4-2018

PCCF: Periodic and continual temporal co-factorization for recommender systems

Guibing GUO

Feida ZHU
Singapore Management University, fdzhu@smu.edu.sg

Shilin QU

Xingwei WANG

Citation

GUO, Guibing; ZHU, Feida; QU, Shilin; and WANG, Xingwei. PCCF: Periodic and continual temporal co-factorization for recommender systems. (2018). *Information Sciences*. 436-437, 56-73.
Available at: https://ink.library.smu.edu.sg/sis_research/4860

Published in Information Sciences, 436–437, April 2018, Pages 56-73.
https://doi.org/10.1016/j.ins.2018.01.019

Accepted version

PCCF: Periodic and continual temporal co-factorization for recommender systems

Guibing Guo[a], Feida Zhu[a,b], Shilin Qu[a], Xingwei Wang[a,*]

[a] *Software College, Northeastern University, China*
[b] *School of Information Systems, Singapore Management University, Singapore*

Keywords:
Recommender systems
Temporal model
Rating timestamps
Periodic effect
Continual effect
Co-factorization model

A B S T R A C T

Rating-only collaborative filtering has been extensively studied for decades with great improvements achieved in predicting a user's preference on a target item at a particular time point. Yet, it remains a research challenge on how to capture users' rating patterns which may drift over time. In this article, we propose a time-aware matrix co-factorization model, called *PCCF*, which considers two types of temporal effects, i.e., *periodic* and *continual*. Specifically, periodic effects refer to the impact of discrete periodic time slices with which users' preferences may be associated, and continual effects refer to the impact of continuous gradual time over which users' preference patterns may change. The fact that users exhibit different preference patterns with respect to different time aspect has been further confirmed by our analysis on three real-world data sets. Together with time-based user biases, we integrate the two kinds of temporal effects into a unified matrix factorization model. Experimental results on the three data sets demonstrate the effectiveness of both kinds of temporal effects for rating prediction as well as the superiority of our approach's performance over that of the other counterparts.

1. Introduction

Nowadays, people are facing the challenge of overwhelming choices over the Internet, which is known as the *information overload* problem. It becomes more and more challenging for users to effectively select the items (e.g., movies, music, products) that suit their preferences. For example, as there are hundreds of thousands of movies in a system, it will be very difficult for a specific user to search for interesting movies. Recommender systems have thus become an essential toolkit for electronic commerce (e-commerce) applications, aiming to improve users' satisfaction and experience by automatically suggesting items of interest in the light of their past behaviors and feedback. Recommendations are often presented in the manner of "People who watched this movie also watch..." or "You watched these 10 movies, so you might also like to watch... ".

There are two important recommendation tasks: *rating prediction* and *top-N item recommendation*. The former task attempts to generate prediction for an active user and a given item, for instance, to predict how many stars a user may give to a movie. The latter task is to select and recommend the top-N items upon which the user is likely to act (e.g., watch,

* Corresponding author.
E-mail addresses: guogb@swc.neu.edu.cn (G. Guo), wangxw@mail.neu.edu.cn (X. Wang).

purchase, read). For example, recommend a user the top-10 most interesting movies along with their ranking orders in the recommendation list. In this article, we work on the first task, i.e., to produce accurate rating prediction for active users.

Collaborative filtering (CF) [3,6] is one of the most effective methods to predict a user's rating towards a given item. The basic idea is to aggregate the preferences of like-minded users (a.k.a, memory-based approaches), or to learn a preference pattern from historical rating data (a.k.a., model-based approaches). It is generally agreed that model-based approaches outperform memory-based ones in terms of predictive accuracy [11,12]. Our work follows the same line of model-based approaches for recommender systems. One of the challenges of traditional CF lies in that it fails to capture users' behavior (rating) patterns which may drift over time in the following two ways. First, users may prefer different kinds of items in different time periods (e.g., busier hours vs. off-work time) [17] regularly. It implies that users may periodically switch their preferences. Second, users' view of items may develop as their expertise levels upgrade from amateur to connoisseur over time [25]. In other words, users may gradually and progressively change their preferences over items.

To handle the scenario where user preferences drift over time, many time-aware recommender systems [12,15,16,18,23–25] have been proposed recently. These studies model rating time from several different perspectives, including global and local temporal effects, periodic and continual rating patterns, etc. Broadly speaking, they take "time" into account either from the view of gradual drifting over time or from the view of periodic time cycle patterns. Based on our closer observations and analysis on three real-world data sets (see Section 3), we argue that a user's preference drift is a complex procedure that cannot be simply explained by either way alone. For example, some users may better enjoy movies on weekends while preferring TV dramas during weekdays, a pattern of strong periodicity. On the other hand, as time goes by, users may also change their taste towards different kinds of movies and dramas, a trend of continuity. As another example of temporal change of user preference, we take Github[1], one of the most well-known service providers for open-source repositories. It presents users' coding preferences (i.e., statistics of commits) both in consecutive months (continually) and in each day of a week (periodically).[2]

In this article, we presume that users' preferences are a combination of periodic and continual temporal effects at any particular time point. Specifically, periodic effects refer to the impact of discrete time slices on preference patterns, such as weekdays and weekends. It reflects relatively regular, stable temporal effects. In contrast, continual effects refer to the impact of continuous time over which users' preferences may gradually change. It reflects relatively dynamic, transient temporal effects. We integrate both kinds of temporal effects into a unified matrix factorization model, aiming to provide more accurate rating prediction. In particular, each kind of temporal effects would result in a predicted rating. A linear combination of both predictions is taken to make a proper rating prediction, in which users' time-based rating biases are incorporated as well. Experimental results on three real-world data sets (i.e., Epinions, Ciao, MovieLens) show that our approach performs better than other counterparts, demonstrating the effectiveness of combining both kinds of temporal effects.

Summary of contributions. Our work makes four key contributions. The first contribution is to propose that a user's rating towards a given item is influenced by a combination of gradually changing factors and periodically reoccurring factors. It differs from other models which treat rating time in an either continual or periodic manner. The second contribution is to provide an insightful data analysis of temporal effects of user preferences on three real-world data sets in Section 3. We show that user preference can be characterized from different perspectives of continual and periodic time. The rating patterns are distinct with respect to different perspectives. The third contribution is to propose a novel time-based recommendation approach, called *PCCF* that incorporates both temporal effects for rating prediction. The fourth contribution is to conduct extensive experiments to evaluate the effectiveness of the proposed approach on the three real-world data sets. We demonstrate that our approach achieves better performance than other counterparts in predictive accuracy.

Outline. The rest of this article is organized as follows. Section 2 provides a brief overview of time-aware recommender systems in the literature. Section 3 presents a data analysis of temporal effects of user ratings on three real-world data sets. With the conclusion drawn in Section 3, Section 4 describes our approaches in detail, where three different models are proposed and discussed. A series of experiments are conducted in Section 5 to evaluate the effectiveness of the proposed approaches. Finally, Section 6 concludes the present work and outlines future research.

2. Related work

Time-aware recommender systems have been widely studied recently, given the fact that user preferences may drift over time. We can classify these works according to the recommendation tasks, i.e., methods for rating prediction and methods for item recommendation. In this article, we focus on the first kind of recommendation approaches, which can be further split into memory-based and model-based ones. Earlier studies are often memory-based, applying heuristic rules to identify a proper neighborhood for an active user and then to aggregate their preferences for rating prediction. For example, Ding et al. [4] design a time weighting factor to decay the similarities to previously rated items if time difference increases relative to the prediction time. The underlying assumption is that the most recent ratings can better reflect users' real preferences at the current time. However, this assumption is not applicable to the users who have long-term or periodic preferences.

[1] https://github.com/.

[2] Such statistics of a repository can be viewed by clicking 'Graphs' and then going to 'Commits'.

Lathia et al. [14] propose an adaptive time function to select proper K values in K-nearest neighbor (KNN) algorithm. In particular, the K values that minimize the errors in different time intervals are selected. Campos et al. [1] propose a time-based UserKNN method based on the assumption that many movie preferences remain only in a short time span. Although rating prediction is generated by the most recent user ratings, user similarity is computed by all user ratings without a proper consideration of time.

Most recent work is model-based approaches where users' temporal rating patterns are modeled and learned according to their historic rating data. Then, rating prediction is generated by applying the learned model to other unrated items for a specific user. The literature has shown that model-based approaches generally outperform memory-based ones in terms of predictive accuracy. Chu et al. [2] take into account the dynamics of item contents (e.g., popularity, click-through rate, freshness) rather than user preferences. They propose a predictive bilinear regression model to provide accurate personalized recommendations. Users' profiles are composed of demographic information and a summary of users' activities in Yahoo! properties. However, those information may not be available in real practice due to the concern of privacy. This article focuses on the time information associated to users' rating data and other additional information will not be used. Li et al. [15] propose a cross-temporal domain predictive model from the perspective of user and item groups. In particular, they assume that the preferences of user (item) groups are relatively stable. Then, the preferences of individual user (item) are a combination of preferences of multiple user (item) groups that the user (it) belongs to. The relatedness of user and item groups between two successive time steps[3] is fed as prior knowledge into an existing bi-clustering graphic model. However, even the preferences of user groups may not always keep unchanged over time. Yin et al. [24] contend that users' rating behaviors are influenced both by users' intrinsic preferences and by temporal context. An item is likely to be less popular as time goes by, such as news. The authors propose a generative model where an item is selected based on either user's intrinsic or temporal topics. An item weighting factor is designed to enhance the proposed model by exploiting the popularity distribution and temporal distribution of items. The authors focus on global temporal effects and the task of item recommendation. In contrast, we consider both global and local temporal effects and work on the task of rating prediction. A number of temporal recommenders for item recommendation has also been proposed in the literature [8,17,18,22].

