Temporal Dynamics in Recommender Systems

Cheng Luo

A thesis in fulfillment of the requirements for the degree of

Doctor of Philosophy

UNSW
AUSTRALIA

School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

March 2016

THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: **Luo**

First name: **Cheng** Other name/s:

Abreviation for degree as given in the University calendar: **PhD**

School: **School of Computer Science and Engineering** Faculty: **Faculty of Engineering**

Title: Temporal Dynamics in Recommender Systems

Abstract 350 words maximum

In real-world scenarios, user preferences for items are constantly drifting over time as item perception and popularity are changing when new fashions or products emerge. The ability to model the tendency of both user preferences and item attractiveness is thus vital to the design of recommender systems (RSs). However, conventional methods in RSs are incapable of modeling such a tendency accordingly, leading to an unsatisfactory recommendation performance.

This thesis proposes a framework for the temporal dynamics problem in RSs. The temporal properties and dynamic information in user preferences and item attractiveness derived from user feedback over items are modeled, learned and applied to predict user preferences on items over time. The framework provides original solutions to improve the performance of RSs by incorporating and exploiting this significant but traditionally neglected information.

Firstly, a novel probabilistic temporal model for RSs is developed to tackle the inherent nonlinear and non-Gaussian dynamic problem with the complex and diverse real-world recommendation scenarios. It tracks simultaneously latent factors that represent user preferences and item attractiveness. A learning and inference algorithm combining a sequential Monte Carlo method and the Expectation-Maximization algorithm for this model is developed to tackle the top-k recommendation problem over time. Secondly, a novel probabilistic personalized and item-wise temporal model is proposed to solve the cold start transition (CST) problem by collaborative tendencies without any prior assumptions about the structure of the dynamics. The CST problem is first defined in this thesis, which is a result that users often leave feedback on an item only once and on only one period, preventing from learning any dynamics directly. Finally, a Bayesian Wishart matrix factorization method is proposed to model the temporal dynamics of variances due to sudden changes and other local temporal effects among user preferences and item attractiveness. It combines the collapsed Gibbs sampling method and the elliptical slice sampling method.

The presented models and learning algorithms are validated experimentally on several real-world public benchmark datasets. The experimental results demonstrate that those models and algorithms significantly outperform a variety of state-of-art methods in RSs.

FOR OFFICE USE ONLY Date of completion of requirements for Award

Originality Statement

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Cheng Luo
March 29, 2016

Copyright Statement

Cheng Luo
March 29, 2016

Authenticity Statement

Cheng Luo
March 29, 2016

Abstract

In real-world scenarios, user preferences for items are constantly drifting over time as item perception and popularity are changing when new fashions or products emerge. The ability to model the tendency of both user preferences and item attractiveness is thus vital to the design of recommender systems (RSs). However, conventional methods in RSs are incapable of modeling such a tendency accordingly, leading to an unsatisfactory recommendation performance.

This thesis proposes a framework for the temporal dynamics problem in RSs. The temporal properties and dynamic information in user preferences and item attractiveness derived from user feedback over items are modeled, learned and applied to predict user preferences on items over time. The framework provides original solutions to improve the performance of RSs by incorporating and exploiting this significant but traditionally neglected information.

Firstly, a novel probabilistic temporal model for RSs is developed to tackle the inherent nonlinear and non-Gaussian dynamic problem with the complex and diverse real-world recommendation scenarios. It tracks simultaneously latent factors that represent user preferences and item attractiveness. A learning and inference algorithm combining a sequential Monte Carlo method and the Expectation-Maximization algorithm for this model is developed to tackle the top-k recommendation problem over time. Secondly, a novel probabilistic personalized and item-wise temporal model is proposed to solve the cold start transition (CST) problem by collaborative tendencies without any prior assumptions about the structure of the dynamics. The CST problem is first defined in this thesis, which is a result that users often leave feedback on an item only once and on only one period, preventing from learning any dynamics directly. Finally, a Bayesian Wishart matrix factorization method is proposed to model the temporal dynamics of variances due to sudden changes and other local temporal effects among user preferences and item attractiveness. It combines the collapsed Gibbs sampling method and the elliptical slice sampling method.

The presented models and learning algorithms are validated experimentally on several real-world public benchmark datasets. The experimental results demonstrate that those models and algorithms significantly outperform a variety of state-of-art methods in RSs.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Dr. Xiongcai Cai for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor and mentor for my Ph.D study.

I thank my fellow teammates, Nipa Chowdhury and Shaobo Dang, for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years.

Last but not the least, I must express my very profound gratitude to my wife, to my parents and to my parents-in-law for providing me with unfailing support and continuous encouragement throughout my years of study and through writing this thesis and my life in general. This accomplishment would not have been possible without them.

Thank you.

Abbreviations

BPR	**B**ayesian **P**ersonalized **R**anking
CF	**C**ollaborative **F**iltering
ELM	**E**xtreme **L**earning **M**achine
EM	**E**xpectation **M**aximization
GWP	**G**eneralized **W**ishart **P**rocess
MAP	**M**aximum **A** **P**osterior
MF	**M**atrix **F**actorization
PF	**P**article **F**iltering
PMF	**P**robabilistic **M**atrix **F**actorization
POI	**P**oint-**O**f-**I**nterest
RSs	**R**ecommender **S**ystems
SGD	**S**tochastic **G**radient **D**escent

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to Recommender systems

As the information technology is pervasive in our daily life, the abundance of information available to individuals in their online activities, such as digital content distribution and electronic commerce, leads to the difficulty of individuals to efficiently locate objects, such as documents and movies, to meet their requirements. Technologies to precisely target user interests and efficiently facilitate their decision-making processes become increasingly important in this era of information overload. For most of scenarios, users only have some general idea about their preferences and this level of thoughts cannot be easily transformed to some specific keywords and requirement. Therefore, an information filtering tool is needed to help users to overcome the problem of information overload.

Personalized recommender systems (RSs) are such powerful tools that automatically identify various user preferences and item characteristics and yield personalized item recommendations to match the interests of users. The term "item" is a general notation utilized to denote any object that the system recommends to its users, for example, products, people [49, 132] or other objects [7, 168, 113, 31]. This characteristic of personalized RSs is different from that of information retrieval systems in which users should explicitly feed

the systems with the inputs that compactly and precisely describe the information that users are seeking. Meanwhile, the conventional techniques for information retrieval are developed without taking a special care of personalization, which makes them unable to deliver precisely the exact services to satisfy an individual's personal taste. In addition to personalized RSs, there are also RSs that provide users with non-personalized recommendations. The non-personalized recommendations are much simpler to generate [115, 184]. While this kind of recommendation is useful and usually found in news recommendations [11, 10, 115], it is not typically addressed by the research of RSs and it is not the focus of this thesis. Without the specific statement, the term "personalized" will be omitted in the following discussion for the purpose of clarity.

Resnik and Varin developed the first recommender system in 1997 [190]. Since then, RSs have constantly been evolving to meet the requirement of higher recommendation accuracy and user satisfaction. With the help of RSs, the user can quickly reach the point of his interests without going through irrelevant information. Currently, RSs are an integral component of web-based applications, such as book recommendations in Amazon [148], news recommendation in Google [68], and movie recommendations in Netflix [129]. In the annual Securities and Exchange Commission filing, Netflix has partially contributed its success of business model to the effective use of its recommender system [194]. A relevant and effective recommender system increases user satisfaction which in turn increases the probability to take positive actions (such as rating in explicit feedback; click in implicit feedback) by the user on the recommended items.

1.1.1 Recommendations and User Feedback

Personalized recommendations are usually offered as a ranked list of items for the user. Items are ranked by RSs based on the estimation of user preferences under some constraints. In order to fulfill such a computational task, the historical and context information relating to user preferences and item attractiveness is constantly collected by RSs. Generally speaking, there are two forms of input data for RSs. They are explicit feedback

and implicit feedback. The commonly used explicit feedback in RSs is numeric ratings, such as 1-5 stars provided by customers in Amazon's online book reviews; the binary preference, such as the like button clicked by Facebook users; the ordinal ratings, such as, one of the terms "strongly like, like, neutral, dislike" selected by users to express their preferences. Implicit feedback is usually collected transparently to front-end users. This type of feedback is usually recorded by the back-end servers in the form of the server logs, such as the clicking history of users on items or their viewing history over web pages. When this type of implicit feedback is exploited, RSs usually assume that users prefer those items that they have touched with to those items whose logs cannot be found.

1.1.2 Categories

Conventional methods in RSs can be roughly classified into three categories: content-based or context-based filtering approach, collaborative filtering (CF) approach, and their hybrid approach.

Collaborative filtering The basic idea behind methods in CF approach is based on a simple and straightforward observation: individuals usually resort to recommendations provided by others for generating daily and routine decisions [39, 115]. This observation is especially true when individuals lack sufficient personal experience or competence to evaluate the potentially overloaded number of items exposed to them. For example, it is a common practice for a consumer to peruse the reviews provided by other consumers or professional review websites before making a purchase decision; individuals often ask some advice from their friends about places of attractions to visit or restaurants to get together.

In order to model this observation, the developed methods in CF predict a user's interests over all the unseen items by exploiting its historical feedback and the historical feedback of "like-minded" users. Specifically, if the user in the system agreed in the past with some other users, the past items preferred by these similar users should also be appreciated by

the active user. The similarity computed from the historical feedback between the active user and its similar users can, to some extents, reflect the preferences that are passed from the other users to the active user. This is the reason why this approach is named after collaboration.

Content-based approach Content-based or context-based RSs are constructed to recommend the active user items that are matched with the ones that are preferred by the user in the past [198, 204, 216, 57, 67]. The similarity of items is usually computed from the features that are extracted from the content or context information describing these items. If the properties of user profiles that store user interests and preferences are also constructed and matched up with the features of the item description, the personalized recommendation can be achieved. For example, when users that are coming from the similar geographical regions or classified into the same age or cultural group demonstrate the preferences over some books or movies in one genre, it is highly possible for those users to like some other books or movies coming from the same genre.

Note that in the classical way to classify the types of RSs, there are three other separate approaches in RSs, which are the demographic-based approach [166], community-based approach [87, 93, 96], and knowledge-based approach [208, 78]. The demographic-based approach assumes that different recommendations should be generated for different demographic groups. The community-based approach assumes that the friends explicitly identified by the active user could generate more meaningful feedback to learn the preferences of the active user than relying on the recommendations from similar but anonymous users. The knowledge-based approach is usually case-based [115]. It recommends items to the active user on the basis of specific domain knowledge about how the user preferences and requirements are satisfied by some specific item features. Because all of those three approaches actually exploit the information sourced from the contents or contexts of either users or items, they are classified into the content-based approach.

Hybrid methods The third approach in RSs combines the above two approaches in order to take advantages of both CF and the content-based approach [46, 94, 70]. The content or context information not only mitigates the cold start problem [13, 203], which will be discussed in next subsection, in CF approach but also supplies the approach with more intuitive and broaden constraints to be explored. For example, RSs exploiting information from social networking, such as Twitter and Facebook, are an active field of recommender system research and receiving more and more attention in recent years. Methods using social relations of the active user can also be roughly classified into the hybrid approach that combines CF and content-based approach. Social networks introduce social relations of users. When this kind of relations is exploited by RSs, the close friends are assumed to share similar opinions and preferences and they have some profound influences on the preferences of the active user. This practice shares the concept of "like-minded" users in CF. However, rather than learning the similarity from the historical data, social relations are usually extracted from a graph model that is constructed from the content information relating to users.

1.1.3 Comparison

Because users are usually exposed to overwhelming information on the web, the gathered datasets in real-world scenarios are usually extremely sparse. For explicit feedback in the systems, only an extremely tiny portion of items have been rated by users and most users only provide the system with a little feedback. Meanwhile, as users do not bother to revise their feedback, the feedback is commonly non-repetitive. Even though more information could be collected through users' viewing and clicking history, this phenomenon is still largely true for implicit user feedback. This is known as the problem of data sparsity in RSs [115]. Therefore, for each user or item, instance based techniques in data mining and machine learning may not obtain sufficient training data to yield personalized recommendations. CF, which effectively shares the historical feedback from other users and items, has been shown to be efficient at mitigating the problem of data sparsity.

The information applications incorporating RSs are also dynamic in nature [128]. There are always new users starting to embrace the systems and new items emerging. For those newly emerged items, CF usually does not collect enough feedback from users. Similarly, those newly appeared users may still be inactive and do not generate enough feedback over items. For those users and items, RSs adopting CF approach are incapable of generated accurate recommendations. These problems are known in the community of RSs as the problems of cold start users [146, 38] and cold start items [202], respectively. In this case, content-based or context-based approach, which usually does not rely on the gathering of the interaction information from users and items, is still able to provide users with some systematic recommendations. Compared with CF that does not rely on side information, the automatic analysis of content or side information relating to users and items, such as feature extraction and selection, usually introduces some extra burden into RSs. Meanwhile, the content information is usually expensive to obtain and there may be some risks to exploit, which imposes some extra constraints on the applicability of this approach. For example, even though all the user information had been carefully made anonymous, the Netflix company had to cancel its second competition in developing a more accurate recommender system for personalized movie recommendations [2].

In addition to the problem of data sparsity and the problems of cold start users and cold start items, conventional methods in RSs also encounter other issues, such as limited coverage and lack of diversity for personalized recommendations. In order to leverage the collaboration from similar users, the active user must share a set of commonly rated items with other users. In order to impose constraints on social relations, the active user should not be an isolated node. Diversity [224, 136] is usually considered as an opposite measure to similarity. To some extents, this problem is reminiscent of the well-known trade-off between exploration and exploitation [180, 142] in the study of machine learning and data mining. Therefore, it may not be very interesting for users to have been recommended a set of items with similar characteristics and attractiveness, because it will take users longer to explore the range of unseen items. Since the current research in RSs is still focusing on generating recommendation lists with higher accuracy to meet personalized satisfaction, those problems will not be explored in detail in the following discussion.

1.2 Motivation

Matching item characteristics with user preferences is essential to the success of RSs. User preferences for products are constantly drifting over time as they are gaining more knowledge and experience and becoming more mature in a long-term period or simply changing their moods under various circumstances in a short-term period. Meanwhile, product perception and popularity are changing when new fashions or products emerge or seasonal influences become evident. The varying nature of user preference and item attractiveness is especially prominent in cases where users are repeatedly exposed to consume some products of certain categories, such as movies, news and music [145, 158]. For example, users may watch different types of movies with their children from those with their partners at different times; movies may lose their popularity because they are filmed a while ago but get popularity again because they win awards or are closely relating to current affairs.

Because the user feedback is continuously gathered by applications that incorporate RSs in the real-world scenarios, the tendency of not only user preferences but also item attractiveness is actually residing in the input data to RSs. For example, Figure 1.1 and Figure 1.2 excerpt from [128] demonstrate two temporal effects within the Netflix dataset for movie recommendations. Figure 1.1 shows that there is a sudden jump of the average movie rating, and Figure 1.2 illustrates that the average rating for each movie tends to keep increasing since the first day when the movie had been rated in the system. In this regard, the non-stationary user and item factors actually require the real-world RSs to be continuously updated to adapt to the current tendency of user preferences and item popularity. Note that similar to the situation in the static scenario, it is usually not the case that users repeatedly consume the same item over time.

1.2.1 Temporal dynamics in recommender systems

Traditional methods in RSs mainly focus on the generation of personalized recommendations for users whose preferences are assumed to be static. This static assumption is

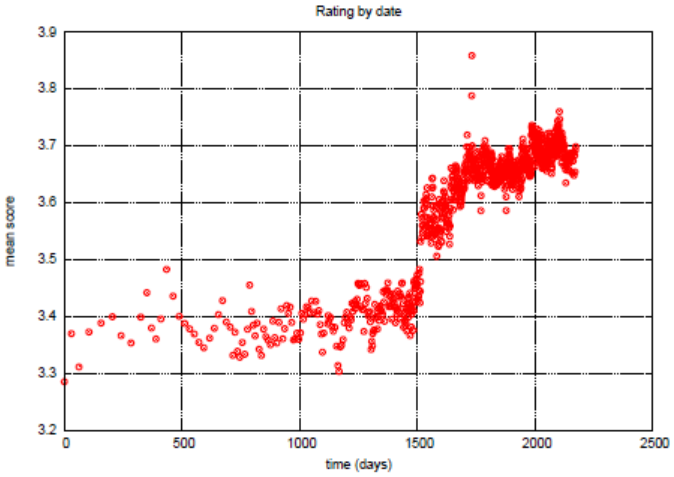


Figure 1.1: The first temporal effect on the Netflix dataset [128].

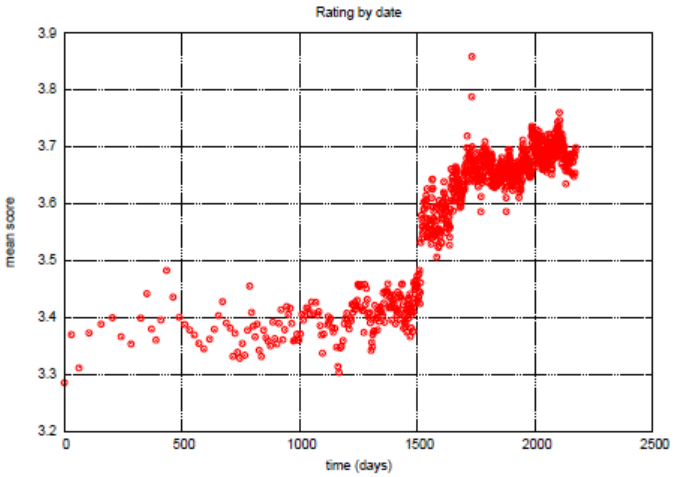


Figure 1.2: The second temporal effect on the Netflix dataset [128].

such a strong assumption that it may cause problems when users and items exist in the systems over a long period. Although those systems may have successfully learned user preferences and item popularity from the past data in the training period, the predicted user preferences and item attractiveness may be no longer valid when those estimated characteristics have changed during the testing period or in the real-world deployment. More importantly, static methods in RSs are fitted into the feedback without taking into account that the data are actually generated by a dynamic process. Thus, the learned models only attempt to yield the best average models that aim to capture the global pat-

terns of user and item behaviors in the past. As time goes, it is very difficult for those RSs constructed under static assumption to refrain their recommendations from drifting away their ground truth.

Due to their incapability of modeling such a tendency accordingly, those methods lead to lower-than-expected predictive power and thus unsatisfactory recommendation performance in many real-world deployments. Therefore, the ability to model the tendency of both user preferences and product attractiveness is vital to the design of RSs [128, 145, 158, 80, 215, 7]. In fact, it is shown [128] that the performance of recommendation system can be improved by explicitly modeling the temporal behaviors of user preferences in the underlying model.

Categories In general, methods in temporal or dynamic RSs can be divided into four approaches in terms of how the temporal and dynamic information is utilized [160]: heuristic approach, binning-based approach, online updating approach and dynamic-based approach. The detailed review of temporal dynamic methods in RSs will be covered in Chapter 2.

Heuristic approach As the current user preferences may be better represented by the current feedback than the old one, methods in the heuristic approach exploit the temporal influences by discounting the temporal effects of the past feedback on the future behaviors of users and items while preserving the long-term trends that are inherent in the data. Usually, a pivot point will be determined empirically or learned from the historical data [73, 229]. Concept drift methods have been used to detect a pivot point when user's interests are dramatically changing. Meanwhile, there are also some methods of this approach that extract the temporal influences from user feedback and use them as the input data or features for later usages. Therefore, methods belonging to heuristic approach can be regarded as a pre-processing stage that re-weights available information to make the original algorithms temporal-aware. Even though the time-based weighting approach is easy to understand and simple to implement, it is prone to information loss and tends to undervalue the past data.

For instance, the time instance when the rating prediction occurs is selected as the pivot point in [73]. By taking as input the time distance between the pivot point and the timestamp associated with a given rating, the exponential function is utilized to discount the temporal influence of the rating. The user similarity in CF is then weighted by the calculated temporal weights.

Binning-based approach Methods in the binning-based approach appear in various forms, but they all bucketize the data by time intervals to which the data are collected [128, 117, 170]. Hence, both training and testing data in this approach could be from the same interval and they are usually generated by a random split of all the data in this interval. Thus, the predictions for a user's interests to the unseen items are actually *post hoc* predictions about what her interests *would have been* in the past, rather than what her interests would be in the future. While this approach demonstrates its predictive ability when user preferences have periodic attributes, it is not a general approach that is able to be applicable to many other real-world scenarios. Meanwhile, in practice, it is too rigorous to consider only the periodic effects on the trends in user preferences and item attractiveness while excluding other temporal effects. For example, user preferences over clothes are highly influenced by seasonal effects but it is unwise to exclude the influence of the current fashion that is irrelevant to seasons.

For example, after grouping user feedback into time intervals and randomly splitting training and testing data in each time interval, the local temporal efforts within any time interval are captured by associated latent factors in the procedure of matrix factorization [128]. The temporal dynamics across time intervals are largely neglected in the model.

Online updating approach Methods in the online updating approach aim at modeling how a recommender system would be used in practice, i.e. only information in the past is used to predict the unknowns that are in the future. The underlying model should be updated to reflect the newly arrived feedback in the system. Online updating mechanisms are thus developed to update model parameters dynamically [71, 149, 14]. However, methods of this approach usually place an emphasis on the scalability of the updating stage and ignore the modeling of dynamics of user taste and item attractiveness.

For example, in order to catch up the latest trends of user preferences, similarity computation in CF has been online updated when newly arrived user feedback is available [154]. In order to avoid rebuilding the whole model from scratch and reducing the computational complexity, the updating procedure is somewhat similar to the procedure of updating model parameters in incremental learning.

Dynamic-based approach The dynamic-based approach explicitly models the temporal dynamics in the feedback to track the tendency of user preferences and item attractiveness. A stochastic state space model or temporal regressions are usually adopted in this approach to track the tendency of user preferences [158, 199]. However, item attractiveness in this approach is assumed to be static in the state space model, leading to a temporal linear model. Meanwhile, to be tractable, distributions in the system have to comply with Gaussian distributions, which may oversimplify the real situations in practice and could lead to low predictive performance.

For example, the first order Gaussian random walk, which is a non-stationary stochastic process with linear and Gaussian assumptions, is utilized to model the dynamics of user latent factors that compactly represent user preferences [80, 158, 55]. The generative procedure for users' ratings on items is also assumed to follow Gaussian distribution. By further assuming the item attractiveness is relatively static over time, the Kalman filtering is applied to track the probabilistic linear model reflecting the tendency of user preferences.

1.2.2 Existing Problems

As briefly mentioned in Section 1.2.1 and will be described in Chapter 2 in detail, diverse approaches and techniques have been investigated and proposed in order to properly integrate the temporal and dynamic information into RSs. With those newly developed techniques, the performance of RSs has been greatly improved under temporal context.

However, there are still some major problems in terms of how the temporal dynamics are being explored and exploited in those methods as follows,

1) The tendency of user preferences is usually assumed to be linear and Gaussian.

2) Item characteristics or attractiveness is usually assumed to be static.

3) The generative process for user feedback is usually assumed to be Gaussian distributed.

4) The model structure to capture the tendency of user preferences and item attractiveness is either predefined or learned from a linear system.

5) A universal dynamic model is commonly adopted across all users and items in the system, rather than using personalized and item-wise dynamic systems.

6) The temporal dynamics of variation of user preferences and item attractiveness are largely neglected. Almost all of existing work on RSs focus on the modeling of temporal dynamics of average behaviors of user preference and item attractiveness.

Problems $1 \sim 5$ described above may oversimplify the real-world scenarios in two-folds. Firstly, problems $1 \sim 3$ restrict the range of targets under investigation. Secondly, when it comes to modeling the temporal dynamics, problems 4 and 5 hamper the flexibility and diversity of the developed model. The 6-th or last point described above illustrates the weakness of the current methods from an algorithmic point of view.

1.3 Aims

The work in this research attempts to overcome some of existing problems in RSs that work under temporal context and improves their performance by finely modeling the temporal and dynamic information from user feedback. The developed methods and algorithms in this research aim to provide original solutions to improve the performance of RSs by incorporating and exploiting this significant but traditionally neglected information. Specifically, the research aims are shown as follows,

1) Based on the dynamic-based approach mentioned in Section 1.2.1, a novel probabilistic temporal bilinear model for RSs has been developed to tackle the nonlinear and non-Gaussian dynamic problems, which is inherent in the tendency of user preferences and item attractiveness in the complex and diverse real-world recommendation scenarios.

2) A novel probabilistic personalized and item-wise temporal model has been developed to solve the *cold start transition* problem and learn the personalized and item-wise dynamics by collaborative tendencies without any prior assumptions about the structure of the dynamics of users and items. The *cold start transition* problem is a result that users often leave feedback on an item only once and for only one period, preventing from learning any dynamics directly.

3) A Bayesian Wishart MF method has been developed to model the temporal dynamics of variances of model parameters, which models the sudden changes and other local temporal effects among user preferences and item attractiveness.

1.4 Methodology

Based on the survey of the study of temporal dynamics in RSs, some existing problems of RSs exploiting temporal and dynamic information have been identified in this section. In order to tackle with those problems, methods are proposed and briefly discussed in this section.

1.4.1 Overview of the Research Work

The whole research work will be divided into three related components. Specifically, the first part of this research focuses on the finely modeling of the tendency of user preferences and item attractiveness simultaneously, which also releases the Gaussian assumption on the generative procedure of user feedback. The second part attempts to learn the personalized

and item-wise model structures of dynamic systems from user feedback in order to flexibly capture the tendency of user preferences and item attractiveness. Meanwhile, a new problem in the learning of personalized and item-wise dynamic systems, namely, the *cold start transition* problem, has been identified and solved. The last part of this research models the temporal dynamics of variations of model parameters in RSs due to the sudden changes and other local temporal effects among user preferences and item attractiveness. This part of work also aims to investigate a new perspective to model the tendency of user preferences and item attractiveness.

1.4.2 Three Components

In order to tackle the problems listed in Section 1.2.2, the strategies adopted in this research are briefly discussed in the following, which also includes the contributions made by this research.

Modeling Temporal Dynamics Flexibly In order to overcome the problems listed in Section 1.2.2, this research makes the following contributions.

- A particle filtering for MF method is developed to track both the tendency of user preferences and item popularity efficiently, which could be of both non-linear and non-Gaussian. The designed observation function is non-Gaussian and proper for the Top-N recommendation [66] under temporal context.
- A novel probabilistic temporal bilinear model is developed to enforce the temporal interaction between user preferences and item attractiveness and dynamically adjust significance on different dimensions of latent factors.
- A novel self-training mechanism is developed to cooperate with particle filtering to solve the problems of data sparsity and scalability in CF.
- An effective and efficient learning algorithm is developed for the probabilistic temporal bilinear model by combining particle filtering and EM framework [35].

- The proposed models and learning algorithms are experimentally shown to achieve significant improvement in the performance of top-k recommendation.

Learning Personalized and Item-wise Temporal Dynamics In order to solve the *cold start transition* problem in dynamic RSs, this research makes the following contributions.

- A dynamic model is developed for the tendency of user preferences and item popularity in a personalized and item-wise fashion, where no assumption on dynamical model structure is imposed,
- A collaborative inference and learning algorithm is developed. This algorithm explicitly considers the uncertainties of model structure and dynamically updates model parameters to track user preferences and item attractiveness accurately,
- An algorithm exploiting the temporal dynamics of "like-minded" users and similar items is developed to learn the cold start transition models.
- The proposed models and learning algorithms are experimentally shown to achieve significant improvement in the performance of Top-N recommendation.

Modeling and Learning Temporal Dynamics of Variations In order to solve the last problem listed in Section 1.2.2, this research makes the contributions as follows,

- A novel Bayesian MF method for RSs is developed to model the tendency of user preferences and item attractiveness effectively via the direct controlling of temporal dynamics of covariances of user and item latent vectors.
- An effective and efficient inference algorithm for the Bayesian model is developed, which combines the collapsed Gibbs sampling method and elliptical slice sampling method [35].

- It is experimentally shown that the proposed model and learning algorithm lead to the improvement in the temporal performance of RSs for rating prediction on public benchmark datasets.

1.5 Publications

- Luo, C., Cai, X., and Chowdhury, N. Self-training temporal dynamic collaborative Filtering. In Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining (2014), pp. 461-472. [160]
- Luo, C., Cai, X., and Chowdhury, N. Probabilistic temporal bilinear model for temporal dynamic recommender systems. In Proceedings of 2015 International Joint Conference on Neural Networks (2015), pp. 1-8. [161]
- Luo, C., Cai, X., and Chowdhury, N. Towards solving the cold start transition problem in dynamic recommender systems. In 2015 IEEE 12th International Conference on e-Business Engineering (2015), pp. 95-100. [162]
- Luo, C., and Cai, X. Bayesian wishart matrix factorization. Submitted to the journal track of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD) (2016). Under revision. [159]

1.6 Organization

The rest of this thesis is organized as follows,

1) Chapter 2 will conduct an intensive literature review on existing methods in RSs involving the modeling of temporal and dynamic information from user feedback.
2) Chapter 3 will discuss the research in detail on how probabilistic bilinear models are constructed to track the tendency of both user preferences and item attractiveness

simultaneously. Their learning and inference algorithms will be also discussed in this chapter before showing the experimental results on three real-world benchmark datasets.

3) In Chapter 4, the necessities of constructing the personalized and item-wise dynamic system for RSs will be discussed at first. Then, the learning and inference method to construct such a dynamic system will be derived in detail. Finally, experimental results will be demonstrated and discussed.

4) Chapter 5 discusses research on modeling the temporal dynamics of variations of user preferences and item popularity. The learning and inference algorithms are also discussed in detail for the developed model. Then, experimental results will be illustrated and discussed.

5) Chapter 6 concludes this research and discusses the possible directions of future work.

Chapter 2

Background

Methods that exploit temporal and dynamic information in recommender systems (RSs) can be roughly divided into four categories. Generally speaking, each category takes a different perspective when exploiting such temporal and dynamic information. In this chapter, detailed discussions relating to these methods in each category will be presented. Meanwhile, besides discussing the techniques involving the modeling of temporal and dynamic information in RSs, it is necessary to discuss some methods developed for static or conventional RSs. Those static methods are so fundamental that on the basis of which almost all the RSs exploiting temporal and dynamic information are developed.

2.1 Fundamental methods in Conventional Recommender Systems

Although there are numeric methods that tackle various problems in conventional RSs, there are three types of methods closely related to the modeling of temporal and dynamic information in RSs. They are, memory-based collaborative filtering (CF), probabilistic matrix factorization (PMF) and Bayesian personalized ranking (BPR). This section discusses these fundamental methods in conventional RSs.

2.1.1 Memory-based Collaborative Filtering

As mentioned in Section 1.1.2 in Chapter 1, the rationale behind CF is that if users agreed in the past, then the recommended items by similar users should also be favored by the user to be recommended items. The user-based CF method [215, 76, 48, 115, 68] is thus developed to capture this concept for explicit user feedback.

The similarity between user u and user v can be computed based on their rating history. Let $r_{u,i}$ denote the numeric rating that user u gives on item i. By treating the past ratings of user u as a vector that is filled with zeros when its entry is unknown and otherwise $r_{u,i}$, many similarity measurements have been proposed, such as distance measures [6, 119], cosine similarity or $L2$ norm [111, 220, 215] and Pearson correlation coefficient [18, 198]. Among those measurements, the Pearson correlation is commonly used due to its simplicity and efficiency, which is defined as follows,

$$w(u,v) = \frac{\sum_{i \in \mathcal{I}_{u,v}} (r_{u,i} - \bar{r}_u)^2 (r_{v,i} - \bar{r}_v)^2}{\sqrt{\sum_{j \in \mathcal{I}_u} (r_{u,j} - \bar{r}_u)^2} \sqrt{\sum_{j \in \mathcal{I}_v} (r_{v,j} - \bar{r}_v)^2}}, \tag{2.1}$$

where $\bar{r}_u$ is the average of ratings given by user u on items $\mathcal{I}_u$, $\mathcal{I}_u$ the set of items that have been rated by user u and $\mathcal{I}_{u,v}$ the set of items that have been rated by both user u and user v.

After obtaining the similarity among users, the preference of user u on its unseen item j can be estimated by assembling its similar users' preferences on item j. This computation can be expressed as follows,

$$\hat{r}_{uj} = \bar{r}_u + \frac{\sum_{v \in \mathcal{U}_j} w(u,v)(r_{v,j} - \bar{r}_v)}{\sum_{v \in \mathcal{U}_j} w(u,v)}, \tag{2.2}$$

where $\mathcal{U}_j$ denotes the set of users that have rated item j. The denominator of the above equation is used to normalize the computation. The contributions from similar users are adjusted by removing its average rating. This practice is somehow used to consider the fact that users may adopt different numeric rating values to express the identical preference over an item. Eq (2.2) does not filter out those users that are less similar to user u, which may introduce unnecessary noise and degrade the performance of prediction. Therefore,

it is common to assemble the preferences from the nearest neighbors of the user. After identifying the K nearest neighbors $\mathcal{N}(u)$ of user u, the prediction can be improved by replacing $\mathcal{U}_j$ with $\mathcal{U}_j \cap \mathcal{N}(u)$ in Eq (2.2).

The above CF method is constructed based on user-user similarity and it needs to store the rating history of all users in memory. Symmetrically, it is also possible to take the perspective of items. Then, the method computes item-item similarity and assembles user preference on the unseen item from similar items. This approach is advocated by Amazon and named as item-based CF [148, 201, 214]. Roughly speaking, those two approaches do not result in a significant performance difference. However, when the number of items is much less than the number of users in the system, the item-based approach has a smaller computational complexity than user-based one.

The memory-based CF is easy to understand and simple to implement. The algorithm also enjoys a low computational complexity and has easily interpretable results. However, it is not easy for this approach to select a proper set of neighbors to compose the user's preferences accurately. Meanwhile, the empirical adjustment in Eq (2.2) can only partially mitigate the problem of diverse rating criteria among users.

2.1.2 Probabilistic Matrix Factorization

Probabilistic matrix factorization [197], as a model-based approach to CF, has been widely used due to its simplicity and efficiency. This method relies on the usage of latent variables or factors to represent the complicated reasons behind user preferences over item characteristics compactly.

In particular, assuming that there are N users and M items, let $R \in \mathcal{R}^{N \times M}$ be a user-item preference matrix with an entry $r_{u,i}$ representing the rating given by user u to item i. The model also assumes that rating $r_{u,i}$ is generated by a Gaussian distribution $P(r_{u,i}|U_u, V_i)$ conditioned on K-dimensional vectors U_u and V_i from user and item latent matrices $U \in \mathcal{R}^{N \times K}$ and $V \in \mathcal{R}^{M \times K}$, where K is the number of latent factors adopted. In addition,

prior distributions $P(U)$ and $P(V)$ are formulated to contain regularization terms [196]. These latent variables are further assumed to be marginally independent while any rating $r_{u,i}$ is assumed to be conditionally independent given latent vectors U_u and V_i for user u and item i [196]. Figure 2.1 shows the graphical model of PMF. The likelihood distribution over preference matrix R is defined as follows,

$$P(R|U,V,\alpha) = \prod_{u=1}^{N}\prod_{i=1}^{M} I_{u,i} \cdot \mathcal{N}(r_{u,i}|U_u V_i^T, \alpha^{-1}), \tag{2.3}$$

where $\mathcal{N}(x|\mu,\alpha^{-1})$ is a Gaussian distribution with mean μ and precision α, and $I_{u,i}$ is an indicator variable with value 1 when the rating $r_{u,i}$ is not missing and value 0 when the rating is not observed. Priors $P(U)$ and $P(V)$ are given as follows,

$$P(U|\alpha_U) = \prod_{u=1}^{N} \mathcal{N}(U_u|0,\alpha_U^{-1}I), \qquad \text{and} \qquad P(V|\alpha_V) = \prod_{i=1}^{M} \mathcal{N}(V_i|0,\alpha_V^{-1}I). \tag{2.4}$$

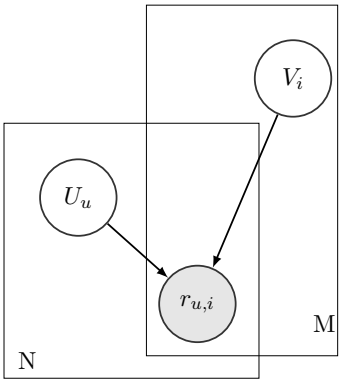


Figure 2.1: The graphical model of probabilistic matrix factorization method.

Maximizing the log-posteriors over U and V is equivalent to minimizing the sum-of-square error function with quadratic regularization terms for MF [196], leading to the following objective function,

$$E = \frac{1}{2}\sum_{u=1}^{N}\sum_{i=1}^{M} I_{u,i}(r_{u,i} - U_u V_i^T)^2 + \frac{\lambda_U}{2}\sum_{u=1}^{N}||U_u||_{Frob}^2 + \frac{\lambda_V}{2}\sum_{i=1}^{M}||V_i||_{Frob}^2, \tag{2.5}$$

where $\lambda_U = \alpha_U/\alpha$, $\lambda_V = \alpha_V/\alpha$, and $||\cdot||_{\text{Frob}}$ denotes the Frobenius norm. The stochastic gradient descent (SGD) method [40, 41, 85] is a popular approach to learning those latent factors for the above objective function. Meanwhile, the alternating least squares method [221] has also been developed to conduct this learning task.

There are no constraints on the domains of latent factors, which makes the results of PMF difficult to interpret. For example, by considering the signs of user and item latent factors, the specific user preference on a hidden dimension cannot be read off from the value of user latent factors on that dimension. This limitation can be overcome by restricting the domain of latent factors to be non-negative. It results in non-negative MF [59, 137, 103], whose factors are easily interpretable. Although it is possible to conduct dynamic non-negative MF [175, 80], the non-negative constraint is either easily violated [80] or computationally expensive to maintain [80, 175], considering a large number of users and items in RSs. Hence, this constraint is usually neglected [80].

2.1.3 Bayesian Personalized Ranking from Implicit Feedback

Traditional methods in RSs usually focus on the explicit feedback from users. BPR aims to conduct personalized Top-N recommendation by exploiting the rich implicit feedback, such as view times and monitored clicks, which are automatically tracked by the system [188, 95, 179]. The traditional approaches usually either ignore the missing entries in the feedback or treat all the missing entries as the negative values. This measure cannot reflect the real world scenario because the non-observed user-item pairs are the results of the mixing of the real negative feedback and the missing values. Moreover, the traditional treatment of non-observed data makes the learning methods suffer from the data imbalance problem [188, 99]. That is, the input data to the learning methods either contain only one class of instances or have much more negative instances than the positive ones due to the problem of data sparsity [115] in RSs.

To overcome these issues, BPR constructs tuples of items from the implicit feedback, from which the personalized preference over items can be inferred. Let $\mathcal{I}$ denote the universe of items and $\mathcal{I}_u^+$ denote the set of items that are seen by user u. By assuming that user u is more interested in items in $\mathcal{I}_u^+$ than unobserved items $\mathcal{I} \setminus \mathcal{I}_u^+$, the implicit feedback can be used to construct the training data D_S as $D_S = \{(u, i, j) | i \in \mathcal{I}_u^+ \text{and } j \in \mathcal{I} \setminus \mathcal{I}_u^+, \text{for all } u \in \mathcal{U}\}$, where $\mathcal{U}$ denotes the set of all users.

Based on the constructed data, the personalized recommendation problem can be converted to the personalized pairwise ranking problem. Let $\hat{x}_{uij}$ denote an arbitrary real-valued function with model parameter Θ, and it should be designed to capture the preference relation among user u, item i and item j. By keeping this function abstract, a generic optimization criterion, named as BPR-OPT, is proposed as follows [188, 82],

$$BPR-OPT = \ln P(\Theta|D_S) = \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) + \ln P(\Theta) = \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta ||\Theta||^2, \tag{2.6}$$

where λ_Θ works as a regularization coefficient as defined in the PMF method, and the function ln denotes the natural logarithm function. The function $\sigma(\cdot)$ is the sigmoid function [171]. This optimization criterion is derived from the maximum posterior estimator for optimal personalized pairwise ranking [188]. It is also shown that the optimization of this criterion equals to the maximization of the area under the AUC curve [188].

For the personalized Top-N recommendation, two models, the MF model and the K-nearest neighbor model [33], are proposed to implement $\hat{x}_{uij}$. Only the MF model will be discussed here because it is more close to the research work conducted in this thesis.

Let vector U_u represent the K-dimensional latent vector of user u and vector V_i represent the K-dimensional latent vector of item i. The function $\hat{x}_{uij}$ can be defined as $\hat{x}_{uij} = \sum_{k=1}^{K} U_{uk}V_{ik} = U_u^T V_i$. Then, the model parameter is $\Theta = \{U, V\}$ by stacking over all the user and item latent factors into matrices.

The method learns parameter Θ using the SGD method [188]. In order to speed up the convergence of the SGD method, the bootstrap sampling approach [98, 36] is also adopted, which selects a subset of training data D_S. This sampling approach can also alleviate the data imbalance problem by avoiding cycling through the full data D_S [188].

