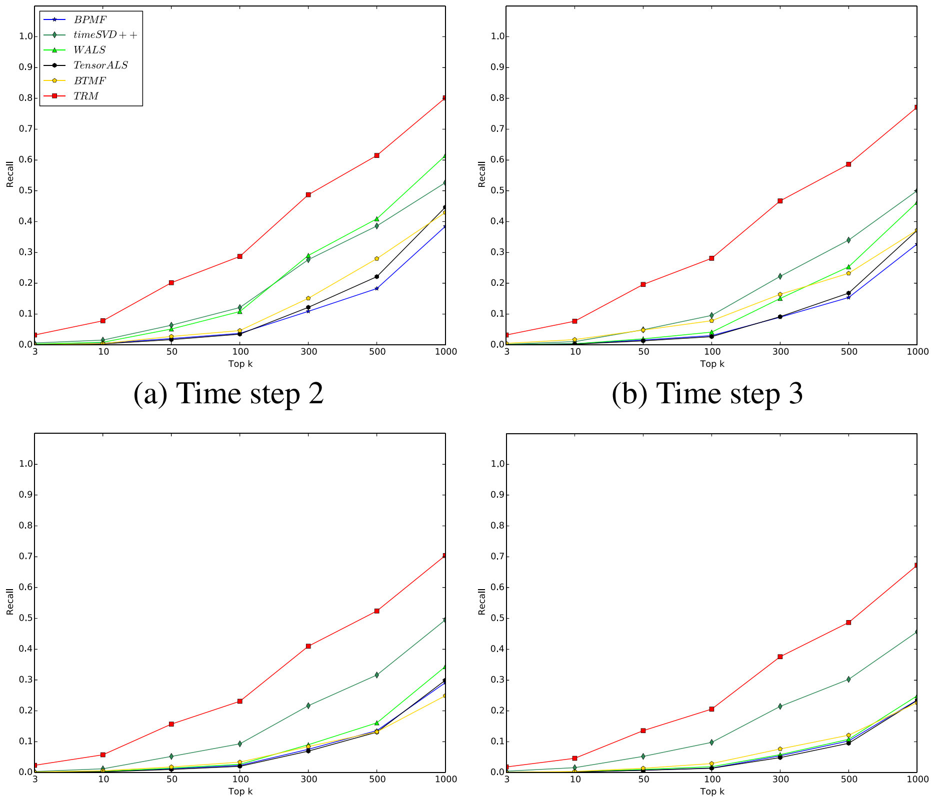


Figure 6 NDCG@k for all comparative models on CiteULike dataset at time step 2–5 and vary k from 3 to 1000. Higher values are better



(a) Time step 2 (b) Time step 3

(c) Time step 4 (d) Time step 5

Figure 7 Recall@k for all comparative models on MovieLen dataset at time step 2–5 and vary k from 3 to 1000. Higher values are better

evaluation metrics (i.e., *NDCG*@k and *Recall*@k). This comparison result is an indication of how temporal feature of user interests can remarkably contribute to the performance of recommender system. The fact that TTARM outperforms TRM at all time steps implies that incorporating the temporal topic similarity between users and items into our model achieves great success. Moreover, the introduction of similarity between topics benefits solving the cold start problem and the sparsity of dataset.

TensorALS and WALS's performance is very close and better than timeSVD++, BPMF and BTMF on CiteULike dataset at time step 2–5 since they both are based on alternating least squares method (ALS). However, timeSVD++ beats TensorALS and WALS on MovieLens dataset. This phenomenon indicates that timeSVD++, TensorALS and WALS's performance depends on dataset more or less. Simultaneously, static methods, like BPMF, simply use all previous ratings will dismiss user interests' transmit over time step and performs not that well in our experiments.