The most relevant models to our work are as follows. Koren [12] introduces a time-aware matrix factorization model called *timeSVD++* which performs better than other non-temporal models in terms of predictive accuracy on the Netflix data set. The basic idea is that user preferences will gradually change over different time bins. However, periodic temporal effects are not considered in their work. Karatzoglou et al. [10] propose a Multiverse recommendation model where additional types of context information are treated as separate dimensions as a tensor. In this way, the relationships between any two dimensions are taken into consideration and learned for better recommendation. However, its drawback is the inability to accommodate continual temporal effects. Karatzoglou [9] proposes two matrix factorization models—a multiplicative model and an addictive model—by considering the sequential order of items rated by users. Specifically, a user can be modeled by the feature factors both at the current t and previous $t - N$ steps ($N \geq 1$). Their experiments show that better performance can be achieved by integrating time information, and the additive model outperforms the multiplicative model. The best time step t is 2, indicating that the previous user preferences are most indicative to the current user preferences. However, periodic temporal effects are not taken into account. Xiong et al. [23] propose a Bayesian probabilistic tensor factorization (BPTF) model to learn global temporal effects on user-item interactions. In particular, a three-way tensor of user-item-time ratings is decomposed into three latent feature matrices of users, items and time aspects, respectively. The basic idea is that a user's rating is not only dependent on the feature factors of users and items, but also dependent on how these feature factors match the current global trends (i.e., the latent feature vectors of time aspects). However, Zhang et al. [25] argue that the time dimension for recommendations is more likely to be local than global (across all user-item pairs). They propose a Bayesian temporal matrix factorization (BTMF) model under the assumption that user preferences gradually change over time. Specifically, a preference transition matrix is learned, which represents users' time-invariant preference patterns of evolving over time steps. However, we argue that user preferences may not be changed with fixed patterns, but vary distinctly in different time steps. Besides, this model cannot capture repeatable user preferences over different time slices. Vaca et al. [21] propose a time-based collective factorization model, called *Joint Past Present* (JPP) decomposition model, to discover evolving topics and new trends in news. The authors assume that a collection of documents (e.g., news) arrive continuously in batches, and the current topic distribution can be linearly explained by the previous one with the help of a topic transition matrix. However, separating data into batches would cause *data sparsity*[4] even more severe, and thus greatly deteriorate the performance of recommendations.

To sum up, we draw the following conclusions from the literature review. First, temporal information is useful to improve recommendation performance. Second, temporal effects could be either periodic and continual. Third, there is no previous work that combines both periodic and continual temporal effects for rating prediction. We claim that users' temporal preferences cannot be simply explained by either one of temporal effects, but a combination of them.

[3] In this paper, we refer time steps as to time bins for continual (or gradual) temporal effects, and to time slices for periodic temporal effects.

[4] Data sparsity refers to the problem that users generally have only rated a small number of items, resulting in a sparse matrix of user-item interactions. The matrix sparsity is often greater than 90%.

Table 1
The specifications of used data sets.

Features	Epinions	Ciao	MovieLens
# users	22,164	2248	6040
# items	296,277	16,861	3706
# ratings	912,441	3583	1000209
Density	0.0139%	0.0945%	4.4684%
Rating scale	[1, 5]	[1, 5]	[1, 5]
Max # ratings per user	5337	862	2314
Avg # ratings per user	41.17	15.94	165.60
Max # ratings per item	1742	170	3428
Avg # ratings per item	3.08	2.13	269.89
Beginning date	1999-07-05	2000-06-01	2000-04-26
Ending date	2011-05-09	2011-04-12	2003-03-01

3. Data analysis

This section provides a data analysis of temporal effects of user preferences based on three real-world data sets, and shows that both temporal effects capture different characteristics of users' rating behaviors.

3.1. Data sets

Three publicly available data sets, i.e., Epinions, Ciao[5] and MovieLens are used for data analysis and later the experiments in Section 5. The first two data sets are provided by Tang et al. [20], consisting of product reviews (i.e., ratings) along with rating timestamps. The products (i.e., items) cross over multiple product categories, such as movies, computers, sports, etc. More specifically, Epinions contains 22,164 users, 296,277 items and 912,277 ratings along with corresponding timestamps. In Ciao, 2248 users have issued 35,835 ratings over 16,861 items. MovieLens is offerd by GroupLens[6],and it contains 1,000,209 ratings from 6040 users and 3706 movies between April, 2000 and February, 2003, with the restriction that each user has at least 20 ratings. The time units of all data sets are seconds. Note that the three data sets have only the timestamps of users rating items, but no temporal information of items, e.g., being released. Detailed specifications of the three data sets are summarized in Table 1, from which we note that Epinions and Ciao data sets are much sparser in ratings than MovieLens. The time span of the first two rating data are over 10 years and that of the third is around 3 years.

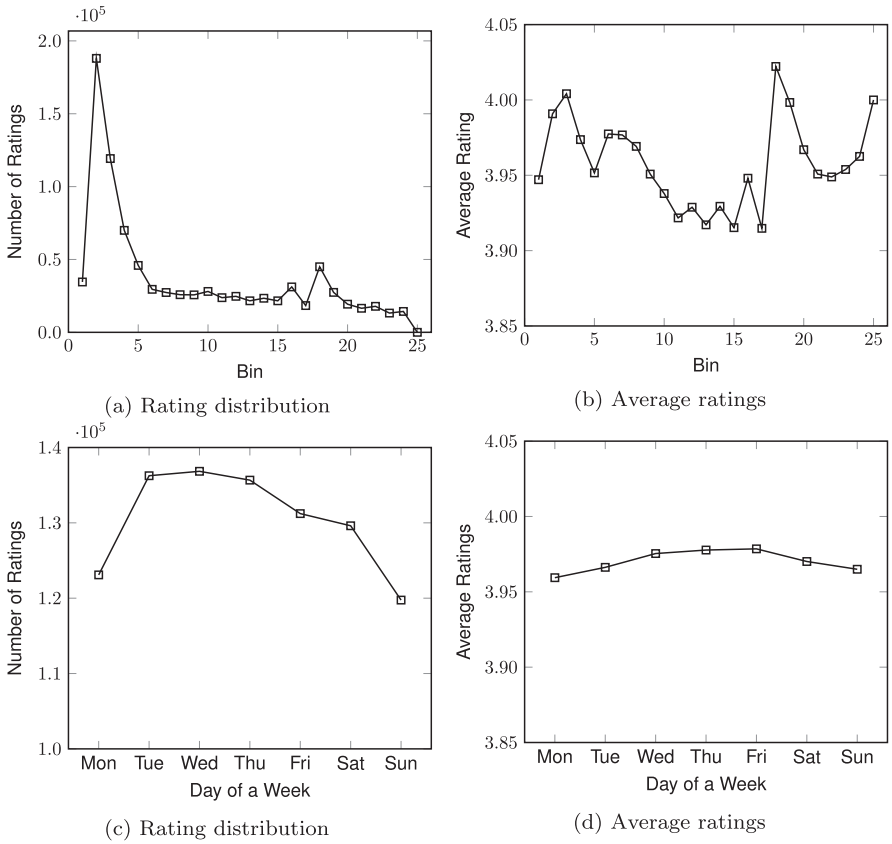
3.2. Case study: Epinions

Two series of experiments are conducted to summarize the statistics of ratings from two different perspectives: all users as a whole and a specific user as individuals. Specifically, on one hand the whole time span is equally split into 25 bins, where each bin covers around half a year (5.68 months to be exact). We believe that user preference may not change very frequently, and half a year would be appropriate to measure their changes. On the other hand, the time span is split into seven time slices each corresponding to a day of a week. Other time slices could be used as well, such as hour of a day, week of a month, etc. Recall that we refer time steps as to time bins for continual (or gradual) temporal effects, and to time slices for periodic temporal effects. We investigate the number of ratings distributed at each time step as well as the average rating values.

Fig. 1 illustrates the results of rating distributions and average ratings over all users on Epinions, where sub figures (a, b) and (c, d) correspond to data split in consecutive and periodic, respectively. Specifically, Fig. 1(a) shows that users were very active in giving ratings since time step 2, and then the activeness continually decreased to a normal level at time step 5. The variations in the first 5 time steps are very large. Fig. 1(c) implies that the numbers of ratings given on Monday and Sunday are smaller than that of ratings on the other days, and the variations are much smaller than that in Fig. 1(a). For average ratings, by comparing Fig. 1(b) with (d), users preferences over days are more stable than those over time bins. Note that the vertical axis of both sub figures has the same range of scale, making it easy to compare the rating variation.

Fig. 2 depicts the results of rating distributions and average ratings over a specific user on Epinions. Specifically, we randomly select a user who have more than 300 ratings. As a result, a user with 367 ratings is chosen. The time bins are ranged from 4 to 19. Although user activities may vary from bin to bin (see Fig. 2(a)), the average rating continually increases (see Fig. 2(b)). Such trend is more significant in Fig. 2(c, d). In particular, the user is less active on Friday (e.g., possibly busier to finish her work before the coming weekend), and she is more generous in giving higher ratings during weekends, especially on Sunday (e.g., due to good mood or entertainment).

[5] http://www.cse.msu.edu/~tangjili/trust.html.
[6] https://grouplens.org/.



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 1. Rating distributions and average ratings of **all users on Epinions**, where (a, b) are results by splitting rating data into continuous 25 bins (i.e., 5.68 months/bin), and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (1.167, 1.169, 1.165, 1.168, 1.166, 1.172, 1.174) corresponding to each day of a week, indicating all days have almost the same deviations overall.

3.3. Case study: Ciao

Similarly, we conduct two sets of experiments on the Ciao data set to understand the nature of temporal effects. The first case is regarding the preferences of all users and the second one is about the preferences of a specific user. The same settings of rating split are adopted for the Ciao data set. The results are presented in Figs. 3 and 4.