2.1.4 Summary

In general, user feedback can be classified as either explicit or implicit feedback. Compared with explicit feedback, implicit feedback is usually regarded as being easier to obtain and

less sparse. BPR is thus designed to handle implicit feedback to predict user preferences over items. In contrast, explicit feedback is usually thought of conveying more information relating to users' opinions on items, which makes memory-based CF and PMF very popular for RSs. Meanwhile, because BPR is a pairwise approach to rank user preferences over items, some methods based on the list-wise ranking [205] are also proposed in RSs. However, it may not be straightforward to apply them to implicit feedback. Moreover, considering the computational complexity of those methods, temporal extensions of those methods may be inapplicable to real-world scenarios in RSs.

Therefore, most of the existing work in exploiting temporal and dynamic information in RSs is based on the approaches and ideas adopted by memory-based CF and PMF, which are discussed in detail in the following sections.

2.2 Heuristic Approach in Recommender Systems with Temporal and Dynamic Information

As briefly discussed in Chapter 1, the heuristic approach is a relatively simple approach in RSs that exploit temporal and dynamic information from user feedback. Usually, this approach can be regarded as a pre-processing stage that computes the temporal influences of input data to make the static methods in RSs time-aware. Methods in this approach usually discount the importance of user feedback. Those methods are named as importance reweighting based methods in this thesis. In addition, some methods of this approach calculate the temporal influences of user feedback and then feed them as the input data into algorithms at later stages in RSs. For those methods, they are referred as methods with temporal importance as inputs.

2.2.1 Importance Reweighting Based Methods

For classic methods in RSs, such as item-based CF, all the user feedback is given equal weights, even if the feedback may be generated at different time instances. Intuitively speaking, the recent feedback, which is given by users on items, should have a larger impact on the predictions of user's current interests than the feedback that is collected a long time ago. In this regard, the performance of CF should generate more precise predictions if the temporal importance of user feedback is explicitly taken into account.

Time decaying function in collaborative filtering The time function is used to assign different weights to ratings to reflect the fact that the recent ratings are more important than the older ratings. Item-based CF is thus extended by assigning the ratings with different weights based on the personalized time functions [73]. Let $r_{u,i}$ denote the true rating of item i given by user u from the training data and $\hat{r}_{u,i}$ denote the preference prediction of user u on its unseen item i. This predicted preference in item-based CF can be extended as follows,

$$\hat{r}_{u,i} = \frac{\sum_{j \in \mathcal{N}(i)} r_{u,j} \cdot sim(i,j) \cdot f(t_{u,j})}{\sum_{j \in \mathcal{N}(i)} sim(i,j) \cdot f(t_{u,j})}, \tag{2.7}$$

where $sim(i,j)$ measures the similarity between item i and item j, and $\mathcal{N}(i)$ represents the set of k nearest neighbors of item i which can be found out based on the given rating history. The function $f(t)$ is the time function that reduces monotonically with time t, and $t_{u,j}$ represents the timestamp of rating $r_{u,j}$. The function $f(t)$ is usually defined as the exponential function as $f(t) = exp(-\lambda \cdot t)$, where exp is the exponential function and λ is the decay rate interpreted as the reciprocal of the half-life parameter T_0 with the following relation, $f(T_0) = \frac{1}{2} f(0)$. There are some alternative functions to the exponential function used in the time decaying approach. Nevertheless, the decaying effects are emphasized on the most recent data to reflect the user's latest interest. Hence, the exponential function, whose gradient near zero value is steeper than its gradient far away from zero, properly serves this purpose.

It is not feasible to compute T_0 for every user and item. Therefore, the method assumes

that the same user has the identical decay rate of preferences over a group of similar items. The half-life parameter T_0 is then computed for each user and each cluster of items. The clusters could be obtained by the K-means clustering method [116, 22] based on the rating history of items. For the user u, the half-life parameter over one cluster of items can be learned from the training data by minimizing the following mean absolute error (MAE) [53], $\arg\min MAE(T_0) = \arg\min \frac{\sum_{i=1}^{M_c} |\hat{r}_{u,i} - r_{u,i}|}{M_c}$, where M_c is the number of items in the cluster c. As the above optimization problem is difficult to solve due to its non-concave property, an approximation technique is used to obtain the optimal solution by doing grid searching [30, 219] over some selected discrete values of T_0.

Instead of numerical ratings, explicit user feedback in RSs can also appear in other forms, such as, tags attached by users to some resources. Tags and times are two main factors to reflect the process of tagging behaviors of users in social tagging systems [135, 253, 51]. A more frequently used tag for a user usually means that the tagged resource is more interesting to the user [168, 109]. Meanwhile, more recently tagged resources should, to some extents, reflect the user's current interests. Those tags should be placed more emphasis on the predictions of users' future preferences. Therefore, personalized recommendations are expected to be improved by exploiting the tag and time information in the social tagging systems. The user-based CF method could be extended by integrating the tag and time information in user feedback [253, 51]. The time weight strategy adopts the time decaying function to reflect the idea that more recent tags should receive more emphasis.

When incorporating the temporal information from user feedback into RSs, the temporal factor that captures the time sensitivity between the purchase time and the recommendations could also be exploited to improve the performance of RSs [229]. The temporal information exploited in this way is ubiquitous in the post-purchase recommendation, which is common in e-commerce websites [230, 229]. It recommends users the relevant items after the user made a purchase. Then, the time weight mechanism is adopted to give the weight to the category relevance and the product-level behavioral relevance. Instead of adopting a decaying function, the unit step function about time could be used. Hence, for the category relevance, the number of purchases is only counted for those purchases

that are made within a predefined time distance from the active purchase. Similarly, for the product-level relevance, the step function is multiplied by it in order to only count the historical purchases within the time window.

The approach discussed above utilizes the time decaying function that tends to undervalue the importance of the historical data. Meanwhile, the linear combination may not only introduce extra burdens to the model complexity control but also be too simple to capture the relationship between the tag and its temporal information.

Kernel-based Time Decay Function Because users may have a sudden change of interests, the continuous smoothing scheme over the timeline may result in suboptimal predictions. Hence, user behaviors are explicitly modeled and divided into a sequence of sessions to overcome this issue, where the session-like behaviors capture the user's short-term interests [245].

The temporal effects of user interests in the tagging systems are also investigated [131, 248, 246, 130]. Assume that the tagging behaviors of different users are independent. Then, the tags attached to post p for user u at time t are defined to follow a multinomial distribution as [246], $P_t(p|u) \propto \prod_{\tau \in \mathrm{T}_p} \theta_{\tau,u}^{c(\tau|p,u)}$, where T_p represents the set of tags attached to post p, $c(\tau|p,u)$ is the number of times that tag τ is attached to post p and $\theta_{\tau,u}$ denotes the probability that user u prefers tag τ. The standard kernel smoothing techniques [36, 98] are used to construct the likelihood function based on $P_t(p|u)$ to model the temporal effects of user interests. Given user u's historical data D_u, its likelihood with respect to all the users and tags is given as $l(\eta|D_u) = \sum_{t' \in \mathcal{T}} K(t-t') \sum_{p' \in P_{t,u}} \sum_{\tau \in \mathrm{T}_{p'}} c(\tau, p'|u) \log \eta_\tau$, where $\mathcal{T}$ is the set of time steps for all records, $P_{t,u}$ is the set of posts tagged by user u at the t-th time step and $c(\tau, p'|u)$ is the number of times that tag τ attached to post p' by user u. The function $K(\cdot)$ is the kernel function and η_τ denotes the set of $\theta_{\tau,u}$ over all the tags for user u. Assuming the lifetime of the short-term interests follows an exponential distribution [246], $K(\cdot)$ is defined as the probability of the emerged topics remaining alive, which is 1 minus its cumulative function.

The maximization of the likelihood function of D_u defined above is achieved by using the method of Lagrangian multipliers [32] with the constraint that all of $\theta_{\tau,u}$ in the multinomial distribution should sum to 1. Then, the probability that user u applies tag τ at time t can be estimated, which is actually the fraction of occurrences weighted by the kernel function. There are also some situations that topic switches occur. Therefore, for each post p, the kernel smoothing will only be effective from the post p_i from which the latest session begins. The similarity between two consecutive posts p_{i-1} and p_i is computed by the Jaccard's coefficient [173] from the tags attached to those posts. Let J_i denote the computed similarity and $t_i = t_{p_i} - t_{p_{i-1}}$ be the time interval between those two posts. The topic switch can then be identified by using the nearest neighbor method by taking the inputs as the set of pairs (J_i, t_i) over all the posts.

Kernel-based smoothing techniques are used to track the temporal effects of user interests, where the exponential function is used as the kernel function. In this regard, this approach reweights the temporal importance of user feedback. It also focuses on the modeling of the short-term interests, which may be insufficient to capture many other real-world scenarios in which the importance of historical data should not be neglected. Meanwhile, the kernel parameters, which are numerous and play vital roles in modeling the temporal dynamics, are not learned from the data but only empirically constructed.

Summary As discussed above, the methods based on importance reweighting to exploit temporal and dynamic information in RSs are intuitive to understand and easy to implement. For almost all of memory-based methods in RSs, it is straightforward to extend them to incorporate the temporal and dynamic information by using this approach, and the computational complexity introduced by this kind of extension is usually negligible. However, this approach tends to oversimplify the complicated scenarios in the tendency of user preferences and item attractiveness. Too much emphasis has been placed on the recent user feedback, and the importance of the past feedback is thus underestimated.

2.2.2 Methods with Temporal Importance as Inputs

In addition to utilizing the time decaying mechanism to penalize temporal influences of user feedback, there are also methods in the heuristic approach to process temporal information as input data or features for later stages.

Combine Temporal and Location Information The temporal information and geometric information can also be combined to improve the performance of personalized point-of-interest (POI) recommendation [247]. In particular, the check-in times of user activities and the locations of POIs are integrated into user-based CF to fulfill this task. The temporal information is usually assumed to have periodic properties and it is thus bucketized into time frames with a fixed length. Hence, this approach also shares some similarities with methods in a binning-based approach.

The preference $c_t^{u,l}$ of user u over a POI l at time t is a linear combination of its temporal behavior and geometric behavior as [247], $c_t^{u,l} = \alpha c_{t,(te)}^{u,l} + (1-\alpha) c_{t,(sp)}^{u,l}$, where the parameter α controls the trade-off between the preference for check-in data $c_{t,(te)}^{u,l}$ and the preference from location data $c_{t,(sp)}^{u,l}$.

Assume that there are L POIs and T time intervals in the system. The temporal preference $c_{t,(te)}^{u,l}$ of user u over a POI l at time t is defined as, $c_{t,(te)}^{u,l} = \frac{\sum_v w_{u,v}^{(te)} \sum_{t'} \tilde{c}_{t'}^{v,l} \cdot \rho_{t,t'}}{\sum_v w_{u,v}^{(te)}}$, where $\rho_{t,t'}$ captures the similarity between time intervals t and t' across all users and all locations. The temporal similarity $w_{u,v}^{(te)}$ between user u and user v is defined as,

$$w_{u,v}^{(te)} = \frac{\sum_{t=1}^{T} \sum_{l=1}^{L} \tilde{c}_t^{u,l} \tilde{c}_t^{v,l}}{\sqrt{\sum_{t=1}^{T} \sum_{l=1}^{L} (\tilde{c}_t^{u,l})^2} \sqrt{\sum_{t=1}^{T} \sum_{l=1}^{L} (\tilde{c}_t^{v,l})^2}}, \tag{2.8}$$

where $\tilde{c}_t^{u,l}$ is the smoothed $c_t^{u,l}$ based on $\rho_{t,t'}$. The temporal similarity is not defined across all the time intervals in order to alleviate the problem of data sparsity [247].

The geometric similarity is based on the idea that the user's willingness to a POI is reducing as the distance increases. This is captured by using a power law [64] to model and using the least-square regression [36] to learn its parameters from the historical data.

The geometric or spatial preference is defined based on the conditional probability and its derivation is based on the assumption that any historically visited POI is conditionally independent given a target POI l.

Combine Temporal Information and Social Activeness The probabilistic LSA model [101] could also be used to model the community (topic) distribution jointly over users and items [244]. In order to consider the social influence and the tendency of user interests, the social activeness and the temporal feature of users are also incorporated into this model. Then, the MF-based CF model is constructed for each community based on the computed results from the previous model to make the Top-N recommendations. This is a two-phase strategy to generate the personalized Top-N recommendations.

Let $R \in \mathcal{R}^{N \times M}$ denote a user-item rating matrix for N users and M items. By treating user as word and item as document, the likelihood function of probabilistic LSA model is defined as $L(U) = \log P(U|\theta) = \sum_{u=1}^{N} \sum_{i=1}^{M} r_{u,i} \cdot \log[\sum_{k=1}^{K} P(u|z_k)P(z_k|i)]$, where $P(z_k|i)$ is the probability that item i on community z_k, $P(u|z_k)$ is the probability that community z_k contains user u and $z_k \in \{1, \ldots, K\}$. EM [35] is used to learn $P(z_k|i)$ and $P(u|z_k)$. All the items are then clustered into K communities based on $P(z_k|i)$.

Instead of using $P(u|z_k)$ to cluster users, a measurement of POI could be used to incorporate the temporal information and the activeness of users. A time decay function is used to exploit the tendency of user preference as $W_t = \frac{1}{1+\beta|t_{u,i}-t_u^{last}|}$, where $t_{u,i}$ is the timestamp of rating $r_{u,i}$ and t_u^{last} is the timestamp of the latest rating given by user u. Let $W_a(i)$ denote the social activeness of user u. It is defined as $W_a(u) = \frac{1}{1+\log(\text{Rank}(u))}$, where Rank$(u)$ is the rank of user u derived from the PageRank algorithm [181] whose transition matrix is defined as, $PageRank(u) = \frac{1-d}{M} + d\sum_{v \in \mathcal{N}(u)} \frac{PageRank(v)}{L(v)}$. The $\mathcal{N}(u)$ is the set of users that user u trusts and $L(v)$ is the cardinality of the set of users who trust user v. The overall effect W is then defined as a linear combination of W_a and W_t as $W = \alpha W_t + (1-\alpha)W_a$, where parameter α controls the trade-off between those two components. The POI-measurement for user u_i on community z_k is then defined as, $POI(u, z_k) = \sum_{i=1}^{M} W_{ui} r_{u,i} P(i|z_k)$.

In the perspective of MF, the above approach can be regarded as a pre-processing stage, which clusters users and items on the combinations of ratings, the social activeness of users and the tendency of user interests. The problem of data sparsity is partially mitigated by learning latent factors per community. However, due to the usage of probabilistic LSA, users and items are jointly clustered. Hence, this approach is incapable of recommending items to users if they do not belong to the same community.

Combining Temporal Information and Taxonomy User feedback could be grouped into T discrete time periods or epochs to model the current tendency of user preferences. The predictions of transactions for user a are then conducted based on memory-based CF with data from the latest epoch. Similar transactions of all but the user from any epochs are identified to alleviate the problem of data sparsity under the temporal context. Because both the items transacted by users and the topics estimated from the user's transactions can change dramatically from one epoch to another, the taxonomy of items [107, 250], which is manually labeled and relatively stable, is used to identify similarities among epochs [178].

The temporal interests of user u are modeled by two vectors. Vector $\mathbf{i}_t^u$ denotes the item transactions of user u at epoch t, where its j-th entry $i_t^{u,j}$ represents the frequency of item j's transactions. Vector $\mathbf{c}_t^u$ denotes the class transactions of user u at epoch t, where its k-th entry $c_t^{u,k}$ represents the frequency of transactions under class k in the taxonomy. The frequency accumulates counts of items belonging to all the descendants of class k. Let C denote the set of classes available in the taxonomy, C_j denote its j-th class, and $C_{j,t}(u)$ the subset of classes in class C_j which are transacted by user u at epoch t. For user a and any other user u, the class and item similarities between any epoch $t \in 1, \ldots, T$ and the latest epoch T for class transactions are calculated. For example, for user a and any other user u, their similarity between any epoch $t \in 1, \ldots, T$ and the latest epoch T for class transactions is defined by $S_c(a, u, t)$ as follows,

$$S_C(a, u, t) = \sum_{C_j \in C} \frac{\sum_{C_k \in \{C_{j,T}(a) \cap C_{j,t}(u)\}} \min(c_T^{a,k}, c_t^{u,k})}{|\{C_{j,T}(a) \cup C_{j,t}(u)\}|}. \tag{2.9}$$

The min operation in the above equation is used to filter out the transaction noise. The item similarity can be similarly derived. Based on the above similarities, the prediction of frequency transacted by user a on item j in user-based CF can be expressed by considering the set of users whose epochs of transactions are most similar to the current transactions of user a and the overall similarity between user a and user u at epoch t.

However, this method does not consider the temporal interaction and collaboration between user preferences and item attractiveness when exploiting the temporal information. In addition, users tend to purchase multiple items in one transaction, while the relations among those items are also usually ignored in this approach.

Summary As discussed above, methods in this subsection pre-process the user feedback by utilizing the temporal information and using them as the transformed feedback for later stages in the methods. In order to fully exploit various information available in RSs, temporal information is usually combined with other types of information, such as locations from users' check-in data and taxonomy from items' meta data. As a by-product of this approach, the problem of data sparsity in RSs could, to some extents, be alleviated. Although it is usually infeasible or expensive to obtain that heterogeneous information, the performance of RSs is expected to be improved when the data fusion is deliberately achieved. However, similar to the approach using temporal information to reweight the importance of user feedback, the temporal information exploited in methods discussed in this subsection usually ignore the significance of the past data and oversimplify the tendency of user preferences and item attractiveness in the real-world scenarios.

2.2.3 Discussion

The experimental results from the methods discussed above show that the performance of RSs can be improved by simply introducing the time weights for preference prediction. However, these approaches tend to ignore the importance of the historical data, because the monotonic decreasing function oversimplified the complicated tendency of user prefer-

ences and item attractiveness. Meanwhile, the half-life parameters are only learned from the training data and not dynamically updated when new feedback is available. This simplification may make the proposed methods incapable of catching up with the tendency and the sudden changes of user preferences and item attractiveness during the period of prediction. Furthermore, when temporal influences are combined with other factors, such as tag importance and content relevance, the proposed models involve empirical components and introduce a huge burden to model complexity control.

The performance of RSs can also be improved by utilizing the temporal information to construct the user feature vector or input data that will be used by later stages in RSs. However, the usage of temporal information is usually heuristic, which usually undervalues the temporal influence of past user feedback. Meanwhile, the fusion between temporal preference and other information, such as spatial preference, usually resorts to a simple linear combination, which may not capture the trade-off between various components across time intervals flexibly and accurately.

Furthermore, the dynamic information and the stage to update the model parameters are not explicitly considered. Methods in this direction usually focus only on the short-term user preference. Hence, the models may not cope with the problem of data sparsity when users do not provide enough feedback. Moreover, those methods usually do not consider the modeling of item popularity over time.

2.3 Binning-based Approach in Recommender Systems with Temporal and Dynamic Information

In the binning-based approach, the effects of temporal dynamics in user preferences and item attractiveness are usually explicitly modeled while training and testing data could be from the same interval. The prediction for users' interests is actually *post hoc* about what interests *would have been* in the past, rather than what interests would be in the future. In the context of Bayesian recursive estimation, the binning-based approach can

be regarded as resembling the smoothing based estimation.

2.3.1 Matrix Factorization with Temporal Information

By deliberately designing the interpretations of latent factors in MF methods, various and diverse rationals behind user preferences and item attractiveness in RSs can be represented compactly by those methods. In this regard, many methods in the binning-based approach exploit the temporal information in RSs by using latent factors in MF to capture the tendency of user preferences or the temporal interactions between users and items from side information.

Modeling Temporal Influences The first method under investigation in this approach is the SVD++ method [128, 29, 123]. In this method, latent factors are added for each time interval. A rating $r_{u,i}$ rated by user u to item i can be statically decomposed as follows,

$$r_{u,i} = \mu + b_i + b_u + V_i^T(U_u + |\mathcal{I}(u)|^{-\frac{1}{2}} \sum_{j \in \mathcal{I}(u)} Y_j), \tag{2.10}$$

where $\mathcal{I}(u)$ represents the set of items that have been rated by user u and μ is the average of ratings. The latent factors b_u and b_i are used to capture the user biases and item biases for user u and item i, respectively. The latent factors V_i are used to capture the item characteristics of item i, and the latent factors U_u are included to capture the user preference for user u. The average influence from similar items is also considered by incorporating the latent effects (factors) Y_j from all the rated items of user u, which is expressed by the term $|\mathcal{I}(u)|^{-\frac{1}{2}} \sum_{j \in \mathcal{I}(u)} Y_j$.

For modeling the temporal dynamics in Eq (2.10), the basic idea is to align all the latent factors along the time index. By dividing the ratings into time intervals according to their timestamps, the time index for each time interval is denoted as t. The temporal item bias term $b_i(t)$ is expressed as $b_i(t) = b_i + b_{i,Bin(t)}$, where $b_{i,Bin(t)}$ is the core component to capture the temporal behavior. Due to the problem of data sparsity, the time intervals

are further grouped into coarser bins that have longer span, which is identified as $Bin(t)$. The concrete number of bins depends on the application.

A linear function is used to capture the gradual drift of user bias. The temporal user bias is defined as $b_u(t) = b_u + \alpha_u \cdot dev_u(t) + b_{u,t}$, where parameter α_u controls the importance of the time deviation, and the time deviation of ratings is defined as $dev_u(t) = sign(t - t_u) \cdot |t - t_u|^\beta$, where t_u denotes the mean date of ratings given by user u. The function $sign(t - t_u)$ gives the sign of the time distance between t and t_u. The parameter β is set by the cross-validation method [36, 83].

Furthermore, user preferences are also assumed to change constantly over time. The K-dimensional latent factors $U_u = (U_{u,1}, \ldots, U_{u,K})$ can also be extended to model the temporal dynamics as follows,

$$U_{u,k}(t) = U_{u,k} + \alpha_{u,k} \cdot dev_u(t) + U_{u,k,t} \ \forall k \in 1, \ldots, K. \tag{2.11}$$

The latent factor $U_{u,k}$ captures the stationary effects and the term $\alpha_{u,k} \cdot dev_u(t)$ is used to capture the possible linear dynamics over time. The local and short-term variability is modeled by the latent factor $U_{u,k,t}$.

Finally, ratings in Eq (2.10) can be extended to model the temporal dynamics at time t as $r_{ui}(t) = \mu + b_i(t) + b_u(t) + V_i^T(U_u(t) + |\mathcal{I}(u)|^{-\frac{1}{2}} \sum_{j \in \mathcal{I}(u)} Y_j)$. The learning of latent factors is conducted by minimizing the associated squared error function with regularization terms on the training data. An SGD method is utilized to conduct the minimization.

Although the temporal information relating to user preferences is exploited in this approach, item latent factors are assumed to be static, which neglects the temporal interactions between user preferences and item attractiveness. In addition, the dynamics of user preferences across consecutive time frames are also omitted, which ignores the evolution processes of user preferences and leads to the learned latent factors focusing more on the local effects of user preferences.

Combine Temporal and Location Information Given a POI i, its user preference $f_u(i)$ by user u, spatial preference $f_l(i)$ at location l and temporal preference $f_t(i)$ at time t can also be combined to improve the performance of RSs [144]. All of these three components return a score that measures the preference of POI i. PMF is adopted to learn the components of user preference and temporal preference. The spatial preference is built from the distance information and review scores from some reference websites. The outputted scores from those three components for each location i are used as its feature vector for a learning-to-rank method [140, 45] to get the final preference score for POI i.

Let $c(u, i)$ denote the check-in frequency of user u over POI i, which is computed from the training check-in data. Let $U_u \in R^{K_{UP}}$ represent the latent factors of user u and $V_i \in R^{K_{UP}}$ represent the latent factors of POI i. The user preference $f_u(i)$ is defined as $f_u(i) = U_u^T V_i$. The factorization method for temporal preference aims to model the weekly periodic patterns of check-in behaviors. Therefore, the check-in data are bucketized according to different days of the week. Let $H_t \in R^{K_{TP}}$ represent the latent factors of t-th hour of the week and $W_i \in R^{K_{TP}}$ represent the latent factors of POI i. The temporal preference $f_t(i)$ is defined as $f_t(i) = H_t^T W_i$. Because the dimensionality K_{UP} is usually different from K_{TP}, the above equations use different latent factors W_i to represent the attractiveness of POI. In addition, the frequencies of all users' check-in data conditioned on the hour of the week and the hour of the day are used as the temporal features of the later learning-to-rank problem [140, 44].

For the spatial preference component, it is computed by not only considering the distance $dist(i, l)$ between the query location l and the location of POI i but also using the popularity scores obtained from some mainstream review websites.

BPR [188] is used to formulate the optimization problems to learn latent factors of user and temporal preferences. SGD learns those latent factors from the historical check-ins. Then, for each POI, its scores of both user and temporal preference are computed. By combining with the spatial preference, those computed scores are used as the feature vector for the POI. With the preference label in the training data, the method uses the ListNet method [52] to obtain a learning-to-ranking model to generate the final preference score.

The approach discussed above only exploits the temporal information to construct the user feature vector that will be used by the ListNet method. The MF methods are thus used as a pre-processing stage. The temporal model is also used to model the periodic patterns in users' check-in history data, which can be regarded as a binning-based approach because the predictions are made with retrospective information. Meanwhile, the dynamics of user preferences and item attractiveness are also ignored.

Modeling User Evolution The personal development of users could also be explored and exploited to model the user evolution in RSs. Traditional RSs usually assume that the user community evolves over time and has different preferences at different time stages [128, 239, 145]. However, two users may have similar preferences for a product if they have the similar experience levels, even if their responses to the products occur temporally far away [170]. In other words, the personal clock is considered when modeling the temporal effects of user evolution. Meanwhile, the user gains its experience as time goes, so the experience is assumed to be monotonically non-decreasing. The different evolution stages are therefore divided by users' experiences.

Assume that there are E stages existing in the user evolution. Due to various background associated with users, there is no requirement that a user needs to achieve all the experience levels. For each rating $r_{u,i}$ given by user u to item i, there is a latent variable $e_{u,i} \in \{1, \ldots, E\}$ that identifies the experience level of user u at the time $t_{u,i}$ when the rating $r_{u,i}$ is given. A simple monotonicity constraint on time and experience is imposed as $\forall u, i, j, \quad t_{u,i} \geq t_{u,j} => e_{u,i} \geq e_{u,j}$.

The latent factor model in CF is adopted to model rating $r_{u,i}$, which is defined as $rec(u, i) = \alpha + \beta_u + \beta_i + U_u^T V_i$, where the latent factor α is a global offset, the latent factor β_u and β_i represent the bias for user u and item i, respectively. The latent factors U_u and V_i represent the user preferences and item properties for user u and item i, respectively. The above equation is modified to incorporate the personalized experience level as $rec(u, i) = \alpha(e_{ui}) + \beta_u(e_{ui}) + \beta_i(e_{ui}) + U_u^T(e_{ui}) \cdot V_i(e_{ui})$.

Let $\Theta = \{\alpha(e), \beta_u(e), \beta_i(e), \gamma_u(e), \gamma_i(e)\}$ and $\Xi = \{e_{ui}\}$. Those latent factors and experience variables can be learned by minimizing the mean squared error of the predictions on a set of training rates R as follows,

$$(\hat{\Theta}, \hat{\Xi}) = \arg\min_{\Theta,\Xi} \sum_{r_{u,i} \in R} \frac{1}{|R|}(r_{u,i} - rec(u,i))^2 + \lambda\Omega(\Theta), \tag{2.12}$$

$$\text{s.t. } t_{u,i} \geq t_{u,j} => e_{u,i} \geq e_{u,j}, \tag{2.13}$$

where $\Omega(\Theta)$ is a regularization term to prevent overfitting and λ is its regularization coefficient. Because similar experience levels should have similar parameters, a smooth constraint is imposed on adjoined experiences as $\Omega(\Theta) = \sum_{e=1}^{E} ||\Theta_e - \Theta_{e+1}||_2^2$, where Θ_e is all the parameters defined with experience level e.

The above optimization problem is non-convex. By alternatively optimizing against Θ and Ξ, the local optimal solution can be obtained. Given the categorical variable Ξ, Θ can be obtained at each iteration by using the L-BFGS method [150], which is a quasi-Newton method for non-linear optimization problems with many variables. Then, by fixing the learned Θ, the discrete variable Ξ can be obtained by solving a longest common subsequence problem [100] in dynamic programming [65].

The approach discussed above can also be regarded as a binning-based approach with a special treatment for time intervals with the personalized and variable length. Therefore, it is suitable for prediction in situations that exhibit periodic properties. Although it imposes the personalized timeline that corresponds to different personal experience levels, the approach only considers the user evolution in terms of experience gain and ignores the modeling of dynamics of user preferences and item attractiveness. It does not explicitly consider the temporal information relating to item attractiveness either.

Incorporate Various Temporal Effects The four main types of time effects in MF-based CF are also studied in detail [238]. The first three temporal effects are related to the bias term in the model: the time effects of time bias, user bias and item bias. The last effect is related to the shifting of user preference over time. In addition to those four main

types of time effects, there are other types of simple time effects, such as item-month effect and the effects of user activity. After learning the latent factors using MF, the predicted rating values are post-processed by taking into account those simple time effects.

Let $r_{u,i}$ denote the rating given by user u to item i. A regularized MF model with bias is defined as $r_{u,i} = \mu + b_u + b_i + U_u^T V_i$, where μ is the average rating of all ratings. The latent factors b_u and b_i represent user bias for user u and item bias for item i, respectively. The latent factors U_u and V_i are used to capture user preference for user u and item characteristics for item i, respectively. The time bias model is then derived by extending the above equation as $r_{u,i} = \mu + b_u + b_i + b_t + U_u^T V_i$, where the latent factor b_t with $t = t_{u,i}$ represents the time bias at the time that rating $r_{u,i}$ is presented. The user bias model enhances the previous equation as $r_{u,i} = \mu + b_u + b_i + b_{t_{u,i}} + U_u^T V_i + X_u^T Z_\tau$, where the time interval $\tau = \tau_{u,i} = t_{u,i} - t$. Latent factors X_u capture the user bias for user u, and latent factors Z_τ capture the temporal influences at time interval τ. The item bias model enhances the previous equation as $r_{u,i} = \mu + b_u + b_i + b_{t_{u,i}} + U_u^T V_i + X_u^T Z_\tau + S_i^T Y_\omega$, where $\omega = \omega_{u,i} = t_{u,i} - t$, and $S_i^T Y_\omega$ represents the fluctuation of item i's popularity within time interval ω. By incorporating the shifting of user preferences into the above equation, the time-dependent MF model is finally defined as follows,

$$\hat{r}_{u,i} = \mu + b_u + b_i + b_{t_{u,i}} + U_u^T V_i + X_u^T Z_\tau + S_i^T Y_\omega + \sum_k G_{u,k} L_{i,k} H_{\tau,k}, \tag{2.14}$$

where latent factors G_u, L_i and H_τ are introduced for user u, item i and time interval τ. By minimizing the mean squared error function with respect to the retrospective data, those latent factors can be learned by using the SGD method.

Then, the simple time effects are used to post-process the predicted rating. For example, if the item-month effect is used, the finally predicted rating is defined as $r_{u,i} \leftarrow r_{u,i} + ave_{i,t_m}(u,i)$, where $ave_{i,t_m}(u,i)$ is the average prediction error of item i at month t_m as $ave_{i,t_m}(u,i) = \frac{\sum_{(u,i)\in\mathcal{K}, t_m(u,i)=t}(r_{u,i} - \hat{r}_{u,i})}{n_{i,t_m}}$, where n_{i,t_m} is the number of ratings of item i at month t_m and $\mathcal{K}$ denotes all the user and item pairs in user feedback.

Through carefully categorizing various types of temporal effects on user feedback, the temporal information in RSs could be captured more finely by latent factors in MF. When the

user feedback does demonstrate those kinds of properties being modeled, it is reasonable to expect the performance of RSs can be improved. However, the temporal dynamics of user and item latent factors are still largely ignored, which shifts away from the modeling of the tendency of user preferences and item attractiveness in RSs.

Infusing Heterogeneous User Feedback The temporal information in RSs could also be fused with heterogeneous user feedback and social information from a friendship-based social network [151, 50, 126] to improve the performance of RSs. For this approach, the CF problem in personalized recommendations is usually treated as a ranking problem for each user and the focus is shifted on modeling the relative ordering of items for each user rather than the absolute rating values [153]. The heterogeneous user feedback is combined by transforming both explicit and implicit feedback into a unified pairwise preference-based representation. The user and item latent factors are learned at every time intervals to exploit the temporal information from user feedback. Meanwhile, a temporal smoothness regularization is adopted to ensure those latent factors change slightly over successive time frames. The information from social friendship network is also explored and exploited to alleviate the problem of data sparsity and the problem of cold-start users in CF.

The Bradley-Terry model [106] is exploited to construct the probability of user preference. Let U denote the user latent matrix and V denote the item latent matrix. The user u's feedback on item i can be predicted as $\hat{x}_{ui} = U_u^T V_i$. Let $\hat{\delta}_{u,i,j} = \hat{x}_{u,i} - \hat{x}_{u,j}$. The Bradley-Terry model assigns the probability of pairwise preference as, $P(\delta_{u,i,j} = 1) = \sigma_\gamma(\hat{\delta}_{u,i,j})$, $P(\delta_{u,i,j} = -1) = \sigma_\gamma(-\hat{\delta}_{u,i,j})$, and $P(\delta_{u,i,j} = 0) = 1 - \sigma_\gamma(\hat{\delta}_{u,i,j}) - \sigma_\gamma(\hat{\delta}_{u,i,j})$, where σ_γ is the logistic sigmoid function with additional parameter γ to control the probability of ties as $\sigma_\gamma(x) = \frac{1}{1+\gamma e^{-x}}$. By dividing the feedback into T time intervals, a sequence of user latent factor matrices $\{U_1, \ldots, U_T\}$ is used to learn the user preferences at every time interval. The item latent factors are assumed to be static. Then, for a rating $x_t^{u,i}$ given by user u to item i at time t, it can be predicted as $\hat{x}_t^{u,i} = (U_t^u)^T V_i$.

By using X and $\hat{X}$ to denote the stacking of $x_{u,i}$ and $\hat{x}_{u,i}$ accordingly, user and item latent factors can be learned by optimizing the Bradley-Terry model-based loss function

$L_{bt}(X, \hat{X})$, which is inspired by BPR. The optimization problem incorporating temporal information is defined as $L = \sum_{t=1}^{T} L_{bt}(X_t, \hat{X}_t) + R(\Theta)$, where $R(\Theta)$ is a temporal smoothness constraint to avoid large fluctuation of latent factors across successive time intervals. The temporal constraint is defined as follows,

$$R(\Theta) = C_1 \sum_{t=1}^{T} ||\Theta_t||^2_{Frob} + C_2(\sum_{t=2}^{T} ||\Theta_t - \Theta_{t-1}||^2_{Frob}), \tag{2.15}$$

where parameters C_1 and C_2 are regularization coefficients. Then, the SGD method can be applied to learn those latent factors.

Although the temporal smoothness regularization is imposed on the optimization problem across time intervals, the dynamics of user latent factors are not modeled in the approach discussed above. Meanwhile, item latent factors are assumed to be static over time. It also neglects temporal interactions between user preferences and item attractiveness and their influences on the evolution processes of user and item latent factors.

Modeling the effects of Temporal Order Sometimes, the temporal information in RSs can appear in an opaque manner. The quality of items that have been previously exposed to the user should have some influences on the perceived quality of the current item. In this regard, the temporal order of ratings could be exploited to improve the performance of RSs. Two MF-based CF methods are proposed to incorporate the type of temporal information [117].

Let the user-item interaction matrix $R \in \mathcal{R}^{N \times M}$ denote ratings given by N users over M items. Let K denote the dimensionality of latent factors in the method. The user preferences are represented by a $N \times K$-dimensional latent factor matrix U, and the item preferences at the current time t are represented by a $M \times K$-dimensional latent factor matrix V_t. The currently predicted rating between user u and item i is $F_{u,i} =< U_u, V_t^i >$ where $< \cdot, \cdot >$ is the inner product operator.

For the currently perceived item i for user u, without loss of generality, it is assumed that the user's last rated item is k and the item rated n times by the user before the current

item is item l. By a slight abuse of notation, let V_{t-1}^{k} represent the latent factors of item k and V_{t-N}^{l} represent the latent factors of item l.

The first model is named as a multiplicative model as $F_{u,i} =< U_u, V_t^i, V_{t-1}^k, \ldots, V_{t-N}^l >$, and the second model is named as an additive model, $F_{u,i} =< U_u, V_t^i > + < U_u, V_{t-1}^k > + \cdots + < U_u, V_{t-N}^l >$. The regularized objective function is defined as $R[U, V_t, V_{t-1}, \ldots, V_{t-N}] = \frac{1}{2}(F - R)^2 + \lambda_U ||U||_{Frob}^2 + \lambda_{V_t} ||V_t||_{Frob}^2 + \cdots + \lambda_{V_{t-N}} ||V_{t-N}||_{Frob}^2$, where parameters $\lambda_{\{\cdot\}}$ are regularization coefficients. The SGD method is used to learn those latent factors.

Considering the number of regularization coefficients, it may be not a trivial task for model complexity control in the methods discussed above. Meanwhile, the models actually set the time step per rating, which may make the problem of data sparsity extremely severe. In addition, the dynamics of user and item latent factors are not modeled and the temporal information relating to user preferences is also ignored.

2.3.2 Non-negative Matrix Factorization with Temporal Information

As mentioned in Section 2.1.2, although non-negative MF [63, 59, 137] has clear interpretations over its latent factors, non-negative constraints usually imposes extra burdens on computational resources and could be easily violated when considering the temporal information in RSs. For the completeness of this survey, methods in this subsection, which may have high computational complexities for RSs, are also included.

Temporal Information as Feedback The temporal information could be used as a significant source of user feedback in non-negative MF to discover the themes associated with item groups. A group is usually considered as a group of a mixture of themes, where a theme consists of a set of items that share some similar patterns of context and content in items [147].

Let $\mathcal{P}$ denote the set of photos in the system. For each photo, its color, texture, shape and SIFT [156] are used as its content feature that is set as a D-dimensional vector. Then, a

photo-feature matrix can be obtained as $W^F \in \mathcal{R}^{|\mathcal{P}|\times D}$, where the i-th row represents the feature vector of photo i [147].

For context information, the following contexts are extracted to construct photo-context matrices: the owners or users of each photo, the tags attached to each photo and the uploaded time of each photo. Let $\mathcal{U}$ denote the set of users that have posted photos in the system. The photo-user matrix $W^U \in \mathcal{R}^{|\mathcal{P}|\times|\mathcal{U}|}$ is defined as $W^U_{i,u} = 1$ if user u owns photo i and $W^U_{i,u} = 0$ otherwise. Let $\mathcal{Q}$ denote the set of tags that have been attached to photos. The photo-tag matrix $W^Q \in R^{|\mathcal{P}|\times|\mathcal{Q}|}$ is defined as $W^Q_{i,j} = 1$ if tag j is attached to photo i and $W^Q_{i,j} = 0$ otherwise. By dividing the timestamps into $|\mathcal{S}|$ disjoint but consecutive time intervals, the photo-temporal matrix $W^T \in \mathcal{R}^{|\mathcal{P}|\times|\mathcal{S}|}$ is defined as $W^T_{i,t} = 1$ if photo i is uploaded within the t-th time interval and $W^T_{i,t} = 0$ otherwise. All of those context matrices are also normalized to ensure that $\sum_i W_{i,\cdot} = 1$.

Let $P \in R_+^{|\mathcal{P}|\times K}$ denote the photo latent matrix. Its entry $P_{i,j}$ captures the probability that photo i belongs to theme j. Let $Z^F \in \mathcal{R}_+^{K\times D}$ denote the theme-feature latent matrix. Its entry $Z^F_{i,j}$ captures the relation between theme i and feature j. Those latent factors can be learned by solving the following optimization problem,

$$\min \text{ J(P, } Z^F) = \min D(W^F||P \cdot Z^F), \tag{2.16}$$

$$\text{st. } P \in R_+^{|P|\times K}, Z^F \in R_+^{K\times D}, \sum_j P_{i,j} = 1 \quad \forall i, \tag{2.17}$$

where $D(A||B)$ is the KL-divergence [98, 36] between matrix A and matrix B, which is defined as $D(A||B) = \sum_{i,j}(A_{i,j} \log \frac{A_{i,j}}{B_{i,j}} - A_{i,j} + B_{i,j})$. Similarly, the context latent factor matrix can be learned by using the objective functions, $J(P, Z^U) = D(W^U||P \cdot Z^U)$, $J(P, Z^Q) = D(W^Q||P \cdot Z^Q)$, and $J(P, Z^T) = D(W^T||P \cdot Z^T)$, where Z^U, Z^Q and Z^T represent the user latent factor matrix, tag latent factor matrix and temporal latent factor matrix, respectively. The joint optimization problem over both the context and content features can be thus formulated [147]. This optimization problem is no longer concave. A local optimal can be found by a recursive updating method inspired by [137].

Although the method discussed above exploiting temporal information to generate recommendations, it does not consider the personalized recommendation, which should be the

focus of the current study of RSs. Meanwhile, it does not consider the dynamics of latent factors, which shares the weaknesses of the methods discussed in previous sections.

Temporal Information as Regularization With new documents keeping arriving in the system, the existing topics are evolving and new topics are emerging. The non-negative MF method, which has a clear interpretation for topic extraction and clustering, has been used to discover and cluster latent topics for documents at every time step. Meanwhile, the existing topics should evolve smoothly over time while the new topics should be encouraged to emerge. In this regard, the temporal regularization is developed to impose the constraint over the temporal behaviors of latent factors representing the relations between documents and topics and the relations between topics and terms [193].