TTARM beats all other methods both in Figures 5 and 6 on the CiteULike dataset. The items in this dataset are research articles, which have tight relation with topics, hence incorporating topic to TTARM contributes a lot. However, our proposed model TRM which doesn't introduce topic feature beats all other models under both metrics and datasets as

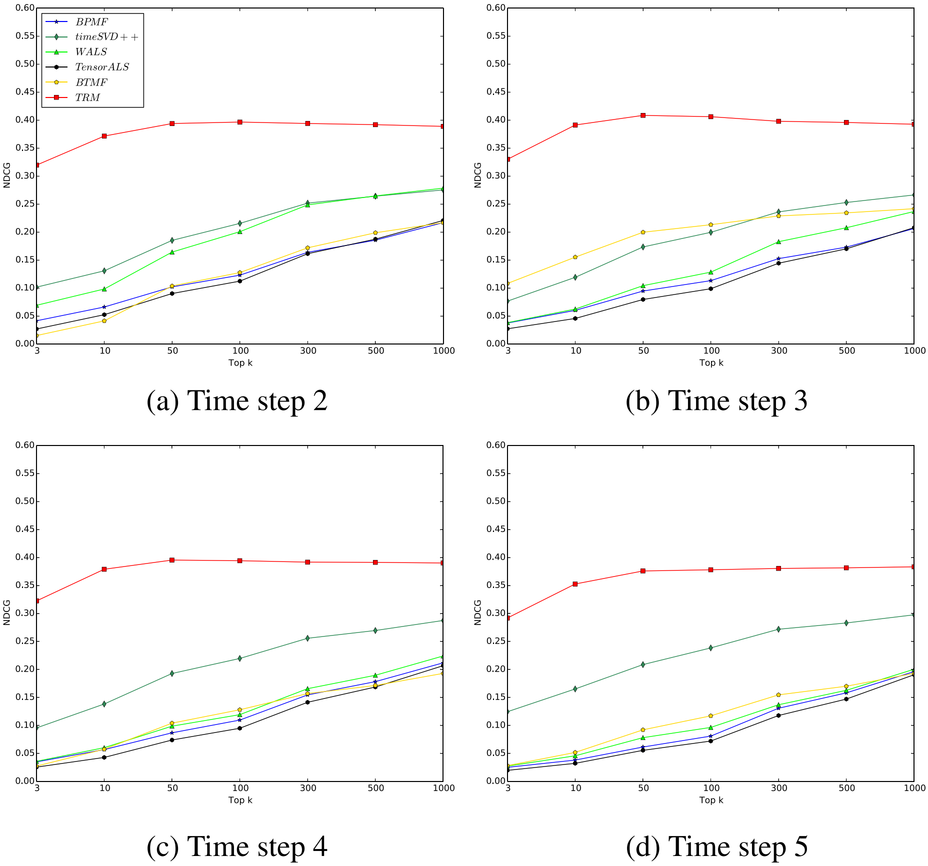


Figure 8 NDCG@k for all comparative models on MovieLen dataset at time step 2–5 and vary k from 3 to 1000. Higher values are better

well. These evidences suggest that considering the effect of temporal information and topic features contributes to the performance of recommendation.

We also investigate the trend of comparative models' performance under NDCG by adjusting the number of recommendations, i.e., the parameter k in $NDCG@k$. Figure 6 shows the influence of k on the seven comparative models by considering $NDCG@k$ on CiteULike dataset. We can see that at time steps 2–5 when the parameter k grows, the performance of our TTARM model is stable, while TRM and all comparative methods increase slightly, illustrating that their recommended items are becoming accurate in ranks, as NDCG is related with the position of "hit" items in the ranked list. Figure 8 shows six models' NDCG@k performance on MovieLens dataset at time steps 2–5. We observe that with k growing, TRM's performance trends to a stable status, while other models increase all the time.

In Table 2, we show comparative models' NDCG@100 performance on CiteULike and MovieLens datasets at time steps 2-6. The boldface and italic highlight the best and second best performers, respectively. Each value in this table is the mean value and standard error, which calculated by running 5 times experiments with same parameters. On CiteULike dataset, our TTARM model performs best since it introduces the topic feature of items and can track user interests over time. Moreover, TRM beats other comparative methods all

Table 2 NDCG@100 Performance (mean ± standard error) comparison of TRM, TTARM and comparative methods. The best performer is in **boldface** and the second is in *italic*. These experimental mean, standard error results are statistics on 5 times experiments with same parameters