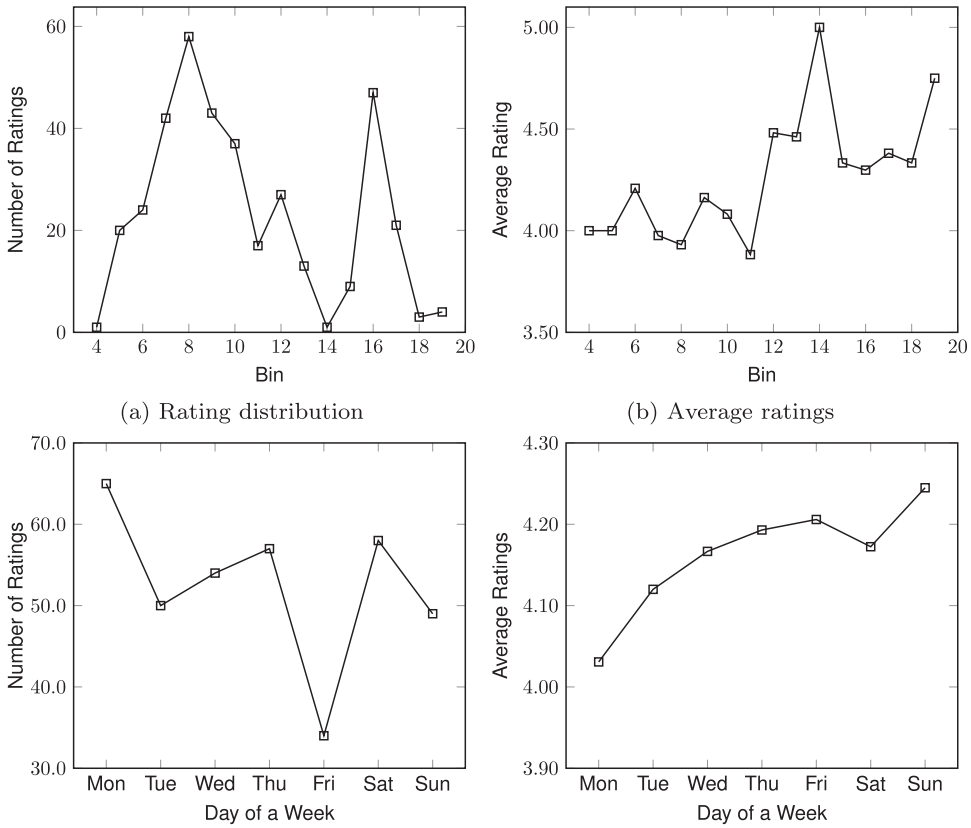
Specifically, Fig. 3(a) is similar to Fig. 1(a) in that user activities are enhanced within the first 5 time bins, but differs in that Ciao users get more and more active after time step 5 whereas Epinions users get decreased continually. Fig. 3(c) illustrates similar characteristics with Fig. 1(c), that is, users participate less active on Monday and Sunday. Fig. 3(b) shows more transient preferences than that in Fig. 3(d), which remains relatively stable.

Fig. 4 represents a user's rating distribution and average ratings over different time steps. The user is randomly chosen by thresholding the number of ratings greater than 100. The ratings cover 5 time bins in Fig. 4(a), and there are 188 ratings in total. The selected user is more active on Tuesday and Friday than on the other days (See Fig. 4(c)). The data shown in Fig. 4(b) are less indicative since the number of time bins is small. Even though, from Fig. 4(d) we can find that the user generally gives lower ratings on weekends than on workdays. It is exactly opposed to the phenomenon shown in Fig. 2(d), indicating the differences of the two data sets to some extent.

3.4. Case study: MovieLens

Same as the previous two data sets, we conduct two experiments on the MovieLens data set to investigate the effect of temporal effects. The first case is regarding the preferences of all users and the second one is about the preferences of a specific user. The same settings of rating split are adopted for this case study. The results are presented in Figs. 5 and 6.

Specifically, Fig. 5(a) shows that the number of user actions is varied a lot in the first 5 time bins, and peaked at time step 5. After that, the trend is declining and reaching a stable state quickly. Fig. 5(c) illustrates similar characteristics with Fig. 1(c) and Fig. 3(c), that is, users participate less actively on Monday and Sunday. Fig. 5(b) shows more transient preferences than that in Fig. 5(d), which remains relatively stable.



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 2. Rating distributions and average ratings of **a specific user on Epinions** (with 367 ratings), where (a, b) are results by splitting rating data into continuous 25 bins, and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (1.067, 1.032, 0.660, 0.867, 0.932, 0.746, 0.937) corresponding to each day of a week. These deviations are much smaller than those of all users, and relatively small considering the rating scale is in [1, 5].

Fig. 6 represents a user's rating distribution and average ratings over different time steps. The user is randomly chosen by thresholding the number of ratings greater than 800. The ratings cover 19 time bins in Fig. 6(a), and there are 822 ratings in total. The selected user is more active on Monday and Saturday than on the other days (See Fig. 6(c)). From Fig. 6(b) and Fig. 6(d), we find it similar to Fig. 5(b) and Fig. 5(d) in that users usually have a lot of temporary preferences but the periodic behavior characteristics is still very stable.

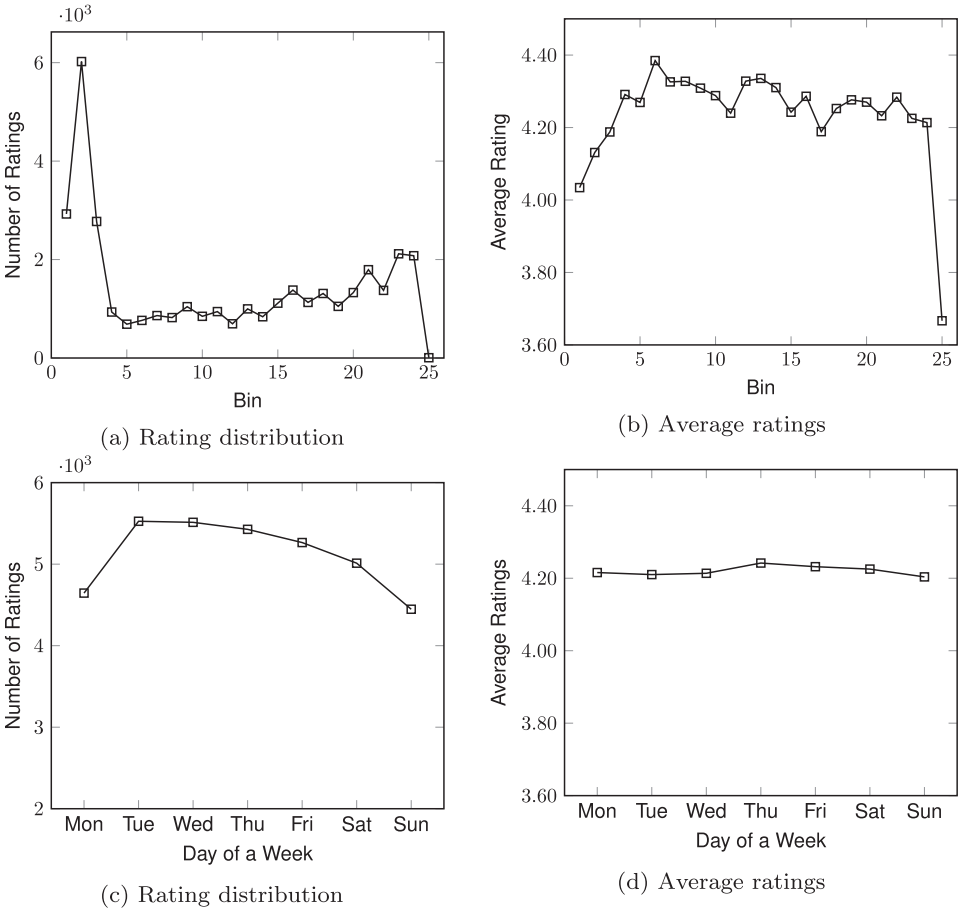
3.5. Summary

The following conclusions can be drawn from the previous case studies on Epinions, Ciao and MovieLens. First, users' continual or gradual preferences are likely to be dynamic and transient. Although user activities may vary a lot in different time steps, the average ratings between continual time steps differ only in a small range. This is consistent with the claim given by Zhang et al. [25]. Second, users' periodic preferences tend to be relatively stable, while a specific user's rating patterns are clearer in periodic time slices than those in continual time bins. The difference in user activities is relatively small among time slices. It is safe to say that users' rating behaviors can be modeled distinctively with different time split manners. Therefore, it is possible to better model user preferences, especially their changes over time by considering both kinds of temporal effects. We next present our approaches based on these conclusions.

4. Periodic and continual temporal matrix factorization models

In this section, we will elaborate two temporal matrix factorization models by integrating both periodic and continual temporal effects, followed by a discussion on items' temporal effects on predictive accuracy.

Specifically, our work builds upon latent factor models which consider bias terms, user- and item-feature matrices at the same time. We then add the effect of both periodic and continual terms on the prediction of user-item ratings. In addition,



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 3. Rating distributions and average ratings of **all users on Ciao**, where (a, b) are results by splitting rating data into continuous 25 bins (i.e., 4.8 months/bin), and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (0.668, 0.653, 0.654, 0.668, 0.657, 0.659, 0.677) corresponding to each day of a week, indicating all days have almost the same deviations overall.