Let $X(t) \in R^{N(t)\times D}$ denote the observed document-term matrix at time t, where $N(t)$ represents the number of documents existing in the system and D is the number of terms. The (d, r)-th entry of $X(t)$ is statistically weighted using its standard term frequency. Let $X(t_1, t_2)$ denote the vertical concatenation of the document-term matrix $X(t)$ from time t_1 to time t_2. At the current time t, the system only considers the data streaming in a time window of size ω. That is, $X(t - \omega + 1, t)$.

Assume that there are $K(t)$ topics at time t, and the topics are not removed even if its popularity diminished. Therefore, $K(t)$ is monotonically non-decreasing. Let W^* denote the document latent factor matrix at time t, which has the same number of rows as the number of documents accumulated in $X(t - \omega + 1, t)$. The last $N(t)$ rows of W^* are the weight matrix $W(t)$ that corresponds to the document-term matrix $X(t)$. Let $H(t)$ denote a topic latent factor matrix, where the number of rows is set to be $K(t)$. At time t, $K^{em} = K(t) - K(t-1)$ is the emerging topics and $K(t-1)$ is the evolving topics. They have corresponding topic latent factor matrices H^{em} and H^{ev}.

The Hodrick-Prescott trend filtering [187] is used to impose temporal constraints on emerging topics. It estimates the unobserved latent variables at each time step from a series of scalar observations in data. The emerging topics w_i, which is the i-th col-

umn in the document latent matrix, are encouraged to grow over time. Let the matrix $S \in \{0,1\}^{T \times N}$ denote the time-document matrix. Its (i,j)-th entry equals to 1 if document j appears in the i-th time interval and 0 otherwise. The objective function is defined as $\min_{W,H} ||X(t-\omega+1,t) - WH||^2_{Frob} + \mu L(Sw_i)$, where μ controls the importance of the hinge loss L that has a weight c_t at each time t, and Sw_i accumulates the temporal contribution of topic i over documents. The optimization problem defined above is solved by cycling through w_i and h_i while keeping other variables fixed, which is inspired by K-SVD [15] for dictionary learning.

The above approach takes a special care of newly observed documents and imposes smooth constraints across time frames. Meanwhile, the temporal information is also exploited to encourage the survival of the emerging topic. However, no dynamics relating to document and term latent factors have been modeled. The temporal constraint over the emerging topic is also under a linear assumption which may not capture real-world scenarios that topics are able to grow in a very fast and abrupt pace. The number of topics at different time steps is dynamic and has a profound influence on the emergence of topics, but those numbers are only empirically and manually determined.

Summary There are few studies on incorporating temporal information in non-negative MF, especially for the binning-based approach. Although non-negative MF achieves huge success in clustering and topic discoveries, the computational burdens introduced by this approach may not trade well with its benefits for clear interpretations of latent factors and the improvement of temporal performance of RSs.

2.3.3 Tensor Decomposition with Temporal Information

The tensor factorization [145, 182] extends MF by introducing latent vectors on the specific time dimension to take care of the temporal information. Unlike other methods that model the temporal influence of user and item latent factors, those methods usually focus on the overall effects of temporal information that are shared across all users and items.

Therefore, this approach usually assumes that latent factors representing user preferences and item attractiveness are static.

This approach adopts tensor factorization [63, 133], which maximizes the ranking-based measure for prediction of the possible temporal-spatial correlations for tourists [231]. Let $U \in \mathcal{R}^{N\times K}$ be the latent matrix representing tourist preferences among N tourists, $V \in \mathcal{R}^{M\times K}$ be the latent matrix representing location attractiveness among M POIs, and $T \in \mathcal{R}^{L\times K}$ be the latent matrix representing the characteristics of temporal context among L time slots. The preference $X_{u,s,t}$ of user u to the s-th POI at the t-th time slot can be described by the inner products of their corresponding latent vectors as $X_{u,s,t} = \sum_{k=1}^{K} U_{u,k}V_{s,k}T_{t,k}$.

In real world scenarios, the tourists' interests of POIs are usually captured by graded relevance. In this regard, the latent vectors described above can be learned by optimizing an objective function that captures list-wise preference of users. The NDCG metric [98, 36], which is a cumulative and multilevel measure of ranking quality, is adopted to form the objective function. Let $P_{u,i,t}$ denote the ranked position of the i-th POI in the recommendation list of user u at the t-th time slot. By treating $X_{u,i,t}$ as the relevance score, the NDCG for user u at the t-th time slot is defined as $NDCG_{u,t} = Z_{u,t}\sum_{i=1}^{n}\frac{2^{X_{u,i,t}}-1}{\log(1+P_{u,i,t})}$, where n is the number of places of attractions in the recommendation list and $Z_{u,t}$ is the normalization factor. Then, the overall NDCG for all users and all time slots is defined as $NDCG = \frac{1}{N\cdot L}\sum_{u=1}^{N}\sum_{t=1}^{L} NDCG_{u,t}$.

The objective function can be thus formulated as follows,

$$L(U,V,T) = \frac{1}{N\cdot L}\sum_{u=1}^{N}\sum_{t=1}^{L} Z_{u,t}\sum_{i=1}^{n}\frac{2^{X_{u,i,t}}-1}{\log(1+(1+\sum_{j=1,j\neq i}^{n} I(X_{u,j,t}\geq X_{u,i,t})))}, \tag{2.18}$$

where $I(X_{u,j,t} \geq X_{u,i,t})$ equals to 1 if $X_{u,j,t} \geq X_{u,i,t}$ and 0 otherwise. Because of the non-smoothness of the above indicator function, the objective function is non-differentiable with respect to latent vectors U, V and T. Therefore, the logistic function is used to approximate the indication function. In order to avoid overfitting during the learning, the Frobenius norms are added in the objective function. The latent parameters can then

be obtained by optimizing the above objective function. The steepest gradient descent method [36] is adopted to achieve this optimization.

Although the temporal information is exploited, it is only used as a kind of context information. The aspects of dynamic information residing in the temporal data are not considered. For example, to model the dynamic information, the tensor factorization approach in [145], which will be discussed later, has imposed a multivariate normal random walk over the consecutive time latent vectors.

2.3.4 Latent factors with Temporal Information

Conventionally, the temporal information is integrated into the procedure of MF to improve the performance of RSs. However, the latent factors alone could be used to represent temporal information in other optimization problems. By adopting latent factors, the personalized click shaping is developed to extend the user segment-based click shaping method [11, 10]. Instead of learning the allocation plan between each user segment and every article to be recommended, a personalized allocation plan will be learned to provide the personalized recommendation of items. The temporal data are bucketized by their timestamp, and only data from the current time interval are used to generate the allocation plan for the next time interval [12].

However, the personalized recommendation cannot be directly generated by solving the multi-objective program due to two main challenges. Firstly, for user segment-based click shaping, unseen users can be classified into the existing user segments that are assumed to be static in the model. However, this is not the case in the personalized situation, because it is difficult to predict who will visit in the next time interval. For unseen users, it is challenging to compute personalized allocation plans. Secondly, it is computationally expensive to solve the linear programming problem, considering the huge number of users and items.

To overcome these problems, the localized multi-objective program that contains a large

number of user dependent primal variables will be converted to its dual problem [163] which only contains a few user independent dual variables [12]. The user specific allocation plan can then be computed on the fly from the dual optimal solution. Meanwhile, some constraints are added to the optimization problem to ensure the convertibility. These constraints make the optimization problem become strong convexity. As there is not any information about which users will visit in next time interval in advance, the solution to the quadratic programming problem [79] defined in the optimization problem can only be approximated. The sampling method is used to select a set of users randomly. The personalized allocation plan is only considered for those sampled users. Then, the dual solution is obtained by solving a smaller quadratic programming problem. The personalized recommendation is generated from personalized allocation plan that is reconstructed from the solution of the dual problem of the personalized multiple objective programming problems with respect to click-through rate and time-spent.

Apart from only recommending one item in the above approach, no collaboration, whether it is temporal or not, among users and items are exploited. In addition, the model is constructed based on the information from current time interval. Therefore the dynamics of user preferences and item popularity are also neglected.

2.3.5 Temporal Information with Cross-domain Matrix Factorization

The cross-domain collaborative framework [139, 138] is used to model the drifting of user interests over time. For any user u, its interests at different time intervals are modeled by its multiple counterparts over those time intervals. The group level rating matrix, which is a compact rating pattern representation and its entry expresses the expected rating from one user prototype (group) over one item prototype (group), is considered as static and shared across the temporal domain. The user preference and item popularity can, therefore, be expressed as the convex combination of these prototypes. By assuming that the user distributions and item distributions over those prototypes are changing over time, the user interests can be modeled by the user distributions over user prototypes at each

time interval. In order to alleviate the problem of data sparsity, it is also assumed that the user and item counterparts of successive time intervals are closely related.

Let $R \in \mathcal{R}^{N \times M}$ denote the rating matrix for N users and M items whose ratings span T time intervals. The matrix Y, which has the same size as R, denotes the time index of each rating in R as $Y_{ij} \in \{1, \ldots, T\}$. Let R_t denote the ratings in the t-th time interval or temporal domain. For each user u_i, $\{u_1^i, \ldots, u_T^i\}$ denotes its T counterparts.

Inspired by rating-matrix generative model in [138], a group-level rating matrix $B \in \mathcal{R}^{K \times L}$ is shared across the temporal domain for K user prototypes and L item prototypes. The (k, l)-th entry in matrix B denotes the expected rating of user prototype k over item prototype l. The discrete user distribution of user counterpart u_t^i at time t is denoted by p_t^i with $\sum_{k=1}^{K} p_t^i[k] = 1$. Meanwhile, the discrete item distribution of item counterpart v_t^j at time t is denoted by q_t^j with $\sum_{q=1}^{L} q_t^j[l] = 1$. Then, a rating given by user i to item j can be expressed as $r_{i,j} = (p_t^i)^T B q_t^j$ [139].

The Bi-LDA method [185], which is a Bayesian latent factor model for MF, can be used to treat the rating-matrix generative model with a fully Bayesian approach. It is straightforward to apply the Bi-LDA model for each user counterparts and item counterparts across the T time intervals. However, due to the problem of data sparsity, the performance of the direct usage of Bi-LDA is unsatisfied [139]. Based on the assumption that counterparts across successive time intervals are similar, the generative process for p_t^i and q_t^j at time t can be modified as follows,

$$p_t^i \sim Dirichlet(\lambda p_{t-1}^i), \qquad \text{and} \qquad q_t^j \sim Dirichlet(\lambda q_{t-1}^j), \tag{2.19}$$

where the parameter λ is used to avoid the concentration of the distributions on few components [37, 233]. Then, the collapsed Gibbs sampling [152, 186] can be used to obtain the posteriors of those latent factors that are necessary to make predictions of $r_{i,j}$. However, the dynamics of user preferences and item attractiveness are ignored, and user and item distributions over their corresponding prototypes are assumed to be changing slightly, which may not catch up with the sudden change of user preferences and item attractiveness in RSs.

2.3.6 Adaptive Multi-hyperplane Machine with Temporal Information

By using the adaptive multi-hyperplane machine method [232], user preferences over categories of web events can be transformed into a label ranking problem [105, 225]. The temporal information relating to user activities is used to generate those feature vectors that contain temporally related information, such as, the intensity and recency computed from the server logs. Meanwhile, the names of categories of web events are regarded as the classes (labels) to be ranked [74].

Let the function $g(i, x)$ denote a scoring function for instance x and class i. It is defined as [74], $g(i, x) = \max_j w_{i,j}^T x$, where $w_{i,j}$ is the j-th class weight (i.e., hyperplane) for the i-th class (label). The overall weight matrix W for L classes is defined as $W = [w_{1,1} \dots w_{1,b_1} | w_{2,1} \dots w_{2,b_2} | \dots | w_{L,1} \dots w_{L,b_L}]$, where $\{b_1, b_2, \dots, b_L\}$ is the set of the number of class weights for each of L classes. The adaptive multi-hypeplane machine method minimizes the following convex problem for the i-th training instance x_t,

$$L^t(W|z) = \frac{\lambda}{2}||W||_{Frob}^2 + l(W; (x_t, y_t); z_t), \tag{2.20}$$

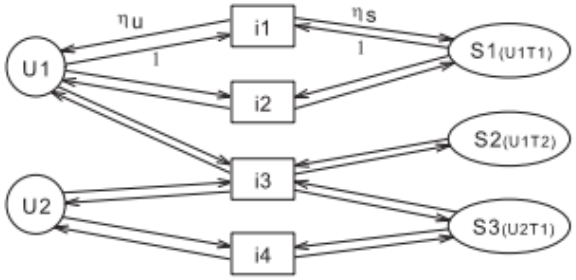
where $l(W; (x_t, y_t); z_t) = \max(0, 1 + \max_{i \in \mathcal{Y} \setminus y_t} g(i, x_t) - w_{y_t, z_t}^T x_t)$, $\mathcal{Y}$ represents the set of all labels, y_t is the true label of instance x_t and z_t is calculated as an index of a true class weight that provides the highest score. The SGD method is used to compute W.

The label ranking problem is different from the learn-to-rank problem because the latter problem ranks over items rather than labels. Meanwhile, a set of hyperplanes is learned for each label. Therefore, it may be computationally too expensive to apply this method to personalized recommendations. The collaboration among users and items are not taken into account either.

2.3.7 Graph model with Temporal Information

Graph model is also investigated and constructed to incorporate the temporal information explicitly in RSs. By taking into accounts all related factors, a fully observed graph model

Figure 2.2: An example of a session-based temporal graph [239].



is built and recommendations are generated by using random walking algorithms inspired by the PageRank method. Meanwhile, the hidden graph model, such as the hidden Markov model, is also exploited to represent the user preferences by its latent states and capture the evolutions of user preferences by their transitional relations.

Fully Observable Graph Model *Long-term and Short-term Interests* After empirically dividing the timestamps associated with the interactions between users and items into bins (sessions), a session-based temporal graph can be constructed based on the transformed input feedback that consists of pairs of $< user, item >$ and $< session, item >$ [239]. The constructed graph is a directed bipartite graph $G(\mathcal{U}, \mathcal{S}, \mathcal{I}, \mathcal{E}, w)$, where $\mathcal{U}$, $\mathcal{S}$ and $\mathcal{I}$ denote the set of user nodes, session nodes and item nodes in the graph, respectively. In general, user node u connects to any item node i in which item i has been viewed by user u. Those connected edges represent users' long-term interests. A session node s connects to those users and items that have interacted within the s-th time session. Those connected edges represent users' short-term interests. Figure 2.2 shows the constructed session-based temporal graph. The edge weight function $w(n, n')$ in w assigns a non-negative weight to any edge $e(n, n')$ in the set of edges $\mathcal{E}$, which is defined as follows,

$$w(n, n') = \begin{cases} 1 & n \in \mathcal{U} \cup \mathcal{S}, n' \in \mathcal{I}, \\ \eta_u & n \in \mathcal{I}, n' \in \mathcal{U}, \\ \eta_s & n \in \mathcal{I}, n' \in \mathcal{S}. \end{cases}$$

The different weights η_u and η_s are used to emphasize the different influences from long-term and short-term effects.

Meanwhile, the transition matrix for the constructed graph that captures the propagation relations among those nodes can be defined as follows,

$$\phi(n,n') = \begin{cases} \frac{1}{|out(n)|^\rho}, & n \in \mathcal{U} \cup \mathcal{S}, n' \in \mathcal{I}, \\ (\frac{\eta}{\eta|out(n)\cap\mathcal{U}|+|out(n')\cap\mathcal{S}|})^\rho, & n \in \mathcal{I}, n' \in \mathcal{U}, \\ (\frac{1}{\eta|out(n)\cap\mathcal{U}|+|out(n')\cap\mathcal{S}|})^\rho, & n \in \mathcal{I}, n' \in \mathcal{U}, \end{cases}$$

where $\eta = \frac{\eta_u}{\eta_s}$. The set $out(n)$ is defined as $\{n' \in \mathcal{V} : e(n,n') \in \mathcal{E}\}$. The parameter ρ is the tuning parameter to control the impact of out-degree in the propagation over the constructed graph. Then, an inference method based on the PageRank method could be used to conduct the inference and recommendation, which is conceptually similar to the random walking algorithms over the graphs developed in [141, 143].

The approach discussed above explicitly considers both long-term and short-term interests of users. However, it still belongs to a binning-based approach, which is the *post hoc* about the users' interests in the past and more suitable for recommending items having periodic properties. Moreover, although temporal interactions between user preferences and item attractiveness are captured via designated session-nodes, the dynamic information across time frames is not explicitly modeled by those session-nodes, whose influences may be reduced during the transition propagation via item nodes in the model.

Similarity-based Method It is possible to exploit temporal information and item taxonomy based on a graph model to generate personalized recommendations [102]. The purchase history of users is uniformly divided into consecutive segments with a predefined time interval T. For any segmented time frame or stage t, user u's profile R_t^u is described as a weighted category vector. The i-th entry represents the i-th category in the item taxonomy. Its value is the number of times that items belonging to the category and its subcategories have been purchased by user u within this time frame.

Based on the computed user profiles, three types of similarities are computed: user-user similarity, item-item similarity and category-category similarity. Then, a multi-modal graph can be constructed with user nodes, item nodes and category nodes. A block-wise adjacent matrix $\mathcal{W}$ of the built graph is defined by the above three types of similarities. The

random walk with restart method [183] can be used to compute a converged distribution in this graph. Except for the usage of the restart node that represents the user under investigation, the learning procedure is similar to the iteration stage used to obtain the stationary distribution from a discrete time discrete value stochastic process induced from a direct acyclic graph with transition matrix $\mathcal{W}$. Because the state in the state space of the stochastic process consists of users, items and categories, the recommended items for user u can be generated from similar items, the most recently purchased item from the similar users and the most popular items in the favorable categories.

The approach discussed above neglects all the available information outside of the current time frame, because it constructs user profiles and item profiles from the data within the t-th time frame to generate recommendations in it. In addition, it considers no temporal dynamics in the model, which may lead to overlooking the influences of the long-term tendency of user preferences and item attractiveness.

Pairwise Preference-based Method The methodology of dynamic pairwise learning can be applied to improve the performance of RSs [34]. The pairwise learning method is developed based on the implicit feedback extracted from users' actions on portal services. In order to generate recommendations, user segments are constructed by clustering users with their feature vectors. The graph-based model is built for each user segment with the training data that is only caused by users in that segment. The identical recommendations will be given to all the users belonging to the same segment.

In order to extract the pairwise preferences from user actions over time in the method, the logs of user actions are divided into a sequence of sessions as $S_t^u =< u; t; (d_1, d_2, \cdots, d_P); C >$, where S_t^u denotes the grouped actions for user u's at time interval t, $(d_1, d_2, \cdots, d_P)$ denotes the contents that are exposed to user u at time t and the set C represents the exposed contents that are clicked by user u. Then, user u prefers content j over content i at time t if $j \in C$ and $i \notin C$, which is denoted as $d_i < d_j$.

For each user segment C, a directed preference graph at time interval t is denoted as $G_t^C =< V_t, E_t, W_t^C >$, where V_t is the set of content nodes at time t. The directed edge

$e_{i,j} \in E_t$ is connected from node V_t^i to node V_t^j if $d_i < d_j$ at time t. The weight $W_t^{C<i,j>}$ associated with the edge $e_{i,j}$, which measures the strength of the preference, is defined as $W_t^{C<i,j>} = \rho W_{t-1}^{C<i,j>} + \delta^{C_{[t,t-1]}}$, where $\delta^{C_{[t,t-1]}}$ denotes the number of times that content j is preferred over content i by any user in user segment C between time t and $t-1$. The parameter ρ controls the time decay rate of user preference over time. After building the dynamic preference graph, an algorithm based on the PageRank method [181, 19] can be used to compute the preferences S_t^C of user segments over V items at every time interval t by initializing it as $S_t^{C<i>} = \frac{1}{|V|}$ for $i = \{1, \ldots, V\}$.

The approach discussed above does not exploit the collaboration among users. In addition, it imposes linear and Gaussian assumptions on the dynamics of hidden scores, which may oversimplify the real-world scenarios in RSs. Meanwhile, the preferences of user segments over items are directly modeled by hidden scores rather than the interactions between user preferences and item attractiveness. This modeling approach may be not coping with the problem of data sparsity in RSs either.

Hidden Markov Model A hidden Markov model [35, 211] is developed for generating personalized recommendations to model the sequence of user visits to different blog articles in each month. The model assumes that its state describes the classes of all the users. Class specific observation models are also global. By initializing each user with a different state distribution over latent classes, personalized recommendations at every time step are achieved by computing the latent class distribution for each user using its historical data [195].

Because the users browser various number of blog articles in each month, the observations in every month are generated by using two distributions. Let N_t^u denote the number of articles consumed by user u in month t. This count variable N_t^u is modeled as a set of class specific negative binomial distributions [5], each of which has a pair of parameters $\{(a_k, b_k)|1, \ldots, K\}$ with K hidden classes.

Let $\mathcal{I}_t^u$ denote the set of articles visited by user u at time step t. The item selection

process is modeled by a set of class specific multinomial distributions across all items. The multinomial distribution in the k-th class has parameter θ_k, and each article in the N_t^u observations is selected from those multinomial distributions.

The parameters and latent class states are learned from the training data by using EM. The MAP extension of EM [35] is used to avoid overfitting. The model adopts Dirichlet distributions [36, 98] as the priors for the rows of the transition matrix and the initial state distribution [195]. The parameters of those priors are tuned based on the cross-validation method [124]. After training the proposed model, the probability that user u visits article i at next time step $t+1$ is defined as follows,

$$P(i \in \mathcal{I}_{t+1}^u) = \sum_k P(Z_{t+1} = k) P(i \in \mathcal{I}_{t+1}^u; a_k, b_k, \theta_k), \tag{2.21}$$

where Z_{k+1} denotes the latent class at time $t+1$. Then, the article with the highest probability is recommended to user u.

Although the dynamics of user preference is modeled by the transition of its hidden class, there is no updating procedure considered in the model when new observations arrive. The model is also constructed based on the hidden classes that a user belongs to. Because this is a global assumption adopted during the design of the model, there is no collaboration conducted. Meanwhile, the order that articles are observed is not modeled within the time frame. The temporal order is only considered from one time step (month) to another. Meanwhile, even if the individual user demonstrates the different paths of state transitions, all of those behaviors are controlled by the identical model. Finally, the tendency of item attractiveness is not taken into account.

Summary As discussed above, the graph model can be utilized to not only construct interpretable recommendations to suit personal interests but also incorporate temporal information in user feedback explicitly and seamlessly to RSs. However, it is challenging to model the temporal influences and dynamics across time frames that are caused by the tendency of user preferences and item attractiveness. Due to the random walking algorithms adopted in recommendations, graph-based RSs are expected to achieve satis-

fiable diversity for recommended results, but the accuracy of recommendations may be compromised.

2.3.8 Discussion

In the binning-based approach, existing methods rely on future information to make the predictions on unrated items in the past. Therefore, this direction can make proper predictions for those temporal scenarios that exhibit periodic properties. Most of the methods in the binning-based approach are based on model-based CF. In particular, latent factors are utilized to represent user preferences, item characteristics, and temporal influences. Meanwhile, the developed models are usually based on PMF, and the SGD method is a common practice to learn those model parameters.

The developed methods in this direction place more emphasis on the local effects of temporal dynamics and tend to capture them through the introduction of some extra latent factors. Although the temporal information in user feedback is widely exploited in this approach, most of the methods in the approach do not consider the dynamics of user preferences and item attractiveness. In other words, the dynamic systems of latent factors across time intervals are largely neglected, and the item characteristics or attractiveness are usually assumed to be static over time. This practice may be inadequate to cope with the complexity of real-world scenarios. For methods considering the dynamics such as [139, 193, 170], those dynamics are working as the temporal constraints or regularization under the binning-based framework. Those temporal dynamics cannot work independently if the underlying models are not extended to new time intervals by introducing extra latent factors and retraining.

In real-world deployments, RSs are continuously collecting user feedback, it is necessary to handle those newly observed data in the developed methods. However, there are no such updating stages for methods in the binning-based approach. Instead, the models are usually retrained with all the ratings up to date to make predictions over next time interval. Furthermore, this approach usually introduces too many latent components to

capture various causes behind user preferences and item attractiveness. Considering the number of regularization coefficients, it may add too much burden on model complexity control.

2.4 Online Updating Approach for Recommender Systems with Temporal and Dynamic Information

In real-world scenarios, it is not only necessary to consider temporal influences intrinsic in the past user feedback but also essential to update model parameters to reflect the trends introduced by the newly collected user feedback. In this regard, methods of the online updating approach for RSs are developed.

2.4.1 Online Updating with Matrix Factorization

Similar to methods in the binning-based approach, a large portion of methods in the online updating approach is also developed based on the MF-based CF. Some of the representative methods in this direction will be discussed in detail in this subsection.

Online version for Probabilistic Matrix Factorization The online updating methods for the PMF method and the Top-one probability based ranking MF method [205] are developed. The online updating methods [149] convert the batch based learning methods, which are widely used in the learning of user and item latent factors, into a recursive updating procedure, in which user and item latent factors are updating based on the previously learned latent factors and the newly arrived ratings. Meanwhile, two popular learning methods are exploited. They are the SGD method and dual averaging method [242, 240].

Let $r_{u,i}$ denote the rating given to item i by user u. Without loss of generality, it is assumed to be newly arrived. By using some simple algebraic rewriting, the gradient of

the objective function in PMF with respect to latent factors can be recursively expressed as $U_u \leftarrow U_u - \eta((g_{u,i} - r_{u,i})g'_{u,i}V_i + \lambda_U U_u)$, and $V_i \leftarrow V_i - \eta((g_{u,i} - r_{u,i})g'_{u,i}U_u + \lambda_V V_i)$, where U_u and V_i represent the latent factors for user u and item i, respectively. The parameters λ_U and λ_V are regularization coefficients for regularization terms to avoid overfitting. $g_{u,i}$ represents the estimated value of $r_{u,i}$, which is the product of corresponding latent factors and expressed as $U_u^T V_i$. $g'_{u,i}$ denotes the derivative of $g_{u,i}$ with respect to user or item latent factors which are identified according to the context.

Let $\mathcal{I}_u$ denote the set of items that have been rated by user u. Let Y_{U_u} denote the average gradient of user u with respect to latent factors of user u in the dual average method. It can be recursively written as $Y_{U_u} \leftarrow \frac{t_u - 1}{t_u} Y_{U_u} + \frac{1}{t_u}(g_{u,i} - r_{u,i})g'_{u,i}V_i$, and $Y_{V_i} \leftarrow \frac{t_i - 1}{t_i} Y_{V_i} + \frac{1}{t_i}(g_{u,i} - r_{u,i})g'_{u,i}U_u$, where t_u and t_i denote the number of times that user u have rated an item and item i have been rated, respectively. The average gradient Y_{U_u} is also an approximation of $(\sum_{i \in \mathcal{I}_u}(g_{u,i} - r_{u,i})g'_{u,i}V_i)/t_u$, which is the definition of the average gradient of the square loss with respect to user latent factor U_u.

By solving the following equations for the average gradients of latent factors, user and item latent factors are obtained in the average gradient method,

$$U_u = \arg\min_w Y_{U_u}^T \cdot w + \lambda_U ||w||_2^2, \qquad \text{and} \qquad V_i = \arg\min_w Y_{V_i}^T \cdot w + \lambda_V ||w||_2^2, \tag{2.22}$$

where λ_U and λ_V are regularization coefficients.

Similarly, the recursive formulation of the Top-one probability based ranking MF can be derived. The only exception is the updating formula for item i, because the Top-one probability is only defined in terms of users. Let $Y_{t_i}^{V_i}$ denote the gradient of the objective function in the Top-one probability with respect to latent factors of item i when the t_i-th rate is assigned to item i. The updating formula is approximated as follows,

$$Y_{t_{u+1}}^{V_i} \leftarrow (1 - \alpha^{c \cdot t_i}) Y_{t_i}^{V_i} + \{\frac{exp(g_{u,i})}{\sum_{k \in \mathcal{I}_{t_{u+1}}^u} exp(g_{u,k})} - \frac{exp(r_{u,i})}{\sum_{k \in \mathcal{I}_u^{t_{u+1}}} exp(r_{u,k})}\}, \tag{2.23}$$

where $\mathcal{I}_{t_{u+1}}^u$ denotes the set of items that have been rated by user u when there are t_{u+1} rates. The decaying factor gradually approaches 1 as the time goes. The rationale behind

this approximation is based on the observation that the changes in the Top-one probability are getting smaller when more ratings are revealed on the item.

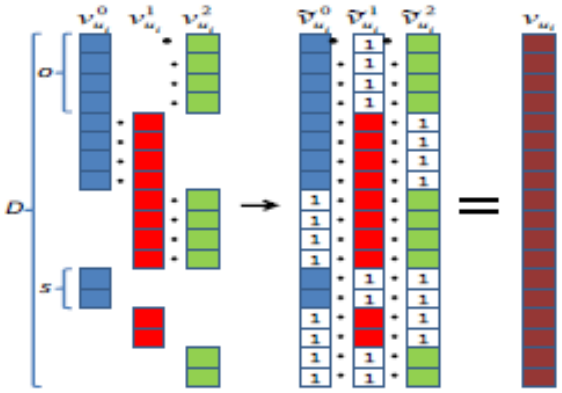
Although the approximation used for item latent factors under the Top-one probability presents some interesting techniques, the recursive formula in the approach discussed above does not differentiate the SGD method in practice. Because the SGD method usually takes one sample at each iteration and it is initialized by the previously learned values. Meanwhile, even though the approach presents some useful techniques to avoid retraining the whole model when new observations are being accumulated, it actually does not model any temporal and dynamic information on user feedback.

Online version of Matrix Factorization with One-pass updating An online training mechanism is usually required for the scenarios that usually run for a short period [14, 16]. Therefore, each observed sample is used to update the model parameters only once. This strategy makes the SGD method in learning procedure become a one-pass optimization method.

A real-world scenario for one-pass updating, such as a single ad campaign, is demonstrated in [14]. Instead of modeling a user as a linear combination of feature vectors, the interdependency among user features is also explored. By assuming that all the users only appear once, it is not necessary to use a projection matrix to map from user feature space to user latent vector space. In this regard, user latent factors are completely constructed from the context information.

Let K and D denote the dimensionality of latent factors and user feature vectors, respectively. For user u_i, its feature vector v_{u_i} is constructed from D standalone values and $\binom{D}{2}$ overlapped values from feature vectors. Let s and o be the dimensionality of standalone values and overlapped values respectively, and $K = D \cdot s + \binom{D}{2} o$. Let $\tilde{v}^j_{u_i}$ denote user u_i's j-th feature vector whose entries are set to 1 except for those entries reserved for standalone and overlapped components. Then, $v_{u_i} \in R^K$ is constructed as $v_{u_i} = \prod_{j=1}^{D} \tilde{v}^j_{u_i}$. Figure 2.3 shows the brief idea to construct user latent factors from feature vectors.

Figure 2.3: An example of constructing a user latent vector from its feature vector with $D = 3$ [14].



By treating the CF problem as a ranking problem, the probability of a click on the item should be maximized. Let $v_{a_j} \in \mathcal{R}^K$ denote the latent factors of item a_j. Then, the score of the match is defined as $S(u_i, a_j) = v_{u_i}^T v_{a_j}$. Following the approach in [17], the probability $PC(u_i, a_j)$ of a click for (u_i, a_j) is defined as follows,

$$PC(u_i, a_j) = |\mathcal{C}| \cdot \frac{exp^{S(u_i,a_j)}}{\sum_{(u_k,a_l)\in\mathcal{N}} exp^{S(u_k,a_l)}}, \tag{2.24}$$

where $\mathcal{C}$ denotes the set of $(user, item)$ pairs that result in a click, $\mathcal{NC}$ denotes the set of $(user, item)$ pairs that do not result in a click on the item, and $\mathcal{N} = \mathcal{C} \cup \mathcal{NC}$. Item latent factors can then be learned by maximizing the log-likelihood over all the clicks as $\Theta = \arg\max \quad \log \prod_{(u_i,a_j)\in\mathcal{C}} PC(u_i, a_j)$. The SGD method with one pass is used to learn those latent factors.

There is no temporal and dynamic information being exploited in the model. The online aspect of the model only focuses on the strategy to update the model parameters. It does not consider a latent factor matrix that projects user features to latent space. Therefore, the applicability of the proposed method should be not very wide.

Online Version of Bayesian Personalized Preferences The static BPR approach in Section 2.1.3 can be extended to an online version to make the real-time Top-N recommendations. For social network applications like Twitter, a continuous stream of incoming

tweets is arriving at a relatively high rate. By equipping with the online pairwise ranking approach [71], the real-time Top-N recommendations for a set of tweets can be made feasible based on individual preferences that are learned from the user's past interactions with Twitter.

Specifically, a pairwise ranking objective function is developed for PMF as follows [71],

$$\underset{\theta=\{U,V\}}{\arg\min} \quad L(P,U,V) + \frac{\lambda_U}{2}||U||_2^2 + \frac{\lambda_V}{2}||V||_2^2, \tag{2.25}$$

where the matrix U represents the stacking of user latent factors and the matrix V is the stacking of item latent factors. Parameters λ_U and λ_V are regularization coefficients. The set P is the set of tuples constructed from the data stream to reflect the pairwise preferences of users, which follows the common practice in [188]. The SVM loss [36, 98], or the hinge loss is used in Eq (2.25). This loss function is advocated by RankSVM [54] for learning in the pairwise ranking task. The loss function L is then defined as follows,

$$L(P,W,H) = \frac{1}{|P|}\sum_{p\in P} h(y_{u,i,j}\cdot < U_u, V_i - V_j >), \tag{2.26}$$

where $h(z) = \max(0, 1-z)$, and $y_{u,i,j} = sign(x_{u,i} - x_{u,j})$. In the above equations, $h(z)$ is the hinge loss function, $x_{u,i}$ represents the preferences of user u over item i, U_u is the user u's latent factors and V_i represents the latent factors for item i.

User and item latent matrices U and V can be obtained by applying the SGD method to Eq (2.26). Due to the nature of the online streaming, the learning algorithm should also be bounded in both space and time. In this regard, the sampling approach is adopted to reduce both space and time complexity in the inference procedure. Three sampling mechanisms can be exploited to achieve this purpose. They are, single pass, user buffer and reservoir sampling. In particular, the single pass sampling method does not remember previously observed instances and takes a single pair from the input stream. The update of the model is performed at every iteration using the sampled pair. The user buffer method retains the latest n instances per user to update the model. Finally, the reservoir sampling, which demonstrates the best performance in the experiments, retains a fixed size of instances in a reservoir for all the users. This sampling mechanism has also been

proposed as an efficient solution to the problem of online AUC maximization for binary classification [252].

The proposed method extends the static BPR model to an online model. However, the model also neglects the modeling of temporal and dynamic information in the streaming data. Meanwhile, although the experimental results demonstrate that the proposed method outperforms some other baseline methods, such as, the weighted regularized MF method, the comparison is not convincing because the compared methods are not updated whenever the new data are available.

Online updating and Offline training In real-world scenarios, new items are continuously entering the RSs and the characteristics of old items keep changing over time. An online component should be useful to cope with the newly arrived items and an offline component could be used to update the characteristics of the old items. For controlling of model complexity, both of online and offline components are usually based on the identical model [155]. The differences between those two components arise from the usage of the underlying model. In particular, those two models are feed by different datasets and updated at different frequencies. The online component frequently updates the online model so that the newly arrived items can be timely recommended to the users. The offline component is only updated when a batch of the sets of newly arrived items is merged into the offline component. Meanwhile, the online component stores the sets of the newly arrived data while the offline component stores all the historical data.

Meanwhile, in most existing RSs that use latent Dirichlet allocation (LDA) to exploit the context information [8, 228, 169], the number of topics to be modeled is set to be the dimensionality of latent factors in the MF [155]. Meanwhile, latent factors are usually initialized or bounded by the associated topic distributions learned from the corpus. This approach implicitly assumes that the context information is the dominant effect that influences the learning of latent factors in MF [155], which is usually not true in practice.

In order to release this arguably strong coupling between topic distributions and item

latent factors, a topic specific latent matrix $X \in \mathcal{R}^{K \times P}$ is introduced as an indirection layer, where P is the number of topics to be modeled and K the dimensionality of latent factors. Let $\theta_{d,i}$ denote the probability that item i belongs to topic d. The item specific latent factor matrix $V \in \mathcal{R}^{K \times M}$ can be decomposed as $V = XY$, where the matrix $Y \in \mathcal{R}^{P \times M}$ represents the topic distributions over M items with its entry $Y_{d,i} = \theta_{d,i}$. The topic distribution matrix Y is learned from LDA in which the corpus consists of the set of item descriptions. Let $U \in \mathcal{R}^{K \times N}$ denote the user specific latent matrix over N users. The predicted rating given by user u over item i is described as $r_{u,i} = U_u^T X V_i$.

For explicit feedback in RSs, the inputs in the pairwise-based learn-to-rank approach are generated by comparing the relevance of any pair of rated items for each user. After obtained the input data, latent factors are optimized to minimize the pairwise loss that is described by the cross entropy cost function [120] as follows,

$$L = \min_{U,X} \sum_{u=1}^{N} \sum_{(i,j) \in \Omega_u} -\frac{e^{r_{u,i} - r_{u,j}}}{1 + e^{r_{u,i} - r_{u,j}}} \cdot (\hat{r}_{u,i} - \hat{r}_{u,j}) + \log(1 + e^{\hat{r}_{u,i} - \hat{r}_{u,j}}) + \frac{1}{2}(||U||^2 + ||V||^2), \tag{2.27}$$

where $\hat{r}_{u,i}$ denotes the predicted rating given by user u on item i as $r_{u,i}$, and Ω_u denotes the set of pairwise preferences built for user u. The parameter λ is the regularization coefficient for regularization terms to prevent overfitting. The SGD method is used to obtain the local optimal solutions to this optimization problem. However, the underlying model is basically a static one, which attempts to fuse the context information via LDA with MF. Finally, the personalized recommendation list for user u is generated probabilistically from both the online and offline components.

Although the approach discussed above presents a practical framework to cope with the streaming of user feedback in RSs, it actually does not utilize any incremental algorithm to implement the updating of either online or offline component. The updating stage is still conducting by retraining those components with the expanded dataset. Meanwhile, no temporal and dynamic information in user feedback has been exploited to model the tendency of user preferences and item attractiveness.

Summary For MF-based online updating methods in RSs, the key observation is that the learned latent factors should be updating when enough amount of observations have arrived in the systems. For the purely incremental approach, methods aim to utilize the newly available information to update the constructed model. The major focus has been placed on the reduction of computational complexity to comply with the online context. However, the temporal dynamics of user preferences and item attractiveness are largely ignored in this direction. In contrast, the other approach in this direction models the temporal dynamics of user preferences to reflect the current trends of users' interests. The newly arrived observations are usually used as the driven inputs to derive the predictions of the nearest future interests of users. However, some separate stages are usually required to accumulate the arrived data and update the models used in predictions offline. Meanwhile, the temporal dynamics of item attractiveness are usually neglected and the temporal interactions between user preferences and item attractiveness are not modeled.

2.4.2 Online Updating with Memory-based Collaborative Filtering

Section 2.2 shows that the temporal information plays an important role in improving the performance of memory-based collaborated filtering method in RSs. Hence, incremental updating algorithms have also been developed [154] to make the developed models catch up the latest tendency of user preferences.

Similar to the approach used in 2.2.1, a time-based decaying function is used to generate the temporal relevance $f_{\alpha,t}^{u,i}$ for user u on item i at time t. This decay function is defined as shown in Section 2.2.1. By incorporating the temporal relevance defined, the similarity between item i and item j is defined as follows,

$$S_t^{i,j} = \frac{\sum_{u \in \mathcal{U}_t^i \cap \mathcal{U}_t^j} (f_{\alpha,t}^{u,i} \cdot r_{u,i})(f_{\alpha,t}^{u,j} \cdot r_{u,j})}{\sqrt{\sum_{u \in \mathcal{U}_t^i} (f_{\alpha,t}^{u,i} \cdot r_{u,i})^2 \sum_{u \in \mathcal{U}_t^j} (f_{\alpha,t}^{u,j} \cdot r_{u,j})^2}}, \tag{2.28}$$

where $r_{u,i}$ denotes the rating that user u gives to item i and $\mathcal{U}_t^i$ represents the set of neighbors of user u at time t. Thus, the similarity places more emphasis on the recent ratings. Meanwhile, those items identified by this similarity measure are inclined to have

linked users that assign similar ratings at time t. The score prediction $\hat{r}_t^{u,i}$ for user u over item i at time t can be expressed as a weighted average of ratings on the subset of the nearest neighbors of item i as follows,

$$\hat{r}_t^{u,i} = \bar{r}_t^u + \frac{\sum_{j \in \mathcal{N}_t^i \cap \mathcal{I}_t^u} S_t^{i,j} \cdot f_{\beta,t}^{u,j} \cdot r_{u,j}}{\sum_{j \in \mathcal{N}_t^i \cap \mathcal{I}_t^u} S_t^{i,j} \cdot f_{\beta,t}^{u,j}}, \tag{2.29}$$

where $\mathcal{N}_t^i$ is the set of the nearest neighbors of item i at time t and $\mathcal{I}_t^u$ represents all the items rated by user u at time t. The $\bar{r}_t^u$ denotes the average rating of user u at time t. Note that a separate decaying factor β is used in the score prediction to model the temporal behaviors of user preferences more finely.