Algorithm	Time step 2	Time step 3	Time step 4	Time step 5	Time step 6
	CiteULike				
BPMF	0.0161 ± 0.0040	0.0104 ± 0.0024	0.0089 ± 0.0040	0.0078 ± 0.0003	0.0075 ± 0.0004
timeSVD++	0.0144 ± 0.0018	0.0211 ± 0.0009	0.0104 ± 0.0008	0.0117 ± 0.0005	0.0117 ± 0.0015
WALS	0.1316 ± 0.0092	0.1008 ± 0.0070	0.0924 ± 0.0005	0.0942 ± 0.0102	0.0700 ± 0.0069
TensorALS	0.0856 ± 0.0029	0.0589 ± 0.0039	0.0613 ± 0.0043	0.0535 ± 0.0032	0.0454 ± 0.0074
BTMF	0.0386 ± 0.0000	0.0462 ± 0.0000	0.0442 ± 0.0000	0.0425 ± 0.0000	0.0637 ± 0.0000
TRM	*0.3535* ± 0.0017	*0.2537* ± 0.0019	*0.2358* ± 0.0044	*0.1874* ± 0.0023	*0.2212* ± 0.0020
TTARM	**0.7013** ± 0.0000	**0.6805** ± 0.0000	**0.6967** ± 0.0000	**0.6788** ± 0.0000	**0.6610** ± 0.0000
	MovieLens				
BPMF	0.1231 ± 0.0015	0.1133 ± 0.0023	0.1096 ± 0.0023	0.0810 ± 0.0018	0.0751 ± 0.0037
timeSVD++	*0.2156* ± 0.0000	0.1996 ± 0.0000	*0.2197* ± 0.0001	*0.2385* ± 0.0001	*0.2044* ± 0.0000
WALS	0.2008 ± 0.0022	0.1285 ± 0.0026	0.1193 ± 0.0010	0.0965 ± 0.0022	0.0732 ± 0.0011
TensorALS	0.1125 ± 0.0035	0.0988 ± 0.0016	0.0950 ± 0.0026	0.0721 ± 0.0038	0.0586 ± 0.0026
BTMF	0.1278 ± 0.0006	*0.2132* ± 0.0044	0.1281 ± 0.0196	0.1172 ± 0.0049	0.1341 ± 0.0027
TRM	**0.3966** ± 0.0032	**0.4061** ± 0.0024	**0.3943** ± 0.0032	**0.3781** ± 0.0016	**0.3469** ± 0.0020

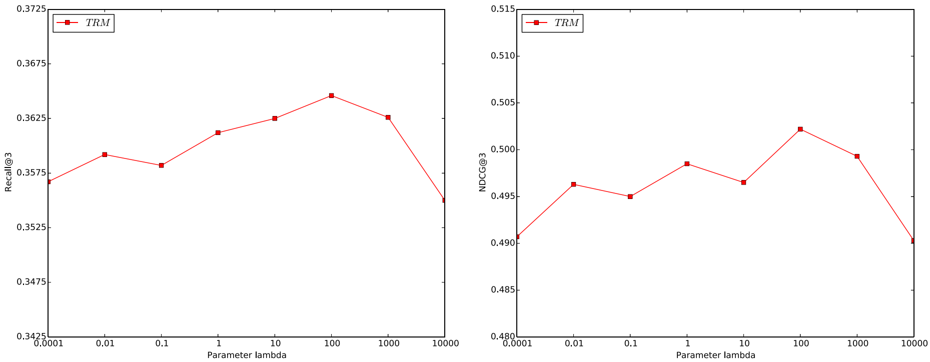
the time. And TTARM can not run on MovieLens dataset for it needs items' content information, which this dataset doesn't involve. However, our TRM model also achieve the best performance. In addition, timeSVD++ performs much better than BPMF, WALS, TensorALS and BTMF at time step 2, 4, 5, 6. However, BTMF beats timeSVD++ at time step 3.

In summary, the proposed TTARM and TRM outperform comparative models in most cases. Our models can learn user interests and topic feature very accurately and this fact contributes to the performance of recommender system.

4.6 Parameter study

In the our models' learning process introduced in Section 3.4, the parameter $\lambda \in (0, \infty)$ is used to balance the model between past and present factors. In order to study the impact of λ on the performance of our proposed model, we conduct experiments with a set of λ values on TRM (i.e., set $\eta = 0$ in TTARM). The results are shown in Figure 9, where the *Recall*@3 and *NDCG*@3 are their mean value on all time steps. Through Figure 9, we can see that TRM performs best when λ is set to be 100, when past and present oriented information is balanced.