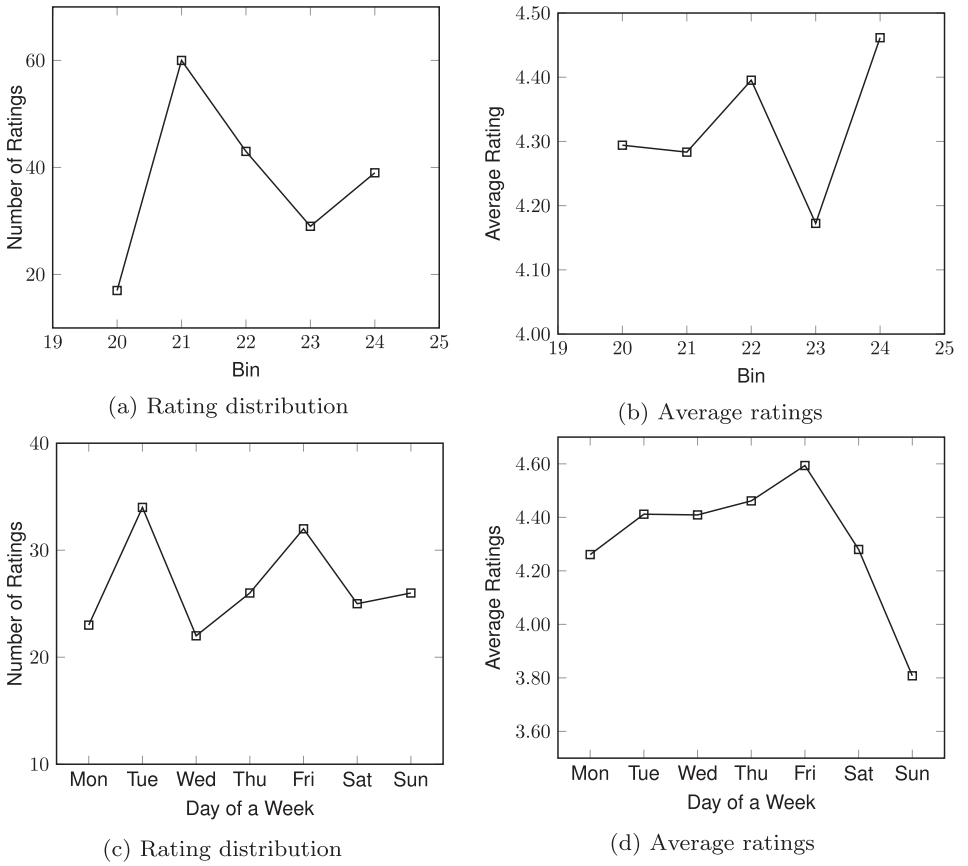
we analyze the relations between two consecutive time steps whereby a time-aware regularization term can be made of, to further smooth the learning of our proposed models. More details will be given in the following subsections.

4.1. Preliminary and notations

To facilitate discussion, we will introduce a number of notations. Let $R = [r_{u,i}]_{m\times n}$ be a user-item rating matrix, where $r_{u,i}$ is a rating given by user u on item i. The number of users and items is m and n, respectively. For simplicity, we preserve symbols u, v for users and i, j for items. Let $T = [t_{u,i}]_{m\times n}$ be a rating time step matrix, where $t_{u,i}$ is the time step associated with rating $r_{u,i}$ regarding user u and item i. The time unit could be real timestamps, such as seconds, millionseconds, or index of time bins or time slices split by time intervals. In this article, we use *time bin* to indicate the time step if continuous time is split (for continual time effects), and *time slice* to imply the time step if discrete time is split (for periodic time effects). Both matrices R and T are very sparse, since users only rated a small portion of items in general. The sparsity is usually greater than 90% (see Table 1). The recommendation task in question is that: given a rating matrix R and associated time step matrix T, generate accurate rating prediction for the missing entries in the rating matrix R.

Matrix factorization [13] has been demonstrated an effective technique to generate accurate rating predictions. Comparing with memory-based approaches, matrix factorization is able to handle large-scale data sets and make prediction more accurate and efficient. We next briefly introduce the basic idea of matrix factorization for rating prediction. The underlying assumption is that both users and items can be characterized by a small number of latent factors in the same feature space. By decomposing the rating matrix R into two low-rank user-feature matrix $P \in \mathbb{R}^{m\times k}$ and item-feature matrix $Q \in \mathbb{R}^{n\times k}$, where k is the number of latent features. Then, a rating prediction can be generated by the inner product of feature vectors of the corresponding user and item. Specifically, the rating prediction is computed by:

$$\hat{r}_{u,i} = P_u^\top Q_i,$$



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 4. Rating distributions and average ratings of **a specific user on Ciao** (with 188 ratings), where (a, b) are results by splitting rating data into continuous 25 bins, and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (0.376, 0.365, 0.407, 0.366, 0.375, 0.385, 0.397) corresponding to each day of a week. These deviations are much smaller than those of all users, and relatively small considering the rating scale is in [1, 5].

where $P_u \in \mathbb{R}^k$ is the latent feature vector of user u, $Q_i \in \mathbb{R}^k$ is the latent feature vector of item i, and $\hat{r}_{u,i}$ denotes the predicted rating for user u on a target item i that she has not rated before.

Koren [13] advocates that the use of user and item biases can greatly improve predictive accuracy. This is due to the fact that some users tend to give higher (lenient users) or lower (strict users) ratings, and some items are likely to receive higher (popular items) or lower (niche items) ratings. In other words, these bias terms can help capture the general trend of users and items in giving (or receiving) ratings. Formally, the rating prediction is enhanced by:

$$\hat{r}_{u,i} = \mu + b_i + b_u + P_u^\top Q_i,$$

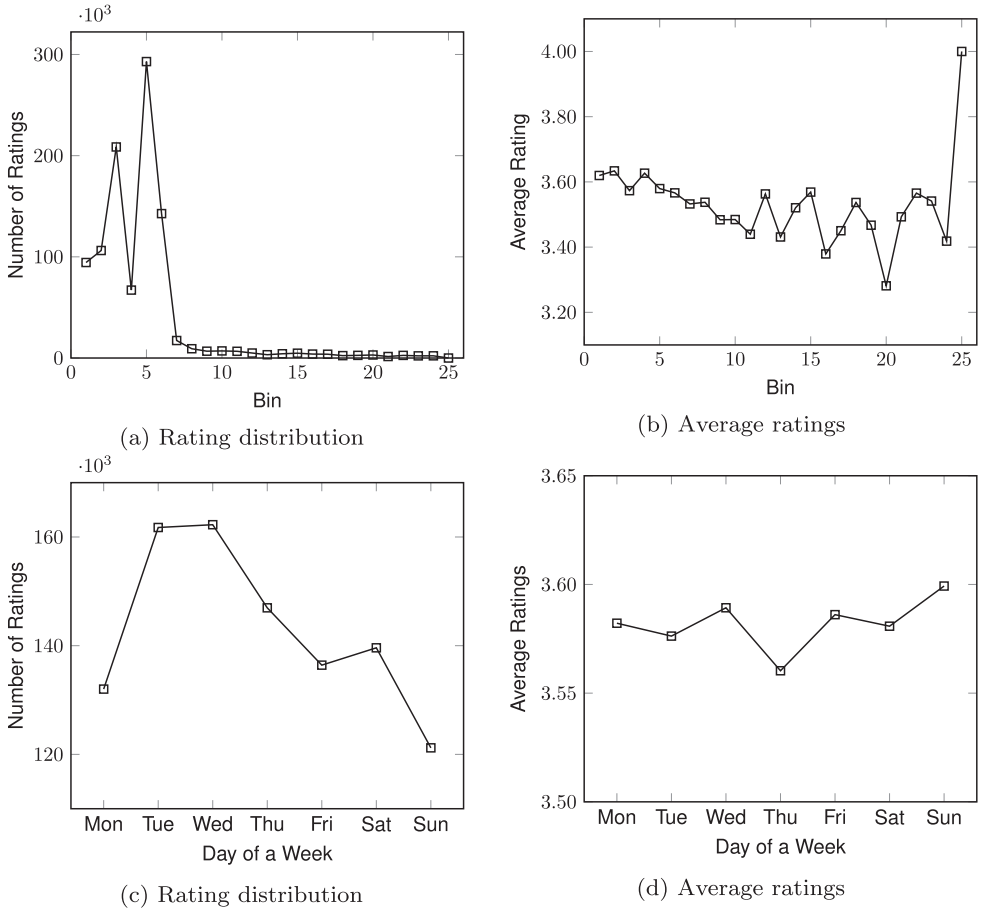
where μ is the global average rating, b_i and b_u are the biases of item i and user u, respectively.

The objective is to ensure that the predicted rating $\hat{r}_{u,i}$ is close to the ground truth $r_{u,i}$. In other words, we'd like to minimize the difference between the two ratings, defining the objective function as follows.

$$\mathcal{J} = \frac{1}{2}\sum_u \sum_i \delta(r_{u,i})(\hat{r}_{u,i} - r_{u,i})^2 + \frac{\lambda_b}{2}\Big(\sum_u b_u^2 + \sum_i b_i^2\Big) + \frac{\lambda_u}{2}\sum_u \|P_u\|_F^2 + \frac{\lambda_i}{2}\sum_i \|Q_i\|_F^2,$$

where $\delta(r_{u,i})$ is an indicator function which equals 1 if user u rated item i and 0 otherwise, $\|\cdot\|_F$ is the Frobenius norm[7], and λ_b, λ_u, λ_i are regularization parameters to help avoid over-fitting. Stochastic gradient descent (SGD) algorithm is usually adopted to learn a local minimum of variables P, Q, B_u, B_i, which will be elaborated in Section 4.4.

[7] https://en.wikipedia.org/wiki/Matrix_norm#Frobenius_norm.



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 5. Rating distributions and average ratings of **all users on MovieLens**, where (a, b) are results by splitting rating data into continuous 25 bins (i.e., 1.4 months/bin), and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (1.106, 1.130, 1.109, 1.123, 1.115, 1.124, 1.108) corresponding to each day of a week, indicating all days have almost the same deviations overall.