In order to enable online updating in memory-based CF, the temporal relevance should be recursively computed as $f_{\alpha,t+1}^{u,i} = e^{-\alpha} \cdot f_{\alpha,t}^{u,i}$. Let $P_t^{i,j} = \sum_{u \in \mathcal{U}_t^i \cap \mathcal{U}_t^j} (f_{\alpha,t}^{u,i} \cdot r_{u,i})(f_{\alpha,t}^{u,j} \cdot r_{u,j})$, and $\mathcal{U}_t^u$ denote the set of items that are rated by user u at time t, and $\triangle \mathcal{U}_t^u = \mathcal{U}_t^u \setminus \mathcal{U}_{t-1}^u$ denote the newly rated items by user u at time t. The quantity $P_t^{i,j}$ can be recursively computed from $P_{t-1}^{i,j}$ as follows,

$$\begin{aligned} P_t^{i,j} = & \sum_{u \in \triangle \mathcal{U}_t^i \cap \mathcal{U}_{t-1}^j} (r_{u,i}) \cdot (f_{\alpha,t}^{u,j} \cdot r_{u,j}) + \sum_{v \in \triangle \mathcal{U}_t^j \cap \mathcal{U}_{t-1}^i} (f_{\alpha,t}^{v,i} \cdot r_{v,i}) \cdot (r_{v,j}) + \\ & \sum_{k \in \triangle \mathcal{U}_t^i \cap \triangle \mathcal{U}_t^j} (r_{k,i} \cdot r_{k,j}) + \sum_{l \in \mathcal{U}_{t-1}^l \cap \mathcal{U}_{t-1}^j} (f_{\alpha,t}^{l,i} \cdot r_{l,i})(f_{\alpha,t}^{l,j} \cdot r_{l,j}) \\ = & \triangle P_t^{i,j} + e^{-2\alpha} \cdot P_{t-1}^{i,j}, \end{aligned} \tag{2.30}$$

where the term $\triangle P_t^{i,j}$ captures the incremental quantity incurred by the newly rated items.

Although the approach discussed above focuses on the incremental learning of user preferences from the newly arrived observations, it can be classified as an importance reweighting approach that emphasizes the contributions from the recent ratings. Hence, it also suffers from the problem that the significance of past data may be undervalued.

2.4.3 Online Updating with Tensor Factorization

A tensor factorization method that incorporates three facets, which are user preferences, item characteristics, and temporal influences, has been developed at each time interval

[114]. Instead of learning the latent factors representing the projection between each facet and the corresponding dimension of the core latent component, an incremental learning or online updating method is developed to adapt the latent factors to the current observations on the basis of the newly observed inputs and the latent factors learned from the previous time interval [114]. To alleviate the problem of data sparsity, the regularization terms, which are forms of Laplacian matrices [172], are imposed on the latent factors to capture the similarities of the entities. Those Laplacian matrices are constructed from the context information in the training data.

Take the tweets dataset in [114] as an example. The dataset consists of a list of tuples (s, u, w, t), where s denotes a tweet source, u is the target user mentioned by using "@username", w denotes the word that is included in the body of the tweet and t denotes the time that the tweet is published. All the data are grouped by their timestamps into T time intervals. Let $\mathcal{U}$, $\mathcal{S}$ and $\mathcal{W}$ denote the sets of users, sources, and words, respectively. Then, the data are modeled as a three-order tensor sequence $X_t \in \mathcal{R}^{|\mathcal{S}|\times|\mathcal{U}|\times|\mathcal{W}|}$, where $|\mathcal{S}|$, $|\mathcal{U}|$ and $|\mathcal{W}|$ are the number of sources, users and words, respectively. Meanwhile, to reduce the data sparsity under the temporal context, $\mathcal{X}_t$ contains all the tuples (s, u, w, t') where $t' \leq t$.

The tensor factorization of X_t is then described as $X_t = Y_t \times_{\mathcal{S}} S_t \times_{\mathcal{U}} U_t \times_{\mathcal{W}} W_t$, where $Y_t \in \mathcal{R}^{D_{\mathcal{S}}\times D_{\mathcal{U}}\times D_{\mathcal{W}}}$ is the core tensor sequence at time t with latent dimensions $D_{\mathcal{S}}$, $D_{\mathcal{U}}$ and $D_{\mathcal{W}}$. Matrix $S_t \in \mathcal{R}^{|\mathcal{S}|\times D_{\mathcal{S}}}$, $U_t \in \mathcal{R}^{|\mathcal{U}|\times D_{\mathcal{U}}}$ and $W_t \in \mathcal{R}^{|\mathcal{W}|\times D_{\mathcal{W}}}$ are the latent factor matrix for sources, targets and words, respectively. The users' social relations, such as the number of common friends, can be used to construct the Laplacian matrix $L^{\mathcal{S}}$. Similarly, other context information can be used to construct the flexible regularization terms in terms of the Laplacian matrices $L^{\mathcal{U}}$ and $L^{\mathcal{W}}$. To reduce the computational complexity, an approximation method to learn those latent factors incrementally is developed based on some simple linear algebra operations with the assumption that all high order (greater and equal to 2) perturbation terms in the formula expansion are trivial and safe to ignore. The concrete updating equations after some linear algebra operations are omitted here for clarity. The interesting reader could refer to [114] for more information.

Although an incremental method is developed to conduct tensor factorization over the temporal domain, the regularization terms are still static in terms of only exploiting some static information. Moreover, temporal smoothness constraints are usually not imposed on the regularization. Hence, the method is an online updating method for tensor factorization with static regularization terms.

Online updating with Assumed Density Filtering In RSs, user feedback can demonstrate in various forms. Three types of observation functions are also developed to cope with various forms of user feedback [213]. They are observation function for rating based user feedback, observation function for the binary feedback and observation function for a set of ordinal ratings on a user-specific scale. When the observation models are not conjugate with the transition models, some approximations have been used to make the computation tractable. Expectation propagation [174, 84] is used to fulfill this task. To adapt the model to the latest observations, an updating stage based on assumed density filtering (ADF) is also used.

Let the tuple (x, y, ϕ, r) denote the rating based user feedback. Vectors $x \in R^n$, $y \in R^m$ and $\phi \in R^f$ represent the n-dimensional user feature vector, the m-dimensional item feature vector and the f-dimensional other context feature vector extracted from the meta data, respectively. The scalar $r \in R$ represents the rating. Following the approach in PMF, the rating r is assumed to follow a Gaussian distribution as follows,

$$P(r|s, t, b) = N(r|s^T t + b, \beta^2), \tag{2.31}$$

where β is the standard deviation of the observation noise. Vector $s \in R^K$ is a K-dimensional trait vector defined as $s = Ux$ where $U \in R^{K \times n}$ is the user latent factor matrix. Similarly, vector $t \in R^K$ is a K-dimensional trait vector defined as $t = Vy$ where $V \in R^{K \times m}$ is the item latent factor matrix. The bias b is defined as $b = \phi^T w$ where the latent factors w model the weights over the context information ϕ. Meanwhile, the method places independent Gaussian priors over each factor of those latent factors.

For observation function for a set of ordinal ratings on a user-specific scale, the interpre-

tation of scale may be different from user to user and the mapping from ordinal rating to user internal rank may be nonlinear. Therefore, for each user u, a user-specific threshold $b_u \in R^{L-1}$ is maintained, which has L consecutive intervals $(b_{u(i-1)}, b_{u(i)})$ with varying lengths. As a fully Bayesian approach, it is also assumed that the latent threshold has independent Gaussian prior as $P(b_{u,i}) = N(b_{u,i}|\mu_i, \sigma_i^2)$. The observation function with binary user feedback can be regarded as a special case of the ordinal user feedback.

To handle with various forms of observation functions, an inference algorithm that combines the variational message passing [237] and expectation propagation is developed, which approximates the posteriors of latent variables when the analytical solutions are not available. Meanwhile, ADF [227, 26] is also adopted to conduct the online updating that incrementally takes care of newly observed feedback.

To be capable of modeling the tendency of user preferences and item attractiveness, the dynamics of latent factors can also be modeled. For the online updating conducted by ADF, the dynamics can be incorporated by assuming that the variance of a latent factor follows a first order Gaussian random walk. For the full expectation propagation stage, latent variables in the above observation models are repeated for each time interval. At each time step, the backward messages and marginals are updated from future observations for the posterior of latent factors. It thus conducts a smoothing estimation.

Although expectation propagation and ADF are used to approximate the inference when it is applicable, the dynamics are still under Gaussian and linear assumptions. For the inference conducted by the full expectation propagation, temporal dynamics are only exploited in the way as the binning-based approach does. The separate ADF only works as online updating procedure. Meanwhile, the approach assumes that the dynamics introduced by variances on every latent factor are independent of each other, and the effects of interactions among user and item latent vectors are largely neglected.

2.4.4 Online Updating with Sampling Scheme

A set of informative user feedback could also be dynamically constructed to capture the current tendency of user preferences and item attractiveness. A representative set of the dataset is kept in a reservoir [72, 226]. User and item latent factors in MF-based CF are updated by only using samples from this reservoir and the newly observed data [58].

In particular, the reservoir contains the most informative data. Instead of maintaining an accurate sketch of all the historical data in the reservoir with a constraint of a fixed size [72, 226], the sampling mechanism aims to maintain the current interests of users. Let c denote the size of the reservoir. The t-th data will be added to the reservoir with a probability of $1 - \frac{c}{t}$. This probability reflects the observation that it is not necessary to discard the input data immediately after the initialization. When the size limit of the reservoir is reached, an old data instance in the reservoir will be replaced by the newly observed data. For any instance s_i in the reservoir S_{t-1}, the probability that s_i will be replaced is defined as $P(s_i \notin S_t) = 1 - P(s_i \in S_{t-1})$, where $P(s_i \in S_{t-1}) \propto e^{\frac{1}{t-i}}$. The value $t-i$ measures the time distance between the current time order t and the time order i of the old instance s_i added into the reservoir. This probability reflects the idea that the older the data is, the more possible it will be replaced.

After building the current reservoir, positive instances used to update the latent factors will be sampled from the reservoir. There are also two sampling mechanisms commonly used to obtain negative samples. The first strategy will give high probability over unseen items that have largely predicted ratings. Under the assumption that unobserved items are usually disliked by users, those items with largely predicted ratings should be corrected with a high priority. The second strategy will sample unseen items for a user with high probability if the unseen item has been rated by other users recently. Items with such a property imply that they have some characteristics to distinguish the user that do not rate them with other users that have rated. Hence, more emphasis has been placed on the recent data, which may underrate the importance of past data. Meanwhile, after a long period, the reservoir may be dominated by the recent user feedback.

2.4.5 Discussion

Methods in online updating approach attempt to adapt model parameters to the latest trends of user preference and item attractiveness by incorporating the temporal influences of newly observed user feedback. However, compared with the dynamic approach that will be discussed in next section, methods of this approach usually neglect the modeling of the dynamics of the tendency of user preferences and item preferences.

Because the key component of methods in this direction is to develop some mechanisms to cope with the newly observed user feedback, the concept of incrementally updating the model parameters is widely adopted in those methods. Usually, it is not feasible to decompose the learning and inference procedures into incremental ones. Therefore, some assumptions and approximations are necessary to facilitate the derivation. Meanwhile, it is also possible to rely on sampling mechanisms to keep the underlying models updated.

2.5 Dynamic-based Approach for Recommender Systems with Temporal and Dynamic Information

In this section, methods in the dynamic-based approach for exploiting temporal dynamics in RSs will be discussed in detail. Although methods share a lot of similarities with the fundamental techniques adopted in other three approaches, the characteristic of methods in this approach is that the dynamics of the tendency of user preference and item attractiveness will be explicitly taken into account.

2.5.1 State-space Approach

For methods based on the state-space approach, the Bayesian filtering techniques [26], such as the Kalman filtering [158], are intensively involved in tracking the tendency of user preferences. However, almost all the existing methods in the state-space approach rely on

the linear and Gaussian assumptions for user preference, which oversimplify the real-world scenarios in RSs. Roughly speaking, methods in this approach can be classified into latent factors-based methods, which are developed to track latent vectors, and distribution-based methods, which aims to track the distribution parameters.

Latent Factors based Methods Because latent factors are able to represent user preferences compactly, most of the methods in the state-space approach rely on utilizing latent factors as the states. However, there are few studies conducted in this direction, which are discussed in detail in this section.

A Spatio-Temporal Approach

A spatio-temporal model is developed based on model-based CF [158]. The model adopts a state space approach, where the latent factors in MF are treated as the state [218, 217]. The spatial model attempts to capture the correlations among users and items based on implicit feedback and context information. The temporal component models the temporal and dynamic information in user feedback through the adoption of dynamic systems to model the process evolution.

Let U_u denote the latent factors that represent the preference of user u. A Gaussian Markov random field [192] can be used as a prior to imposing an informative prior over user latent factors as follows,

$$f_U(U) \propto \prod_{u,v} e^{-\frac{\alpha}{2} W_U^{u,v} ||U_u - U_v||^2}, \tag{2.32}$$

where matrix U is the stacking of all the user latent factors and $W_U^{u,v}$ is the similarity between latent factors of user u and user v. The parameter α is predefined, which controls the strength of this prior. According to Hammersely-Clifford theorem [134], the conditional distribution of user u's latent factors on latent factors of all other users can be reduced from the joint distribution in Eq (2.32) as follows,

$$P(U_u | U_{(-u)}) \sim \mathcal{N}(\frac{\sum_{v \in \mathcal{U}(u)} W_U^{u,v} U_v}{\sum_{v \in \mathcal{U}(u)} W_U^{u,v}}, (\alpha \sum_{v \in \mathcal{U}(u)} W_U^{u,v})^{-1} I), \tag{2.33}$$

where $U_{(-u)}$ denotes all the user latent factors except the latent factors of user u, and $\mathcal{U}(u)$ denotes all the neighbors of user u that are identified by the Markov random field. The matrix I is a proper identity matrix. Therefore, the dependencies among user latent factors are modeled by ellipsoidal constraints that are functions of covariance matrices.

By combining the priors over user and item latent factors, the latent parameters U and V can be optimized using the objective function below. This optimization objective function is constructed identically in the way that is utilized in PMF to convert the probability distribution to a deterministic objective function.

$$\sum_{u=1}^{N}\sum_{i=1}^{M}(r_{u,i}{U_u}^T V_i)^2 + \lambda(||U||^2 + ||V||^2) + \alpha(tr(U^T \Delta_U U) + tr(V^T \Delta_V V)), \tag{2.34}$$

where λ is the standard deviation in the priors of latent factors and it works as the regularization coefficient in the objective function. The term Δ_U is the graph Laplacian [69] that is constructed from the similarity matrix W_U as $\Delta_U = D_U - W_U$, where the diagonal matrix D_U is defined as $D_U^{(u,u)} = \sum_v W_U^{u,v}$. The trace term $tr(U^T \Delta_U U)$ can be expressed as $tr(U^T \Delta_U U) = \sum_{u,v} W_U^{u,v} ||U_u - U_v||^2$. This term penalizes the discrepancy between user u and user v. The similarity matrices W_U and W_V are empirically constructed from the context information or ratings in the training data.

Meanwhile, user latent factors are assumed to follow a first order Gaussian random walk. The dynamic system for user u at time t is defined as follows [158, 217],

$$U_t^u = U_{t-1}^u + w_t^u, \tag{2.35}$$

and the observation function for user u at time t is given as follows,

$$r_t^u = H_t^u U_t^u + v_t^u. \tag{2.36}$$

Under this assumption, the Kalman filtering can be used to update latent factors U and V. For simplicity, $w_t^u = \sigma I$ and $v_t^u = \beta I$ with $\sigma, \beta \in \mathcal{R}^+$ for all users over all time steps.

After combining both spatial prior and the temporal prior, the complete likelihood of user latent factors $\{U_\tau\}_{\tau=1}^t$ and ratings $\{r_\tau\}_{\tau=1}^t$ is defined as follows,

$$P(\{U_\tau\}_{\tau=1}^t, \{r_\tau\}_{\tau=1}^t | \{W_{U,\tau}\}_{\tau=1}^t, \theta) = \sum_{\tau=1}^{t} P(U_\tau | U_{\tau-1}, \theta) P(r_\tau | U_\tau, \theta) P(U_\tau | W_{U,\tau}, \theta), \tag{2.37}$$

where the observation function $P(r_\tau|U_\tau, \theta)$ is defined in Eq (2.36), the transition distribution $P(U_\tau|U_{\tau-1}, \theta)$ is defined in Eq (2.35) and the spatial prior $P(U_\tau|W_{U,\tau}, \theta)$ is defined in Eq (2.32).

Due to the introduction of the similar matrix $W_{U,t}$, the latent factors U_t^u cannot be updated independently of other latent vectors in the Kalman filtering. Hence, the Kalman filtering [77] has to update the state that is the concatenation of all the user latent vectors simultaneously. It is also necessary to reduce the computational complexity of updating the covariance matrix $\bar{\Sigma}_t$, which is a $Nk \times Nk$ matrix. Then, the mean field approximation [36] is adopted to approximate the posterior $P(U_t|\{r_\tau\}_{\tau=1}^t, \{W_{U,\tau}\}_{\tau=1}^t, \theta)$. The developed spatio-temporal model for CF does not model the temporal behavior of item latent factors. This incapability is due to the linear and Gaussian assumptions imposed on the model. If the item latent factors were also modeled into the state space, the observation model, which is no longer linear, will violate the assumption. The Kalman filtering is thus not able to be applied to track the tendency of user latent factors.

Kalman filtering with Non-negative Matrix Factorization in User Adoption

The behaviors of users adopting items across different time steps are modeled and the personalized recommendations are made based on the evolving user preferences [80]. The problem of user adopting items is different from the rating prediction problem in RSs. Unlike the setting of rating prediction in which users only rate each item once, the users in the context of temporal adoption can adopt the identical item more than once and each adoption may have different quantity numbers.

Let $Y \in R^{N \times M \times T}$ denote the adoption matrix, where N is the number of users, M the number of items and T the number of time intervals that contains the timestamps when the adoptions occur. Like the user-item interaction matrix in RSs, the adoption matrix is highly sparse. Due to the nature of adoption matrix, it also makes sense to collapse it along the time dimension and obtain a matrix $Y^* \in R^{N \times M}$ where $Y^*_{u,i} = \sum_{t=1}^T Y_{u,i,t}$.

Let $\{X_t \in R^{K \times M} | t = 1, \dots\}$ denote a set of latent factor matrices over time, where K is

the dimensionality of the latent factor. Then, user preferences at each time step t can be represented by user latent factors at time t. Let $X^* \in R^{K \times M}$ denote user latent factor matrix for the collapsed adoption matrix Y^*. Similarly, let $C \in R^{K \times N}$ denote the item latent factor matrix. Note that item latent factors are assumed to be static over time as usual. The first-order linear dynamic system [60] is used to model the tendency of user latent factors. In particular, four types of dynamic systems are proposed. The first type of dynamic system of user u is defined as $X_t^u = A_t^u X_{t-1}^u + w$ and $Y_t^u = C X_t^u + v$, where A_t^u is the projection matrix for user u at time t. The item latent matrix C is separately learned by using the non-negative MF where the prediction is defined as $Y^* = C \cdot X^*$. The variables w and v are uncorrelated Gaussian noise defined as $v \sim N(0, R)$ and $w \sim N(0, Q)$, where the covariance matrices R and Q are set to be $0.1 * I$ in the method.

In non-negative MF, the collapsed adoption matrix Y^* is composed of the contribution of both C and X^*. In the above model, C is learned from Y^* and remains constant while user latent factors are changing over time to reflect the temporal adoptions Y. Therefore, X_t has to be adjusted downwards to compensate the spread of Y^* over time. In this regard, it is also possible to spread to the contribution of C across time, which leads to the second type of dynamic system as $Y_t^u = C^u X_t^u + v$, where $C^u = \frac{C}{\text{the number of observed item steps for user } u}$. Note that the transition function in the dynamic system keeps unchanged.

In the above models, the projection matrix A_t^u is different for each user at each time step. Due to the problem of data sparsity in RSs, those transition functions may introduce too many free variables and cause the parameter learning prone to overfitting. Therefore, the third type of dynamic system will adopt a global projection matrix A for all users at every time step, $X_t^u = A X_{t-1}^u + w$. The fourth type of dynamic system combines the idea of spreading the effect of C and the global projection matrix A, which has the following dynamic system, $X_t^u = A X_{t-1}^u + w$ and $Y_t^u = C^u X_t^u + v$.

The dynamic systems learn their parameters dynamically at each time step. In particular, the projection matrix $A_{n,t}$ is dynamically updated using Kalman or RTS smoothing method [210, 80, 217]. Meanwhile, user latent factors are tracked by using the Kalman

filtering. Although the model is based on the non-negative MF, the usage of RTS smoothing and Kalman filtering will violate the non-negative constraint, which makes the learned latent factors hard to be interpreted again. If non-negative constraints are adopted and Lagrange optimization is used, the computational complexity will be too high to be applicable in RSs [80].

The approach discussed above imposes linear and Gaussian assumptions on the dynamic system, which may not be adequate to catch up with sudden changes in the tendency of user preferences. Meanwhile, the tendency of item attractiveness is ignored. Furthermore, all the users share the identical parametric form of dynamic systems. By considering the diversity of user preferences and a large number of users in the systems, this approach may be not flexible enough to be tailored to user's specific tendency.

Binary Feedback with Poisson Factorization In order to model the tendency of user preferences and item attractiveness in the implicit (binary) feedback, the Poisson distribution is also exploited [55]. Let K denote the dimensionality of latent factors in the system. Similar to the latent factor representation used in [80, 145], $U^t \in \mathcal{R}^{K\times N}$ and $V^t \in \mathcal{R}^{K\times M}$ are used in [55] to represent the latent matrices over N users and M items at time interval t, respectively. For user u and item i at time interval t, the implicit feedback $y_t^{u,i}$ can be defined as follows,

$$y_t^{u,i} \sim Poisson(\sum_{k=1}^{K} e^{(U_t^{u,k}+\bar{U}_u)(V_t^{i,k}+\bar{V}_i)}), \tag{2.38}$$

where $\bar{U}_u$ and $\bar{V}_i$ represent the latent factors irrelevant to temporal effects for user u and item i, respectively.

Similar to the linear dynamic systems used above, the first order Gaussian random walk models the temporal dynamics of latent factors across time frames. However, to capture the idea inspired by the Poisson factorization [89], the transition model is imposed for every latent factor of user u as $U_{k,t}^u = U_{k,t-1}^u + \epsilon_U$, where the Gaussian noise ϵ_U is defined as $\epsilon_U \sim \mathcal{N}(0, \sigma_U^2)$ over all the users. The transition model for each latent factor in item latent matrix is defined symmetrically. Due to the usage of the Poisson distribution to capture the binary feedback, the posterior distributions of latent factors are no longer

analytically solvable and the Kalman filtering is actually not employed to update the inference in [55]. Therefore, the variational inference is adopted to approximated those posteriors. After obtaining the estimated posteriors, the prediction of user u's preference over unseen item i at time t can be estimated via the following formula, $E[y_t^{u,i}|\mathcal{Y}] = E_{\text{posterior}}[e^{(U_t^{u,k}+\bar{U}_u)(V_t^{i,k}+\bar{V}_i)}]$, where the set $\mathcal{Y}$ represents the historical binary feedback up to the current time interval.

Compared with methods in previous subsections, the method developed in [55] focuses on the accurate modeling of the binary user feedback in RSs. The latent factors are used to represent the user preferences and item attractiveness over time. Similar to previous methods, this method still imposes linear and Gaussian assumptions over the transitional relations over latent factors. Due to the non-conjugate property of the adopted observation model, the Kalman filtering cannot be used to update the inference over those latent factors across time frames. Therefore, to estimate the tendency of user preferences and item attractiveness over time, this approach has to retrain the underlying model based on all the historical data and the newly arrived feedback, which may not catch up with the sudden changes in user preferences and item attractiveness. Meanwhile, due to the adoption of Poisson factorization, the interactions and collaboration among user and item latent factors can only happen to one particular latent dimension, which does not capture the effects across latent dimensions and may require a higher degree of freedom in the underlying model than that of models in [80, 158].

Pairwise preferences with Dynamics User segments are constructed by clustering users by their feature vectors to enable personalized recommendations. All the users belonging to the same segment will be provided with the same recommendations. In order to extract the pairwise preferences from user actions over time, the logs of user actions are divided into a sequence of sessions as shown in Section 2.3.

A probabilistic model based on Bayesian hidden score method [222] is developed to generate user segment-based ranking and recommendations [34]. Let $r_t^{i,j;c}$ denote the perceived preference between content i and content j for user segment c at time interval t. The

preference is assumed to have the following distribution,

$$r_t^{i,j;c} \sim P(r_t^{i,j;c}|s_t^{i;c}, s_t^{j;c}, \sigma) = N(r_t^{i,j;c}|s_t^{j;c} - s_t^{i;c}, \sigma), \tag{2.39}$$

where σ denotes the standard deviation of the Gaussian distribution. The latent factors $s_t^{i;c}$ and $s_t^{j;c}$ represent the intrinsic attractiveness of content i and content j for user segment c at time t, respectively.

Each latent factor $s_t^{i;c}$ is assumed to follow a first order Gaussian random walk as follows,

$$s_t^{i;c} \sim P(s_t^{i;c}|s_{t-1}^{i;c}, \lambda) = N(s_t^{i;c}|s_{t-1}^{i;c}, \lambda), \tag{2.40}$$

where λ is the standard deviation of Gaussian noise. Given the set of perceived preferences R, the likelihood function of Bayesian hidden score can then be defined as follows,

$$P(R; \alpha, \lambda) = \prod_t \prod_{r_t^{i,j;c} \in R} P(r_t^{i,j;c}|s_t^{i;c}, s_t^{j;c}, \sigma) P(s_t^{i;c}|s_{t-1}^{i;c}, \lambda) P(s_t^{j;c}|s_{t-1}^{j;c}, \lambda). \tag{2.41}$$

The SGD is used to maximize the logarithm of the likelihood function $P(R; \alpha, \lambda)$ to learn the latent factors $s_t^{i;c}$ that correspond to the preference of user segment over contents.

Although dynamics are modeled by those methods, there are no recursive estimation methods involved, such as Kalman filtering or ADF. Meanwhile, the method does not exploit the collaboration among users. The preferences of user segments over items are directly modeled by hidden scores rather than the interaction between user preferences and item attractiveness. In this regard, the modeling approach taken above may be too coarse for the personalized recommendation.

Non latent factor-based method An individual level hazard model is developed to estimate the user preferences from the time duration that the user has spent listening to a song [62]. Meanwhile, it is assumed that the duration that a user listens to a song is proportional to the utility that the user derives from the song. A log-normal distribution [64, 110] is used to model the listening duration $y_{u,i}$ of user u for song i as follows,

$$f(y_{u,i}|\mu_u, \sigma_u) = (2\pi)^{-\frac{1}{2}} (y_{u,i}\sigma_u)^{-1} exp\{-\frac{(\log(y_{u,i}) - \mu_u)^2}{2\sigma_u^2}\}, \tag{2.42}$$

where μ_u is the mean for user u and σ_u the standard deviation for user u. Because only the listening duration up to the length of the song in question is observed, a survival function $S(y_{u,i}|\mu_u, \sigma_u)$ is also applied. Then, the likelihood function over user u's observations y_u is defined as follows,

$$L(\mu_u, \sigma_u|y_u) = \prod_{i=1}^{n_u} f(y_{u,i}|\mu_u, \sigma_u)^{\delta_u} S(y_{u,i}|\mu_u, \sigma_u)^{(1-\delta_u)}, \tag{2.43}$$

where n_u represents the number of observation for user u, and the parameter δ_u is the censor indicator for user u. The mean μ_u is decomposed as $\mu_u = X^T\beta_u$, where the matrix X denotes song attributes and genre dummies from meta information and β_u denotes the latent coefficients for user u. An aggregate model is also used to capture the impact of the songs' unique characteristics that are not reflected in X.

The Gibbs sampling method [86] that is equipped with Metropolis-Hasting MCMC method [21] is used to sample the posterior distributions of model parameters σ_u, latent variables β_u and their covariance Ω_{β_u}. Because there are numerous exploratory variables, the variable selection method [97] is applied to each individual model to remove the redundant variables. This variable selection stage is not applied to the aggregate model because all song-specific constants are expected to be estimated. The goal of variable selection is to eliminate the inclusion of j-th property in X_u if $\beta_{u,j}$ is close to 0. Let $\Psi_u = (\Psi_{u,1}, \ldots, \Psi_{u,P})$ denote the indicator variables, where $\beta_{u,p}$ is set to 0 if $\Psi_{u,p} = 0$ and $\beta_{u,p}$ is set to 1 if $\Psi_{u,p} = 1$. The Gibbs sampling method used in the above procedure is thus extended by incorporating the posterior distribution $P(\Psi_u^{new}|\beta_{u\Psi_u}, \sigma_u^2, \Psi_u^{old}, y_u)$, where $y_u = y_{u,\cdot}$ and $\beta_{u\Psi_u}$ represents the reduced vector of latent coefficients.

To dynamically update model parameters σ_u, variables β_u and Ψ_u, the particle filtering (PF) [200, 60], which is one of the Sequential Monte Carlo methods [75], is used to track and update them whenever new observations arrive. The observation function in PF is based on the reduced set of variables obtained from the variable selection stage. The transition function adopts the first order Gaussian random walk.

Although the PF method is used to update model parameters and latent coefficients when new observations arrive, the observation function only models the individual's observa-

tions and thus no collaboration involved. Meanwhile, the collaborative customization only serves as the average over the uncertainty of individual models and the aggregate model, which does not share the ideas of CF. Moreover, the dynamics of item attractiveness are neglected.

Summary As discussed above, the dynamic approach based on latent factors in RSs not only represent user preferences and item attractiveness compactly but also explicitly tracks the tendency of user preferences via the evolutions of user latent factors. However, those models usually impose some restrict assumptions about the evolution process, which may not capture the real-world scenarios accurately. In addition, the temporal interactions and collaborations between user preferences and item attractiveness are largely neglected, which may not catch up with the sudden changes in user preferences and item attractiveness across time frames. For methods based on non-latent factors, the parameters of some distributions are usually treated as the states and tracked. Because a proper distribution capturing user feedback is sometimes application dependent, methods in this approach are not as general as methods in dynamic approach based on latent factors. Similar to its counterpart, dynamic approach based on non-latent factors usually ignores to model the temporal information relating to item attractiveness and temporal interactions between user preferences and item attractiveness.

2.5.2 Dynamics Modeled by Temporal regression

The linear regression model [176, 167] is also adopted to model the dynamics of user preferences for exploiting the temporal dynamics in RSs. The learned models are usually fixed after the training period, which may not capture the trends in user preferences and item popularity over time. Meanwhile, compared to the first order random walk that functions as a random search in state space or parameter space, the learned regression model is a universal one that guides the dynamics across all users. This modeling approach may be too coarse to fit into the diversity of user preferences in RSs.

Linear Regression Models A simple regression approach for each item could be used to model the short-term dynamics of the item. The short-term evolution of the probability that items are rated is modeled by a temporal regression model [42].

Let $p_i(t)$ denote the popularity of item i at time t. Then, for movie i, its popularity $p_i(t)$ at time t is modeled as follows,

$$p_i(t) = \alpha_{i,0} + \alpha_{i,1}(t - t_0) + \cdots + \alpha_{i,K}(t - t_0)^K, \tag{2.44}$$

where t_0 is the time index of the beginning time interval in the training data. Model parameters $\{\alpha_{i,0}, \ldots, \alpha_{i,K}\}$ are learned from the training data that only span a short period, and model complexity K is selected from empirical studies, such as the cross-validation method.

In order to incorporate the temporal information in personalized RSs, the preference $score(u, i, t)$ of user u on item i at time t is defined as follows,

$$score(u, i, t) = \frac{1}{\sqrt{|\mathcal{R}(u, t)|}} \sum_{k \in \mathcal{R}(u,t)} w_{k,i} + p_i(t), \tag{2.45}$$

where $\mathcal{R}(u, t)$ represents the set of items that have been rated by user u at time t, and the latent factor $w_{k,i}$ represents the contribution from item k to item i. The parameters in $p_i(t)$ and latent factor $w_{k,i}$ over all users and items are jointly learned with the constraint that $score(u, i, t) \geq score(u, j, t)$ if item i has been rated at a rate greater or equal to 3 while item j are not. A ranking loss, which is an instance of a non-separable loss in [234], is used to relax the constraints and form the objective function of the optimization problem. Then, the SGD method is used to learn those parameters and latent factors. Finally, the personalized recommendation list is generated by ranking items via the predicted $score(u, i, t)$ over the next time interval.

As discussed above, there may be too many free parameters to learn a model with a satisfiable ability of generalization. This problem may be getting worse when the available data are very sparse, which is not uncommon in the context of exploiting temporal dynamics in RSs. Meanwhile, this approach does not exploit the collaboration among users, which is essential to cope with the problem of data sparsity in RSs.

Online learning and Offline initialization A fast online learning stage is developed to track item latent factors that represent the item characteristics over time [9]. The parameters of the linear regression model used for online tracking are initialized by an offline learning stage, which is conducted much less frequent than the online stage.

Both online and offline components share the same observation models. For binary feedback, they use the Bernoulli distribution to model an observation as $y_t^{i,j} \sim Bernoulli(p_t^{i,j})$, where $y_t^{i,j}$ is the binary feedback given by user i on item j at time t and $p_t^{i,j}$ is the success probability of this distribution. The logarithm of the likelihood function of all the observations $\{y_t^{i,j}\}$ is given as follows,

$$l(\{y_t^{i,j}\}; \{p_t^{i,j}\}) = \sum_{i,j,t} (y_t^{i,j} \log(p_t^{i,j}) + (1 - y_t^{i,j}) \log(1 - p_t^{i,j})). \tag{2.46}$$

Let $s_t^{i,j} = \log \frac{p_t^{i,j}}{1-p_t^{i,j}}$ denote the log-odds of $p_t^{i,j}$. For numeric feedback, the model assumes that the observation $y_t^{i,j}$ follows a Gaussian distribution with mean $s_t^{i,j}$ and standard deviation σ. Its logarithm of the likelihood function of all the observations is given as follows,

$$l(\{y_t^{i,j}\}; \{s_t^{i,j}\}, \sigma) = -\frac{R}{2} - \sum_{i,j,t} \frac{(y_t^{i,j} - s_t^{i,j})^2}{2\sigma^2} + \mathcal{C}, \tag{2.47}$$

where $\mathcal{C}$ is the constant irrelevant to model parameters and R is the number of feedback. For easy exposition, the following discussion is only based on the numeric feedback. The situation of binary feedback can be similarly derived.

Let $U_t^u \in \mathcal{R}^{K \times 1}$ and $V_t^i \in \mathcal{R}^{K \times 1}$ denote the latent factors representing the preferences of user u at time t and the characteristics of item i at time t, respectively. Let $X_t^{u,i}$ denote the feature vector for both user u and item i at time t, which is computed offline from the context information related to user u at time t and item i. Because most of the contextual information related to items is static, the item feature vectors are assumed to be fixed over time as usual. The feature vector of item i is denoted as X_i.

The offline component adopts a bilinear regression model to model the relationship between the feature vectors and the hidden state $s_t^{i,j}$ used in Eq (2.47). In particular, the regression

model is defined as $s_t^{u,i} = (X_t^{u,i})^T b + (U_t^u)^T A X_i + (U_t^u)^T V_t^i$, where b and A are model parameters that will be learned offline from the historical data. A Bayesian approach is also taken to enhance the bilinear regression model in the above equation. The prior of user latent factors U_t^u at time t is assumed to follow a multivariate normal distribution as $Y_t^u \sim \mathcal{MVN}(G \cdot X_t^u, \sigma_U^2 I)$, where model parameters G and σ_U will also be trained beforehand. To further reduce the computational complexity, the item latent factors V_t^i at time t are assumed to be further decomposable as $V_t^i = B\theta_t^i$, where the matrix B represents an offline learned projection matrix and item latent factors θ_t^i represent item properties more compactly. The prior of θ_t^i is also assumed to follow a multivariate normal distribution as $\theta_t^i \sim MVN(0, \sigma_V^2 I)$ with standard deviation σ_V.

EM is used to learn model parameters with all the available historical data stored in the offline component. The Gibbs sampling method is used to approximate the posterior of latent factors whenever the expectations are not analytically derivable. After learning parameters η from the offline component, the current item latent factors can be updated through the online component. The online regression model is defined as $s_t^{u,i} = (X_t^{u,i})^T b + (U_t^u)^T A X_i + (U_t^u)^T B\theta_t^i$. This regression model is almost identical to the offline model except it only has one parameter θ_t^i. By treating this linear regression model as the observation function in a dynamic system, the Kalman filtering can update latent factors θ_t^i with the assumption that the transition system of θ_t^i follows a Gaussian random walk.

Item latent factors are tracked based on the initialized regression model, which is used as the observation function, from the offline component. However, the regression model used in the offline component does not model the dynamics of latent factors across time frames. Model parameters are learned in a sense that they are averaged over observations from a set of time frames without considering any ordering constraints. Moreover, the online component only tracks item latent factors and ignores user latent factors.

2.5.3 Discussion

Almost all the methods in dynamic-based approach for RSs have the linear and Gaussian assumptions, which may not be adequate to catch up with the sudden changes in the tendency of user preferences. Meanwhile, the tendency of item attractiveness is largely neglected. Under the linear and Gaussian assumptions used in the dynamic model, it is not straightforward to jointly track both the user and item latent factors, because the observation functions will become non-linear ones.

Furthermore, all the users share the identical parametric form of dynamic systems. This approach may not be flexible enough to tailor the models to user's specific tendency, considering the diversity of user preferences and item attractiveness and a large number of users and items in the systems.

2.6 Miscellaneous

There are some other studies relating to exploit temporal and dynamic information in RSs. Those miscellaneous studies, which do not fit well into previous four approaches, are discussed in detail in this section.

2.6.1 Personalized marginal utility with Temporal Information

Existing product RSs usually only consider the order of items that are purchased by users [128, 189, 239]. However, the time intervals between purchases can also greatly influence the users' decisions in future [118][1]. Intuitively speaking, within a short period and for some items, users do not tend to purchase the similar items after the latest purchase.

[1]Nevertheless, in [118], only the possible time instances that a user will probably make a purchase on one particular item has been studied and inferred. Item preferences at each time instances for the user are largely neglected. Actually, experiments do not distinguish various item transitions for one particular kind of item.

This consumer behavior is known as the law of diminishing marginal utility [27]. Hence, the purchase interval information from the purchase history of users is combined with the marginal utility to generate product recommendations [251].

In practice, users have different preferences for items. Hence, the classic utility measure is extended to be personalized one as $\delta_{u,i} = \sum_k \beta_u^k * c_i^k$, where $\delta_{u,i}$ denotes the utility of item i for user u, β_u^k represents the k-th latent factor of user u's preferences and c_i^k is the k-th latent factor of the characteristics of item i. The utility surplus [27] for user u on item i can be then defined as $US(u,i) = \delta_{u,i} - \alpha_u \cdot price_i$, where the parameter α_u controls the sensitivity of user u to the price of item i, and $price_i$ is the price of item i.

Let T_i and T_j denote the set of timestamps at which item i and item j are purchased by user u, respectively. Let $\Phi = \{[t_{i_1}, t_{j_1}], \ldots, [t_{i_n}, t_{j_n}]\}$ denote the pairs of timestamps that are selected by the following criterion: for each item i with a timestamp t_{i_l} in the set T_i, its counterpart in T_j has the smallest interval $t_{j_l} - t_{i_l}$ and the time interval is also less than a predefined threshold w. Then, the average purchase interval $d_{u,i,j}$ representing the average time interval between the post-purchase on item j after the purchase on item i for user u is defined as follows,

$$d_{u,i,j} = \frac{\sum_{[t_{i_r}, t_{j_r}] \in \Phi} t_{j_r} - \frac{t_{i_r}}{\log(2+count(t_{j_r}, t_{i_r}))}}{\sum_{[t_{i_r}, t_{j_r}] \in \Phi} \frac{1}{\log(2+count(t_{j_r}, t_{i_r}))}}, \tag{2.48}$$

where $count(t_{j_r}, t_{i_r})$ measures the number of items purchased by user u between time t_{j_r} and time t_{i_r}. Due to the problem of data sparsity, it is impossible to generate the average time interval $d'_{u,i,j}$ between any pair of items for a user, and $d'_{u,i,j}$ is then estimated collaboratively based on $d_{u,i,j}$ to alleviate this problem.

After obtaining the average time intervals, the utility surplus can be extended by incorporating them as follows,

$$US^+(X_i, i, u) = \delta_{u,i} \cdot ((X_i + 1)^{\gamma_i} - X_i^{\gamma_i}) \cdot (1 + \max_{j \in H_u} PI(u, j, i))^{\mu_i} - \alpha_u \cdot price_i, \tag{2.49}$$

where H_u is the purchase history of user u, and parameters γ_i, μ_i and α_u could be learned from the data beforehand. The function $PI(u, j, i)$ is defined as $PI(u, j, i) = \frac{1}{\log(|t - t_i - d'_{u,i,j}| + 2)}$. This function offsets the effect of purchase intervals.