In addition, $\eta \in [0, 1]$ is also a crucial parameter in TTARM, as it is used to adjust the weight of topic feature in TTARM. Consequently a series of experiments about η are designed. Similar to parameter λ, we calculate the performance of TTARM under different η with the mean *Recall*@3 and *NDCG*@3 value on all time steps. The results are shown in Figure 10. And from Figure 10, we can see that TTARM works best when η is set to be 0.3-0.6. Furthermore, the performance shows a sharp decline when $\eta < 0.3$ or $\eta > 0.6$ and almost not change when $\eta > 0.3$ and $\eta < 0.6$. This phenomenon explains that both temporal user interests and dynamic topic similarity are crucial factors in our proposed model, overweight or underweight their contributions will hamper the effectiveness of our proposed model.



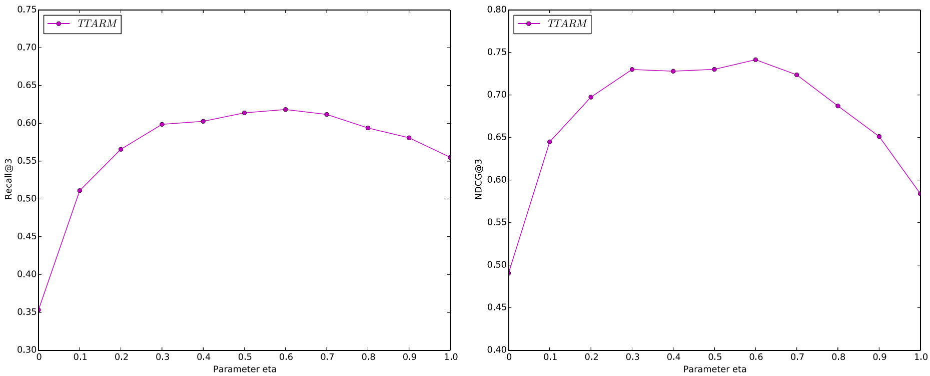
(a) Recall@3 (b) NDCG@3

Figure 9 λ study

We also study the impact of latent factors' number D on our proposed model TTARM. The experiments with different D are conducted on CiteUlike dataset and we evaluate the performance of TTARM under different D with the mean *Recall*@3 and *NDCG*@3 value on all time steps. The results in Figure 11 show that variation of D has little effect on Recall but NDCG decreases a bit with the rise of D. Therefore increasing parameter D can not make a better performance, but increase model's compute complexity on the contrary.

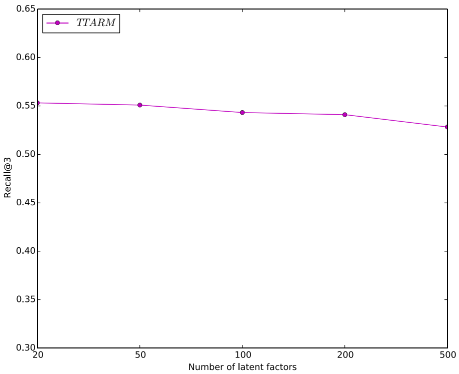
5 Conclusions and future work

In this paper, we propose a *Temporal and Topic-Aware Recommender Model*, namely TTARM, based on collective factorization to model temporal user interests and dynamic topic similarity over time for the purpose of making a better recommendation at current time. When only considering the temporal information, Temporal Recommender Model (TRM) is a special case of TTARM, and it is a dynamic collaborative filtering recommender model. After incorporating topic similarity, the designed TTARM method is a hybrid recommender model which joints *Collaborative Filtering* (CF) and *Content-based* recommender

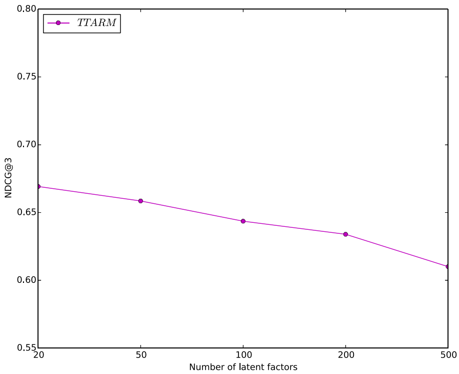


(a) Recall@3 (b) NDCG@3

Figure 10 η study



(a) Recall@3



(b) NDCG@3

Figure 11 Latent factor number study

methods and can form promising recommendations about both existing and newly published items. By applying on two large datasets (i.e. CiteULike and MovieLens), our proposed model outperforms competitive recommender algorithms, which demonstrates that temporal user interests and topic similarity features are crucial factors in recommender systems.