4.2. PCCF: periodic and continual co-factorization

The first factor we consider is the continual temporal effect where users' preferences drift over continuous time bins. It captures users' dynamic and transient preferences at a specific time step, as described in Figs. 1(b) and 3(b). Specifically, let $P_u^g \in \mathbb{R}^k$ be a latent feature vector of user u at time bin g, where k is the number of latent features. Then, the rating prediction for user u on item i can be derived by:

$$\hat{r}_{u,i} = \mu + b_i + b_u^g + (P_u^g)^\top Q_i, \tag{1}$$

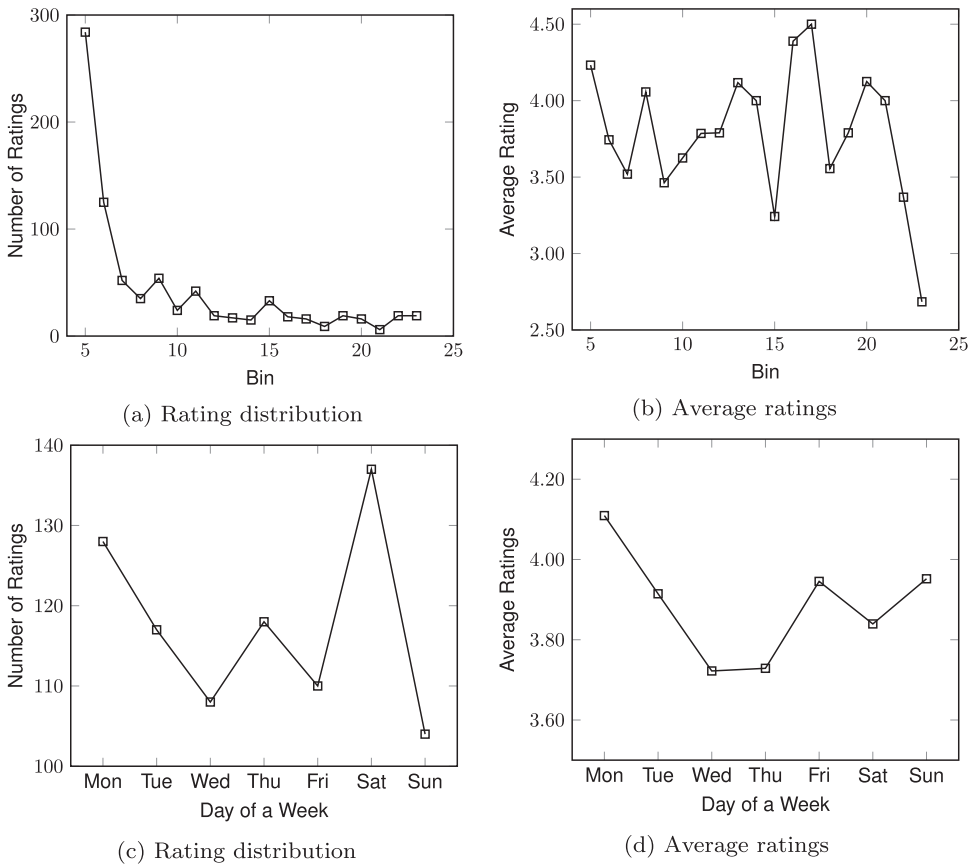
where b_u^g indicates the user bias at time bin g, intending to capture the features illustrated in Figs. 2(b) and 4(b). Note that we opt not to consider the time-aware feature vector Q_i^g and instead we remain the time-invariant vector Q_i. We will defer the explanations to Section 4.5. For now, let us focus on time-aware preferences of users rather than those of items.

The second factor we adopt is the periodic temporal effect where users' preferences can be represented by a number of latent features associated with a time slice. It captures users' cyclic and thus relatively stable preferences at a specific time slice, as described in Figs. 1(d) and 3(d). Similarly, let $P_u^p \in \mathbb{R}^k$ be a latent feature vector of user u at period (i.e., time slice) p. Hence, the rating prediction for user u on item i can be generated as follows:

$$\hat{r}_{u,i} = \mu + b_i + b_u^p + (P_u^p)^\top Q_i, \tag{2}$$

where b_u^p denotes the user bias at time slice p, aiming to model the characteristic illustrated in Figs. 2(d) and 4(d).

Section 3.5 has concluded that combining both kinds of temporal effects may better model users' preferences at a specific time step since they each reflect time-aware user preferences from different perspectives. In other words, user u's preferences at time step (time bin g, time slice p) can be represented by a latent feature vector of both P_u^g and P_u^p. We adopt a convex combination to combine them together due to its simplicity. More complex and non-linear combinations can be



(a) Rating distribution (b) Average ratings

(c) Rating distribution (d) Average ratings

Fig. 6. Rating distributions and average ratings of **a specific user on MovieLens** (with 822 ratings), where (a, b) are results by splitting rating data into continuous 25 bins, and (c, d) are results by splitting rating data into days of a week. The standard deviations in (d) are (1.048, 1.009, 1.016, 0.953, 0.961, 1.075, 0.913) corresponding to each day of a week. These deviations are much smaller than those of all users, and relatively small considering the rating scale in [1, 5].

investigated in the future work. Therefore, the rating prediction for user u on target item i can be written by:

$$\begin{aligned}\hat{r}_{u,i} &= \mu + b_i + \delta(r) b_u^g + \delta(1-r) b_u^p \\ &\quad + r * (P_u^g)^\top Q_i + (1-r) * (P_u^p)^\top Q_i,\end{aligned} \tag{3}$$

where $\delta(x)$ is an indicator function that yields 1 if $x > 0$ and 0 otherwise, and $r \in [0, 1]$ represents the relative importance of continual temporal effect for modeling user preferences; the symbol $*$ denotes the multiplication of two scalars. In particular, if $r = 1$, Eq. (3) will be degraded into Eq. (1), i.e., prediction by continual user preferences only. If $r = 0$, Eq. (3) will be equivalent with Eq. (2), i.e., prediction by periodic user preferences only. As a result, if $r \in (0, 1)$, both temporal effects are linearly considered.

Other than convex combination, we also consider an alternative method—affine combination, given by:

$$\begin{aligned}\hat{r}_{u,i} &= \mu + b_i + \delta(r) b_u^g + \delta(s) b_u^p \\ &\quad + r * (P_u^g)^\top Q_i + s * (P_u^p)^\top Q_i,\end{aligned}$$

where r, $s \in [0, 1]$ represents the importance of continual and periodic temporal effects, respectively, which are empirically determined in our experiments. Note that the requirement $r + s = 1$ of convex combination is not applicable in this case. However, we have empirically find that the affine combination works slightly worse than the convex combination, although the former manner has more degrees of freedom in parameter tuning than the latter one. This implies that the continual and periodic temporal effects are not totally independent, but related to some extent. For our approach, we take Eq. (3) to generate rating predictions. We are aware that other non-linearly combination approaches are possible for an even further study.

Section 3 also shows that user preferences between continual times bins vary only in a small range in general (see Fig. 1(b) and Fig. 3(b)), although greater variations can be observed in a first few time bins. Hence, we design the following

regularization term to avoid over-fitting in model learning.

$$R(g) = \sum_u \sum_g f_u(g, g-1) \| P_u^g - P_u^{g-1} \|_F^2, \tag{4}$$

where $g > 1$, and $f_u(g, g-1) \in (0, 1]$ is a function to measure the correlation between two consecutive time bins for user u. The regularization term intends to add such a constrain that a user's preference vectors among two consecutive time bins are relatively stable. The value of $f_u(g, g-1)$ indicates how much user preferences are changed between two continuous time bins. In particular, $f_u(g, g-1) = 1$ means users do not change preferences while the smaller $f_u(g, g-1)$ value implies the greater changes in user preferences. We propose two ways to define function $f_u(g, g-1)$. The simplest way is to set $f_u(g, g-1) = C$, where $C \in (0, 1]$ is a constant. In other words, it means that we treat the correlation as irrelevant to specific users and time bins. In our case, we take the value $C = 1$. The other way is based on item similarity, given by:

$$f_u(g, g-1) = \begin{cases} \frac{1}{\Delta} \sum\limits_{j \in I_u^t} \sum\limits_{i \in I_u^{t-1}} s(j, i) & \text{if } I_u^t, I_u^{t-1} \neq \emptyset; \\ 1 & \text{otherwise}, \end{cases} \tag{5}$$

where I_u^t, I_u^{t-1} denote the set of items rated by user u at time step t and $t-1$, respectively. The similarity between two items i, j is represented by $s(j, i) \in [0, 1]$, and Δ is a normalization term. The intuition behind is that if items rated in consecutive time bins are highly similar, it indicates that a user's preference has little or small changes, and vice versa. The item similarity can be computed by some similarity measure, such as the Pearson correlation coefficient, cosine similarity or Bayesian similarity [5,7] based on all the user ratings across over all the time bins (to alleviate the data sparsity problem). In our experiments, we adopt the first approach, i.e., $f_u(g, g-1) = 1$ for simplicity, and leave the exploration of the best similarity functions as a part of our future work. In this article, we focus on the two different kinds of temporal effects.

Note that the same intuition of Eq. (4) cannot be applied to periodic temporal effects. In Section 3, we have shown that a specific user can have different preference patterns in different time slices, such as workdays and weekends. Hence, no similar regularization term of periodic preferences is designed in our approach.

Finally, the overall objective function to minimize is derived as follow.

$$\begin{aligned} \mathcal{J} = & \frac{1}{2} \sum_p \sum_g \sum_u \sum_i \delta(r_{u,i}, p, g)(\hat{r}_{u,i} - r_{u,i})^2 \\ & + \frac{\lambda_U}{2} \left(\delta(r) \sum_g \sum_u \| P_u^g \|_F^2 + \delta(1-r) \sum_p \sum_u \| P_u^p \|_F^2 \right) \\ & + \frac{\lambda_G}{2} \left(\delta(r) \sum_g \sum_u f_u(g, g-1) \| P_u^g - P_u^{g-1} \|_F^2 \right) \\ & + \frac{\lambda_B}{2} \left(\delta(r) \sum_g \sum_u (b_u^g)^2 + \delta(1-r) \sum_p \sum_u (b_u^p)^2 \right) \\ & + \frac{\lambda_I}{2} \sum_i \| Q_i \|_F^2 + \frac{\lambda_B}{2} \sum_i b_i^2 \end{aligned} \tag{6}$$

where $\delta(r_{u,i}, p, g)$ is to indicate if user u has rated item i at time bin g and (or) time slice p, $\hat{r}_{u,i}$ is computed by Eq. (3), and variables λ_U, λ_I, λ_G, λ_B are hyper-parameters to avoid over-fitting in model learning, elaborated in Section 4.4.

4.3. PCMF: periodic and continual matrix factorization

The second model we introduce is an alternative matrix factorization model, by mixing latent feature vectors of users with both time bins and time slices. Specifically, let $P_u^{p,g}$ denote user u's latent feature vector at time slice p and time bin g. Different from PCCF, a user's feature vector is related with both time aspects, i.e., time bins and time slices. Then, the rating prediction for user u on item i can be computed by:

$$\hat{r}_{u,i} = \mu + b_i + b_u^{p,g} + (P_u^{p,g})^\top Q_i, \tag{7}$$

where $b_u^{p,g}$ is the user bias for user u at time step (p, g).