Similar to PMF, the joint distribution defined over parameters α_u, μ_i, γ_i, β_u and c_i can be converted to an optimization problem via maximum likelihood estimation. The above approach assumes that prior distributions of all the parameters follow the Gaussian distributions whose standard deviations work as coefficients in the regularization.

The above approach explicitly considers the time interval between user purchases to improve the performance of RSs. However, only temporal information is exploited. The dynamics between latent factors across time frames are not considered. Meanwhile, there is no updating stage adopted in the model. After learning, the model can only make predictions based on the model parameters that are fitted to the training data.

2.6.2 Dynamic Item Weighting and Selection for Collaborative Filtering

From the perspective of the classic instance-based learning approach [235], users in RSs can be treated as instances, and ratings that users give to items can be regarded as feature vectors that describe instances. Intuitively, some features are more important and informative than others. Therefore, dynamic feature weighting and dynamic feature selection can be applied to user-based CF to improve the performance of recommendation by introducing the weights when computing the similarities among users [25].

Specifically, two groups of methods are studied to generate item weights. The first group of methods yields the item weights based on some statistical methods, such as, variance and mutual information [164] that use frequency counts as the empirical estimation of the probability distribution. The second group exploits the tags attached to items to compute the weights based on the empirically defined formula that measures the number of common tags shared between items. After introducing the weighting mechanism, the similarity, such as Pearson correlation coefficient [115], is modified as follows,

$$WPCC(u,v,j) = \frac{\sum_i (w_{j,i}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_i (w_{j,i}(r_{u,i} - \bar{r}_u))^2 \sum_i (w_{j,i}(r_{v,i} - \bar{r}_v))^2}}, \tag{2.50}$$

where $w_{j,i}$ is the weight between item i and item j, and $r_{u,i}$ is the rating that user u gives to item i with user u's average rating $\bar{r}_u$.

Some empirical criteria are also adopted as feature selection methods. Among them, the dynamic best-f-per-overlap [235], which selects f items with largest weights presented in both the target user profile and profiles of its neighbors, outperforms other methods.

The above approach attempts to improve the performance of user-based CF by dynamically selecting important features for each user. However, the temporal information which is vital to model the tendency of user preferences and item attractiveness has been neglected in the proposed method. Meanwhile, the experiments conducted demonstrate that little improvement can be made through the proposed feature weighting and feature selection methods when computing the similarities between users.

2.6.3 Evaluating the dynamic properties of recommendation algorithms

In order to measure the temporal performance of RSs, a temporal leave-one-out technique is also proposed in [47]. This measurement is inspired by the methodology of the leave-one-out method [124] for training static methods. The temporal leave-one-out technique requires that the prediction of a rating $r_t^{u,i}$ for user u on item i at time t is made only based on the ratings prior to time t. By sorting the temporal order of ratings and applying this methodology to each rating, a temporal MAE metric is obtained, where MAE represents mean absolute error metric [115].

Meanwhile, the number of ratings perceived by users is constantly increasing as the time passes. For dynamic RSs, it is essential to measure the temporal behaviors of these methods when more and more ratings are arriving at the system over time. Therefore, a metric called ProfileMAE [47] is proposed to measure the evolution of the average prediction error as the number of inputs keeps increasing over time.

Although these evaluation metrics have some interesting properties, their computation requires much more computational resources compared with commonly used metrics, such as RMSE, precison@N and recall@N, in RSs. Meanwhile, these metrics are not widely tested and verified and not widely noticed and acceptable in the study of RSs.

2.6.4 Dynamic Negative Item Sampling

Existing work on pairwise-based ranking for CF methods relies on either random sampling over unseen items or some heuristic approach to select negative items to construct the training pairs [188, 191, 157]. Either approach is prone to introducing the bias in the built training data and leading to a prolonged training period. Meanwhile, the traditional BPR-based methods optimize area under curve or AUC curve [252, 36], which may be insufficient for personalized RSs when the relative positions of recommended items in the recommendation list do matter. Hence, a dynamic negative item sampling method is developed to overcome these problems by sampling negative instances that aim to maximize the NDCG measure of the recommended list. The dynamic nature of the sampling mechanism is from its procedure to obtain various negative samples at each iteration of the SGD method that is used to learn the latent factors in the model.

The idea of LambdaRank method [45] is borrowed to ensure that the samples are optimized against the NDCG metric. The gradient of the loss function C in the pairwise ranking can be then expressed as follows [249],

$$\frac{\partial C(< i, j >_u)}{\partial \omega} = f(\lambda_{i,j}, \zeta_u)(\frac{\partial \hat{r}_{u,i}}{\partial \omega} - \frac{\partial \hat{r}_{u,j}}{\partial \omega}), \tag{2.51}$$

where $f(\lambda_{i,j}, \zeta_u)$ is the lambda weight function for the pair of items i and j, and ζ_u is the generated ranking list for user u. The notation $< i, j >_u$ denotes the pairwise preference input for user u on item i and item j. The parameter $\lambda_{i,j}$ is the learning rate for this pair. For the purpose of NDCG maximization, the function $f(\lambda_{i,j}, \zeta_u)$ is defined as $f(\lambda_{i,j}, \zeta_u) = \lambda_{i,j} \triangle NDCG_{i,j}$, where $\triangle NDCG_{i,j}$ represents the difference between the NDCG values when the ranks of item i and item j are switched.

It is computationally expensive to compute $f(\lambda_{i,j}, \zeta_u)$, considering the huge number of users and items in the RSs. Therefore, a negative item sampling scheme is developed to approximate $\lambda_{i,j} \triangle NDCG_{i,j}$, which efficiently obtains negative samples without cycling through all the items. The idea is based on the observation that the higher ranked unseen items, regardless of its real preferences by users, have a large impact on the computation

of $\lambda_{i,j} \triangle NDCG_{i,j}$. Therefore, it is reasonable to place more emphasis on those negative samples that will be ranked in the top positions.

Two sampling mechanisms are proposed. Both of them are based on a two-phase sampling scheme. That is, a subset of unseen items is randomly drawn, which is used as the candidate samples in the first phase. In the second phase, one negative item that has a high rank with large probability will be sampled to construct the training pair. The differences between those two sampling mechanisms lie at the number of candidate samples in the first phase and how to reject one but all candidates in the second phase.

The sampling schemes at each iteration in the optimization procedure not only enhance the pairwise ranking-based MF by optimizing against NDCG metric but also develop an efficient and systematic way to obtain negative samples. However, it is very specific to BPR-based methods and no temporal information in user feedback is exploited.

2.7 Summary

In this chapter, an intensive literature review has been conducted for existing methods that exploit temporal and dynamic information in RSs. As mentioned in the chapter, those methods can be roughly categorized into four approaches: the heuristic approach, binning-based approach, online updating approach and dynamic-based approach. The technical details and their advantages and weaknesses are also briefly discussed.

Roughly speaking, most of the methods in the heuristic-based approach are easy to understand and implement. However, the temporal dynamics of user preferences and item attractiveness are largely neglected in this approach. Meanwhile, the temporal interactions between user preferences and item attractiveness are also not considered. Although those methods usually underestimate the importance of past feedback, they do not tend to introduce any extra computational burden to their counterpart static methods and are suitable for real-world deployments that encounter a huge number of users and items and do not have a rigorous specification of the way to exploit the temporal information.

For methods in the binning-based approach, their great computational complexity involved in the training stage may impede their deployment as responsive tools for real-world deployments. However, their low computational complexity for prediction still enables them to be applied by repeatedly retraining their models offline with all the data up to the current timestamp. Methods of this approach tend not to model the temporal information relating to item attractiveness and the dynamics and temporal interactions between user preferences and item attractiveness. In addition, by borrowing the temporal information after the timestamp when the predictions should be made, the recommendations generated in this approach are actually *post hoc* about what interests *would have been* in the past, instead of predicting the future interests of those users. However, for user feedback that does demonstrate some periodic properties, methods in this approach are more proper than reweighting the importance of user feedback.

Methods in the online updating approach take into consideration the continuous nature of the gathering of user feedback in RSs. For some methods that incrementally update their constructed models, this approach is able to reduce the computational complexity of generating satisfiable recommendations and make the original methods more responsive and attractive. For some methods that consider exploiting the temporal information from the continuously arrived observations, they are more capable of catching up with the current tendency of user preferences whenever new observations arrive in the systems. However, most of the methods in this approach do not consider modeling the dynamics and temporal interactions between user preferences and item attractions. Meanwhile, the tendency of item attractiveness, which is also vital to the success of RSs, is also largely neglected in this approach.

Most of the methods in previous three approaches can be regarded as somewhat simplified versions of methods developed in the dynamic-based approach by fixing some of their components as static. In other words, it is possible to deduct from methods in the dynamic-based approach to approximate some methods in previous three approaches. However, no matter whether the state space has been adopted or not, methods in the dynamic-based approach tends to place linear and Gaussian constraints on both the dynamics of

user preferences and the distributions of user feedback. Meanwhile, the tendency of item attractiveness is usually ignored and assumed to be static over time. Furthermore, the temporal influences and interactions between user preferences and item attractiveness are not explicitly considered.

2.8 Discussion

Existing methods of the four approaches discussed above have explored and exploited the temporal information in various ways. Methods in heuristic-based and online updating approaches can find their origins in other fields, such as concept drifting [81], incremental learning and regression in machine learning and data mining. Methods in the dynamic-based approach also share some similarities with methods in object tracking and signal processing. However, as discussed above, the characteristics and requirements of RSs, such as the problem of data sparsity and the large number of users and items in the systems, impose some new challenges and problems when exploiting and transferring those techniques to RSs. Methods in the binning-based approach are quite unique for the study of RSs, but they usually ignore the temporal dynamics of user preferences and item attractiveness and are not as applicable as methods in other three approaches.

Although there exist some studies on utilizing temporal and dynamic information in RSs, existing methods are still under some rigorous assumptions that simplify the real-world scenarios. Meanwhile, even though the temporal dynamics of user preferences have been covered in some of the methods discussed above, the temporal dynamics of item attractiveness is not investigated in those existing methods. The methodology of collaboration plays a significant role in the success of CF in RSs. However, for those existing methods, the temporal collaboration and interactions between users and items are also neglected. Furthermore, almost all of existing methods in those four approaches merely focus on the modeling of temporal dynamics of average behaviors of user preference and item attractiveness. The temporal dynamics of variations of user preferences and item attractiveness are not taken into consideration.

As discussed above, methods of each approach have their advantages and weaknesses. However, compared with other three approaches, methods in the dynamic-based approach demonstrate more interesting traits from the algorithmic perspective. Hence, the research in this thesis will focus on the dynamic-based approach at first. The work conducted at Chapter 3 and Chapter 4 aims to advance the methods in this category by overcoming their shortages while preserving their advantages as much as possible. Then, Chapter 5 focuses on exploring and taking advantages of the temporal dynamics of variations of user preferences and item attractiveness. Specifically, a fine modeling of the tendency of user preferences and item attractiveness will be conducted at first, which not only tracks those two components in RSs simultaneously but also releases the Gaussian assumption on the generative procedure of user feedback. After equipped with this model, the personalized and item-wise model structures of dynamic systems from user feedback is investigated. This model aims to capture the tendency of user preferences and item attractiveness flexibly. In order to achieve this goal, a new problem in the learning of personalized and item-wise dynamic systems, namely, the *cold start transition* problem, has been identified, defined and solved. Finally, the temporal dynamics of variations of model parameters in RSs, which is usually overlooked, will be explicitly modeled. This modeling approach aims to cope with the sudden changes and other local temporal effects among user preferences and item attractiveness. In terms of algorithmic perspective, it also aims to explore a new approach to model the tendency of user preferences and item attractiveness.

Chapter 3

Tracking the Tendency of User Preferences and Item Attractiveness

3.1 Introduction

In real-world deployments, user feedback is continuously gathered over a long period. As depicted in Figure 1.1 and Figure 1.2 in Chapter 1, this feedback, to some extents, reveals a tendency of user preferences and item attractiveness. As discussed in Section 1.2.2 in Chapter 1, existing methods exploiting temporal dynamics in RSs largely neglect the modeling of temporal dynamics in both user preferences and item attractiveness. Even though there exist some methods that attempt to exploit temporal dynamics, they usually neglect to model the dynamic nature corresponding to the tendency of item attractiveness. Hence, the inherently and significantly temporal interactions between users and items are also ignored. In addition, user feedback is usually assumed to be Gaussian distributed, which may oversimplify real-world scenarios. This chapter thus focuses on modeling these important aspects intrinsic in user feedback to improve the performance of RSs over time.

3. Tracking the Tendency of User Preferences and Item Attractiveness

Because the datasets in practice are usually extremely sparse and the feedback is commonly non-repetitive, a user's interests over all of its unseen items are usually predicted by exploiting its historical feedback and the historical feedback of "like-minded" users. This idea is the exact mechanism adopted by collaborative filtering (CF) [115], which is widely used in RSs. Inspired by this motivation, under temporal context, users' tastes over their unknown items within a concrete time frame are also extracted by exploiting the feedback of users and other "like-minded" users on similar items up to the time frame. Furthermore, as discussed in Section 1.1.3 in Chapter 1, the content or context related information in RSs is usually unavailable and contains relatively static information. Taking all these factors into consideration, the methods discussed in this chapter are developed on the basis of CF. Specifically, those developed methods enhance MF-based CF with the capability to exploit temporal dynamics in RSs. The fine modeling of the temporal and dynamic dependencies among user preferences and item attractiveness across time frames is also emphasized, which aims to explore the intrinsic temporal interaction in RSs as much as possible.

As discussed in Chapter 2, the dynamic-based approach shows some advantages over previous three approaches, i.e., heuristic based, binning-based and online updating based approaches. This approach explicitly models temporal dynamics by either a stochastic state space model or temporal regressions. However, in such an approach, item attractiveness is usually assumed to be static and Gaussian distributed. Hence, in order to overcome these problems, the particle filtering [200] could be utilized as a dynamic technique to model non-Gaussian observed behaviors and track latent factors representing user preferences and item attractiveness. Instead of using a huge concatenated state vector consisting of all the user and item latent factors, to make the developed method tractable, those latent factors are separately tracked for individual user and item at each time frame. Although this enhancement improves the performance of recommendations in general, specific care must be taken as simply applying this compromised approach could lead the model to develop in inaccurate direction. For example, the separation of user and item latent factors in the state space makes the developed method prone to the divergence of particle filtering, especially when the tracked tendency is multimodal and highly nonlinear. As this situation is not uncommon for user preferences and item popularity, new models and

learning algorithms are further required to enhance the developed particle filtering-based method to improve the temporal performance of RSs.

Meanwhile, under temporal context, the problem of data sparsity [115] becomes more challenging as many users would be inactive for some consecutive time slots. Exploiting additional information, such as contextual information or common patterns [115], or imputing missing data are two common approaches to mitigate the problem of data sparsity. However, as discussed above, the gathering of extra information is usually infeasible in practice. Hence, side information is not considered in this chapter. Alternatively, missing data are usually imputed as negative [212, 191] or other values [207, 112]. However, these methods are only developed for static context on the basis of some heuristic rules or point estimators. Furthermore, all these methods impute all or most missing data, which not only heavily reduces the scalability of the underlying models but also reduces the recommendation accuracy.

The work described in this chapter attempts to solve the above problems by [160, 161],

1) making both user preferences and item attractiveness time-varying, and tracking both of their representations by exploiting temporal and dynamic structures in user feedback, which could be non-Gaussian distributed,

2) creating a new dynamic model based on particle filtering and introducing factor weights to not only enforce the interactions among user preferences and item attractiveness but also dynamically weight different dimensions of latent factors according to their temporal importance,

3) utilizing the self-training principle [254] to construct the training data dynamically based on the distributions of the current personalized prediction of user preferences,

4) proposing a new learning algorithm, without the introduction of nontrivial extra computational complexity, to dynamically update the model in order to alleviate the problems when separately tracking user and item latent factors.

Specifically, the proposed system uses two sets of latent vectors to represent user preferences and item attractiveness compactly and respectively at each time step, whose initial settings are learned by a probabilistic matrix factorization method. Based on these representations, the system employs particle filtering to track separately the tendency of user preferences and item attractiveness over time. In order to enforce the temporal and dynamic interactions among user and item latent factors, the system utilizes a novel probabilistic temporal bilinear model to improve further the temporal recommendation performance. In order to mitigate the problems of data sparsity and scalability, the system also adopts a novel self-training mechanism to construct the training data dynamically. The system then dynamically updates all the model parameters taking as inputs the temporally constructed training data. For a Top-N recommendation, a personalized recommendation list for each user is generated based on the predicted user preferences for items, which are computed using the updated model parameters and the current user and item latent vectors.

The rest of this chapter is organized as follows. Related work is presented in Section 3.2. The particle filtering for MF method, the probabilistic temporal bilinear model and the self-training scheme are discussed in Section 3.3, Section 3.4 and Section 3.5, respectively. Section 3.6 covers the developed learning algorithm. The pseudo code and the analysis of the computational complexity of developed methods are given in Section 3.7. The performance of proposed methods and a variety of baseline methods is presented and analyzed in Section 3.8. Finally, Section 3.9 will present the summary.

3.2 Related Work

In this section, some of the closely related work to the methods presented in this chapter is briefly discussed as a reminder. The interested readers are suggested to use the detailed discussion in Chapter 2 as a reference. There have been few studies [158, 80, 128, 241, 195, 55] on exploiting temporal dynamics to improve the performance of RSs where [158, 80] are closely related to the work conducted in this chapter. However, item latent factors are

only updated but not tracked in [158]. In [80], item latent factors are assumed to be static and only user latent factors are tracked by Kalman filters. As the transition functions are not learned offline in this chapter, the method in [158] can incorporate [80] by removing spatial priors and fixing item latent factors. In addition, the usage of Kalman filters restricts the dynamics and observation functions to be linear and Gaussian which may not be the case in practice. In [55], the Poisson distribution is employed to focus on the modeling of the binary (implicit) feedback at every time intervals. Meanwhile, the linear and Gaussian assumptions over the transitional relations of latent factors are still imposed. The Kalman filtering cannot be applied to update the inference over latent factors due to the non-conjugate property introduced by the observation model. The particle filtering method has also been used to update a log-normal distribution dynamically that models user preferences in music recommendation [62], assuming the staticness of item popularity. Nonetheless, the method is not based on latent factors, and very application-specific (otherwise, no proper features). Since [158, 80] provide the state-of-the-art approach for the temporal recommendation, the proposed methods are compared to them as described in later sections.

Conventional CF with imputation [115] suffers from the domination of imputed ratings. Sampling missing data is only used in the non-temporal context in one class CF [191], which can be deducted from the problem to be tackled in this chapter by setting relevant ratings as positive examples. User or item oriented sampling schemes, which only based on the times that items or users present, are proposed in [191]. However, the recommendation accuracy is compromised to boost the scalability of the underlying method. Samples are also selected based on pairwise estimation for one class CF to train the model iteratively [249]. All of these sampling methods are developed under static context. Moreover, unlike the developed methods in this chapter, these algorithms do not solve the problems of scalability and sparsity at the same time.

The developed learning algorithm in this chapter has a close relationship with the algorithm in [92], which exploits the EM method to learn a probabilistic bilinear model for vision tracking. However, the methods developed in this chapter have a different gen-

erative model. Meanwhile, the dynamic system in [92] is an identity function. Particle filtering in [92] actually becomes an importance sampling method. In other words, no dynamic and temporal information is used and the model can be regarded as a subset of the model developed in this chapter. In [61], a predictive model is developed to make recommendations of dynamic contents. This model is not based on latent factors. Instead, it directly constructs user and item profiles based on static and temporal side information. In addition, its parameters are only learned offline.

In [9], an online regression method has been exploited to update both user and item latent factors under the Gaussian assumption. However, both regression weight matrices and projection matrices are fixed and not updated online. Unlike the dynamic approach adopted in this chapter, latent factors in the method are not dynamically updated or tracked but only retrained with data in the current time frame. Therefore, information from latent factors at previous time frame are ignored, which makes this method function as a binning-based approach.

Almost all of these methods utilizing temporal information improve recommendation performance under error metrics, such as RMSE [115]. Instead of directly measuring user preferences over the recommendation, error metrics compare the actual user ratings in the test datasets with the numerical recommendation scores that are generated by these methods. In contrast, the developed methods in this chapter exploit temporal and dynamic information to improve the temporal performance of RSs on accuracy metrics, such as top-k hitrate [212], which attempt to measure recommendation relevance or user satisfaction directly.

3.3 Particle Filtering for Matrix Factorization

A state space approach [200] is adopted in this chapter to represent compactly and track the tendency of user preferences and item popularity. Even with linear and Gaussian assumptions, it is not easy to define the state as a joint vector of user and item latent

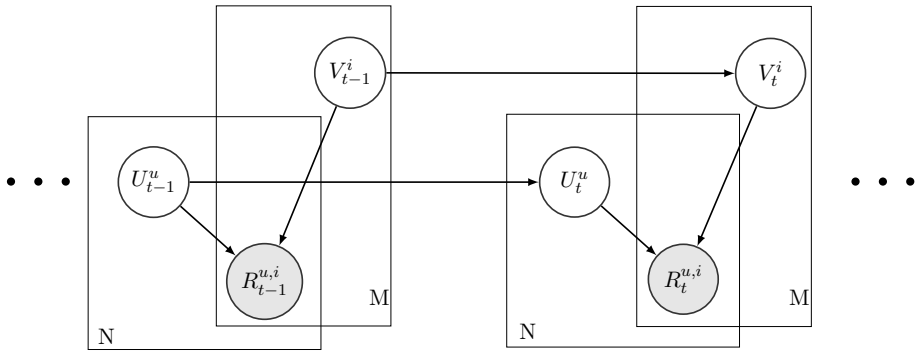


Figure 3.1: A slice of the graphical model of particle filtering for matrix factorization at time $t-1$ and t.

factors. This is because the coexistence of user and item latent factors can easily make the observation function bilinear, which lacks analytical and tractable solutions. Furthermore, it is shown in [196] that empirical distributions of the posteriors should be non-Gaussian. Therefore, the particle filtering is utilized to track these latent factors simultaneously. The particle filtering iteratively approximates regions of high density as discrete sample points. As the number of particles goes to infinity, the approximation converges to the true distribution [200]. In practice, given d-dimensional state space, the number of required particles should be $O(2^d)$ to achieve a satisfactory result [200]. In order to make a compromise between the accuracy of user and item representation and tractability of particle filtering, latent factors for each user and item are separately tracked. Figure 3.1 shows a slice of the graphical model of particle filtering for MF. Let $R^{u,i}_t$ denote the rating given by user u on item i at time t, and U^u_t and V^i_t denote the latent factors for user u and item i at time t, respectively. These latent variables are assumed to be marginally independent while any rating $R^{u,i}_t$ is assumed to be conditionally independent given latent vectors U^u_t and V^i_t [196].

3.3.1 The transition function

Because of the lack of prior knowledge, it is assumed that the transition functions of user and item latent factors follow a first-order random walk driven by multivariate normal noise. However, with the non-Gaussian observation model shown later, this assumption

does not imply that the tendency of user preference and item attractiveness should follow a normal distribution. These functions impose smooth constraints on the underlying dynamics, but the usage of particle filtering enables the estimated latent factors to catch up with a sudden change of the tendency. The transition functions at time t are defined as follows,

$$U_t^u = U_{t-1}^u + c_t^u, \tag{3.1}$$

$$V_t^i = V_{t-1}^i + d_t^i, \tag{3.2}$$

where $c_t^u \sim \mathcal{N}(0, \sigma_U I)$ and $d_t^i \sim \mathcal{N}(0, \sigma_V I)$ are defined as unrelated Gaussian process noises. Intuitively, the scalar values σ_U and σ_V should be different, considering the different evolving pace of user preferences and item attractiveness. Because their associated Gaussian noises mathematically work as the step size of the randomly searching in the latent factor space, they are set as $\sigma_U = \sigma_V = \sigma$ in this chapter for simplicity.

3.3.2 The observation function

The observation function should reflect the ability of a particle to reconstruct the given feedback (ratings). In PMF, maximizing the log-posteriors over U and V is equivalent to minimizing the sum-of-square error function with quadratic regularization terms for MF [196], leading to the following objective function,

$$E = \frac{1}{2}\sum_{u=1}^{N}\sum_{i=1}^{M} Y_{u,i}(r_{u,i} - U_u V_i^T)^2 + \frac{\lambda_U}{2}\sum_{u=1}^{N}||U_u||_{Frob}^2 + \frac{\lambda_V}{2}\sum_{i=1}^{M}||V_i||_{Frob}^2, \tag{3.3}$$

where $\lambda_U = \alpha_U/\alpha$, $\lambda_V = \alpha_V/\alpha$, and $||\cdot||_{\text{Fro}}$ denotes the Frobenius norm.

The objective function in Eq (3.3) is an immediate candidate to define the observation function as follows,

$$P(R|U, V, \theta) \propto e^{-E}. \tag{3.4}$$

However, this candidate function is sub-optimality for the Top-N recommendation task, because an algorithm attempting to minimize the root-mean-squared-error in prediction does not have a satisfactory performance for the Top-N recommendation [66]. Moreover,

Eq (3.3) assumes that unobserved data in both training and testing cases are missing at random. That is, the probability that a rating to be missing is independent of its value. Nevertheless, it is shown [212] that feedback in RSs is generally not missing at random (NMAR). Low ratings are much more likely to be missing than high ratings because users are free to choose items to give feedback [212].

In order to design a suitable observation function, the key idea is to consider the ranking of *all* the items, no matter whether they are observed or not. By treating all the missing data as negative with weights (wAMAN), the observation function over imputed ratings $\bar{R}_t^u$ for s-th particle of user u is,

$$\begin{aligned} P(\bar{R}_t^u | U_t^{u,(s)}, \{V_t^{i,(s')}\}) = \\ \sum_{s'=1}^{S'} exp\{-\sum_{i=1}^{M} W_{u,i}((\bar{r}_t^{u,i} - U_t^{u,(s)}(V_t^{i,(s')})^T)^2 + \\ \frac{\lambda_U}{2}||U_t^{u,(s)}||_{\text{Frob}}^2 + \frac{\lambda_V}{2}||V_t^{i,(s')}||_{\text{Frob}}^2)\}, \end{aligned} \tag{3.5}$$

where

$$\bar{r}_t^{u,i} = \begin{cases} r_t^{u,i} & \text{if } Y^{u,i} = 1 \\ r_m & \text{if } Y^{u,i} = 0. \end{cases}$$

The value r_m is an imputed value for all the missing data, which is regarded as the average value of ratings in the complete but unknown data. The weight $W_{u,i}$ is defined to reflect the confidence over imputation and it is set as a global constant w_m for the imputed data for simplicity [212]. The latent vectors $U_t^{u,(s)}$ and $V_t^{i,(s')}$ represent the s-th and s'-th samples of user and item latent factors at time t, respectively. The observation function over the s'-th particle of item i is defined similarly. The S' is the number of particles for item i and *exp* represents the exponential function. The exponential loss in the observation function is inspired by the formulation of the energy function in the Markov random field [36]. This loss function will place more emphasis on those particles that can reconstruct user preferences over items more precisely. Other loss functions, such as the squared loss or absolute loss, have also been explored in the trails, but it is difficult for them to filter out well-performed particles for the Top-N recommendation task. For later reference, this method is named as PFUV. Because the point mass approximation for

posterior distributions is obtained via particle filtering, the usage of regularization terms in observation functions can only slightly prevent the PFUV method from overfitting. These terms are ignored in the following discussion for clarity.

3.3.3 Tracking

Assuming that the estimation at time t over all the item latent vectors V_t is given, the posterior distribution of U_t^u is approximated by particle filtering with S particles as follows,

$$P(U_t^u|R_{1:t}, \{V_t^i\}) = \sum_{s=1}^{S} w_{U,t}^{u,(s)} \delta(U_t^u - U_t^{u,(s)}), \tag{3.6}$$

where the latent vector $U_t^{u,(s)}$ and its weight $w_{U,t}^{u,(s)}$ represent the s-th particle of user u at time t. Using the transition priors in Eq (3.1) and Eq (3.2) as the proposal distributions, the weight at time t for all the particles is evaluated recursively as

$$w_t = w_{t-1} \cdot P(R_t|U_t, V_t), \tag{3.7}$$

where $P(R_t|U_t, V_t)$ is the observation function discussed in Section 3.3.2. The indicators for users, items and particles are omitted in the formula for clarity. The particle $U_t^{u,(s)}$ is obtained by propagating the s-th particle $U_{t-1}^{u,(s)}$ using dynamics in Eq (3.1). The estimation of item i's latent factors at time t is obtained analogously. User and item latent factors are estimated alternatively as shown in Algorithm 1.

3.4 Probabilistic Temporal Bilinear Model

In this section, a probabilistic temporal bilinear model will be developed, taking as input the explicit feedback of users over items.

3.4.1 Probabilistic Temporal Bilinear Model

Although the approach discussed in the previous section has been developed to improve the recommendation in general, the separate tracking of user preferences and item attrac-

tiveness in the latent vector space can cause some other problems, such as divergence and degeneration of particle filtering [200]. Hence, the approach is inclined to deviate from the correct tendency of user preferences and item popularity. Meanwhile, different dimensions of latent factors may have different interpretation and importance [115]. In addition to the separate tracking of these factors, it is beneficial to explicitly and finely model the dynamic importance of these latent dimensions over time, because this modeling can deeply exploit the temporal and dynamic relations between user preferences and item attractiveness.

A novel probabilistic bilinear model is thus developed in this chapter to overcome these problems. The model aims to not only mitigate the problems of divergence and degeneration during dynamic modeling but also finely model the temporal and dynamic importance between user and item latent vectors. Figure 3.2 illustrates a slice of the graphical model of this probabilistic model at time $t-1$ and t.

Specifically, the temporal weighting matrix $\{D_t \in \mathcal{R}^{K\times K} | t \in 1, \dots\}$ explicitly models such a dynamic relation among latent dimensions. Any rating at time t is now conditionally independent given D_t and corresponding user and item latent vectors while the latent vectors are still marginally independent as before. Extra degrees of freedom are introduced by these temporal importance matrices, enabling the developed model to depict the tendency of user preferences and item characteristics more finely. Through the matrix that is learned online, the developed model dynamically and adaptively confines the temporal variation of user and item latent factors over time. Each weighting matrix will be optimized when a balance between exploitation and exploration in the latent space is reached. This additional optimization also mitigates the divergence and degeneration of particle filtering over a long period of time. The temporal dynamics at each time frame is carefully extracted by D_t and temporal latent factors of users and items. The developed model is thus also expected to tackle the problem of data sparsity in RSs partially.

The transition function Similar to the PFUV method discussed in the previous section, it is assumed that the transition functions of user and item latent factors follow a first-order random walk driven by multivariate normal noise. Similarly, with the non-

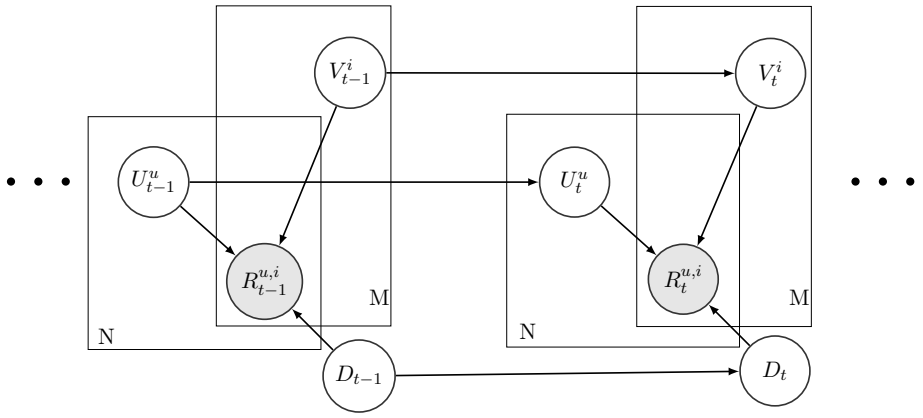


Figure 3.2: A slice of the graphical model of the probabilistic temporal bilinear model at time $t-1$ and t.

Gaussian observation model shown later, this assumption does not imply that the tendency of user preference and item attractiveness should follow a normal distribution. The transition functions of user and item latent factors are identical to the ones described in Eq (3.1) and Eq (3.2), and they are omitted here for clarity.

For ease of exposition, D_t is restricted to a diagonal matrix to reduce the computational complexity of the model in this chapter. This simplification will not impact the following discussion of the developed model. It is straightforward to release the current restriction on D_t by vectorizing D_t and also adopting a first-order random walk as the priors, where MCMC and the spherical Gaussian proposal could be used to sample each component of it [236] during the M-step in Algorithm 1 that will be shown later.

The observation function By adopting the interaction matrix D_t to enforce the temporal importance among user and item latent factors and capture the significance on different latent dimensions, the observation function over imputed ratings $\bar{R}^u_t$ for s-th particle of user u is defined as follows,

$$P(\bar{R}^u_t|U^{u,(s)}_t, \{V^{i,(s')}_t\}, D_t) = \sum_{s'=1}^{S'} exp\{-\sum_{i=1}^{M} W_{u,i}(\bar{r}^{u,i}_t - U^{u,(s)}_t D_t (V^{i,(s')}_t)^T)^2\}. \quad (3.8)$$

Similarly, with S particles for user u, the observation function for s'-th particle of item i is,

$$P(\bar{R}_t^i|V_t^{i,(s')}, \{U_t^{u,(s)}\}, D_t) = \sum_{s=1}^{S} exp\{-\sum_{u=1}^{N} W_{u,i}(\bar{r}_t^{u,i} - V_t^{i,(s')} D_t (U_t^{u,(s)})^T)^2\}, \tag{3.9}$$

where $\bar{r}_t^{u,i}$ is defined in Eq (3.3.2). Similarly, with S particles for user u, the observation function at time t for s'-th particle of item i can be derived.

Tracking Assuming that the estimation at time t over the matrix D_t and all the item latent vectors V_t is given, the posterior distribution of U_t^u is approximated by particle filtering with S particles as follows,

$$P(U_t^u|R_{1:t}, \{V_t^i\}, D_t) = \sum_{s=1}^{S} w_{U,t}^{u,(s)} \delta(U_t^u - U_t^{u,(s)}), \tag{3.10}$$

where δ is the delta function and the latent vector $U_t^{u,(s)}$ and its weight $w_{U,t}^{u,(s)}$ represent the s-th particle of user u at time t, and $U_t^{u,(s)}$ is obtained by propagating the s-th particle $U_{t-1}^{u,(s)}$ using Eq (3.1). By using the Gaussian transition priors as the proposal distributions in the particle filtering, the weight at time t for all the particles is evaluated recursively as

$$w_t = w_{t-1} \cdot P(R_t|U_t, V_t, D_t), \tag{3.11}$$

where $P(R_t|U_t, V_t, D_t)$ is the observation function. The indicators for users, items and particles in it are omitted for clarity. The estimation of item i's latent factors at time t is obtained similarly. User and item latent factors are estimated alternatively. This method is named as PTBM for later reference.

Prediction In order to estimate a particle's weight for U_t^u in Eq (3.11), a weight of a particle for V_t^i is needed. Likewise, a weight of a particle is used for U_t^u to reweight a particle for item latent factors. As the computation of all possible pairs of user and item particles is too expensive, canonical particles [92] $\hat{U}_t^u$ and $\hat{V}_t^i$ are resorted to representing the total effect of particles on the estimation for user u's and item i's latent factors at time t, respectively.

In general, canonical particles $\hat{U}_t^u$ and $\hat{V}_t^i$ can be any proper function taking as input $\{(w_{U,t}^{u,s}, U_t^{u,s})|s \in 1 \dots S\}$ and $\{(w_{V,t}^{i,s'}, V_t^{i,s'})|s' \in 1 \dots S'\}$. In order to avoid the degeneracy problem [200] in particle filtering, particles will be resampled in proportional to their weights. After resampling, the expectations of posterior distributions of U_t and V_t are utilized as canonical particles, which are estimated as $\hat{U}_t^u = \sum_{n=1}^{S} w_{U,t}^{u,(n)} U_t^{u,(n)}$ and $\hat{V}_t^i = \sum_{m=1}^{S'} w_{V,t}^{i,(m)} V_t^{i,(m)}$ for user and item latent factors, respectively. After learning the interaction matrix D_t in next section, user u's preference over item i at time t can be estimated as $\hat{U}_t^u D_t (\hat{V}_t^i)^T$.

3.5 Personalized Self-Training Method

In practice, the user feedback is usually unavailable before the recommendation is made, which implies observation R_t at the current period is not available to estimate the tendency of user preferences and item attractiveness before recommendation. It is straightforward to use all the historical observations $R_{1:t-1}$ to approximate the estimation. However, the ratings would be dominated by the past information and cannot represent the recent tendency. An alternative approximation uses the most recent observation R_{t-1} instead. However, under temporal context, the ratings are too sparse for each user or item to track the current tendency. The data sparsity can be reduced by imputing all the missing data as shown in Eq (3.8). However, the dynamics in this approximation will drift away from the true tendency due to the domination of imputed ratings in $\bar{R}_{t-1}$. Meanwhile, this approximation does not have a satisfactory scalability due to the usage of all the missing data.

Therefore, the self-training principle [254] is exploited to solve the problems mentioned above. Instead of treating wAMAN, for each user at every time step, a subset of missing items is dynamically selected as negative samples to complement the user's most recent observation. This personalized and self-training procedure not only distinguishes the past and recent information but also avoids dominating recent observation with imputed data.

3.5.1 Self-training sampling method

Given user u and its current unobserved items $\mathcal{I}_t^{m,u}$, a set of $N_t^{n,u}$ items $\mathcal{I}_t^{n,u} \subseteq \mathcal{I}_t^{m,u}$ is selected by a multi-nominal distribution. The distribution is defined as follows,

$$P(x_1, \cdots, x_{N_t^{m,u}} | N_t^{n,u}, \theta_1, \ldots, \theta_{N_t^{m,u}}) \propto \theta_1^{x_1} \cdots \theta_{N_t^{m,u}}^{x_{N_t^{m,u}}}, \tag{3.12}$$

where $N_t^{m,u}$ is the number of unobserved items for user u until time t, $\{x_i | i \in \{1, \ldots, N_t^{m,u}\}\}$ represents the times that unobserved item i would be selected as negative, and $\{\theta_i | i \in \{1, \ldots, N_t^{m,u}\}\}$ is the probability that unobserved item i is disliked by user u. Without restricting x_i's to binary variables, this personalized selection is adaptive. An unseen item with a high probability will be selected more frequently than those with lower probability. As the accumulation of w_m for the same negative sample in Eq (3.8), more emphasis will be placed on the sample. By imposing such restriction, $N_t^{n,u}$ different items will be chosen. This restriction will be adopted in the model for ease of exposition.

Confidence estimation A candidate negative sample should have a small prediction error and a large estimation variance if the sample was negative. Given the historical data $R_{1:t-1}$, θ_i is defined as $P(\hat{r}_{u,i} = r_m \wedge var(\hat{r}_{u,i}) | R_{1:t-1})$ where $\hat{r}_{u,i} = r_m$ represents the event that the predicted rating equal to the imputed value and $var(\hat{r}_{u,i})$ represents the variance of prediction. Assuming prediction error and variance are conditionally independent given latent factors U_t and V_t, the probability θ_i can be derived as follows,

$$\begin{aligned}\theta_i &= \int P(\hat{r}_{u,i} = r_m | U_t, V_t, D_t) P(var(\hat{r}_{u,i}) | U_t, V_t, D_t) P(U_t, V_t | R_{1:t-1}, D_t) dU_t dV_t \\ &\sim \sum_{s=1}^{S} \sum_{s'=1}^{S'} w_{U,t}^{u,(s)} w_{V,t}^{i,(s')} P(\hat{r}_{u,i} | U_t^{u,(s)}, V_t^{i,(s')}, D_t) P(var(\hat{r}_{u,i}) | U_t^{u,(s)}, V_t^{i,(s')}, D_t),\end{aligned} \tag{3.13}$$

where the predicted joint distribution of latent factors is estimated using particle filtering described below, S and S' are the number of particles used to track user u's latent factors and item i's latent factors, respectively.

Prediction Combining with canonical particles $\hat{V}_t$ and Eq (3.13), the prediction distribution of user u's preference over item i is estimated as follows,

$$\theta_i \sim \sum_{s=1}^{S} w_{U,t}^{u,(s)} P(\hat{r}_{u,i} = r_m | U_t^{u,(s)}, \hat{V}_t^i, D_t) P(var(\hat{r}_{u,i}) | U_t^{u,(s)}, \hat{V}_t^i, D_t), \tag{3.14}$$

where $U_t^{u,(s)}$ are obtained by propagated $U_{t-1}^{u,(s)}$ as in Eq (3.1). A small distance between the imputed value and the predicted rating usually means a high confidence that the item should be negative. Thus, the probability of the estimated rating with respect to its true value is defined as follows,

$$P(\hat{r}_{u,i} = r_m | U_t^{u,(s)}, \hat{V}_t^i) = exp\{-|U_t^{u,(s)} D_t (\hat{V}_t^i)^T - r_m|\}. \tag{3.15}$$

In terms of variance estimation, the estimated rating with most uncertainty is selected. The probability of prediction variance can be estimated as follows,

$$P(var(\hat{r}_t^{u,i} | U_t^{u,(1)}, \ldots, U_t^{u,(S)}, \hat{V}_t^i, D_t)) = exp\{\frac{1}{S-1} \sum_{s=1}^{S} (U_t^{u,(s)} D_t \hat{V}_t^i - \hat{U}_t^u D_t \hat{V}_t^i)\}. \tag{3.16}$$

3.5.2 Two-Phase Self-Training Method

Considering the large size and high sparsity of user-item preference matrix, the previous sampling scheme using all the unobserved items is infeasible in practice. A two-phase approach is utilized to reduce computational complexity.