There are still several factors worthy of taking into account in the future. Referring to [49], user interests can be divided into user intrinsic and public interests in TTARM in the future. In addition, learning parameters automatically, incorporating the social information between users into TTARM, and visualization of user interests and dynamic topics are also very interesting works. There are also other possible ways to help with recommendation system. In addition to using collaborative filtering or using topic model features, work like automatically generated lexicons could also help with recommendation systems [4, 23, 43]. We will study them in the future.

Acknowledgments This work was supported by National Key Research and Development Program of China (Grant No. 2016YFB1000902), National Program on Key Basic Research Project (973 Program, Grant No.2013CB329600), and National Natural Science Foundation of China (Grant No. 61472040).

Appendix: Derivation of Gradient Equation

Suppose $R = [R_{ij}]_{m\times n}$, $P = [P_{ij}]_{m\times k}$, $Q = [Q_{ij}]_{k\times n}$, $C = [C_{ij}]_{m\times n}$.

$$
\begin{aligned}
&\|R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\|_F^2 \\
&= tR\{\{R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\}^T \\
&\quad \{R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\}\} \\
&= tR\{\{R^{(t)^T} - [(1-\eta)Q^{(t)^T}P^{(t)^T} + \eta C^{(t)^T}]\} \\
&\quad \{R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\}\} \\
&= tR(R^{(t)^T}R^{(t)}) - 2tR\{[(1-\eta)Q^{(t)^T}P^{(t)^T} + \eta C^{(t)^T}]R^{(t)}\} \\
&\quad + tR\{[(1-\eta)Q^{(t)^T}P^{(t)^T} + \eta C^{(t)^T}][(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}]\}
\end{aligned}
$$

$$
\begin{aligned}
&= tR({R^{(t)}}^T R^{(t)}) - 2[(1-\eta)tR({Q^{(t)}}^T {P^{(t)}}^T R^{(t)}) + \eta tR({C^{(t)}}^T R^{(t)})] \\
&\quad + (1-\eta)^2 tR({Q^{(t)}}^T {P^{(t)}}^T P^{(t)} Q^{(t)}) \\
&\quad + 2\eta(1-\eta)tR({Q^{(t)}}^T {P^{(t)}}^T C^{(t)}) \\
&\quad + \eta^2 tR({C^{(t)}}^T C^{(t)})
\end{aligned} \tag{18}
$$

Then we derive:

$$
\begin{aligned}
\nabla_{P^{(t)}} L &= \nabla_{P^{(t)}} \| R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}] \|_F^2 + \nabla_{P^{(t)}} \alpha \| P^{(t)} \|_F^2 \\
&= -2(1-\eta)R^{(t)}{Q^{(t)}}^T + 2(1-\eta)^2 P^{(t)} Q^{(t)} {Q^{(t)}}^T \\
&\quad + 2\eta(1-\eta)C^{(t)}{Q^{(t)}}^T + 2\alpha P^{(t)} \\
&= 2P^{(t)}[(1-\eta)^2 Q^{(t)}{Q^{(t)}}^T + \alpha I] \\
&\quad - 2[(1-\eta)R^{(t)} - \eta(1-\eta)C^{(t)}]{Q^{(t)}}^T
\end{aligned} \tag{19}
$$

Similarly we can derive the other two gradient equation as follows:

$$
\begin{aligned}
\nabla_{Q^{(t)}} L &= \nabla_{Q^{(t)}} \| R^{(t)} - [(1-\eta)P^{(t)}Q^{(t)} + \eta C^{(t)}] \|_F^2 \\
&\quad + \nabla_{Q^{(t)}} \| R^{(t)} - [(1-\eta)s^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}] \|_F^2 \\
&\quad + \nabla_{Q^{(t)}} \beta \| Q^{(t)} \|_F^2 \\
&= 2(1-\eta)[-{P^{(t)}}^T R^{(t)} + \eta {P^{(t)}}^T C^{(t)} + (1-\eta){P^{(t)}}^T P^{(t)} Q^{(t)}] \\
&\quad + 2(1-\eta)[-{P^{(t-1)}}^T {s^{(t)}}^T R^{(t)} + \eta {P^{(t-1)}}^T {s^{(t)}}^T C^{(t)} \\
&\quad + (1-\eta){P^{(t-1)}}^T {s^{(t)}}^T s^{(t)} P^{(t-1)} Q^{(t)}] + 2\beta Q^{(t)} \\
&= 2(1-\eta){P^{(t)}}^T [\eta C^{(t)} + (1-\eta)P^{(t)}Q^{(t)}] \\
&\quad + 2(1-\eta){P^{(t-1)}}^T {s^{(t)}}^T [\eta C^{(t)} + (1-\eta)s^{(t)}P^{(t-1)}Q^{(t)}] \\
&\quad - 2(1-\eta)({P^{(t)}}^T + {P^{(t-1)}}^T {s^{(t)}}^T)R^{(t)} + 2\beta Q^{(t)}
\end{aligned} \tag{20}
$$

$$
\begin{aligned}
\nabla_{s^{(t)}} L &= \nabla_{s^{(t)}} \| R^{(t)} - [(1-\eta)s^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}] \|_F^2 \\
&\quad + \nabla_{s^{(t)}} \gamma \| s^{(t)} \|_F^2 + \nabla_{s^{(t)}} \lambda \| s^{(t)} - I \|_F^2 \\
&= 2(1-\eta)[-R^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T \\
&\quad + (1-\eta)s^{(t)}P^{(t-1)}Q^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T \\
&\quad + \eta C^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T] + 2\gamma s^{(t)} + 2\lambda(s^{(t)} - I) \\
&= 2(1-\eta)[(1-\eta)s^{(t)}P^{(t-1)}Q^{(t)} + \eta C^{(t)}]{Q^{(t)}}^T {P^{(t-1)}}^T \\
&\quad + 2(\lambda+\gamma)s^{(t)} \\
&\quad - 2[(1-\eta)R^{(t)}{Q^{(t)}}^T {P^{(t-1)}}^T + \lambda I]
\end{aligned} \tag{21}
$$

References

1. Bell, R.M., Koren, Y.: Lessons from the netflix prize challenge. ACM SIGKDD Explorations Newsletter **9**(2), 75–79 (2007)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
3. Chen, L., Shang, S., Zhang, Z., Cao, X., Jensen, C.S., Kalnis, P.: Location-Aware Top-K Term Publish/Subscribe. In: 2018 IEEE 34Th International conference on data engineering (ICDE), IEEE (2018)
4. Chen, Z., Cafarella, M., Jagadish, H.: Long-tail vocabulary dictionary extraction from the Web. In: Proceedings of the Ninth ACM international conference on Web search and data mining, pp. 625–634. ACM (2016)
5. Comon, P., Luciani, X., De Almeida, A.L.: Tensor decompositions, alternating least squares and other tales. J. Chemometr. **23**(7-8), 393–405 (2009)
6. Ding, Y., Li, X.: Time weight collaborative filtering. In: Proceedings of the 14th ACM international conference on Information and knowledge management, pp. 485–492. ACM (2005)
7. Ding, Y., Li, X., Orlowska, M.E.: Recency-based collaborative filtering. In: Proceedings of the 17th Australasian database conference, vol. 49, pp. 99–107. Australian Computer Society, Inc. (2006)
8. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Commun. ACM **35**(12), 61–70 (1992)
9. He, L., Wu, F.: A time-context-based collaborative filtering algorithm. In: IEEE International conference on granular computing, 2009, GRC'09, pp. 209–213. IEEE (2009)
10. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth conference on uncertainty in artificial intelligence, pp. 289–296. Morgan Kaufmann Publishers Inc. (1999)
11. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE international conference on data mining, pp. 263–272. IEEE (2008)
12. Jiang, D., Tong, Y., Song, Y.: Cross-lingual topic discovery from multilingual search engine query log. ACM Trans. Inf. Syst. (TOIS) **35**(2), 9 (2016)
13. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 426–434. ACM (2008)
14. Koren, Y.: Collaborative filtering with temporal dynamics. Commun. ACM **53**(4), 89–97 (2010)
15. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
16. Krestel, R., Fankhauser, P., Nejdl, W.: Latent dirichlet allocation for tag recommendation. In: Proceedings of the third ACM conference on Recommender systems, pp. 61–68. ACM (2009)
17. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–791 (1999)
18. Li, B., Zhu, X., Li, R., Zhang, C., Xue, X., Wu, X.: Cross-domain collaborative filtering over time. In: Proceedings of the Twenty-Second international joint conference on artificial intelligence-Volume Volume Three, pp. 2293–2298. AAAI Press (2011)
19. Linden, G., Smith, B., York, J.: Amazon. com recommendations: Item-to-item collaborative. Internet Comput, IEEE **7**(1), 76–80 (2003)
20. Liu, N.N., Zhao, M., Xiang, E., Yang, Q.: Online evolutionary collaborative filtering. In: Proceedings of the fourth ACM conference on recommender systems, pp. 95–102. ACM (2010)
21. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.-C., Zhang, Z.-K., Zhou, T.: Recommender systems. Phys. Rep. **519**(1), 1–49 (2012)
22. Mnih, A., Salakhutdinov, R.: Probabilistic matrix factorization. In: Advances in neural information processing systems, pp. 1257–1264 (2007)
23. Rong, X., Chen, Z., Mei, Q., Adar, E.: Egoset: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion. In: Proceedings of the Ninth ACM international conference on Web search and data mining, pp. 645–654. ACM (2016)
24. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using markov chain monte carlo. In: Proceedings of the 25th international conference on Machine learning, pp. 880–887. ACM (2008)
25. Shang, S., Ding, R., Yuan, B., Xie, K., Zheng, K., Kalnis, P.: User oriented trajectory search for trip recommendation. In: Proceedings of the 15th international conference on extending database technology, pp. 156–167. ACM (2012)
26. She, J., Tong, Y., Chen, L., Cao, C.C.: Conflict-aware event-participant arrangement and its variant for online setting. IEEE Trans. Knowl. Data Eng. **28**(9), 2281–2295 (2016)