Similarly, for the regularization term, we consider the continual changes between two time bins within a time slice. Formally, it is formulated as follows.

$$R(b, g) = \sum_u \sum_{p,g} f_u(p, g, g-1) \| P_u^{p,g} - P_u^{p,g-1} \|_F^2,$$

where $g > 1$, and $f_u(b, g, g-1) \in (0, 1]$ is a function to measure the correlation between time bins g and $g-1$ within a time slice p. The function $f_u(b, g, g-1)$ can be defined in the same way as $f_u(g, g-1)$ (see Eq. (5)).

Hence, we yield the following objective function.

$$\begin{aligned}\mathcal{J} = &\frac{1}{2}\sum_{p}\sum_{g}\sum_{u}\sum_{i}\delta(r_{u,i}, p, g)(\hat{r}_{u,i} - r_{u,i})^2 \\ &+ \frac{\lambda_U}{2}\sum_{g}\sum_{u}\|P_u^{p,g}\|_F^2 + \frac{\lambda_I}{2}\sum_{i}\|Q_i\|_F^2 \\ &+ \frac{\lambda_G}{2}\sum_{p}\sum_{g}\sum_{u} f_u(p, g, g-1)\|P_u^{p,g} - P_u^{p,g-1}\|_F^2 \\ &+ \frac{\lambda_B}{2}\Big(\sum_{p}\sum_{g}\sum_{u}(b_u^{p,g})^2 + \sum_{i} b_i^2\Big)\end{aligned} \tag{8}$$

where $\delta(r_{u,i}, p, g)$ indicates if user u has rated item i at time bin g in time slice p, $\hat{r}_{u,i}$ is computed by Eq. (7), and variables λ_U, λ_I, λ_B, λ_G are regularization parameters for users, items, biases and continual temporal effects, respectively. By comparing with PCCF, this model considers time bins and time slices simultaneously rather than separately. This formalization leads to less parameters (the parameter r in Eq. (6) is not required) and a more compact model. However, the finer granularity of temporal regularization term in Eq. (8) may over specify the temporal effects, resulting in less accurate recommendation performance, which will be demonstrated in Section 5.

4.4. Model learning & analysis

The objective function of our model PCCF can be learned by applying the stochastic gradient descent (SGD) approach on a training data set. Since similar procedure is applicable to learning model PCMF, we focus on model PCCF in this section. For a specific observation $(u, i, p, g, r_{u,i})$, the SGD update rules for variables $b_u^g, b_u^p, b_i, P_u^g, P_u^p, Q_i$ are given as follows.

$$\begin{aligned}\frac{\partial\mathcal{J}}{\partial b_u^g} &= \delta(r)(e + \lambda_B b_u^g), \\ \frac{\partial\mathcal{J}}{\partial b_u^p} &= \delta(1-r)(e + \lambda_B b_u^p), \\ \frac{\partial\mathcal{J}}{\partial b_i} &= e + \lambda_B b_i, \\ \frac{\partial\mathcal{J}}{\partial Q_i} &= e\big(rP_u^g + (1-r)P_u^p\big) + \lambda_I Q_i, \\ \frac{\partial\mathcal{J}}{\partial P_u^g} &= \delta(r)\Big(erQ_i + \lambda_U P_u^g + \lambda_G f_u(g, g-1)(P_u^g - P_u^{g-1}) \\ &\qquad - \lambda_G f_u(g+1, g)(P_u^{g+1} - P_u^g)\Big), \\ \frac{\partial\mathcal{J}}{\partial P_u^p} &= \delta(1-r)\Big(e(1-r)Q_i + \lambda_U P_u^p\Big),\end{aligned} \tag{9}$$

where $e = \hat{r}_{u,i} - r_{u,i}$ is the rating prediction error for user u on item i. For simplicity, we omit the subscripts u, i.

The pseudo-code for model learning and updating is given in Algorithm 1 . To explain, we take as input a training matrix R, time matrices T^g, T^p (resp. time bins and time slices), regularization parameters λ_U, λ_I, λ_B, λ_G, and learning rate γ. We assume that the original time matrix T, with timestamps in some time unit (e.g., seconds), has been preprocessed by formatting the timestamps into a number of time bins (T^g) and of time slices (T^p). First, we initialize the following variables with small random values in (0, 0.01) (line 1), including vectors of user biases $B^g = [b_u^g]_{m\times 1}$, $B^p = [b_u^p]_{m\times 1}$, a vector of item bias $B^i = [b_i]_{n\times 1}$, user-feature matrices $P^g = [P_u^g]_{m\times k}$, $P^p = [P_u^p]_{m\times k}$ and an item-feature matrix $Q = [Q_i]_{n\times k}$, where k is the number of latent factors. If the objective value $\mathcal{J}$ has not converged[8] (line 2) or the maximum number of iterations is not reached, for each observation $(u, i) \in R$ in the training matrix (line 3), we conduct the following operations. The time bins g and time slices p can be retrieved from time matrices T^g and T^p (lines 5–6), respectively. Then, the rating prediction is computed by Eq. (3) (line 7). Once obtaining the rating prediction error e, we proceed to compute the gradients of variable by Eq. (9) (line 8). The variables are then updated accordingly (lines 9–14). Finally, the learned variables are returned as the output of PCCF algorithm (line 15).

The most computational time to learn the PCCF model is mainly taken by evaluating the objective function $\mathcal{J}$ and computing the variable gradients. Specifically, for each iteration the time to compute a rating prediction by Eq. (3) is $O(k)$, i.e., the steps required to calculate the inner product of user-feature and item-feature vectors, where k is the number of latent

[8] The convergence of an objective function in this work means that the objective function has reached its optimal value and status, and further learning cannot provide better solutions.

Algorithm 1: The PCCF algorithm.

Input : $R, T^g, T^p, \lambda_G, \lambda_B, \lambda_U, \lambda_I, \gamma$ (learning rate)

Initialize vectors B^g, B^p, B^i and matrices P^g, P^p, Q with small and random values in $(0, 0.01)$;
while $\mathcal{J}$ *not converged* **do**
  foreach $(u, i) \in R$ **do**
    $r_{u,i} \leftarrow R(u, i)$,
    $g \leftarrow T^g(u, i)$,
    $p \leftarrow T^p(u, i)$,
    compute rating prediction $\hat{r}_{u,i}$ by Equation 3;
    compute variable gradients by Equation 9;
    $b_u^g \leftarrow b_u^g - \gamma \frac{\partial \mathcal{J}}{\partial b_u^g}$,
    $b_u^p \leftarrow b_u^p - \gamma \frac{\partial \mathcal{J}}{\partial b_u^p}$,
    $b_i \leftarrow b_i - \gamma \frac{\partial \mathcal{J}}{\partial b_i}$,
    $Q_i \leftarrow Q_i - \gamma \frac{\partial \mathcal{J}}{\partial Q_i}$,
    $P_u^g \leftarrow P_u^g - \gamma \frac{\partial \mathcal{J}}{\partial P_u^g}$,
    $P_u^p \leftarrow P_u^p - \gamma \frac{\partial \mathcal{J}}{\partial P_u^p}$,
return $B^g, B^p, B^i, P^g, P^p, Q$;

features. Hence, the overall time for objective function $\mathcal{J}$ is $O(k|R|)$, where $|R|$ denotes the number of entries in the rating matrix R. Due to data sparsity, $|R|$ is much smaller than the cardinality of R. On the other hand, all the time to compute gradients $\frac{\partial \mathcal{J}}{b_u^g}$, $\frac{\partial \mathcal{J}}{b_u^p}$, $\frac{\partial \mathcal{J}}{b_i}$, $\frac{\partial \mathcal{J}}{P_u^g}$, $\frac{\partial \mathcal{J}}{P_u^g}$, $\frac{\partial \mathcal{J}}{Q_i}$ is $O(k|R|)$. Hence, the overall computational complexity is linear with the number of rating observations. In other words, our approach can be learned fast and applicable to large-scale data sets.

4.5. Discussion: items' temporal effects

Till now, we have elaborated our approaches in modeling users' preferences over time steps. Another assumption we adopt is that items' characteristics are relatively stable over time. This assumption is also taken by some previous works, including [9,18,23]. However, some other researchers contend that items' characteristics could be also influenced by time [15,16,24,25]. A typical example is that an item may be outdated and get less popular and attention from users, or users may not prefer an outdated item. In other words, the most exemplified case is that the popularity of items may decrease over time, such as news and movies. Even though, these works do not take time-based regularization term towards temporal effects of items. That is, no previous work suggests that items' characteristics between two time steps will not be relatively stable. It is one of the main differences from users' temporal preferences.

Nevertheless, we can model items' temporal characteristic in the form of Q_i^g, where g is a time bin. Then, the rating prediction for user u on item i can be obtained by:

$$\hat{r}_{u,i} = \mu + b_u + b_i^g + P_u^\top Q_i^g,$$

where b_i^g is the bias for item i on time bin g. Alternatively, we can additionally integrate users' temporal preferences. Then, the prediction can be rewritten as follows:

$$\hat{r}_{u,i} = \mu + b_u^g + b_i^g + (P_u^g)^\top Q_i^g.$$

Similar objective function can be designed to learn the model by minimizing the differences between predictions and ground truth.

However, we empirically find that: (1) items' temporal effect has little effect in improving recommendation performance; and (2) combining both users' and items' temporal effects cannot perform better than the case of users' temporal effects only. The results imply that items' temporal characteristics are ineffective, or kept relatively stable. This phenomenon can be explained in two aspects. First, most of items' characteristics are time-invariant, i.e., are not influenced by time. Second, although items' popularity may change from time to time, users' temporal preferences are likely to capture such kind of changes. For example, as previously mentioned, the case of items being less popularity is equivalent with that of users not preferring outdated items.

Therefore, we conclude that items' temporal effects are negligible in our case (based on the used data sets). We focus on the study of users' temporal preferences in this article.