In phase I, for each user u, a subset $\mathcal{I}'^{n,u}_t$ of unobserved items $\mathcal{I}_t^{m,u}$ is sampled. Generally, this sampling scheme can be implemented in terms of any distribution that properly represents NMAR. It is shown [212] that arbitrary data missing mechanism is NMAR as long as missing data happen with a higher probability than relevant ratings do. A uniform distribution of $\mathcal{I}_t^{m,u}$ is utilized to avoid interfering prediction distribution, which has been extensively used to handle some large datasets [249, 191]. For simplicity, it is set that $|\mathcal{I}'^{n,u}_t| = 2 * N_t^{n,u}$.

In phase II, the personalized probability θ_i will be computed only for candidates $\mathcal{I}'^{n,u}_t$. Based on Eq (3.12), negative samples will be selected and then combined with the observed

data R_{t-1} to construct a sparsity reduced data $\bar{R}_t$ at time t. User and item latent factors are thus tracked by using this dynamically constructed data. The PFUV method equipped with this two-phase self-training method is named as ST-PFUV hereafter, and the PTBM method equipped with this self-training scheme is named as ST-PTBM for later reference.

3.6 Learning algorithm

In contrast to user and item latent factors in ST-PTBM, the interaction matrix D_t is treated as model parameters and the EM framework [35] is applied to update its value dynamically. Let $\bar{D}_t$ denote the learned value of the matrix from the previous iteration of the EM algorithm at time frame t. The expectation of log-complete likelihood $\log P(R_t, U_t, V_t | U_{t-1}, V_{t-1}, D_t)$, with respect to the marginal posterior distribution $P(U_t, V_t | U_{t-1}, V_{t-1}, \bar{D}_t, R_t)$ over latent variables U_t and V_t, is derived as follows,

$$\begin{aligned} E &= \int P(U_t, V_t | \bar{D}_t, \bar{R}_{1:t}) \cdot \log P(\bar{R}_{1:t}, U_t, V_t | D_t) \mathrm{d}U_t \mathrm{d}V_t \\ &= \int P(U_t, V_t | U_{t-1}, V_{t-1}, \bar{D}_t, \bar{R}_t) \cdot \\ &\quad (\log P(\bar{R}_t | U_t, V_t, D_t) + \log P(U_t) + \log P(V_t)) \mathrm{d}U_t \mathrm{d}V_t. \end{aligned} \tag{3.17}$$

The prior $P(\cdot) = exp(-0.5 * \lambda || \cdot ||^2_{\mathrm{Frob}})$ functions as Frobenius norms for regularization. At each time frame, on the E-step, the algorithm will reweight each sample that represents the hypothesis over the structure of user and item latent space. On the M-step, the particles will be resampled and D_t will be reestimated to adjust the weights on different latent dimensions dynamically and prevent particle filtering from divergence.

3.6.1 E-step

On the E-step of the algorithm, the posterior distribution of latent factors U_t and V_t is defined as follows,

$$\prod_{u=1}^{M} \prod_{v=1}^{N} P(U_t^u, V_t^i | \bar{D}_t, \bar{R}_t). \tag{3.18}$$

3. Tracking the Tendency of User Preferences and Item Attractiveness

By alternatively estimating user and item latent factors, each joint posterior distribution in Eq (3.18) is approximated as the product of marginal posterior distributions $P(U_t^u|\bar{D}_t, \bar{R}_t)$ and $P(V_t^i|\bar{D}_t, \bar{R}_t)$. For user u, its marginal posterior distribution can be further estimated as follows,

$$\begin{aligned}
& P(U_t^u|\bar{D}_t, \bar{R}_{1:t}) \\
\propto & \int P(\bar{R}_t|U_t^u, V_t^v, \bar{D}_t)P(U_t^u|U_{t-1}^u)P(V_t^i)P(U_{t-1}^u|\bar{R}_{1:t-1}, \bar{D}_t)\mathrm{d}V_t^i\mathrm{d}U_{t-1}^u \\
\approx & \int P(\bar{R}_t|U_t^u, V_t^v, \bar{D}_t)P(U_t^u|U_{t-1}^u)P(\bar{V}_t^i)P(U_{t-1}^u|\bar{R}_{1:t-1}, \bar{D}_t)\mathrm{d}U_{t-1}^u \\
\approx & P(\bar{R}_t|U_t^u, \bar{V}_t^i, \bar{D}_t)P(U_t^u|\bar{U}_{t-1}^u)P(\bar{V}_t^i), \qquad (3.19)
\end{aligned}$$

where $\bar{U}_t^u$ and $\bar{V}_t^i$ represent the estimated values from the previous iteration for user u and item i, respectively. Analogously, the marginal posterior distribution over V_t^i can be derived.

According to the above derivation, these marginal posterior distributions can be estimated using particle filtering and canonical particles as described in previous sections. As mentioned in Section 3.4.1, canonical particles $\hat{U}_t$ and $\hat{V}_t$ are used to represent the estimated latent factors for users and items at time t. Therefore, the expectation of log-complete likelihood in Eq (3.17) can be approximated as,

$$\begin{aligned}
E \approx & \int \prod_u (\sum_{s=1}^{S} w_{U,t}^{u,(s)} \delta(U_t^u - U_t^{u,(s)})) \cdot \prod_i (\sum_{s'=1}^{S'} w_{V,t}^{i,(s')} \delta(V_t^i - V_t^{i,(s')})) \cdot \\
& (\log P(R_t|U_t, V_t, D_t) + \log P(U_t) + \log P(U_t)))\mathrm{d}U_t\mathrm{d}V_t \\
\approx & \log P(R_t|\hat{U}_t, \hat{V}_t, D_t) + \log P(\hat{U}_t) + \log P(\hat{V}_t) + \mathcal{C} \\
\approx & \log P(\hat{R}_t|\hat{U}_t, \hat{V}_t, D_t) + \log P(\hat{U}_t) + \log P(\hat{V}_t) + \mathcal{C}, \qquad (3.20)
\end{aligned}$$

where $\mathcal{C}$ is a constant representing all the terms irrelevant to model parameters D_t, and the observation function is given in the form of Eq (3.22). The second approximation in Eq (3.20) results from the substitution of canonical particles of user and item latent factors as shown in Section 3.4.1.

Intuitively, Eq (3.20) draws only one sample of U_t and one sample of V_t from the posterior distribution $P(U_t, V_t|D_t, \hat{R}_t)$, and these two samples happen to be the canonical

representation of user and item latent factors. Instead of treating all the missing data as negative, the approximation in Eq (3.20) adopts a two-phase sampling mechanism to construct a set of ratings dynamically as the current training data. A subset of missing items is selected for each user at every time step as negative samples to complement the user's most recent observation. The dynamically constructed observations are denoted as $\hat{R}_t$ at time t. This personalized approach not only distinguishes the past and recent information but also avoids dominating recent observation with imputed data. In addition, although ST-PTMB is discussed only on explicit feedback, this sampling scheme enables the model to be a general one that could be applied to (much denser) implicit feedback without increasing the computational complexity.

3.6.2 M-step

After gathering sufficient statistics of model parameters with respect to the posterior distributions of user and item latent factors, D_t is re-estimated at each M-step to maximize the probability of generating the ratings in the training data. Optimization methods, such as stochastic gradient ascent, tend to fit the model to the imputed value as accurate as possible. As a unique imputed value is set to those selected negative samples, this optimization approach can easily overfit the model and make the predicted user preferences over unrated items indistinguishable. Experimental results (not shown in this chapter) also verify this statement. Therefore, on the M-step, a separate particle filtering is utilized to update D_t dynamically at every time step.

Let $diag^{-1}(D_t)$ denote the diagonal elements of matrix D_t. Similar to the transition functions in Section 3.4.1, the dynamics of $diag^{-1}(D_t)$ is assumed to follow a first-order random walk driven by multivariate normal noise,

$$diag^{-1}(D_t) = diag^{-1}(D_{t-1}) + e_t, \tag{3.21}$$

where $e_t \sim \mathcal{N}(0, \sigma_D I)$ is a Gaussian process noise that is unrelated to c_t^u and d_t^i.

The observation function in the particle filtering to track $diag^{-1}(D_t)$ is,

$$P(\hat{R}_t|\hat{U}_t,\hat{V}_t,D_t)=\sum_{s''=1}^{S''} exp\{-\sum_{u,i}(W_{u,i}(\bar{r}_t^{u,i}-\hat{U}_t^u D_t^{(s'')}(\hat{V}_t^i)^T)^2+\alpha||D_t||^2_{\text{Frob}})\}, \quad (3.22)$$

where S'' is the number of particles and α is the regularization coefficient to prevent overfitting. This observation function takes as input the whole set of dynamically constructed samples. Analogous to the definition in Section 3.4.1, the canonical particle for D_t is defined as $\hat{D}_t=\sum_{s''=1}^{S''} w_t^{(s'')} D_t^{(s'')}$.

Because the posterior distributions about the weighting matrix and user and item latent factors are not Gaussian after the introduction of w_m and the imputed value, some approximations have been exploited during the derivation of E-step and M-step in the learning algorithm. The developed EM-based method may not always maximize the likelihood function during iterations. The particle filtering working as an optimization method to search the latent factor space is also used here to alleviate this issue. This optimization method is also suitable for approximating the multimodal distribution. Besides, a properly designed convergence condition also helps the learning procedure achieve a balance between exploitation and exploration. For example, the improvement with respect to the measurement metric on current observations, such as the Top-N recall metric, could be directly used as the convergence condition. Considering the multimodal property in temporal RSs and the local convergence of the EM framework, the developed EM-based method in ST-PTMF is expected to perform well in practice. The experimental results shown below also empirically confirm this expectation.

3.7 Pseudo-code and Complexity

The proposed methods are summarized in Algorithm 1. When selecting the most confident negative samples to build the current training data, the second phase of the sampling scheme in Section 3.5.2 is extended by integrating D_t into it. This modification is not shown in Algorithm 1 for clarity. For easy comparison with baseline methods in Section 3.8, ST-PTBM is initialized with $D_0 = I$ in Algorithm 1.

Each user or item's latent vectors is tracked separately by a particle filter, and every particle in D_t is loosely coupled to other particles. Therefore, these sequential samplers can be easily speeded up via sampling from these distributions in parallel. As shown in Table 3.6, the speedup should be significant, especially when either the number of users or the number of items are large.

3.7.1 Computational Complexity

Recall that user and item latent factors are $U_t \in \mathcal{R}^{N\times K}$ and $V_t \in \mathcal{R}^{M\times K}$ respectively, where $K \ll min(N, M)$. It is set $S = S' = S''$ and $N_t^{n,u} = \hat{N}$ for simplicity.

The time complexity of *personalized* prediction in the proposed methods is $O(KM)$. For computational complexity, at each time step, the PFUV method takes $O(KSMN)$. As the sampling size $\hat{N}$ is usually comparable with $K \ll min(N, M)$, the ST-PFUV method takes $O(KS(M + N)\hat{N}) \approx O(K^2S(M + N))$, which scales efficiently as a linear function of the size of users and items.

The usage of the extra diagonal matrix $D_t \in \mathcal{R}^{K\times K}$ will not change the complexity of the sampling scheme. It takes $O(KS\hat{N})$ to compute the probability $P(\hat{r}_t^{u,i}|\{U_t^{(s),u}\}, \hat{V}_t^i, D_t)$ for item i, and takes the same complexity to compute the probability $P(\hat{r}_t^{u,i}|\{V_t^{(s'),u}\}, \hat{U}_t^u, D_t)$ for user u. The time complexity for updating D_t is $O(S\hat{N}K^2)$. The time complexity of ST-PTBM at each time frame is $O(KS((K + N + M)\hat{N})) \approx O(K^2S(N + M))$, which is comparable to $O(KT_{iter}(N + M))$ of the gradient descent in PMF at each time frame where the number of iterations T_{iter} usually has the same magnitude as S.

3.8 Experiment

The performance of developed methods is tested on the Movielens 100K [1], Hetrec [3] and Netflix datasets [108]. The MovieLens dataset spans 32 weeks with integer rating scale from 1 to 5 while the HetRec dataset spans 12 years with half mark rating scale from 1 to

Input: R_0, ratings before time 0, the sequence of ratings $\{R_1, \ldots, R_T\}$ to present, and the number of missing data to be selected, N_t^{neg}

Output: user latent factors $\{U_t | t \in [1, \ldots, T]\}$, item latent factors $\{V_t | t \in [1, \ldots, T]\}$ and the weighting matrices $\{D_t | t \in [1, \ldots, T]\}$

`/* initialization */`

calculate the initial values of U_0 and V_0 with R_0, and set $D_0 = I$.

foreach *user* u **do**

  for $s \in [1, \ldots, S]$ **do**

    $U_0^{u,(s)} \leftarrow U_0^u$

  end

end

setup particles for item latent factors and the weighting matrix as above.

`/* algorithm */`

while $t \in [1, \ldots, T]$ **do**

  `/* stage of dynamically building the current training data */`

  select N_t^{neg} negative samples for each user and item as in shown in Section 3.5.2, and combine them with R_t to generate the current training data $\hat{R}_t$;

  while *not converged* **do**

    `/* E-stage */`

    `/* stage of tracking user u's latent factors */`

    foreach *user* u **do**

      reweight each particle using $\hat{V}_{t-1}$, Eq (3.11), Eq (3.1) and Eq (3.8) ;

      resampling [200] ;

      calculate canonical particle $\hat{U}_t^u$;

    end

    `/* stage of tracking item i's latent factors */`

    foreach *item* i **do**

      reweight each particle using $\hat{U}_t$, Eq (3.11), Eq (3.2) and the observation function of item i similar to Eq (3.8) ;

      resampling ;

      calculate canonical particle $\hat{V}_t^i$;

    end

    `/* M-stage */`

    `/* stage of learning D */`

    reweight each particle using $\hat{U}_t$, $\hat{V}_t$, Eq (3.11), Eq (3.21) and Eq (3.22) ;

    resampling ;

    calculate canonical particle $\hat{D}_t$;

  end

  `/*prediction*/`

  predict user u's preferences over all the items by $\hat{U}_t^u \hat{D}_t \hat{V}_t$

end

Algorithm 1: The Pseudo-code of Probabilistic temporal bilinear model. The particle filtering for MF (PFUV) method can be reduced from this method.

5. These two datasets are selected to test the performance of the proposed methods for short and long periods of time, respectively. A Netflix dataset is also tested to verify the performance of proposed method on a reasonably large dataset. This sampled version of Netflix dataset spans 27 months.

3.8.1 Protocol

In the experiments, ratings are grouped on the basis of the time frame in which their timestamps are. Ratings before a predefined time instance t_{test} are used as training data, and ratings after it are test data. This setting is preferred over a random split over all the data. As in a real-world deployment, it is infeasible to generate prediction utilizing information in the future. The training periods for MovieLens 100K, HetRec and Netflix are September $\sim$ December 1997, September 1997 $\sim$ December 2007 and the 1-st $\sim$ 15-th months, respectively. Their testing periods are the 1-st $\sim$ 16-th weeks in 1998, January $\sim$ December in 2008 and the 16-th $\sim$ 27-th months, respectively. Different units of time frame are selected to ensure that ratings for each user in a time slot are not too sparse. Based on this setup, MovieLens 100K contains 530 users and 1493 items with the overall sparsity of 93.3% and the sparsity 99.96% in the last frame and HetRec contains 1775 users and 9228 items with the overall sparsity of 95.1% and the sparsity of 99.98% in the last frame. Netflix contains $480,189$ users and $17,770$ items with the overall sparsity 98.84%. A sampled version of Netflix is used. It has the sparsity of 99.98% in the last frame, which is extracted from 4% of the Netflix dataset and 20% users and 20% movies were randomly selected from the whole pool. All the algorithms are repeated 10 times and the means and standard deviations of those results are reported.

Temporal accuracy metrics The metrics precision@k and recall@k [66] are utilized to measure recommendation relevance, which directly assesses the quality of the Top-N recommendations. In order to measure user satisfaction in recommendations, the Top-N hitrate [66, 212] is utilized. As the Top-N hitrate recall is proportional to the Top-N hitrate precision [212], the Top-N hitrate recall is adopted and it is named as top@k for

later reference. In order to test the temporal performance, the temporal extensions of these accuracy metrics are defined. These conventional accuracy metrics adopted to RSs can be found in [66, 115].

For user u, let $prec(k, u, t)$ denote precision@k in month t. During the prediction at time $t-1$, instead of using all the items in testing data as conventional precision@k does, the temporal metrics *only* scan items in month t to determine their relevance to a user. The temporal precision is defined as follows,

$$prec_{temp}(k) = \frac{1}{T * N} \sum_{t=1}^{T} \sum_{u=1}^{N} prec(k, u, t), \tag{3.23}$$

which is the average on all users over all the time frames T. The temporal recall $recall_{temp}(k)$ and temporal hitrate $top_{temp}(k)$ are defined analogously. As a common practice, items with maximum rating value are treated as relevant items, and measure $k = 10$ for precision. For hitrate, it is set that $k = 10$ and each relevant item is mixed with 500 randomly sampled unobserved items to avoid spending too much computational power on evaluation. It is set that $k = 100$ for recall because all the items are being ranked in the temporal context. All the model parameters are tuned under temporal recall and use the identical setting to test the performance of algorithms under temporal precision and hitrate.

Baseline methods In order to test the performance of the proposed methods that enforce and track the temporal and dynamic importance between user preferences and item attractiveness, they are compared with the following five algorithms as part of baseline methods: STKF [158], ST-PTBM-One, TopPopular, PureSVD [66], and AllRank [212].

The method in [158] is implemented to verify the advantages of handling non-Gaussian behaviors and tracking the tendency of item popularity, which dynamically tracks user latent factors based on both spatial and temporal information. The implemented method is named as STKF for later reference. As ST-PTBM method does not rely on side information, users' rating history is utilized to compute the spatial priors for STKF for a fair comparison. It is shown [191] that user-oriented sampling has better performance than uniform sampling, it is, therefore, hybrid with ST-PFUV method, and it is named

as ST-PFUV-User for later reference. In order to demonstrate the efficiency of the developed learning algorithm, the number of iterations is also fixed in the learning algorithm of ST-PTBM to be 1 and this method is named as ST-PTBM-One thereafter.

The above baseline methods are dynamic methods to RSs. To the best of our knowledge, most of the developed algorithms in temporal RSs are compared with static versions of some baseline algorithms. In the following experiments, some static algorithms are introduced and these algorithms are made dynamic. By retraining all these static baseline methods at each time step with all the data up to the time step, as a common practice in a real-world deployment, they are enhanced by exploiting temporal information to make the comparison fair. TopPopular is a non-personalized algorithm that ranks item higher when the more times that the item is rated as relevant. It is chosen to verify that it is beneficial to consider personalized recommendation in a temporal context. PureSVD and AllRank are state-of-the-art algorithms developed to pursuit the Top-N recommendation task. They are selected to illustrate the ability of the developed methods to cope with non-Gaussian behaviors. These dynamic extensions are named as DynTopPopular, DynPureSVD and DynAllRank, respectively. In order to confirm the necessity of exploiting temporal information, PMF is also adopted as the only static method in the experiments, which always predicts the ranking without updating model parameters.

Meanwhile, it is set that $S = S' = S'' = 1000$ and $K = 8$ for all the Monte Carlo-based methods to balance their accuracy, variance and scalability. The imputed value is set to $r_m = 1.95$ which is a lower value than the average observed ratings in all the datasets. All the Monte Carlo-based methods are initialized by AllRank, and STKF is initialized by PMF with spatial priors only. All the parameters in the experiments are tuned by cross-validation method [115]. All other parameters in sampling methods and all the parameters (including the number of latent factors) in other baseline methods are separately tuned to achieve their best performance. The temporal recall metric is used during tuning and the identical settings are applied to test temporal precision and hitrate metrics. It is shown [243] that the effectiveness of RSs with sparse datasets is usually not high, especially the training and testing data are organized in chronological order instead of being randomly

split. Thus, we focus on comparing the relative performance of algorithms instead of comparing their absolute performance. For clarity, some insight into the results will be discussed on the Hetrec and Netflix datasets only. The analysis also applies to the results on the Movielens dataset.

3.8.2 MovieLens 100K

It is set that weight $w_m = 0.05$, regularization constant $\lambda = 0.05$ and $K = 12$ for DynAllRank, $K = 10$ for DynPureSVD and PMF, and $K = 20$ for STKF. It is also set that $\sigma = 0.05$ to reduce variance in particle filtering. The number of selected negative samples in the personalized sampling mechanism is set to be 30 times the number of users and items in ST-PFUV and ST-PTBM. It is also set that $\sigma_D = 0.4$ and $\alpha = 0.4$ for ST-PTBM. There is no need to tune these parameters simultaneously. The parameters for AllRank can be tuned at first, and then the parameters for particle filtering can be selected. The parameters for learning D_t can be determined at last. As the developed models are incrementally constructed, this incremental approach to parameter tuning works well in experiments in this chapter and is not prone to the overfitting of the model.

Results Table 3.1 shows the results of the methods under temporal accuracy metrics as defined in Section 3.8.1. Due to the fact that few relevant items exist for each user in a time frame, low values in the table are expected. Compared with these baseline methods, ST-PTBM has the best performance. Except for the improvement on the ST-PTBM-One method on the precision metric, all the improvements brought by ST-PTBM are statistically significant under both paired and unpaired t-tests [115] with $p = 0.05$. This result verifies that the ability of the proposed method to improve the temporal performance of RSs. This can be ascribed to the fine modeling and learning of temporal dynamics of user preferences and item characteristics in the proposed method.

The ST-PTBM, ST-PTBM-One and ST-PFUV methods outperform all other dynamic algorithms. This result shows that our proposed algorithms can effectively extract the

Method	$prec_{temp}(k)$	$recall_{temp}(k)$	$top_{temp}(k)$
PMF	0.0445±0.0058	0.2479±0.0282	0.1425±0.1117
DynTopPopular	0.0465±0.0000	0.2910±0.0000	0.1782±0.0010
DynPureSVD	0.0233±0.0000	0.3410±0.0000	0.2806±0.0047
DynAllRank	0.0546±0.0059	0.4140±0.0232	0.3162±0.0168
STKF	0.0632±0.0000	0.3040±0.0000	0.1725±0.0046
ST-PFUV-User	0.0458±0.0024	0.3630±0.0094	0.2339±0.0096
ST-PFUV	0.0613±0.0022	0.4403±0.0095	0.3543±0.0095
ST-PTBM-One	0.0646±0.0012	0.4617±0.0156	0.3652±0.0156
ST-PTBM	*0.0651±0.0028*	*0.4843*±0.0112	*0.3758±0.0138*

Table 3.1: Temporal accuracy of methods on the MovieLens dataset under temporal metrics. The best performance is in italic font.

Method	$prec_{temp}(k)$	$recall_{temp}(k)$	$top_{temp}(k)$
PFUV-Rect-wAMAN	0.0423±0.0032	0.3310±0.0169	0.2400±0.0123
PFUV-Hist-User	*0.0447*±0.0040	*0.3648*±0.0141	*0.2439*±0.0106
DynAllRank-User	0.0164±0.0043	0.2358±0.0150	0.1151±0.0184

Table 3.2: Effects of recent and historical data on the MovieLens dataset under temporal metrics. The best performance is in italic font.

important temporal and dynamic information of user preferences and item characteristics, and benefit from the finely modeling among latent factors and the dynamic adjustment of the significance on different latent dimensions. The developed algorithms successfully exploit this vital information and fine structures, even if the dynamics modeling in tracking follows a first order random walk. The ST-PTBM method improves ST-PFUV method over the temporal accuracy measurement by 6.20%, 9.99% and 6.07% as shown in Figure 3.3, respectively. Recall that the developed methods are tuned based on the recall metric during training and the identical settings are applied to precision and Top-N hitrate measurement. Moreover, the ST-PTBM method improves ST-PTBM-One over the temporal recall metric by 4.89%. This improvement confirms the effectiveness of the developed learning algorithm which dynamically learns the interaction matrix D_t and updates user and item latent factors U_t and V_t.

Meanwhile, it is obvious from Table 3.1 that all personalized and dynamic algorithms have better performance than that of the static PMF method under all the temporal metrics. Static PureSVD and static AllRank methods have similar performance and are omitted here. This conclusion is expected because more and more information about

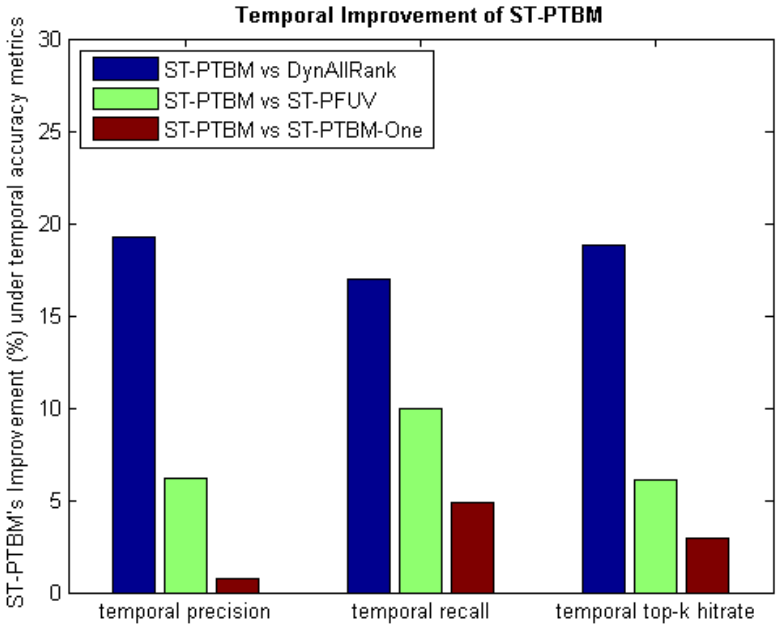


Figure 3.3: The improvement of ST-PTBM over baseline methods on the MovieLens dataset under temporal accuracy metrics.

users is gathered as the time passes, which also demonstrates the importance of temporal dynamic modeling in RSs.

The usage of ratings In order to further illustrate the power of self-training and the importance of distinguishing recent and historical ratings, the PFUV method is tested with two extra settings, where ratings consist of (1) recent observation and wAMAN, (2) all the historical data and missing data sampled by the user-oriented scheme. They are named as PFUV-Rect-wAMAN and PFUV-Hist-User for later reference. Not to discriminate against the baseline methods, the best baseline method DynAllRank is also extended by replacing wAMAN with user-oriented sampling, and name it DynAllRank-User for later reference. Table 3.2 shows the performance of these methods, where the PFUV-based methods outperform the best baseline method with PFUV-Hist-User the best. Combining with those in Table 3.1, results in Table 3.2 further confirm the ability of the developed methods to balance information intrinsic in recent observations and the sparsity reduction introduced by imputation to better incorporate temporal and dynamic information.

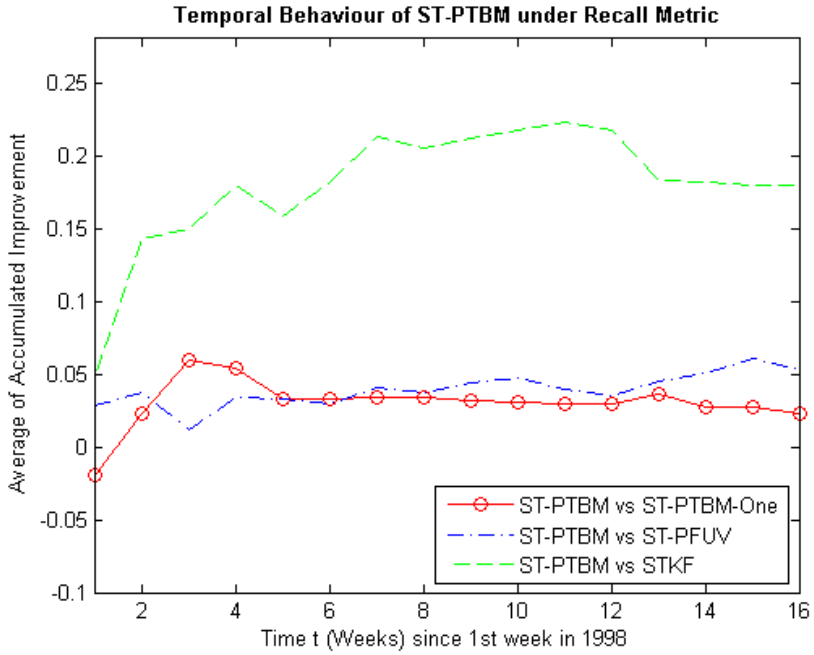


Figure 3.4: The average of accumulated improvement for ST-PTBM on the MovieLens dataset since the 1-st week in 1998.

Temporal behaviors In order to further evaluate temporal behaviors of the proposed methods, the average of accumulated improvement (AAI) over time is defined. Let the performance of any two methods under temporal recall in month t be $Rec_1(t)$ and $Rec_2(t)$, respectively. The AAI in month t_1 is defined as follows,

$$\frac{1}{t_1}\sum_{t=1}^{t_1}(Rec_1(t) - Rec_2(t)). \tag{3.24}$$

Figure 3.4 plots the AAI metric among ST-PTBM, ST-PTBM-One, ST-PFUV and STKF methods. Except in the first month of the red (circle) curve (ST-PTBM vs ST-PTBM-One), all the curves are above zero, showing that the developed method constantly outperforms baseline methods by taking advantage of the fine modeling of temporal dynamics in user preferences and item attractiveness. Meanwhile, compared with ST-PTBM-One, the tendency of the circle curve demonstrates that the developed learning algorithm is more effective at exploiting the underlying temporal patterns. The blue (dash-dotted) curve shows that ST-PTBM constantly improves ST-PFUV by finely modeling and adaptively adjusting the temporal importance among user preferences and item popularity. The green

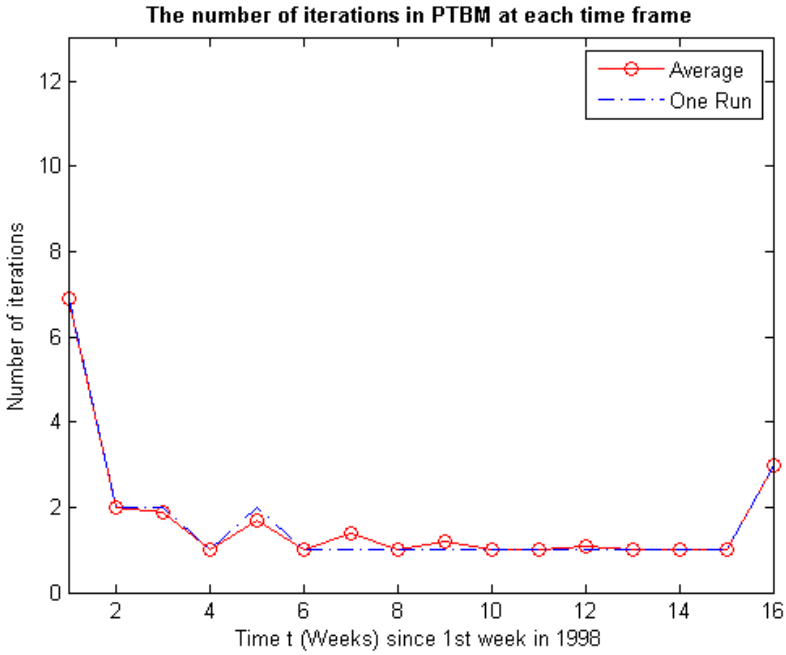


Figure 3.5: The number of iterations in the learning algorithm at each time frame on the MovieLens dataset.

(dash) curve (ST-PTBM vs STKF) shows that the temporal performance can be significantly improved by tracking the tendency of item attractiveness and coping with non-Gaussian behaviors. While baseline methods require longer training period, ST-PTBM performs well even if the training period is short (within 5 time frames) and accumulated amount of ratings is relatively few. Figure 3.5 illustrates a sequence of the average number of iterations in EM under temporal recall metric and one run of the sequence. It shows the robustness of the proposed learning algorithm under the sampling schemes.

3.8.3 HetRec

This experiment focuses on the study of temporal recommendation performance of proposed methods over a longer period. The ST-PTBM, ST-PTBM-One and ST-PFUV methods are compared with the best baseline method, DynAllRank, and the static PMF method. The performance of PMF, DynAllRank, STKF is also included to demonstrate the benefits of personalized and temporal recommendation conducted by our methods.

Similar to previous experiments, ST-PTBM has the best performance under all the temporal metrics as shown below. For PMF, it is set that $K = 40$, the step size for SGD as 0.005, $\lambda = 0.05$, and the weight $w_m = 0.08$. It is set $\sigma_U = \sigma_V = 0.05$ for ST-PTBM and ST-PTBM-One and set $\sigma_U = \sigma_V = 0.1$ for ST-PFUV. The factor of negative samples selection for users and items is set to be 30 in ST-PTBM and 50 in ST-PFUV. It is also set $\sigma_D = 0.1$ and $\alpha = 0.4$ for ST-PTBM. As mentioned before, these parameters are incrementally tuned.

Method	$prec_{temp}(k)$	$recall_{temp}(k)$	$top_{temp}(k)$
PMF	0.0054±0.0012	0.1629±0.0116	0.2577±0.0082
DynAllRank	0.0074±0.0003	0.2263±0.0093	0.4416±0.0078
STKF	0.0110±0.0000	0.1679±0.0000	0.3168±0.0000
ST-PFUV	0.0130±0.0004	0.3036±0.0161	0.5143±0.0042
ST-PTBM-One	0.0126±0.0003	0.3087±0.0040	0.5285±0.0065
ST-PTBM	*0.0149±0.0008*	*0.3325±0.0027*	*0.5392±0.0050*

Table 3.3: Results on the HetRec dataset under temporal metrics. The best performance is in italic font. $\hat{N} = 50$ for ST-PFUV and ST-PFUV-User, and $\hat{N} = 30$ for ST-PTBM and ST-PTBM-One.

$\hat{N}$	10	20	30	40	50	60	70
ST-PFUV-User	0.1754	0.1882	0.1977	0.2026	0.2020	0.2071	0.2041
ST-PFUV	0.2536	0.2826	0.2980	0.3016	0.3036	0.3013	0.3032
ST-PTBM-One	*0.3085*	0.3102	0.3087	0.3088	0.3052	0.3007	0.3033
ST-PTBM	0.3070	*0.3320*	*0.3338*	*0.3323*	*0.3252*	*0.3276*	*0.3245*

Table 3.4: Effect of different sizes of samples on the HetRec dataset under temporal metrics. The best performance is in italic font.

Results Table 3.3 illustrates the results of these methods under temporal metrics. The ST-PTBM method significantly outperforms other methods in terms of temporal accuracy. Particularly, it improves the temporal measurement over that of ST-PFUV by 14.6%, 9.52% and 4.84%, respectively. Meanwhile, it improves the temporal measurement over that of PTBM-One by 18.3%, 7.7% and 2.02%, respectively. Recall that we focus on the improvement instead of the relative values. All the improvements brought by ST-PTBM method are statistically significant under both paired and unpaired t-tests with $p = 0.05$. Compared with results on MovieLens 100K, ST-PTBM has a much greater improvement over baseline methods. This result verifies that ST-PTBM has a much better ability to

exploit temporal and dynamic information, especially over a longer period. Meanwhile, the performance of ST-PTBM-one is only comparable with that of ST-PFUV method. Table 3.4 shows the results for different sizes of negative samples, $\hat{N}$. The key point observed from the table should be the trends in the results, which are introduced by the various number of negative samples adopted. Meanwhile, the standard deviations of the results of ST-PFUV methods are 0.0202, 0.0212, 0.0182, 0.0161, 0.0133, 0.0087, 0.0089 for those various numbers of negative samples. The standard deviations of other methods have similar trends and are omitted in Table 3.4 for clarity. Up to $\hat{N} = 30$, ST-PFTBM method is constantly being improved as the effect of sparsity reduction. A larger number of samples will clutter the built observations with negative samples.

Temporal behaviors Figure 3.6 plots the AAI among ST-PTBM, ST-PTBM-One, ST-PFUV and PMF methods. All the curves in the figure are above zero, showing that our method constantly outperforms baseline methods over time. Intuitively speaking, the learning algorithm of ST-PTBM provides particle filtering with more opportunities to search the latent vector space. The search will terminate when a balance between exploitation and exploration in the latent space is reached. By contrast, ST-PTBM-One only searches the latent space once, and it is prone to the divergence of particle filtering over a long period.

Note that as $D_0 = I$ is set at the initialization, ST-PTBM, ST-PTBM-One and ST-PFUV have the identical initial conditions learned by the AllRank method at time 0. After doing some exploratory analysis on the Hetrec dataset, it is found that the tendency of the blue (dotted dash) curve (ST-PTBM vs ST-PFUV) and the red (circle) curve (ST-PTBM vs ST-PTBM-One) is due to the rapidly decreasing of active users in the dataset over time (only 16.64% active users remained at the final time frame) and the accidentally *retrospective* learning. As there is less new information available at each time frame as the time passes, all these methods are overinflunced by the historical data and they are attempting to extract patterns relating to the temporal importance of the historical data repeatedly. Compared with other methods, ST-PTBM always fully exploited the temporal dynamics

at each time frame to improve the temporal performance of RSs. Therefore, under such a situation, this *retrospective* improvement made by ST-PTBM over ST-PTBM-One and ST-PFUV has been reduced. The almost linear tendency of the green (dash) curve (ST-PTBM vs PMF) also supports this statement because the AAI curve shows that PTBM is still producing improvement across consecutive time frame. Meanwhile, the experimental results on Movielens 100K and Netflix datasets show that even if the dataset is very sparse (Netflix 98.84% overall vs Hetrec 95.1% overall), ST-PTBM can still perform much better than other methods.

Figure 3.7 illustrates a sequence of the average number of iterations in the learning algorithm under temporal recall metric and one run of the sequence. It shows the robustness of the proposed learning algorithm under two-phase sampling scheme and the sequential Monte Carlo estimation. Compared with Figure 3.4 and Figure 3.5, the training period of ST-PTBM is much shorter (within 2 time frames), showing ST-PTMB is more capable of exploiting temporal and dynamic information, which may also be due to more ratings that are accumulated for each user in a time frame.

3.8.4 Netflix

In this subsection, the focus is placed on the temporal recommendation performance of ST-PTBM on a reasonably large dataset. Similar to previous experiments, ST-PTBM and ST-PTBM-One are compared with the best baseline method ST-PFUV. Analogously, the performance of PMF, DynAllRank, STKF methods is also included to demonstrate the benefits of personalized and temporal recommendation conducted by the developed methods in this chapter. The ST-PTBM method also achieves the best performance under all the temporal metrics as shown below. For PMF and DynAllRank methods, they are set to $K = 20$, $\lambda = 0.1$. For ST-PFUV, ST-PTBM-One and ST-PTBM methods, they are set $\sigma = 0.1$ and the weight $w_m = 0.02$. The number of selected negative samples is set to be 30. It is also set $\sigma_D = 0.1$ and $\alpha = 0.4$ for ST-PTBM method. These parameters are also incrementally tuned.

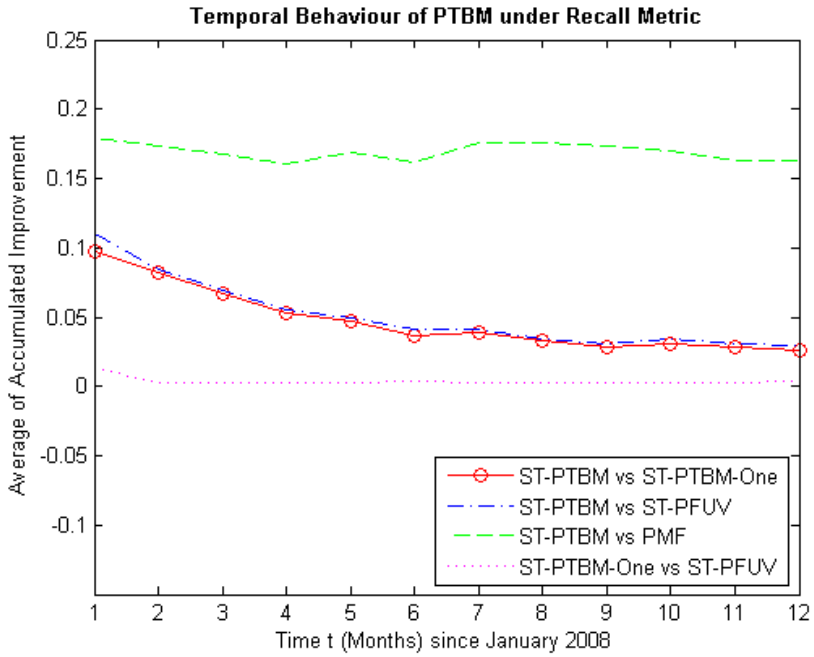


Figure 3.6: The average of accumulated improvement for ST-PTBM on the Hetrec dataset since January 2008.