27. Si, X., Sun, M.: Tag-lda for scalable real-time tag recommendation. J. Comput. Inf. Syst. **6**(1), 23–31 (2009)
28. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 650–658. ACM (2008)
29. Song, D., Lifei, Q., Mingming, J., Lejian, L.: A temporal and topic-aware recommender model. In: 2018 IEEE International conference on big data and smart computing (Bigcomp), IEEE (2018)
30. Tang, T.Y., Winoto, P., Chan, K.C.: Scaling down candidate sets based on the temporal feature of items for improved hybrid recommendations. In: Intelligent techniques for Web personalization, pp. 169–186. Springer (2005)
31. Tong, Y., Cao, C.C., Chen, L.: Tcs: efficient topic discovery over crowd-oriented service data. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 861–870. ACM (2014)
32. Tong, Y., Chen, L., Zhou, Z., Jagadish, H.V., Shou, L., Lv, W.: Slade: a smart large-scale task decomposer in crowdsourcing. IEEE Transactions on Knowledge and Data Engineering (2018)
33. Tong, Y., Chen, Y., Zhou, Z., Chen, L., Wang, J., Yang, Q., Ye, J., Lv, W.: The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1653–1662. ACM (2017)
34. Tong, Y., She, J., Ding, B., Chen, L., Wo, T., Xu, K.: Online minimum matching in real-time spatial data: experiments and analysis. Proceedings of the VLDB Endowment **9**(12), 1053–1064 (2016)
35. Tong, Y., She, J., Ding, B., Wang, L., Chen, L.: Online mobile micro-task allocation in spatial crowdsourcing. In: 2016 IEEE 32Nd international conference on data engineering (ICDE), pp. 49–60. IEEE (2016)
36. Tong, Y., She, J., Meng, R.: Bottleneck-aware arrangement over event-based social networks: the max-min approach. World Wide Web **19**(6), 1151–1177 (2016)
37. Tong, Y., Wang, L., Zhou, Z., Ding, B., Chen, L., Ye, J., Xu, K.: Flexible online task assignment in real-time spatial data. Proceedings of the VLDB Endowment **10**(11), 1334–1345 (2017)
38. Tong, Y.-X., She, J., Chen, L.: Towards better understanding of app functions. J. Comput. Sci. Technol. **30**(5), 1130–1140 (2015)
39. Tuan, C.-C., Hung, C.-F., Wu, Z.-H.: Collaborative location recommendations with dynamic time periods. Pervasive Mob. Comput. **35**(Complete), 1–14 (2017)
40. Umbrath, A.S.R.W.W., Hennig, L.: A hybrid plsa approach for warmer cold start in folksonomy recommendation. Recommender Systems & the Social Web 10–13 (2009)
41. Vaca, C.K., Mantrach, A., Jaimes, A., Saerens, M.: A time-based collective factorization for topic discovery and monitoring in news. In: Proceedings of the 23rd international conference on World wide Web, pp. 527–538. International World Wide Web Conferences Steering Committee (2014)
42. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 448–456. ACM (2011)
43. Wang, R.C., Cohen, W.W.: Language-independent set expansion of named entities using the Web. In: Seventh IEEE International conference on data mining, 2007. ICDM 2007, pp. 342–350. IEEE (2007)
44. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long-and short-term preference fusion. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 723–732. ACM (2010)
45. Xiaojun, L.: An improved clustering-based collaborative filtering recommendation algorithm. Clust. Comput. **20**(2), 1281–1288 (2017)
46. Xiong, L., Chen, X., Huang, T.-K., Schneider, J.G., Carbonell, J.G.: Temporal collaborative filtering with bayesian probabilistic tensor factorization. In: Proceedings of the SIAM international conference on data mining, SDM, vol. 10, pp. 211–222. SIAM (2010)
47. Xu, Y., Chen, L., Yao, B., Shang, S., Zhu, S., Zheng, K., Li, F.: Location-based top-K term querying over sliding window. In: International conference on Web information systems engineering, pp. 299–314. Springer (2017)
48. Ye, M., Liu, X., Lee, W.-C.: Exploring social influence for recommendation: a generative model approach. In: Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval, pp. 671–680. ACM (2012)
49. Yin, H., Cui, B., Chen, L., Hu, Z., Huang, Z.: A temporal context-aware model for user behavior modeling in social media systems. In: Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 1543–1554. ACM (2014)