5. Experiments and results

In this section, we will evaluate the effectiveness of our approaches on three real-world data sets with the aim to study: (1) the influence of number of bins and the ways to slice time; (2) the impact of parameter r for the combination of temporal effects; and (3) the performance of our approaches in comparison with other counterparts.

5.1. Experimental setup

Data sets. The three real-world data sets described in Section 3.1 are used in our experiments. We have tried two different ways to split rating data according to time information. The first way is to preserve the early 80% ratings for each user as a training set and leave the recent 20% ratings as a test set. The second way is the traditional 5-fold cross validation approach, i.e., the original data set is randomly split into 5 folds four of which is used as the training set and the rest as a test set. We empirically find that both approaches produce similar performance trends. Hence, in this article we adopt the 5-fold cross validation to evaluate predictive performance, since it produces different subsets for evaluation at each iteration. The average results of five iterations are taken as the final performance. Cross validation is an often-adopted method to determine model parameters, whereby a model can be adapted to different data sets.

Evaluation metrics. Two well-known metrics are used to measure the predictive accuracy, i.e., mean absolute error (MAE) and root mean square error (RMSE). More formally, they are define by:

$$\mathrm{MAE} = \frac{\sum_u \sum_i |\hat{r}_{u,i} - r_{u,i}|}{N},$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_u \sum_i (\hat{r}_{u,i} - r_{u,i})^2}{N}},$$

where N is the number of ratings in the test set. In general, smaller values of MAE and RMSE indicate better recommendation accuracy.

Comparison methods. A number of comparison models are implemented and compared in our experiments.

- **PMF** [19] is a baseline matrix factorization approach where no additional information is employed.
- **timeSVD++** is proposed by Koren [12] where user preferences are assumed to gradually change over different time bins.
- **TAM** is the additive model proposed by Karatzoglou [9] where the sequential order of ratings is considered in a matrix factorization model.
- **BPTF** is proposed by Xiong et al. [23] where time is regarded as an additional dimension in a tensor, i.e., as a global effect for all users and items.
- **PCCF** is our approach described in Section 4.2 where both continual and periodic temporal effects are linearly combined (with parameter r) and co-factored in a matrix factorization model.
- **PCMF** is our approach described in Section 4.3 where users' preferences are simultaneously associated with time bins and time slices.

PMF and timeSVD++ are provided by an open-source recommendation toolkit, called *LibRec*[9]. TAM and our approaches (PCCF, PCMF) are implemented under the framework of LibRec. BPTF[10] is kindly provided by Xiong et al. [23], who additionally provide an implementation of PMF. We have compared the performance of their version of PMF and LibRec PMF, and found that similar results are obtained by the two implementations. The settings of BPTF (e.g., the number of samples is 50) given by the authors [23] are adopted in our experiments, while the other experimental settings (e.g., the number of latent features, iterations, etc.) are the same as our own approach. The training and test data generated by our approach is used as input to the BPTF, which ensures a fair comparison.

Parameter settings. For each method, there are a number of parameters to tune, such as regularization parameters, learning rate, etc. In our experiments, the most proper values are either determined by empirical results (by trying out different settings) or suggested by the original papers. Besides, we tune the learning rate in {0.0001, 0.001, 0.01, 0.1}, and fix the number of latent features $k = 10$ in order to focus on model comparison. Grid searches are generally conducted to determine parameters. The number of maximum iteration is 300. Each setting will run 10 times to average the performance. The settings leading to the best MAE values are adopted.

5.2. The impact of regularization parameters

For simplicity, we set regularization terms $\lambda_U = \lambda_I = \lambda_B$ and λ_G, and search for their optimal values in the range {0.0001, 0.001, 0.01, 0.1}. In this section, we use the Epinions data set as an example, while the same searching procedure is also used for the Ciao and MovieLens data sets. The settings for other parameters are: $r = 0.5$, periodic time sliced by Hour of a Day, and given 5 continual time bins. Note that they are not optimal settings, and further analysis of each parameter will

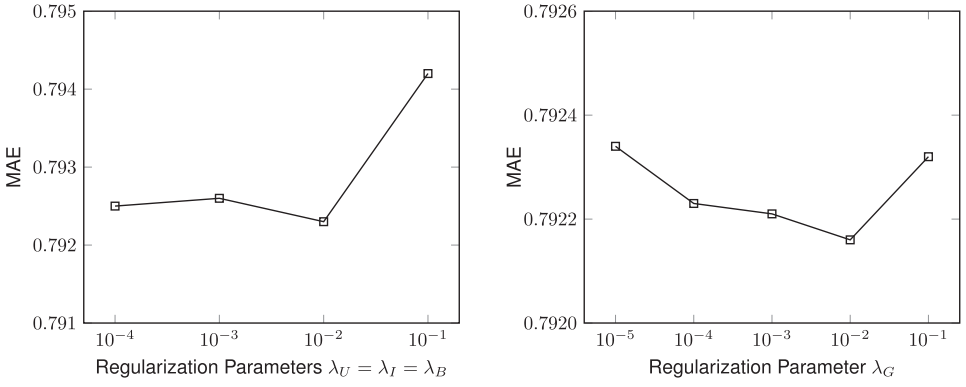


Fig. 7. The impact of regularization parameters on Epinions: (a) λ_U, λ_I, λ_B and (b) λ_G.

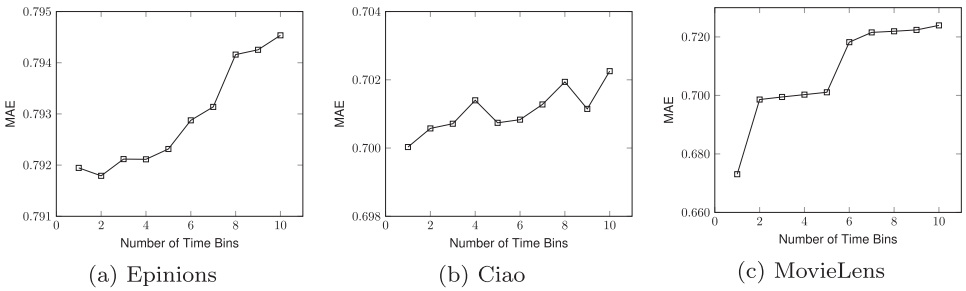


Fig. 8. The impact of time bins on predictive accuracy on three different data sets: (a) Epinions, (b) Ciao and (c) MovieLens.

be given later. The results are depicted in Fig. 7, where the best values for all the regularization parameters are 0.01 on Epinions. Further tuning λ_U, λ_I, λ_B separately may result in better performance.

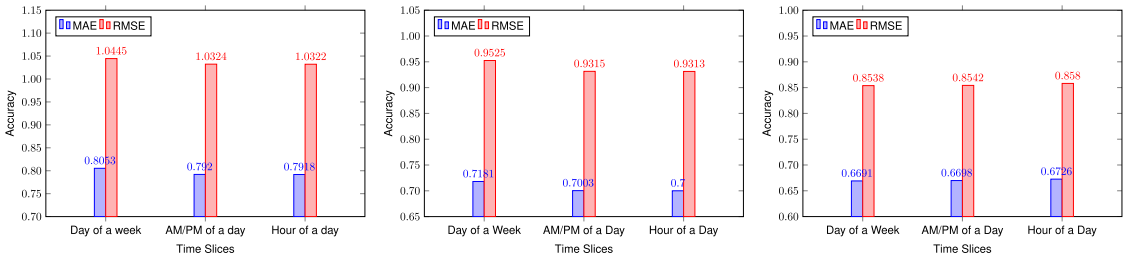
5.3. The impact of time bins

This section studies how the number of time bins influences the recommendation performance of our approach PCCF[11] in terms of MAE[12] The best settings for PCCF are learning rate $\gamma = 0.001$, regularization parameters $\lambda_U = \lambda_I = \lambda_B = 0.01, \lambda_G = 0.01$ on Epinions, settings $\gamma = 0.001, \lambda_U = \lambda_I = \lambda_B = 0.5, \lambda_G = 20$ on Ciao, and settings $\gamma = 0.001, \lambda_U = \lambda_I = \lambda_B = 0.01, \lambda_G = 1000$ on MovieLens. In addition, we select a time slice as hour of a day, and presume the importance of continual effect $r = 0.5$. Then, we vary the number of bins from 1 to 10 stepping by 1. The results are presented in Fig. 8. The trends on Epinions, Ciao and MovieLens are similar in that a greater number of time bins may deteriorate the performance. In particular, Fig. 8 shows the best number of time bins for Epinions, Ciao and MovieLens is 2, 1, 1, respectively. Although it shows that continual time has a small impact on predictive accuracy, the impact could be much greater when the importance parameter r is further tuned as shown in Section 5.5, and the time slices are better selected as shown in Section 5.4. Nevertheless, by carefully choosing a proper number of time bins, a better performance may be achieved by aggregating both continual and periodic temporal effects.

5.4. The impact of time slices

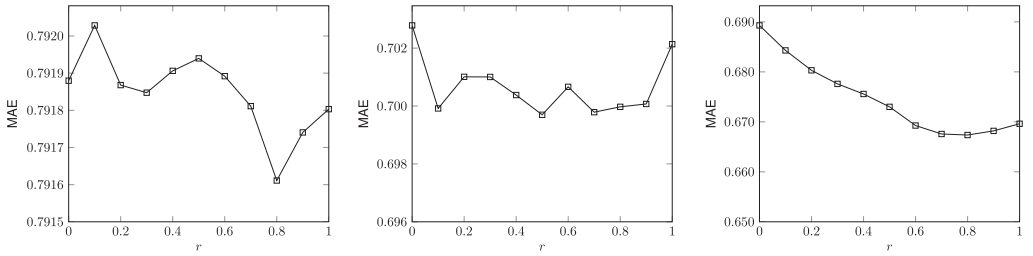
With the previous settings of number of time bins, we proceed to vary the ways to determine time slices and investigate its influence on predictive accuracy. We have considered three manners: Day of a Week, Hour of a Day and AM/PM of a day. Alternative manners include Day of a Month, Week of a Month, etc. The experimental results are illustrated in Fig. 9. Specifically, the best splitting method by Hour of a Day for both Epinions and Ciao works the best. The performance of Hour of a Day and that of AM/PM of a day are very close. However, the best splitting method for MovieLens is by Day of a Week,

[9] LibRec: http://www.librec.net/.
[10] BPTF: http://www.cs.cmu.edu/~lxiong/bptf/bptf.html.
[11] We focus on PCCF rather than PCMF in this article.
[12] The RMSE values follow similar trends with MAE.