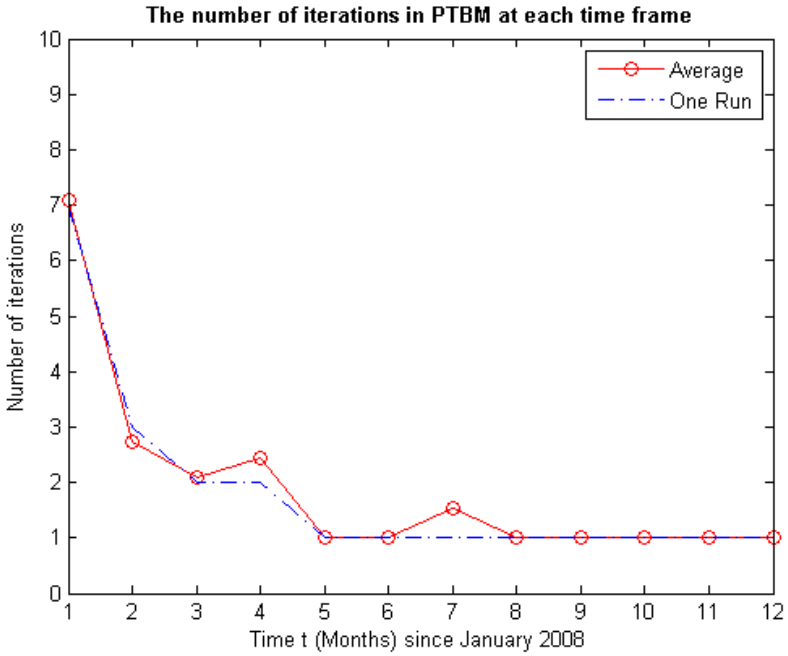


Figure 3.7: The number of iterations in the learning algorithm at each time frame on the Hetrec dataset.

Method	$prec_{temp}(k)$	$recall_{temp}(k)$	$top_{temp}(k)$
PMF	0.0080±0.0000	0.3339±0.0009	0.3011±0.0025
DynAllRank	0.0226±0.0002	0.5050±0.0011	0.4799±0.0012
STKF	0.0087±0.0000	0.3518±0.0000	0.3220±0.0000
ST-PFUV	0.0243±0.0001	0.5351±0.0008	0.5000±0.0004
ST-PTBM-One	0.0250±0.0003	0.5370±0.0012	0.5025±0.0008
ST-PTBM	*0.0292±0.0006*	*0.5752±0.0030*	*0.5215±0.0024*

Table 3.5: Results of methods on the Netflix dataset under temporal metrics. The best performance is in italic font.

Results Table 3.5 shows the results of the methods under temporal accuracy metrics. As mentioned before, the focus is placed on the relative improvement of these methods due to the existence of few relevant items for each user in a time frame. All the improvements brought by the ST-PTBM method are statistically significant under both paired and unpaired t-tests with $p = 0.05$. This result shows that the proposed method works successfully on a reasonably large and sparse real-world dataset. ST-PTBM improves ST-PFUV over the temporal accuracy metrics by 20.16%, 7.50% and 4.30%, respectively. Meanwhile, ST-PTBM improves ST-PTBM-One over temporal recall metric by 7.11%. Similar to the experimental results on the Hetrec dataset, the ST-PTBM method can achieve a better balance between exploitation and exploration when it searches the latent factor space.

Comparison of temporal behaviors Figure 3.8 plots the AAI among ST-PTBM, ST-PTBM-One, ST-PFUV and STKF methods. All the curves are above zero, showing that the developed method constantly outperforms baseline methods over time by taking advantage of the fine modeling of temporal dynamics in user preferences and item attractiveness. Meanwhile, compared with ST-PTBM-One, the tendency of the circle curve demonstrates that the proposed learning algorithm is more effective at exploiting the underlying temporal patterns. The blue dash-dotted curve (ST-PTBM vs ST-PFUV), which almost overlaps with the circle curve, illustrates that ST-PTBM improves ST-PFUV over time by enforcing the temporal importance between user preferences and item popularity and dynamically adjusting weights of latent dimensions. Compared with ST-PFUV, extra

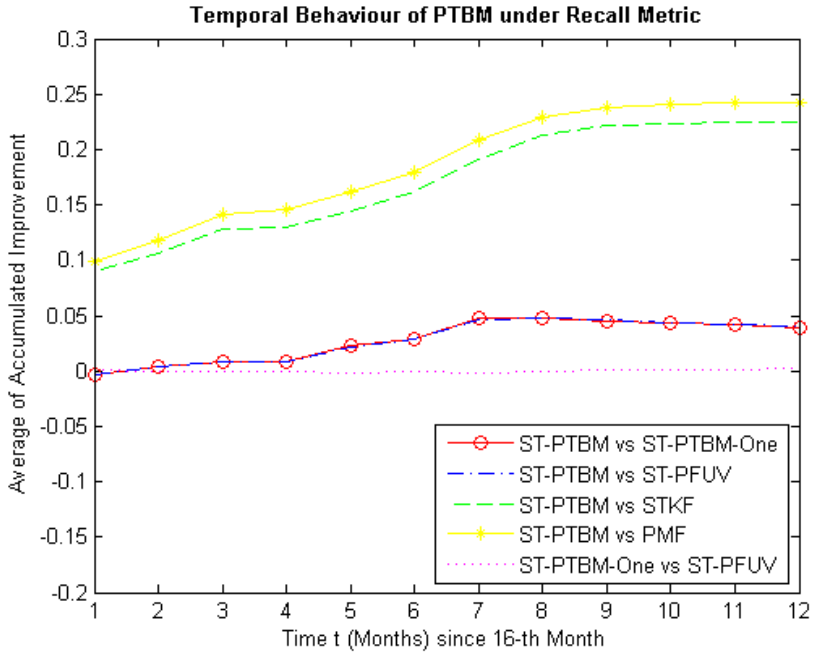


Figure 3.8: The average of accumulated improvement for ST-PTBM on the Netflix dataset since the 16-st month.

degrees of freedom are introduced in the ST-PTBM method. The improvement between the two methods can also be partially explained as these variables are able to depict the tendency of user preferences and item characteristics more finely. Because the interaction matrix is learned online in ST-PTBM, it implicitly constructs temporal importance that confines the variation of user and item latent factors over time.

The green (dash) curve (ST-PTBM vs STKF) shows that the temporal performance of RSs can be significantly improved by tracking the tendency of item attractiveness and coping with non-Gaussian behaviors. The improvement of ST-PTBM over STKF is also attributed to the failure of STKF when it encounters non-Gaussian and NMAR behaviors on user preferences and item popularity. Conversely, the sequential sampling approximation and the learning method enable the ST-PTBM method to track the non-Gaussian and even non-linear tendency of user preferences and item popularity which is common in practice. The light red (dotted) curve (ST-PTBM-One vs ST-PFUV) is flat and very close to zero-axis. This observation shows that only slight improvement could be made when

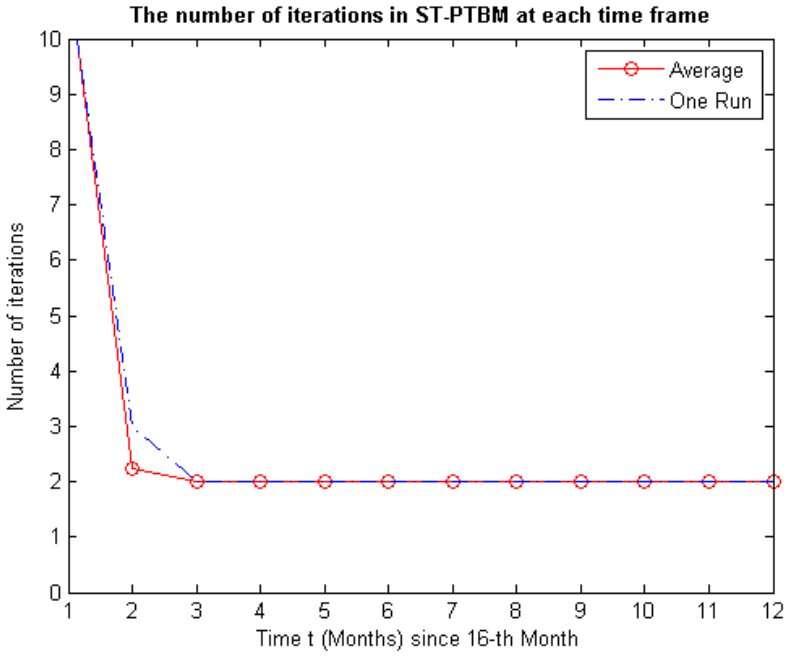


Figure 3.9: The number of iterations in the learning algorithm at each time frame on the Netflix dataset.

temporal importance is not finely modeled and adaptively learned. Figure 3.9 illustrates a sequence of the average number of iterations in EM under temporal recall metric and one run of the sequence.

Method	PMF	ST-PFUV	ST-PTBM-One	ST-PTBM
Hetrec time	432.9	*198.7*	215.3	241.1
Hetrec parallel	-	*133.6*	153.2	172.1
Netflix time	3460.2	*857.3*	1644.2	2316.5
Netflix parallel	-	*523.7*	637.3	907.9

Table 3.6: The average running times of compared methods in the experiments. The unit is second and the best performance is in italic font.

Computational time As a sequential approach, the developed methods dynamically learn model parameters and track user and item latent factors. The approach adopted in the developed methods does not change the computational complexity of prediction. Thus, the total running time of each time frame (including both training or retraining time and prediction time) is averaged to compare the efficiency of these methods. The

algorithms are implemented in Matlab and run on a 4-core machine with 3.3G Hz CPU and 8G memory.

The running times of these methods under temporal recall metric on the Hetrec and Netflix datasets are listed in the first and third lines in Table 3.6, respectively. Because the running times on the MovieLens dataset are quite closed to each other for the compared methods, they are ignored here for clarity. Their running times under other temporal metrics have the similar effects and they are ignored for clarity. As mentioned before, it is straightforward to speed up ST-PTBM by parallelizing the Monte Carlo method and learning algorithm. The running times of these parallelized versions (with 4 threads) are listed in the second and fourth lines in Table 3.6. A parallelized version of PMF is not implemented. The average numbers of iterations for ST-PTBM on the Hetrec and Netflix datasets are 1.74 and 2.75 at each time step, respectively. The key observation in the table is that ST-PTBM can be greatly speeded up by parallelization. Therefore, it is reasonable to expect the running time of ST-PTBM is comparable with baseline methods when ST-PTBM is deployed on more sophisticated machines. Note that PMF has been among the fastest state-of-the-art CF methods.

3.9 Summary

A novel probabilistic temporal bilinear model is presented to improve the performance of RSs over time. The developed model exploits the temporal and dynamic information in users' historical feedback to finely model the tendency of user preferences and item attractiveness. It simultaneously tracks latent factors representing user preferences and item attractiveness for the Top-N recommendation. Meanwhile, this model enforces the temporal interactions between user preferences and item attractiveness and dynamically adjusts significance on different dimensions of user and item latent factors. To simultaneously solve the problems of data sparsity and scalability under temporal context, a novel two-phase self-training mechanism is developed to construct a small but delicate set of observations from missing data dynamically. A new learning and inference algorithm

combining a sequential Monte Carlo method and the EM algorithm is also developed to take advantages of temporal information and dynamic structure in the feedback.

Those methods are evaluated on three real-world benchmark datasets under the temporal extensions of accuracy metrics. The experimental results demonstrate that the developed methods significantly improve the recommendation performance over a variety of state-of-the-art algorithms, which confirms that the developed method can effectively and efficiently learn and track both latent factors and model parameters over time. The experiments also illustrate the advantages of the temporal dynamic model over static ones and the benefits of tracking both user preferences and item attractiveness instead of tracking merely user preferences.

The developed method assumes that the temporal dynamics of latent factors representing user preferences and item attractiveness over time follow the first order Gaussian random walk. Also, it imposes a unique dynamic system, which is not tailored to meet the various and diverse trends of user preferences and item attractiveness in RSs. Chapter 4 will discuss these problems in detail and develop solutions to them.

Chapter 4

On Learning the Dynamics in Temporal Recommender Systems

4.1 Introduction

As mentioned in Section 1.2.2, Chapter 1, the model structure to capture the tendency of user preferences and item attractiveness in recommender systems (RSs) is either predefined or learned to fit a linear system. In addition, rather than using the personalized and item-wise dynamic systems, a universal dynamic model is commonly adopted across all users and items in RSs that aim to exploit temporal dynamics in user feedback. In this chapter, the development of modeling the temporal dynamics in RSs will be continued, especially focusing on solving the above-mentioned problems when trying to model the tendency of user preferences and item attractiveness flexibly.

Recall that temporal RSs can be classified into four approaches in terms of how the temporal and dynamic information is utilized: heuristic, binning-based, online updating and dynamic-based approaches [160]. Generally, the first three approaches can be deducted from the dynamic-based approach, which explicitly models the temporal dynamics by either a stochastic state space approach or temporal regressions. Existing methods of this

approach simply assume that the tendency of user preferences and item attractiveness is known a priori or the model structure of the dynamics is provided. Because it is non-trivial to design meaningful personalized transition models in RSs, some simple transition systems, such as the first order random walk, become popular candidates. However, it is often the case that some users or items are inactive within some time frames. Compared to the aimlessly searching by the random walk, it is reasonable to assume that user preferences and item attractiveness are better governed by the inertia of their dynamics. Meanwhile, instead of using a large concatenated state vector consisting of all the user and item latent vectors, the existing methods [80, 160] separately track latent vectors for each user and item to make the algorithm tractable. This compromise combines with an unintentional dynamic system, such as the random walk model, may lead to the degradation of the temporal performance of RSs. More importantly, considering the diversity of users and items, their properties could evolve in quite diverse manners over time. Instead of just using one model trying to fit all, it is thus more desirable to place fewer assumptions or constraints on the model structures.

For modeling the dynamics, it is always assumed that there is sufficient information to learn the transition models [88, 255]. However, this is not always the case under the personalized and item-wise context. For example, for some users that give feedback only within one time frame in the training period, it is impossible to extract some meaningful dynamic patterns merely based on their own feedback, since this feedback does not span enough time intervals. This problem also exists for some items. In this chapter, this kind of problems is defined as the *cold start transition* problem. It is different from the traditional cold start problem in RSs, where users or items with no feedback exist [1]. For these users or items that suffer from the *cold start transition* problem, they may have provided or received some feedback within a time frame, however, which does not span enough time intervals during the training period to derive transition models. For dynamic[2] RSs, at

[1]The cold start problem is referred in a strict sense. In the wide sense, users or items are allowed to have a little amount of feedback associated with them. For example, when items only receive less than 5 ratings, they are regarded as the cold start items.

[2]The cold start transition problem and the developed solution to it in this chapter are

each time frame, the prediction is made and observations are then used to update models. Hence, no personalized dynamic recommendation can be appropriately obtained for such users without solving the *cold start transition* problem.

In this chapter, the *cold start transition problem* in RSs exploiting temporal dynamics is solved by [162]

1) modeling the tendency of user preferences and item popularity in a personalized and item-wise fashion, where no assumption on dynamical model structure is imposed,

2) proposing a collaborative inference and learning algorithm that explicitly considers the uncertainties of model structure and dynamically updates the model to track user and item latent vectors accurately and alleviate the compromise in the separate tracking, and

3) exploiting the temporal dynamics of "like-minded" users and similar items to learn the cold start transition models.

To the best of the author's knowledge, this is the first work that aims to solve the *cold start transition* problem in RSs. Meanwhile, it is the first work to develop the personalized and item-wise dynamical systems for RSs without structure assumption.

Overall, the proposed system uses two sets of latent vectors to represent user preferences and item attractiveness compactly and respectively at each time frame, whose initial settings are partially learned by a probabilistic matrix factorization (PMF) method. Based on these representations, the system employs a probabilistic personalized and item-wise temporal model to track the tendency of user preferences and item attractiveness over time simultaneously. The model parameters are dynamically updated by the learning algorithm whenever a new batch of feedback occurs. Meanwhile, the posterior distributions involving the temporal dynamics of "like-minded" users and similar items are utilized to generate the initial settings for those users and items that suffer from the *cold start transition* problem.

---

not restricted to temporal data. For ease of exposition, "temporal" and "dynamic" are used interchangeably in this chapter.

For the Top-N recommendation task [115], a personalized recommendation list for each user is generated based on the predicted user preferences for items. The prediction at each time step is adapted by using the updated transition models and the current user and item latent vectors.

The rest of this chapter is organized as follows. Related work is briefly discussed in Section 4.2. Section 4.3 describes the developed model. In Section 4.4, the inference and learning algorithm of this model is covered. The learning algorithm to cope with the *cold start transition* problem is also covered in this section. The performance of proposed algorithms with a variety of baseline methods are compared and discussed in Section 4.6. Finally, the summary will be presented in Section 4.7.

4.2 Related Work

The developed methods in this chapter are closely related to the methods discussed in Chapter 3. For clarity, the related work on particle filtering for MF methods is ignored in this section.

The developed learning algorithm in this chapter is closely related to the learning algorithm in [255, 90], which also relies on the EM framework to identify the model uncertainty. However, the method in [255] only focuses on the modeling of the observation model and assumes the temporal dynamics are predefined rather than dynamically learned over time. The developed method is also different from [90], which is not developed to adjust the dynamics adaptively to new inputs and is only designed to work with data with very low sparsity that is not practical in RSs.

Conventional extreme learning machine (ELM) has been used in [23] to achieve online adaptive object tracking. However, they are used for classification and do not update the kernel parameters. The dynamics are not modeled during the tracking, and the learning algorithm in [23] also does not consider the model uncertainties.

The learning algorithm in [92] exploits the EM framework and particle filtering to learn a probabilistic bilinear model. However, as the dynamic system in [92] is an identity function, there is no dynamic and temporal information being incorporated into that algorithm and the model can be regarded as a subset of the developed method even if it does not include the backward filtering procedure in the developed algorithm.

In [61], a predictive model is developed to recommend dynamic contents. This model is not based on latent vectors. Instead, the method directly constructs some user and item profiles based on both static and temporal side information. Unlike the proposed methods in this chapter, all of these methods are not developed to model the personalized and item-wise dynamics or tackle with the *cold start transition* problem. Furthermore, even if the temporal information is considered, model structure uncertainties in these models are not deliberately considered during the learning procedure.

4.3 Personalized and Item-wise Model

The first step towards solving the *cold start transition* problem is to learn a general transition model for dynamics. A state space approach [200] is adopted to model the tendency of user preferences and item attractiveness in the personalized Top-N recommendation.

Simulation techniques, such as sequential Monte Carlo methods [200], are used to estimate user and item latent factors simultaneously and cope with non-linear and non-Gaussian behaviors. Recall that the number of required particles should be $O(2^d)$ for a d-dimensional state space to achieve a satisfactory result [209]. Following the approach in Chapter 3, latent factors are separately estimated for each user and item to make a compromise between the accuracy of user and item representations and the tractability of simulation techniques. Following [197, 196], these latent variables are still assumed to be marginally independent while any rating $R_t^{u,i}$ is still assumed to be conditionally independent given user u's latent vector U_t^u at time t and item i's latent vector V_t^i at time t.

However, as mentioned before, the dynamical systems in temporal RSs are assumed to

be either known a priori or with known model structures. For instance, the first-order Gaussian random walk with or without specifying the step size is a popular fallback as the transition models for the known model structures. An online adaptive ELM [104] is developed in this chapter to model the dynamics for each user and item, where no assumption on their model structures is imposed. Thus, the learned dynamical systems are tailored to capture the diverse dynamics of the latent vectors representing user preferences and item popularity. With "like-minded" users and similar items as the collaborators, these fully personalized and item-wise dynamical systems are then used to solve the *cold start transition problem* in dynamic RSs. The inertia of the personalized and item-wise dynamics provides the developed method with fine granular control to help predict and distinguish user preferences and item attractiveness within some time frames when there are few observations available for some users and items.

4.3.1 Modeling dynamics

Considering the t-th time window ranging from $t-n+1$ to t with $1 \leq n \leq t$, let $U_m^u \in \mathcal{R}^K$ and $V_m^v \in \mathcal{R}^K$ be latent vectors representing user u's preferences and item v's attractiveness at time m in the time window. The tendency of user preference and item attractiveness is usually modeled by a nonlinear first-order transitional relation between the current and previous latent factors. Specifically, the nonlinear transition models for a user u's preferences and item v's attractiveness in the nonlinear state space based dynamical system are defined as follows,

$$U_m^u = f_u(U_{m-1}^u) + u_m, \tag{4.1}$$

and

$$V_m^v = f_v(V_{m-1}^v) + v_m, \tag{4.2}$$

where u_m and v_m are independent and identically distributed (I.I.D.) noise processes with possibly non-Gaussian distribution.

A major focus of the work in this chapter is to model the completely unspecified nonlinear functions f_u and f_i accurately and efficiently to reflect the tendency of user preferences

and item attractiveness flexibly in order to improve the performance of RSs over time. All the uncertainties in the function f are captured in a set of parameter vectors θ that is treated as an unknown but deterministic set of vectors.

In order to make the transition functions f_u and f_v fully determined by the user feedback, no assumptions or constraints should be placed on the model structure. Therefore, an online adaptive ELM is developed to model the temporal dynamics in RSs. Specifically, for each user u and item v, an ELM with L hidden kernels and K outputs is exploited as the basis to construct the transition model. Note that only the number of hidden kernels and outputs is specified here and the structure of the dynamics of both user and item latent factors will be automatically learned from the user feedback. As Section 4.4.2 will show, the developed ELM is adaptive in a sense that it constantly adjusts its parameters at each time frame to capture the current tendency.

For a typically hidden kernel l in the t-th window, the Gaussian kernel for user u is utilized with the center $\mu_t^{u,l} \in \mathcal{R}^K$, where $l \in 1, \ldots, L$. Its covariance matrix $\Sigma_t^{u,l} \in \mathcal{R}^{K \times K}$ is usually restricted to $\sigma_l I$ with identity matrix I and $\sigma_l \in \mathcal{R}^+$ [104]. To emphasize the correlation between latent vectors, the developed model does not impose such a constraint and uses $\Sigma_t^{u,l} = \Sigma_t^u \in \mathcal{R}^{K \times K}$. This release aims to increase flexibility while avoiding over-parametrization for sparse data in dynamic RSs. Let the interconnection weight matrix for user u be $H_t^u \in \mathcal{R}^{K \times L}$, where $h_{l,t}^u \in \mathcal{R}^K$ is the interconnection weights between the l-th hidden Gaussian kernel and outputs in user u's evolution process in the t-th time window, and the matrix is the column-wise stacking of $h_{l,t}^u$ as $H_t^u = [h_{1,t}^u, \ldots, h_{L,t}^u]$.

The evolution process $f_u(U_{m-1}^u)$ of user u's preferences can be defined as follows,

$$U_m^u = H_t^u \Phi_t^u(U_{m-1}^u) + b_t^u, \tag{4.3}$$

where the vector $b_t^u \in \mathcal{R}^K$ captures the individual's preference bias in the t-th time window. The L-dimensional vector $\Phi_t^u(U_{m-1}^u)$ for user u at time $m-1$ in the t-th time window is defined as follows,

$$\Phi_t^u(U_{m-1}^u) = [\Phi_{t,1}^u(U_{m-1}^u), \ldots, \Phi_{t,L}^u(U_{m-1}^u)]^T, \tag{4.4}$$

where its scalar component $\Phi^u_{t,l}(U^u_{m-1})$ is defined as follows,

$$\Phi^u_{t,l}(U^u_{m-1}) = exp(-\frac{1}{2}(U^u_{m-1} - \mu^{u,l}_t)^T \Sigma^{u,l^{-1}}_t (U^u_{m-1} - \mu^{u,l}_t)), \tag{4.5}$$

Similarly, the transition models for item v can be obtained.

4.3.2 The observation model

In order to make it tractable, the personalized and item-wise temporal model separately models the evolution processes of user preferences and item attractiveness for each user and item. It is the observation model in which enables "like-minded" users and similar items collaborate with each other and convey this collaboration into the dynamics.

Intuitively speaking, in the observation model, a user u's preferences over items are governed by both user u's latent factors and the latent factors of all the items that are also shared by other users when inferences about their latent preferences are made. As discussed in Chapter 3, the feedback in RSs is generally not missing at random [212]. Low ratings are much more likely to be missing than high ratings [212] because users are free to choose items to give their feedback. A proper observation model should consider the ranking of all the items, no matter whether they are observed or not [212]. Instead of treating all the missing data as negative, a subset of missing items is sampled for each user at every time frame as negative samples to complement the user's most recent observation [160]. The dynamically constructed observations for user u at time m are denoted as $\hat{R}^u_m$. This personalized approach not only distinguishes the past and recent information but also avoids dominating recent observation with imputed data.

The observation model over constructed ratings $\hat{R}^u_m$ for user u at time m is defined as follows,

$$P(\hat{R}^u_m|U^u_m,\{V^j_m\}) = exp\{-\sum_{j=1}^{M} w_{im}((\bar{r}^{u,j}_m - U^u_t(V^j_m)^T)^2 + \frac{\lambda}{2}||U^u_m||^2_{\text{Frob}} + \frac{\lambda}{2}||V^j_m||^2_{\text{Frob}})\}, \tag{4.6}$$

where $||\cdot||_{\text{Frob}}$ denotes the Frobenius norm that is used as the regularization term. The constant λ is regularization coefficient and $\bar{r}_m^{u,j}$ equals to $r_m^{u,j}$ if it is observed and equals to r_m otherwise. The value r_m is an imputed value for all the missing data, which is regarded as the average value of ratings in the complete but unknown data. The weight w_{im} is a global constant for imputed data to reflect the confidence over imputation for simplicity [212]. Similarly, the observation model over constructed ratings $\hat{R}_m^v$ for item v at time m can be defined.

The exponential loss in the above observation functions is inspired by the formulation of the energy function in Markov random field [36]. By using particle smoothing shown later, this function will focus more on those particles that can reconstruct user preferences over items more precisely. Similar to loss functions used in Section 3.3.2 in Chapter 3, other loss functions, such as squared loss or absolute loss, have also been explored in the trails but it is difficult for them to filter out well-performed particles for the Top-N recommendation task.

4.4 Inference and Learning

When the transition models of user preferences and item attractiveness are completely given, the estimation problem reduces to a filtering problem. The particle filtering, which can cope with nonlinear and non-Gaussian behaviors in temporal RSs, could be used to inference the hidden states representing user preferences and item attractiveness over time as shown in Chapter 3. When the structure of the transition models is known and the corresponding states and observations are available in the training data, standard methods such as maximum likelihood estimation could be used to learn model parameters. After this pre-processing procedure of system identification, the estimation problem becomes a filtering problem. In temporal RSs, neither of the above two cases can hold as there is no ground truth available for the rich and diverse structures of the tendency of user preferences and item attractiveness. Hence, model uncertainties introduced by removing the structure assumptions in Section 4.3 must be considered during inference and learning.

It is shown [160, 158] that the tendency of user preferences and item attractiveness can be better reflected by recent observations than the historical data. Therefore, it is assumed here that the model parameters are estimated by using current observations and previous $n-1$ observed users' feedback. At time t, user latent factors U_t and item latent factors V_t are forwardly estimated by particle filtering from time $t-n+1$. The particle smoothers are then used to estimate the posterior distributions of these latent factors over such a period. Although the time windowing technique is used here to reduce the influence of historical data, the recursive mechanism of particle smoothing does not make the learning procedure underestimate the importance of these historical data that do not fall within the time window. To this point, it is the E-step and all the estimation is under the condition that the model parameters are given and fixed as the estimated values from the previous time window.

After obtaining the sufficient statistics with respect to the estimated posterior distributions in the E-step, the model parameters are updated in the M-step to maximize the expected complete logarithm likelihood function. Similar to the discussion in Chapter 3, optimization methods, such as the stochastic gradient ascent, tend to fit the model to the imputed value as accurate as possible. As a unique imputed value is set to those selected negative samples, this optimization approach can easily overfit the model and make the predicted user preferences over unrated items indistinguishable. The experimental results (not shown in this chapter) also verify this statement. In this regard, the inference and learning procedures do not conduct the iterative procedure as the conventional EM method does. As the one step optimization framework still aims to maximize the expected complete log likelihood function at each time frame, the inference and learning procedures developed under one step optimization framework are attempting to exploit the benefits of imputation fully while avoiding its artifacts.

In summary, a method that combines an EM-like approach and particle smoothing is developed to estimate the latent vectors and model parameters jointly at each time frame, which blindly incorporates the model uncertainties. At time t (or the last frame in the t-th time window), user and item latent vectors are forwardly estimated by particle filtering

from time $t-n+1$, where $1 \leq n \leq t$. Particle smoothing is then used to estimate the posterior distributions of these latent vectors and model parameters within such a period. For clarity, only the distributions of user latent vectors are shown. It is symmetric to derive the distributions of item latent vectors.

4.4.1 E-step

On the E-step of the algorithm, the forward filtering stage is conducted at first. Let $\theta_t^u = \{H_t^u, \mu_t^u, b_t^u, \Sigma_t^u\}$ and $\theta_t^v = \{H_t^v, \mu_t^v, b_t^v, \Sigma_t^v\}$ denote model parameters of the transition models at the current time window ($t-n+1 \leq m \leq t$) for user u and item v, respectively. Let θ_t denote the union of θ_t^u and θ_t^v at the t-th time window. The posterior distribution of latent factors U_t^u and V_t^v is

$$\prod_{u=1}^{M} \prod_{v=1}^{N} P(U_t^u, V_t^v | R_{1:t}, \theta_t). \tag{4.7}$$

By alternatively estimating user and item latent vectors, each joint posterior distribution over latent vectors is approximated as the product of marginal posterior distributions $P(U_t^u|R_{1:t})$ and $P(V_t^v|R_{1:t})$. For user u, its marginal distribution can be estimated as follows,

$$\begin{aligned}
& P(U_t^u | U_{t-1}^u, V_{t-1}^v, R_{1:t}, \theta_t) \\
\propto & \int P(R_t | U_t^u, V_t^v) P(U_t^u | U_{t-1}^u, \theta_t^u) P(V_t^i | V_{t-1}^v, \theta_t^v) \mathrm{d}V_t^v \\
\approx & \int P(R_t | U_t^u, V_t^v) P(U_t^u | U_{t-1}^u, \theta_t^u) \delta(V_t^v - \bar{V}_t^v) \mathrm{d}V_t^v \\
= & \, P(R_t | U_t^u, \bar{V}_t^v) P(U_t^u | U_{t-1}^u, \theta_t^u),
\end{aligned} \tag{4.8}$$

where $\bar{V}_t^v$ is the estimated value for item v's latent factors from the previous iteration. Particle filtering is used to approximate the posteriors, where transition and observation functions are defined in Section 4.3. The posterior for user u then becomes point mass distribution over S sampled latent vectors $\{U_t^{u,(s)}\}$ and their weights $\{w_{U,t}^{u,(s)}\}$ where $s = 1, \ldots, S$. Similarly, the marginal distribution for item v can be estimated as follows,

$$P(V_t^v | V_{t-1}^v, U_{t-1}^u, R_{1:t}, \theta_t) \propto P(R_t | V_t^v, \bar{U}_t^u) P(V_t^v | V_{t-1}^v, \theta_t^v) \tag{4.9}$$

In order to estimate model parameters, it is also inevitable to have the marginal posterior distribution over U_m^u and the marginal posterior distribution over V_m^v for any time frame m within the time window. By using particle smoothing, the posterior distribution for user u is approximated as the point mass distribution over the latent vectors $\{U_m^{u,(s)}|s = 1, \ldots, S\}$ obtained from particle filtering and their smoothed weights $\{w_{U,m|t}^{u,(s)}|s = 1, \ldots, S\}$, where only the weights are modified by the smoothing. Let $P(U_{m+1}^u|U_m^u, \theta_t^u)$ be the probabilistic form of user u's transition model defined in Section 4.3. Weights $w_{U,m|t}^{u,(s)}$ are backwardly and recursively calculated from time frame t until time frame $t - n + 1$ as follows,

$$w_{U,m|t}^{u,(s)} = \sum_{s'=1}^{S} w_{U,m+1|t}^{u,(s')} \times \frac{w_{U,m}^{u,(s)} P(U_{m+1}^{u,(s')}|U_m^{u,(s)}, \theta_t^u)}{\sum_{q=1}^{S} w_{U,t}^{u,(q)} P(U_{m+1}^{u,(s')}|U_m^{u,(q)}, \theta_t^u)}. \tag{4.10}$$

The initial smoothing weight at time frame t is set as its filtering weight $w_{U,t|t}^{u,(s)} = w_{U,t}^{u,(s)}$. There exist more sophisticated methods for smoothing, such as two filter smoothing [43]. These methods are not covered in this discussion because a concrete smoothing technique is not the focus of this chapter.

As the resampling step is adopted in the forward filtering stage to mitigate the degeneracy problem, if the identical particle propagation is used to approximate the posterior distribution $p(U_m^u|R_{t-n+1:t}, \theta_t^u)$ in both numerator and denominator in Eq (4.10), the smoothing weight can be simplified as follows,

$$w_{U,m|t}^{u,(s)} = \sum_{s'=1}^{S} w_{U,m+1|t}^{u,(s')} \times \frac{P(U_{m+1}^{u,(s')}|U_m^{u,(s)}, \theta_t^u)}{\sum_{q=1}^{S} P(U^{u,(s')_{m+1}}|U_m^{u,(q)}, \theta_t^u)}. \tag{4.11}$$

This simplification can facilitate the vectorization and make the inference more efficient in the concrete implementation.

4.4.2 M-step

After gathering sufficient statistics of model parameters with respect to the posterior distributions of user and item latent factors, model parameters θ_t for t-th time window is re-estimated at each M-step to maximize the probability of generating the ratings in the training data and blindly incorporating the uncertainties on the model structures.

The objective function Recall that for each user u and item v the transition models are given in Eq (4.3) and a similarly derived one respectively. The complete logarithm likelihood function $logP(R_{t-n+1:t}, U_{t-n+1:t}, V_{t-n+1:t}|\theta_t)$ at time t for the t-th time window can be then calculated as follows,

$$\begin{aligned}
&logP(R_{t-n+1:t}, U_{t-n+1:t}, V_{t-n+1:t}|\theta_t) \\
\approx &\sum_{m=t-n+1}^{t} logP(U_{m+1}|U_m, \theta_t^U) + logP(R_{t-n+1:t}|U_{t-n+1:t}, V_{t-n+1:t}) + \\
&\sum_{m=t-n+1}^{t} logP(V_{m+1}|V_m, \theta_t^V) \\
= &\sum_{m=t-n+1}^{t}\sum_{u=1}^{N} logP(U_{m+1}^u|U_m^u, \theta_t^u) + \sum_{m=t-n+1}^{t}\sum_{v=1}^{M} logP(V_{m+1}^v|V_m^v, \theta_t^v) + \mathcal{C},
\end{aligned} \tag{4.12}$$

where θ_t^U and θ_t^V denote the collections of model parameters over all the users and items, respectively. The constant $\mathcal{C}$ represents all the terms irrelevant to model parameters θ_t, and the observation function is given in the form of Eq (4.6). The first approximation is due to the alternative estimation of user and item latent factors and the assumption that there exist initial settings U_0 and V_0.

Hence, the expectation of log-complete likelihood to be maximized can be expressed as

$$\begin{aligned}
E \approx &\sum_{m=t-n+1}^{t} \Big(\int \sum_{u=1}^{N} logP(U_{m+1}^u|U_m^u, \theta_t^u) P(U_{m+1}^u, U_m^u|R_{t-n+1:t}, V_{t-n+1:t}, \theta_t^u) \mathrm{d}U_{m+1}^u \mathrm{d}U_m^u + \\
&\int \sum_{v=1}^{M} logP(V_{m+1}^v|V_m^v, \theta_t^v) P(V_{m+1}^v, V_m^v|R_{t-n+1:t}, U_{t-n+1:t}, \theta_t^v) \mathrm{d}V_{m+1}^v \mathrm{d}V_m^v \Big) \\
= &\sum_{m=t-n+1}^{t} \int \sum_{u=1}^{N} logP(U_{m+1}^u|U_m^u, \theta_t^u) \cdot P(U_{m+1}^u|R_{t-n+1:t}, V_{t-n+1:1}, \theta_t^u) \cdot \\
&\frac{P(U_{m+1}^u|U_m^u, \theta_U^u) P(U_m^u|R_{t-n+1:t}, V_{t-n+1:t}, \theta_t^u)}{\int P(U_{m+1}^u|U_m^u, \theta_t^u) P(U_m^u|R_{t-n+1:t}, V_{t-n+1:t}, \theta_t^u) \mathrm{d}U_m^u} \mathrm{d}U_{m+1}^u \mathrm{d}U_m^u \\
&+ \sum_{m=t-n+1}^{t} \int \sum_{v=1}^{M} logP(V_{m+1}^v|V_m^v, \theta_t^v) \cdot P(V_{m+1}^v|R_{t-n+1:t}, U_{t-n+1:1}, \theta_t^v) \cdot \\
&\frac{P(V_{m+1}^v|V_m^v, \theta_t^v) P(V_m^v|R_{t-n+1:t}, U_{t-n+1:t}, \theta_t^v)}{\int P(V_{m+1}^v|V_m^v, \theta_t^v) P(V_m^v|R_{t-n+1:t}, U_{t-n+1:t}, \theta_t^v) \mathrm{d}V_m^v} \mathrm{d}V_{m+1}^v \mathrm{d}V_m^v,
\end{aligned} \tag{4.13}$$

where the irrelevant constants are omitted in the above equation for clarity. The first approximation is also due to the alternative estimation of user and item latent factors. The second equation exploits the following Bayesian equivalence,

$$\begin{aligned}
&P(U^u_{m+1}, U^u_m | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t) \\
=&P(U^u_m | U^u_{m+1}, R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t) \cdot P(U^u_{m+1} | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t) \\
=&P(U^u_{m+1} | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t) \cdot \frac{P(U^u_m, U^u_{m+1} | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t)}{P(U^u_{m+1} | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_t)}
\end{aligned} \tag{4.14}$$

By approximating the posterior distributions $P(U^u_m | R_{t-n+1:t}, V_{t-n+1:t}, \theta^u_U)$ and $P(U^u_{m+1} | R_{t-n+1:t}, V_{t-n+1:1}, \theta^u_U)$ as described in the E-step, the maximization of Eq (4.13) over the model parameters of users is equal to the minimization of the following function,

$$\begin{aligned}
E_U \approx \sum_{m=t-n+1}^{t} \sum_{u=1}^{N} \sum_{i=1}^{S} \sum_{j=1}^{S} &\| U^{u,(i)}_{m+1} - f_u(U^{u,(j)}_m, \theta^t_u) \|^2 \cdot w^{u,(i)}_{m+1|t} \cdot \\
&\frac{w^{u,(j)}_m P(U^{u,(i)}_{m+1} | U^{u,(j)}_m, \theta^u_t)}{\sum_{q=1}^{S} w^{u,(q)}_m P(U^{u,(i)}_{m+1} | U^{u,(q)}_m, \theta^u_t)},
\end{aligned} \tag{4.15}$$

where $f_u(U^{u,(j)}_m, \theta^u_t)$ is the evolution process for user u as shown in Section 4.3 but with explicitly displayed model parameters θ^u_t. The maximization of Eq (4.13) over the model parameters of items can be similarly obtained.

Adaptive updating In conventional ELMs, the kernel parameters are selected randomly and the output interconnection weights are obtained analytically to achieve an extremely high learning speed. Hence, the M-step can be omitted and the developed learning and inference algorithm is reduced to particle smoothing. However, under temporal context, the static kernel parameters may not cover the proper regions in the latent space to identify the system model correctly, because the latent vectors representing user preferences and item attractiveness are constantly evolving. Therefore, it would be useful to update all the weights and parameters dynamically in transition models.

By exploiting the accuracy of single layer forward radial basis function networks while preserving the fast learning property of ELM, the developed updating procedure con-

ducts the EM loop once and attempts to make the conventional ELM more adaptive and generalization under temporal personalized recommendation context.

Specifically, instead of optimizing against these kernel parameters as much as possible, the developed procedure relies on clustering methods and one step gradient descent technique [88] to update these parameters dynamically.

The K-mean clustering is used for each user and item to regenerate L groups for the centers of hidden kernels at each time step. It is tempted to use all the particles as the current inputs to account the uncertainties existing in the model estimation. However, considering the large number of particles with respect to the number of hidden kernels, the regenerated centers are prone to be overwhelming by the current estimation and underestimating the importance of historical data. Hence, the developed method clusters all the kernel centers μ^u_{m-1} from previous time $m-1$ and the expected latent vectors $\hat{U}^u_m$ at current time m. By this updating procedure, the transition models are able to adapt to the regions where the posteriors of user preferences and item attractiveness have high probabilities. Similarly, the covariance matrix of user u's l-th kernel Σ^u_t is updated by one step gradient descent.