50. Yin, H., Sun, Y., Cui, B., Hu, Z., Chen, L.: Lcars: A location-content-aware recommender system. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 221–229. ACM (2013)
51. Zhang, C., Wang, K., Yu, H., Sun, J., Lim, E.-P.: Latent factor transition for dynamic collaborative filtering. In: Proceedings of the SIAM international conference on data mining, SDM, vol. 14 (2014)
52. Zhang, Y., Tu, Z., Wang, Q.: Temporec: Temporal-topic based recommender for social network services. Mobile Networks and Applications **22**(6), 1182–1191 (2017)
53. Zheng, B., Su, H., Hua, W., Zheng, K., Zhou, X., Li, G.: Efficient clue-based route search on road networks. IEEE Trans. Knowl. Data Eng. **29**(9), 1846–1859 (2017)
54. Zheng, B., Yuan, N.J., Zheng, K., Xie, X., Sadiq, S., Zhou, X.: Approximate keyword search in semantic trajectory database. In: 2015 IEEE 31St international conference on data engineering (ICDE), pp. 975–986. IEEE (2015)
55. Zheng, B., Zheng, K., Xiao, X., Su, H., Yin, H., Zhou, X., Li, G.: Keyword-aware continuous Knn query on road networks. In: 2016 IEEE 32Nd international conference on data engineering (ICDE), pp. 871–882. IEEE (2016)
56. Zheng, K., Shang, S., Yuan, N.J., Yang, Y.: Towards efficient search for activity trajectories. In: 2013 IEEE 29Th international conference on data engineering (ICDE), pp. 230–241. IEEE (2013)
57. Zheng, K., Su, H., Zheng, B., Shang, S., Xu, J., Liu, J., Zhou, X.: Interactive top-K spatial keyword queries. In: 2015 IEEE 31St international conference on data engineering (ICDE), pp. 423–434. IEEE (2015)
58. Zheng, K., Su, H., Zheng, B., Shang, S., Xu, J., Liu, J., Zhou, X.: Interactive top-K spatial keyword queries. In: 2015 IEEE 31St international conference on data engineering (ICDE), pp. 423–434. IEEE (2015)
59. Zheng, K., Zheng, B., Xu, J., Liu, G., Liu, A., Li, Z.: Popularity-aware spatial keyword search on activity trajectories. World Wide Web **20**(4), 749–773 (2017)