(a) Epinions (b) Ciao (c) MovieLens

Fig. 9. The impact of time slices on predictive accuracy on three different data sets: (a) Epinions, (b) Ciao and (c) MovieLens.



(a) Epinions (b) Ciao (c) MovieLens

Fig. 10. The impact of parameter r on predictive accuracy on three different data sets: (a) Epinions, (b) Ciao and (c) MovieLens.

Table 2
Performance of comparison methods on Epinions.

	PMF	timeSVD++	TAM	BPTF	PCMF	PCCF
MAE	1.4224	0.9826	0.8069	0.8196	0.8000	**0.7913**
RMSE	1.8657	1.3230	1.0408	1.0918	1.0410	**1.0306**

the performance of which is very close to that of AM/PM of a day. The results show that a proper manner to slice periodic time can result in better performance. Although Hour of a Day and Day of a Week are the best choices in our experiments, we may have to choose alternative time slices when our approach being applied to some other datasets.

By comparing Fig. 9 with Fig. 8, we observe that the MAE differences among time slices are relatively greater than those among time bins. This is in accordance with our previous conclusion in Section 3.5: users' rating patterns are clearer in periodic time slices than those in continual time bins.

5.5. The impact of parameter r

Our next step is to investigate the impact of parameter r, i.e., the importance of continual effects in Eq. (3). The optimal settings of number of time bins and manner of time slices are adopted according to previous discussion. We vary the value of r from 0.0 to 1.0 stepping by 0.1 on all data sets. The results are presented in Fig. 10. Although MAE values vary in a small range on Epinions, the best value of parameter r is observed at 0.8. In contrast, the trend is clearer on Ciao where $r = 0.5$ outperforms either continual effects only (i.e., $r = 1$) and periodic effects only (i.e., $r = 0$). The curve of MovieLens is quite smooth: the performance increases as r grows, and reaches the extreme when $r = 0.8$. Hence, we conclude that integrating both continual and periodic effects is useful to improve predictive accuracy. To sum up, we have found that a proper integration of both effects is able to improve recommendation performance.

5.6. Comparison with other models

In this section, we will compare our approaches with a baseline (without time information) and other state-of-the-art time-aware recommendation models. The results are presented in Tables 2, 3 and 4, corresponding to the performance on Epinions, Ciao and MovieLens, respectively.

Specifically, when the data set is sparse, time-aware recommenders perform better than time-unaware recommender, i.e., PMF. The best settings for timeSVD++ on three data sets are: 2 time bins and no time decaying weights. The performance of timeSVD++ is the worst among time-aware recommenders, also implying that the continual temporal effects for individual users are less significant. This is confirmed by the comparison with BPTF in which global temporal effects are considered.

Table 3
Performance of comparison methods on Ciao.

	PMF	timeSVD++	TAM	BPTF	PCMF	PCCF
MAE	1.7058	0.8405	0.7044	0.7642	0.7021	**0.6979**
RMSE	2.2222	1.1940	0.9368	1.0521	0.9294	**0.9265**

Table 4
Performance of comparison methods on MovieLens.

	PMF	timeSVD++	TAM	BPTF	PCMF	PCCF
MAE	0.6701	0.6982	0.6950	0.6800	0.6824	**0.6669**
RMSE	0.8606	0.8937	0.8775	0.8811	0.8706	**0.8517**

The best number of time bins for BPTF is also 2, by tuning the value from 1 to 10 stepping with 1. Among the comparison methods, TAM works the best by considering the user preferences at both current and previous time steps. For the results of MovieLens, all the methods show good performance when the data set is relatively dense.

Most importantly, our approaches, i.e., PCMF and PCCF consistently outperform the other counterparts on all data sets, especially when data sets are sparse. That is, the achieved MAE and RMSE values are the lowest ones. Although the improvements are relatively small, Koren [13] has shown that small improvements in predictive accuracy may have a great impact on practical recommendation performance. It is a common practice that relatively small improvements are often achieved step by step in the field of recommender systems. Furthermore, we find that PCCF generally gains better accuracy than PCMF. Therefore, we can conclude that combining both periodic and continual temporal effects is effective in predicting accurate users' ratings on unknown items. Additionally, a convex linear combination works better than a mixture modeling of both kinds of temporal effects.

6. Conclusions and future work

This article made efforts to model users' preferences by simultaneously considering both periodic and continual temporal effects. The periodic temporal effects refer to the impact of periodic time slices on recommendation performance while continual temporal effects indicate the impact of continuous time bins on rating prediction. They described different temporal aspects of user preferences. By conducting a data analysis on three real-world datasets, we found that users had different rating patterns in the light of time bins and time slices as a whole and as individual users. Upon with the observed phenomenon, we designed a co-factorization model by linearly integrating both impacts of periodic and continual temporal effects. The experimental results on the three real-world data sets demonstrated that our approaches worked better than other state-of-the-art time-aware recommendation models.

Our future work will follow two lines of research. The first line is to investigate the influence of similarity measures for model regularization (see Eq. (5)). Although this article focuses on the recommendation task of rating prediction, item recommendation is believed a more optimal task to accomplish for recommender systems. Hence, the second line of future work is to study the temporal effects for top-N item recommendation. In addition, more data sets will be adopted to further enhance the effectiveness of our approach.

Acknowledgments

This work is supported by the National Natural Science Foundation for Young Scientists of China under Grant No. 61702084, the Natural Science Foundation of Liaoning province under Grant No. 20170540319, and the Fundamental Research Funds for the Central Universities No. N161704001.

References

[1] P.G. Campos, A. Bellogín, F. Díez, J.E. Chavarriaga, Simple time-biased KNN-based recommendations, in: Proceedings of the Workshop on Context-Aware Movie Recommendation (CAMRa), 2010, pp. 20–23.
[2] W. Chu, S.-T. Park, Personalized recommendation on dynamic content using predictive bilinear models, in: Proceedings of the 18th International Conference on World Wide Web (WWW), 2009, pp. 691–700.
[3] C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, Recommender Systems Handbook (2011) 107–144.
[4] Y. Ding, X. Li, Time weight collaborative filtering, in: Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM), 2005, pp. 485–492.
[5] G. Guo, J. Zhang, N. Yorke-Smith, A novel Bayesian similarity measure for recommender systems, in: Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI), 2013, pp. 2619–2625.
[6] G. Guo, J. Zhang, N. Yorke-Smith, TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI), 2015, pp. 123–129.
[7] G. Guo, J. Zhang, N. Yorke-Smith, A novel evidence-based Bayesian similarity measure for recommender systems, ACM Trans. Web (TWEB) 10 (2) (2016) 8:1–8:30.

[8] Y. Kabutoya, T. Iwata, K. Fujimura, Modeling multiple users' purchase over a single account for collaborative filtering, in: Proceedings of the 11th International Conference on Web Information Systems Engineering (WISE), 2010, pp. 328–341.
[9] A. Karatzoglou, Collaborative temporal order modeling, in: Proceedings of the 5th ACM conference on Recommender Systems (RecSys), 2011, pp. 313–316.
[10] A. Karatzoglou, X. Amatriain, L. Baltrunas, N. Oliver, Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering, in: Proceedings of the 4th ACM Conference on Recommender Systems (RecSys), 2010, pp. 79–86.
[11] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), ACM, 2008, pp. 426–434.
[12] Y. Koren, Collaborative filtering with temporal dynamics, Commun. ACM 53 (4) (2010) 89–97.
[13] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer (Long Beach Calif) 42 (8) (2009) 30–37.
[14] N. Lathia, S. Hailes, L. Capra, Temporal collaborative filtering with adaptive neighbourhoods, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2009, pp. 796–797.
[15] B. Li, X. Zhu, R. Li, C. Zhang, X. Xue, X. Wu, Cross-domain collaborative filtering over time, in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), 2011, pp. 2293–2298.
[16] N.N. Liu, L. He, M. Zhao, Social temporal collaborative ranking for context aware movie recommendation, ACM Trans. Intell. Syst. Technol. (TIST) 4 (1) (2013) 15.
[17] D. Rafailidis, A. Nanopoulos, Modeling the dynamics of user preferences in coupled tensor factorization, in: Proceedings of the 8th ACM Conference on Recommender Systems (RecSys), 2014, pp. 321–324.
[18] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized Markov chains for next-basket recommendation, in: Proceedings of the 19th International Conference on World Wide Web (WWW), 2010, pp. 811–820.
[19] R. Salakhutdinov, A. Mnih, Probabilistic matrix factorization, in: Advances in Neural Information Processing Systems (NIPS), 20, 2008, pp. 1257–1264.
[20] J. Tang, H. Gao, H. Liu, A. Das Sarma, eTrust: understanding trust evolution in an online world, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2012, pp. 253–261.
[21] C.K. Vaca, A. Mantrach, A. Jaimes, M. Saerens, A time-based collective factorization for topic discovery and monitoring in news, in: Proceedings of the 23rd International Conference on World Wide Web (WWW), 2014, pp. 527–538.
[22] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, J. Sun, 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 (KDD), 2010, pp. 723–732.
[23] L. Xiong, X. Chen, T.-K. Huang, J.G. Schneider, J.G. Carbonell, Temporal collaborative filtering with Bayesian probabilistic tensor factorization, in: Proceedings of the 2010 SIAM International Conference on Data Mining (SDM), 2010, pp. 211–222.
[24] H. Yin, B. Cui, L. Chen, Z. Hu, Z. Huang, 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 (SIGMOD), 2014, pp. 1543–1554.
[25] C. Zhang, K. Wang, H. Yu, J. Sun, E.-P. Lim, Latent factor transition for dynamic collaborative filtering, in: Proceedings of the 2014 SIAM International Conference on Data Mining (SDM), 2014, pp. 452–460.