Let $J = \|U^{u,(i)}_{m+1} - f_u(U^{u,(j)}_m, \theta^u_t)\|^2$. By taking advantages of the concept of α-derivatives [165], the gradient of E_U in Eq (4.15) with respect to Σ^u_t is derived by using matrix-matrix derivatives as follows,

$$\frac{\partial E_U}{\partial \Sigma^u_t} = \sum_{m=t-n+1}^{t} \sum_{i}^{S} \sum_{j}^{S} A(u,i,j) \frac{\partial J}{\partial \Sigma^u_t}, \tag{4.16}$$

where

$$\frac{\partial J}{\partial \Sigma^u_t} = B(u,i,j,m)[(U^{u,(j)}_m - \mu^{u,l}_m)^T \Sigma^{u^{-1}}_t]^T [(U^{u,(j)}_m - \mu^{u,l}_m)^T \Sigma^{u^{-1}}_t], \tag{4.17}$$

and

$$A(u,i,j) = w^{u,(i)}_{U,m+1|t} \frac{w^{u,(j)}_{U,m} P(U^{u,(i)}_{m+1} | U^{u,(j)}_m), \theta^u_t)}{\sum_{q=1}^{S} w^{u,(q)}_{U,m} P(U^{u,(i)}_{m+1} | U^{u,(q)}_m), \theta^u_t)}, \tag{4.18}$$

$$B(u,i,j,m) = (U^{u,(i)}_{m+1} - f_u(U^{u,(j)}_m, \theta^u_t))^T \cdot h^u_{l,t} \cdot (\frac{1}{2} exp\{-\frac{1}{2} \psi_l(U^u_m, \theta^u_t)\}), \tag{4.19}$$

and

$$\psi_l(U^u_m, \theta^u_t) = (U^u_m - \mu^{u,l}_m)^T \Sigma^{u^{-1}}_t (U^u_m - \mu^{u,l}_m). \tag{4.20}$$

Let $\gamma_t^u = [H_t^u \; b_t^u]$ denote the concatenation of interconnection weights and the user bias term for the t-th time window. The parameter γ_t^u over the time frames $\{t-n+1:t\}$ can be analytically computed by maximizing the approximated expectation of log complete likelihood function in Eq (4.15). The updated γ_t^u can be computed as follows,

$$\begin{aligned}\gamma_t^u = \sum_{m=t-n+1}^{t} ((\sum_{i=1}^{S}\sum_{j=1}^{S} A(u,i,j)\cdot U_{m+1}^{u,(i)} \cdot \begin{pmatrix} \Phi_t^u(U_m^{u,(i)}) \\ 1 \end{pmatrix}^T)\cdot \\ (\sum_{i=1}^{S}\sum_{j=1}^{S} A(u,i,j)\cdot (\begin{pmatrix} \Phi_t^u(U_m^{u,(i)}) \\ 1 \end{pmatrix}\begin{pmatrix} \Phi_t^u(U_m^{u,(i)}) \\ 1 \end{pmatrix}^T \\ + \lambda_\Sigma \mathcal{I}_{L+1}))^{-1}), \end{aligned} \tag{4.21}$$

where $\mathcal{I}_{L+1\times L+1}$ is an identity matrix, and the coefficient λ_Σ is the coefficient for the Tikhonov regularization that works as the second regularization to mitigate the singular and ill-posed problem because the input data are not only noisy but also dynamically constructed at each time frame. Similarly, the model parameters of the dynamic system for item v can be derived.

4.4.3 Prediction

As mentioned in Chapter 3, canonical particles [92] $\hat{U}_t^u$ and $\hat{V}_t^v$ are utilized to represent the total effect of particles on the estimation for every latent vector at time t. To be avoided the degeneracy problem [200], particles will also be resampled in proportional to their weights. After resampling, the expectations of posterior distributions of U_t and V_t are utilized as canonical particles. At time t, the expectations of posterior distributions of user and item latent factors U_t and V_t are computed as $\hat{U}_t^u = \sum_{s=1}^{S} w_{U,t}^{u,(s)} U_t^{u,(s)}$ and $\hat{V}_t^v = \sum_{s=1}^{S'} w_{V,t}^{v,(s)} V_t^{v,(s)}$. Then, user u's preference over item i at time t can be estimated as $\hat{U}_t^u (\hat{V}_t^v)^T$.

4.5 Cold Start Problem in Learning Transitions

For ease of exposition, only users providing feedback at only one (the last) time frame T in training data are taken into account. The case of the *cold start transition* problem for items can be similarly derived. This problem is indeed different from both cold start user problem and cold start item problem that have been heavily investigated in RSs. For some users that are classified as cold start users, there is no feedback available. Conventional collaborative filtering (CF) methods are thus unable to extract their preferences when the methods do not resort to side or context information to reduce the sparsity. Similarly, conventional CF methods have difficulty to cope with those items that do not receive any feedback. However, for these users or items for which there is not enough information to initialize their transition models, they do have provided or received feedback. The only issue is that this feedback only exists in the latest time interval. Therefore, the problem that some users or items that are only active in the last time interval in the training data are named as the *cold start transition* problem.

In CF, the behaviors of cold start objects are typically regarded as being similar to normal objects connected by similar side information [13]. Inspired by this idea of tackling the cold start problems, "like-minded" users or similar items are also regarded as having similar dynamics or dynamics learned during training in the *cold start transition* problem, i.e collaborative tendencies.

4.5.1 Objects without cold start transition Problems

Let $R^{tr}_{1:T}$ be training data that are ordered and grouped by their timestamps into T time frames, and $R^{tr}_{1:T-1}$ is obtained by removing the last time frame from $R^{tr}_{1:T}$. Let $\{U_T\}$ and $\{U_{T-1}\}$ denote a set of sampled user latent vectors learned from $R^{tr}_{1:T}$ and $R^{tr}_{1:T-1}$, respectively. For these users that do not have the *cold start transition* problems, these latent vectors can be learned from the posteriors $P(U_T, V_T|R^{tr}_{1:T})$ and $P(U_{T-1}, V_{T-1}|R^{tr}_{1:T-1})$. The posterior distributions of model parameters $\{\theta_T\}$ in the transition models for these

user preferences are then given as follows,

$$P(\{\theta_T\}|\{U_T\},\{U_{T-1}\}) \propto \prod_{u=1}^{N}\prod_{d=1}^{D} P(U^u_{d,T}|U^u_{d,T-1},\theta^u_T)\cdot\prod_{u=1}^{N} P(\theta^u_T), \tag{4.22}$$

where D is the number of samples in the sampling set. The prior distribution $P(\{\theta_T\})$ for the model parameters is set to follow multivariate spherical Gaussian distribution to function as the regularization terms to prevent the estimation from overfitting.

Sampled vectors should be paired to learn meaningful transitions. Each sample obtained from $P(U_{T-1}, V_{T-1}|R^{tr}_{1:T-1})$ is used to initialize the procedure to obtain one corresponding sample from $P(U_T, V_T|R^{tr}_{1:T})$. Ideally, these latent vectors should be sampled from the posteriors $P(U_T, V_T|R^{tr}_{1:T})$ and $P(U_{T-1}, V_{T-1}|R^{tr}_{1:T-1})$ by methods such as MCMC [21]. However, considering the large number of users and items and the slow convergence rate of MCMC, those latent factors are approximated by randomizing the initial condition of AllRank method while leaving other components unchanged. As the number of samples D is usually not large, users that do not have the *cold start transition* problem can learn their initial transition models by using maximum a posterior estimation of their model parameters, which is equivalent to minimize the regularized least squares between latent vectors at T and their corresponding latent vectors transited from $T-1$.

4.5.2 Objects with cold start transition Problems

Recall that for users that do have the *cold start transition* problem, their latent vectors U_T do exist. Let $\{U_T^{\mathbf{N}(u)}\}$ be the set of latent vectors of user u's closest neighbors without the *cold start transition* problem, and $\theta_T^{\mathbf{N}(u)}$ be the set of transition model parameters of these users in the T-th time window. For user u that faces the *cold start transition* problem, its initial parameters of the transition model can be estimated by maximizing the following

joint posterior distribution,

$$
\begin{aligned}
&P(\theta_T^u, \{U_{T-1}^u\} | \theta_T^{\mathbf{N}(u)}, \{U_T^u\}, \{U_T^{\mathbf{N}(u)}\}, \{U_{T-1}^{\mathbf{N}(u)}\}) \\
\propto &P(\theta_T^u, \{U_{T-1}^u\}, \theta_T^{\mathbf{N}(u)}, \{U_T^u\}, \{U_T^{\mathbf{N}(u)}\}, \{U_{T-1}^{\mathbf{N}(u)}\}) \\
= &P(\{U_T^{\mathbf{N}(u)}\} | \{U_{T-1}^{\mathbf{N}(u)}\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u, \{U_{T-1}^u\}, \{U_T^u\}) \cdot P(\{U_T^u\} | \{U_{T-1}^u\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u, \{U_{T-1}^{\mathbf{N}(u)}\}) \cdot \\
&P(\{U_{T-1}^{\mathbf{N}(u)}\} | \{U_{T-1}^u\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u) \cdot P(\theta_T^{\mathbf{N}(u)} | \theta_T^u, \{U_{T-1}^u\}) \cdot P(\{U_{T-1}^u\} | \theta_T^u) \cdot P(\theta_T^u) \\
= &P(\{U_T^{\mathbf{N}(u)}\} | \{U_{T-1}^{\mathbf{N}(u)}\}, \theta_T^{\mathbf{N}(u)}) \cdot P(\{U_T^u\} | \{U_{T-1}^u\}, \theta_T^u) \cdot \\
&P(\{U_{T-1}^{\mathbf{N}(u)}\} | \{U_{T-1}^u\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u) \cdot P(\theta_T^{\mathbf{N}(u)} | \theta_T^u, \{U_{T-1}^u\}) \cdot P(\{U_{T-1}^u\} | \theta_T^u) \cdot P(\theta_T^u) \\
= &P(\{U_T^{\mathbf{N}(u)}\} | \{U_{T-1}^{\mathbf{N}(u)}\}, \theta_T^{\mathbf{N}(u)}) \cdot P(\{U_T^u\} | \{U_{T-1}^u\}, \theta_T^u) \cdot P(\{U_{T-1}^{\mathbf{N}(u)}\} | \{U_{T-1}^u\}) \cdot \\
&P(\theta_T^{\mathbf{N}(u)} | \theta_T^u) \cdot P(\{U_{T-1}^u\}) \cdot P(\theta_T^u) \\
\propto &P(\{U_T^u\} | \{U_{T-1}^u\}, \theta_T^u) \cdot P(\{U_{T-1}^{\mathbf{N}(u)}\} | \{U_{T-1}^u\}) \cdot P(\theta_T^{\mathbf{N}(u)} | \theta_T^u) \cdot \\
&P(\{U_{T-1}^u\}) \cdot P(\theta_T^u),
\end{aligned}
\tag{4.23}
$$

where the proportion is the result of the decomposition of the full joint distributions by Bayesian rules and the conditional independence of transitions.

Intuitively speaking, the conditional independence in $P(\{U_T^{\mathbf{N}(u)}\} | \{U_{T-1}^{\mathbf{N}(u)}\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u, \{U_{T-1}^u\}, \{U_T^u\})$ and $P(\{U_T^u\} | \{U_{T-1}^u\}, \theta_T^u, \theta_T^{\mathbf{N}(u)}, \{U_{T-1}^{\mathbf{N}(u)}\})$ reflects the transitional relations across two consecutive time frames. The conditional independence in $P(\{U_{T-1}^{\mathbf{N}(u)}\} | \{U_{T-1}^u\}, \theta_T^{\mathbf{N}(u)}, \theta_T^u)$ reflects the idea in CF that "like-minded" users behave similarly. The conditional independence in $P(\theta_T^{\mathbf{N}(u)} | \theta_T^u, \{U_{T-1}^u\})$ reflects the previous assumption that "like-minded" users have similar tendencies that are captured by the model parameters in their dynamic systems. The marginal independence in the distribution $P(\{U_{T-1}^u\} | \theta_T^u)$ can also be derived from previous two conditional independence. This is also intuitive because both previous latent factors U_{T-1}^u and the model parameters θ_T^u in the time window are required to predict the current latent factors U_T^u. In other words, U_{T-1}^u and θ_T^u have "explaining away" [36] phenomenon. Meanwhile, the conditional independence leading to $P(\{U_T^{\mathbf{N}(u)}\} | \{U_{T-1}^{\mathbf{N}(u)}\}, \theta_T^{\mathbf{N}(u)})$ is also derived similarly. The last proportion is the result of dropping the terms that are not relevant to the latent factors and model parameters of user u that is under inspection. Figure 4.1 shows a portion of the graphic model relating to this conditional independence in the T-th time window for users. The graphic model

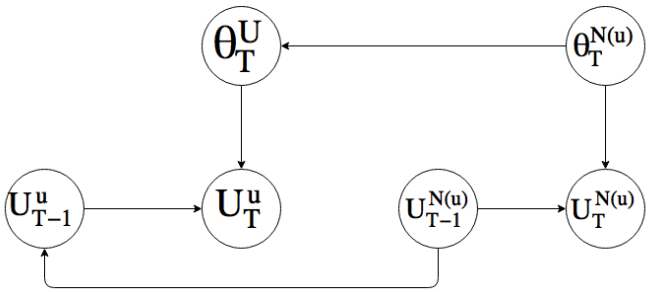


Figure 4.1: A portion of the graphic model for users in the t-th time window.

for items in the T-th time window is symmetric and omitted here for clarity. If there is an edge linking from node $\{U_T^{\mathbf{N}(u)}\}$ to node $\{U_T^u\}$ in the graphic model, this Bayesian network becomes fully connected and there are no conditional independence relations to be exploited to simplify the computational complexity of the learning procedure. Because all the user latent factors at time T, no matter whether the users suffer from the *cold start transition* problem or not, are able to be learned directly from the training data, the edge linking node $\{U_T^{\mathbf{N}(u)}\}$ and node $\{U_T^u\}$ is released in the developed model.

According to collaborative tendencies, the transited user preferences from U_{T-1}^u with θ_T^u should be close to user preferences U_T^u learned from data up to T. The distribution $P(\{U_T^u\}|\{U_{T-1}^u\}, \theta_T^u)$ is thus defined as follows,

$$P(\{U_T^u\}|\{U_{T-1}^u\}, \theta_T^u) = \prod_{d=1}^{D} exp\{-\frac{1}{2}\|U_T^{u,d} - f_u(U_{T-1}^{u,d}, \theta_T^u)\|^2\}. \tag{4.24}$$

As the latent factors of "like-minded" users should be close to each other in latent space, the distribution $P(\{U_{T-1}^{\mathbf{N}(u)}\}|\{U_{T-1}^u\})$ is defined as follows,

$$P(\{U_{T-1}^{\mathbf{N}(u)}\}|\{U_{T-1}^u\}) = \prod_{d=1}^{D} exp\{-\frac{1}{2}\sum_{j\in\mathbf{N}(u)}(w_{uj}\|U_{T-1}^{u,d} - U_{T-1}^{j,d}\|^2), \tag{4.25}$$

where w_{uj} denotes the similarity between users u and j that is computed based on their historical ratings as in user-based CF with the Pearson's correlation [115]. The neighborhood of user u is also determined by w_{uj}.

Based on collaborative tendencies, the distribution $P(\theta_T^{\mathbf{N}(u)}|\theta_T^u)$ is defined as follows,

$$P(\theta_T^{\mathbf{N}(u)}|\theta_T^u) = exp\{-\frac{1}{2}\sum_{j\in\mathbf{N}(u)}(w_{uj}\|\theta_T^u - \theta_T^j\|^2)\}. \tag{4.26}$$

The distributions $P(\{U_T^u\})$ and $P(\theta_T^u)$ are defined to function as regularization terms to prevent overfitting. They are defined as $P(\{U_T^u\}) = exp\{-\frac{1}{2}\sum_{d=1}^{D}\|U_T^{u,d}\|^2\}$ and $P(\theta_T^u) = exp\{-\frac{1}{2}\|\theta_T^u\|^2\}$, respectively.

Derivatives By using the α-derivative similar to Section 4.4.2, the gradients of these parameters with respect to the logarithm of Eq (4.23) can be obtained. Because only the interconnection weights H_T^u and the bias term b_T^u in the transition models have an analytical solution, Eq (4.23) is alternatively maximized over $\{U_{T-1}^u\}$ and θ_T^u via the gradient descent method.

Let J_1 denote the logarithm function of $P(\{U_T^u\}|\{U_{T-1}^u\},\theta_T^u)\cdot P(\{U_{T-1}^{\mathbf{N}(u)}\}|\{U_{T-1}^u\})\cdot P(\{U_{T-1}^u\})$, which contains all the terms relevant to latent factors. Similarly, let J_2 denote the logarithm function of $P(\{U_T^u\}|\{U_{T-1}^u\},\theta_T^u)\cdot P(\theta_T^{\mathbf{N}(u)}|\theta_T^u)\cdot P(\theta_T^u)$, which contains all the terms relating to model parameters to be inspected. Let $\Psi(u,d,l)$ denote $(U_{T-1}^{u,d}-\mu_l^u)^T\Sigma^{u^{-1}}(U_{T-1}^{u,d}-\mu_l^u)$ for user u's d-th sample in the sampled set for l-th kernel.

By using the α-derivative, the gradients of these parameters are computed as follows,

$$\begin{aligned}\frac{\partial J_1}{\partial U_{T-1}^{u,d}} =&(\sum_{l=1}^{L} h_l^u exp\{-\frac{1}{2}\psi_l(U_{T-1}^{u,d},\theta_U^u)\}\cdot(-(U_{T-1}^{u,d}-\mu^{u,l})^T\Sigma^{u^{-1}}))^T\cdot\\ &(U_T^{u,d} - f_u(U_{T-1}^{u,d},\theta^u)) + \sum_{j\in\mathbf{N}(u)} w_{uj}(U_{T-1}^{u,d}-U_{T-1}^{j,d}) + U_{T-1}^{u,d},\end{aligned} \tag{4.27}$$

and

$$\begin{aligned}\frac{\partial J_2}{\partial \mu^{u,l}} =&(\sum_{d=1}^{D}(h_l^u\cdot exp\{-\frac{1}{2}\psi_l(U_{T-1}^{u,d},\theta_U^u)\}(U_{T-1}^{u,d}-\mu^{u,l})^T\Sigma^{u^{-1}})\cdot\\ &(U_T^{u,d} - f_u(U_{T-1}^{u,d},\theta^u))) + \sum_{j\in\mathbf{N}(u)} w_{uj}(\mu^{u,l}-\mu^{j,l}) + \mu^{u,l},\end{aligned} \tag{4.28}$$

and

$$\begin{aligned}\frac{\partial J_2}{\partial \Sigma^u} =& \sum_{d=1}^{D}(\sum_{l=1}^{L}((U_{T-1}^{u,d} - \mu^{u,l})^T {\Sigma^u}^{-1})^T \cdot (U_T^{u,d} - f_i(U_{d,T-1}^u, \theta^u))^T \cdot \\ & -\frac{1}{2} h_l^u exp\{-\frac{1}{2}\Psi(u,d,l)\} \cdot ((U_{T-1}^{u,d} - \mu^{u,l})^T {\Sigma^u}^{-1})) + \\ & \sum_{j \in \mathbf{N}(u)} w_{ij}(\Sigma^u - \Sigma^j) + \Sigma^u. \end{aligned} \tag{4.29}$$

Similar to the derivation in Section 4.4, the interconnection weights and the bias term in the transition models have an analytical solution as follows,

$$\begin{aligned}[H^u \ b^u] =& (\sum_{j \in \mathbf{N}(u)} w_{uj}[H_j \ b_j] + \sum_{d=1}^{D} w_{uj}(U_T^{u,d}[(\Phi_d(U_{T-1}^u))^T \ \ 1]) \cdot \\ & (\sum_{d=1}^{D}[(\Phi^u(U_{T-1}^{u,d}))^T 1]^T [(\Phi^u(U_{T-1}^{u,d}))^T \ \ 1] - (\sum_{d=1}^{D} w_{uj} + 1)\mathcal{I})^{-1}. \end{aligned} \tag{4.30}$$

where $\mathcal{I}_{L+1}$ is a $L+1 \times L+1$ identity matrix. For clarity, the subscript t denoting that these parameters belong to the t-th time window in all the above notations are ignored. Analogously, the learning algorithm for items that suffer from the *cold start transition* problem can be derived.

4.5.3 Computational Complexity

By replacing the summations over smoothed particles in previous equations in previous sections with smoothed canonical particles, those equations can be approximated to reduce their complexity. Similar to the canonical particles for particle filtering, the smoothed expectations for user u and item v at time m in the t-th time window are estimated as $\hat{U}_{m|t}^u = \sum_{s=1}^{S} w_{U,m|t}^{u,(s)} U_{m|t}^{u,(s)}$, and $\hat{V}_{m|t}^v = \sum_{s'=1}^{S} w_{V,m|t}^{v,(s')} V_{m|t}^{v,(s')}$, respectively.

Because PMF [197] is among the fastest state-of-the-art CF methods, its time complexity is used as a reference here. The complexity of PMF is $O(|R| * K' * epoch)$, where $|R|$ is the number of ratings, K' is latent factor dimension in PMF and *epochs* is the number of iterations used. The time complexity of PMF is linear with respect to $|R|$. The time complexity of DynTranPF is dominated by computing the gradient of the covariance matrix, which is

$O(S{*}K^3{*}L)$. Because $\{L, K, t\} \ll min(N, M)$, $K'{*}epoch \approx S{*}K^2$ and $|R| \approx (M{+}N){*}K$, the time complexity of DynTranPF is approximated as $O(|R| * L * K' * epoch)$. This complexity is still linear in terms of $|R|$. Similar to the analysis for DynTranPF, the time complexity of LearnColdTran is $O((N + M) * D * L * K^3) \approx O(|R| * K' * epoch)$ that is also linear in $|R|$.

4.6 Experiments

In order to study the performance of the proposed methods in solving the *cold start transition* problem to flexibly and individually model user preferences and item attractiveness in dynamic RSs, the experiments in this section are divided into two parts. The first part focuses on the evaluation of the proposed method on the Top-N recommendation task, and the second part aims to verify the effectiveness of proposed methods of handling those users and items having the *cold start transition* problem. For both parts, the proposed methods are evaluated on the Movielens 100K (MovieLens) [1], HetRec [3] and Amazon Video Games (VideoGames) datasets [4]. The datasets adopted in this part of experiments are slightly different from the datasets adopted in the experiments conducted in Chapter 3. The commonly used benchmark datasets in RSs, Movielens 100K and HetRec datasets, are retained, which are about movie ratings. In order to explore the performance of the proposed method for datasets with different characteristics, Netflix used in previous experiments is replaced by VideoGames.

4.6.1 Protocol

Ratings are grouped based on the time frame in which their timestamps are. Ratings before a predefined time instance are used as training data, and ratings after it are test data. This setting is preferred over a random split of all the data. As in a real-world deployment, it is infeasible to generate the prediction using information in future.

Data Training periods for MovieLens, HetRec and VideoGames are September 1997 $\sim$ 1-st week of 1998, January 2007 $\sim$ December 2007 and January 1999 $\sim$ December 2011, respectively. Their testing periods are 2-nd week $\sim$ 16-th week of 1998, January 2008 $\sim$ December 2008 and January 2012 $\sim$ December 2012, respectively. Compared with the Hetrec dataset adopted in the experiments in Chapter 3, the training period of HetRec dataset is set to a short period to reduce the computational power that is spent on training. Meanwhile, to enable the proper training of transition models for users that do not have cold start problems, the 1st week of 1998 is also included in the training period of MovieLens dataset. Similar to the setup in Chapter 3, different time units are selected to ensure that ratings in a time slot are not too sparse. For example, after removing these users and items that give or receive less than 10 ratings, VideoGames has $480,189$ users and $17,770$ items, which has the overall sparsity of 98.6% and the sparsity of 99.98% in the last frame. With such units, the time window size n is set to 1 to avoid too much intervention from historical data and rapidly react to sudden changes in the tendency.

Metrics Similar to experiments in Section 3.8.1 in Chapter 3, the metrics precision@k, recall@k and Top-N hitrate recall [66] are utilized to assess the performance of the Top-N recommendations. Meanwhile, items with maximum rating value are treated as relevant items.

In a real world scenario, when the recommendation list itself becomes large, the list will be obsolete since people prefer only top listed items [24, 206]. Hence, it is set to $k = 10$ for all the metrics. For all the sampling-based methods, the number of particles is set to 1000 to balance their accuracy and scalability. For simplicity, the step sizes σ for user and item latent vectors in the proposal distribution in sampling methods are set to be equal. Other parameters in sampling methods and other baseline methods are separately tuned to achieve their best performance. Tuning is conducted under temporal recall and the identical settings are used for other metrics. In order to make the comparison fair, DynTranPF and FixTranPF adopt the same set of parameters. For clarity, only parameters in DynTranPF are presented in the following. It is also set $L = 10$ and

$|\mathbf{N}(i)| = 10$ over all the datasets. Following experiments in Chapter 3, the imputed value is set to $r_{miss} = 1.95$ and $w_{im} = 0.02$. As mentioned in Chapter 3, the effectiveness of RSs with sparse datasets is usually not high [243]. Hence, the focus will be placed on the relative improvement of algorithms instead of comparing their absolute performance.

4.6.2 Baseline Methods

The developed methods are compared with PMF, ST-PFUV [160], STKF [158], FixTranPF, RandColdTran, DynTopPopular, DynPureSVD, and DynAllRank. Latent factors in ST-PFUV and the developed methods are initialized by AllRank [212]. All the methods are repeated 10 times with means and standard deviations reported. Self-training particle filtering for MF (ST-PFUV) is selected to verify whether the explicit modeling and updating of dynamic systems in DynTranPF is able to improve the performance of temporal RSs. It is also chosen to demonstrate the benefits of building the personalized and item-wise transition models that impose no constraints on the structure of the temporal dynamics of user and item latent factors. Note that ST-PFUV adopts the first-order multivariate normal random walk as the transition model for all the users and items. ST-PTBM is also presented in Chapter 3. However, because DynTranPF is developed to enhance the flexibility of modeling the dynamics of ST-PFUV, ST-PTBM is not very closely related to the developed method in this chapter and it is not compared in the following experiments. FixTranPF is identical to DynTranPF except that transitions are fixed to the initial ones. It is chosen to verify the benefits of dynamically updating personalized and item-wise dynamic systems in RSs. As will be shown later, due to the high data sparsity in data, it is essential to adapt the dynamic systems to catch up the tendency of user preferences and item popularity over time. Other baseline methods are those baseline methods used in the experiments in Chapter 3. The discussion related to those baseline methods is omitted here for clarity.

Analogues to the selection of baseline methods in Chapter 3, to confirm the necessity of exploiting temporal information, PMF is also tested as the only static method in the

following experiments. This method always predicts the ranking without updating model parameters. In the second part of experiments, RandColdTran is identical to LearnColdTran in Section 4.5 except that parameters for initial prediction are sampled from their priors instead of learning from objects that do not suffer from the *cold start transition* problem.

4.6.3 Learning Dynamics

Recall that the training data are grouped into T time frames in Section 4.5.1. As neither the cold start user problem nor the cold start item problem is the focus of this chapter, the experiments in this section do not consider these problems.

MovieLens The dimensions of latent factors are $K = 4$ for DynAllRank, $K = 13$ for DynPureSVD, $K = 12$ for PMF, and $K = 4$ for STKF. The number of selected negative samples [160] is 30 times the number of users and items in sampling methods. Other related parameters are set as the step size $\sigma = 0.2$, $K = 4$ and regularization coefficient $\lambda = 0.1$ in both DynTranPF and ST-PFUV, and $\lambda_\Sigma = 0.1$ in DynTranPF.

Results Table 4.1 shows the results of the compared methods under temporal accuracy metrics. The low values in the table are due to the fact that few relevant items exist for each user in a time frame. Compared with these baseline methods, DynTranPF has the best performance. In particular, DynTranPF outperforms all other dynamic methods. This result shows that the proposed method can effectively construct personalized and item-wise transition models and dynamically update the model parameters to improve the temporal performance of RSs on the Top-N recommendation task. This can be ascribed to the fine modeling and learning of temporal dynamics of user preferences and item characteristics in the proposed method. All the improvements brought by DynTranPF are statistically significant under both paired and unpaired t-tests [115] with $p = 0.05$.

Meanwhile, DynTranPF improves ST-PFUV over the temporal accuracy measurement by 11.31%, 5.01% and 16.07%, respectively. Recall that the developed method is tuned based

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
PMF	0.0555±0.0041	0.0910±0.0084	0.0632±0.0096
DynTopPopular	0.0818±0.0000	0.0988±0.0000	0.1238±0.0007
DynPureSVD	0.1028±0.0000	0.1975±0.0000	0.0605±0.0077
DynAllRank	0.0993±0.0038	0.1867±0.0158	0.1441±0.0141
STKF	0.0719±0.0000	0.1341±0.0000	0.1410±0.0042
ST-PFUV	0.1061±0.0024	0.2575±0.0087	0.1972±0.0081
FixTranPF	0.0809±0.0025	0.1751±0.0152	0.1447±0.0148
DynTranPF	*0.1181±0.0039*	*0.2704±0.0091*	*0.2289±0.0090*

Table 4.1: Results of the methods on the MovieLens dataset for temporal metrics (mean±standard deviation, the best performance in italic font).

on the recall metric during training and applying the identical setting to precision and the Top-N hitrate measurement.

Intuitively, the significant improvement brought by DynTranPF can be explained by three reasons. Firstly, the dynamic adaption of the transition models for each user and item at every time frame is capable of finely modeling the dynamics between user and item latent factors across time frames that represent the tendency of user preferences and item attractiveness. Secondly, as DynTranPF does not impose any assumptions on the structure of the tendency of user preferences and item attractiveness, the personalized and item-wise transition models are fully tailored to catch the diverse requirements of the modeled dynamics. Finally, the uncertainty in the model has been explicitly and effectively taken into account by the developed inference and learning algorithms. The improvement between DynTranPF and FixTranPF is also significant, showing the benefits to updating the transition models adaptively while tracking the tendency, which also blindly incorporates the uncertainties of model structures.

Meanwhile, it is obvious from Table 4.1 that all personalized and dynamic algorithms have better performance than that of the static PMF method under all the temporal metrics. This conclusion is expected because more and more information about users is gathered as the time passes.

Comparison of temporal behaviors In order to further evaluate temporal behaviors of DynTranPF, the average of accumulated improvement (AAI) over time defined in Section

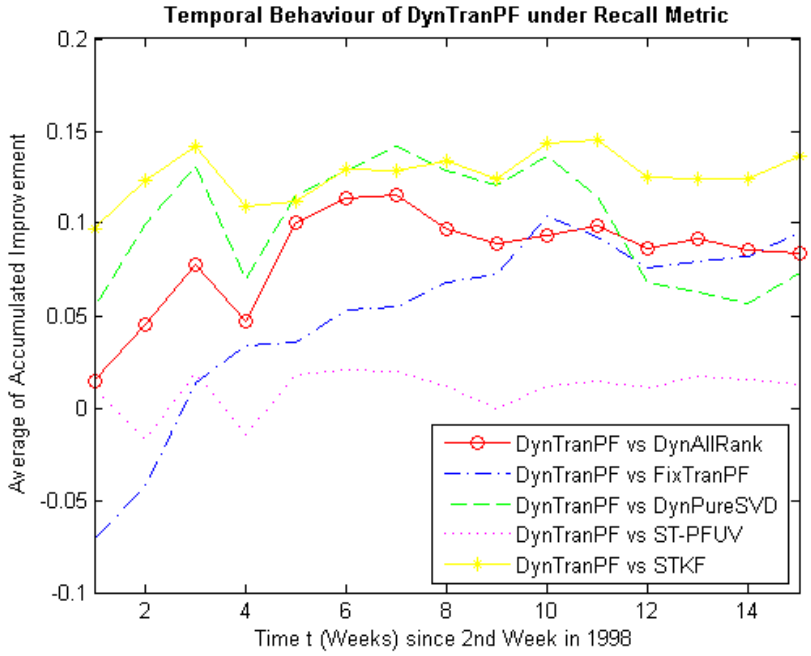


Figure 4.2: The average of accumulated improvement for DynTranPF on the MovieLens dataset since the 2-nd week in 1998.

4.6 in Chapter 3 is used.

Figure 4.2 plots the AAI metric among DynTranPF, FixTranPF, DynPureSVD, DynAll-Rank, ST-PFUV and STKF. Except the second month of the blue (dash-dotted) curve (DynTranPF vs FixTranPF), all the curves in the figure are above zero, showing that the developed method constantly outperforms baseline methods by taking advantage of the tailored and adaptive modeling of personalized and item-size temporal dynamics in user's feedback. The magenta (dotted) curve (DynTranPF vs ST-PFUV) shows that DynTranPF constantly improves ST-PFUV over time by flexibly modeling the temporal correlation between user preferences and item popularity and dynamically adjusting the dynamical systems. In addition, the yellow (stared) curve (DynTranPF vs STKF) shows that the temporal performance of RSs can be significantly improved by tracking the tendency of item attractiveness and coping with non-Gaussian behaviors.

Because the baseline method ST-PFUV significantly outperforms other baseline methods, the results of other baseline methods will be omitted in the following discussion for clarity,

which does not influence the conclusion in the chapter. Note that the results of the devised method FixTranPF will also be included for reference.

Hetrec Similar to previous experiments, DynTranPF has the best performance under all the temporal metrics as shown below. The number of selected negative samples in the two-phase sampling is set to 50. The related settings are σ=0.2, K=4 and λ=0.1 in both DynTranPF and ST-PFUV, and λ_Σ=0.1 in DynTranPF. As mentioned before, these parameters are incrementally tuned.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
ST-PFUV	0.0192±0.0007	0.1208±0.0027	0.0±0.0062
FixTranPF	0.0065±0.0003	0.0322±0.0014	0.0934±0.0029
DynTranPF	*0.0206±0.0005*	*0.1309±0.0033*	*0.3278±0.0051*

Table 4.2: Results of the methods on the Hetrec dataset for temporal metrics (mean±standard deviation, the best performance in italic font).

Results Table 4.2 shows the results of these methods under temporal metrics. DynTranPF significantly outperforms other methods in terms of all the temporal accuracy metrics. All the improvement brought by DynTranPF is statistically significant for both unpaired and paired t-tests [115] with p=0.05. It improves the temporal accuracy measurement over that of ST-PFUV by 2.99%, 7.27% and 3.07%, respectively. Recall that the focus will be placed on the improvement instead of the relative values and the developed methods are tuned based on the recall metric during training and applying the identical setting to precision and the Top-N hitrate measurement. This result verifies that DynTranPF can effectively construct the personalized and item-wise transition models over a long period.

Comparison of temporal behaviors Figure 4.3 plots the AAI among DynTranPF, FixTranPF and ST-PFUV.

All the curves in the figure are above zero, showing that the developed method constantly outperforms baseline methods over time. Compared with ST-PFUV, DynTranPF explicitly considers the temporal dependence among rich and diverse user preferences and item attractiveness across time frames by constructing the dynamical systems. As there are no

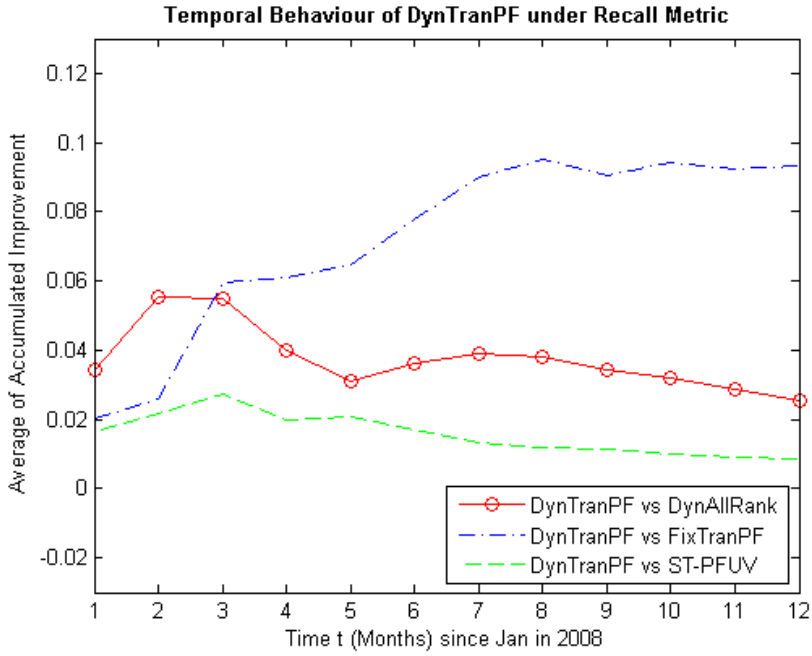


Figure 4.3: The average of accumulated improvement for DynTranPF on the Hetrec dataset since January 2008.

constraints imposed on the structure of the transition model, they are absolutely personalized and item-wise transitions. Moreover, for these users and items that are inactive in some time frames, the inertia of finely modeled dynamics in DynTranPF is more informative and discriminative than the random search conducted in ST-PFUV. In addition to explicitly considering the uncertainties in model structures, the inference and learning algorithm also aims to find a balance between exploitation and exploration in the latent space. By contrast, ST-PFUV only searches the latent space randomly, and it is prone to the divergence of particle filtering over a long period.

After doing some exploratory analysis on the Hetrec dataset, it is found that the slight decrease (maximum 0.0053% between the 11-th month and the 12-th month) of the tendency of the red (circled) curve (DynTranPF vs DynAllRank) is due to the rapidly decreasing of active users in the dataset over time (only 16.64% active users remained in the final time frame). As there is less new information available at each time frame as the time passes, all these methods are overinflunced by the historical data and they are attempt-

ing to extract repeatedly transitional relations relating to the temporal importance of the historical data. Therefore, under such a situation, this *retrospective* improvement made by DynTranPF over DynAllRank and ST-PFUV has been reduced.

VideoGames The factor in the two-phase sampling is set to 20. The related settings of parameters are σ=0.2, K=4 and λ=0.1 in ST-PFUV. For DynTranPF, the settings are λ_Σ=0.001 and σ=0.2, K=4, and λ=0.1.

Results Table 4.3 shows the results for temporal accuracy metrics. DynTranPF achieves the best performance for all metrics. The results show that DynTranPF works successfully on a reasonably large and extremely sparse real-world dataset. Except for temporal hitrate, all the improvement brought by DynTranPF is statistically significant for both unpaired and paired t-tests with $p = 0.05$.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
ST-PFUV	0.0186±0.0002	0.1158±0.0008	0.3250±0.0023
FixTranPF	0.0143±0.0002	0.0885±0.0010	0.2552±0.0024
DynTranPF	0.0208±0.0009	0.1294±0.0011	*0.3264±0.0006*

Table 4.3: Results of the methods on the Amazon VideoGames dataset for temporal metrics (mean±standard deviation, the best performance in italic font).

Comparison of temporal behavior

Except for the initial time frame, both of the curves are above zero in Figure 4.4, showing that the developed method constantly outperforms baseline methods over time by taking advantage of the fine modeling of temporal dynamics in user preferences and item attractiveness. The blue (dash-dotted) curve (DynTranPF vs FixTranPF) illustrates that DynTranPF improves FixTranPF over time by dynamically updating the dynamical systems for user preferences and item popularity. This improvement verifies the benefits of the developed online adaptive ELMs in DynTranPF. As mentioned in Section 4.3, it is essential to update both the kernel parameters and interconnection weights in the machines dynamically to cope with the noisy user feedback and the artifacts introduced by imputation at each time frame. Meanwhile, to tailor the dynamical systems to meet the

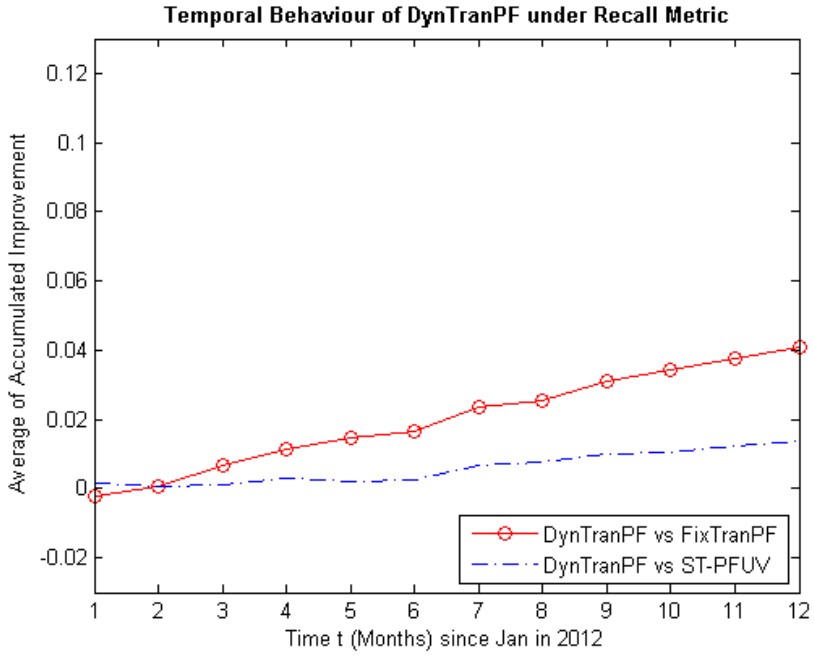


Figure 4.4: The average of accumulated improvement on the VideoGames dataset since January 2012.

specifications of each user and item, it is also necessary to dynamically incorporate the uncertainties of model structures at each time frame as does in DynTranPF.

In addition to the benefits explained from DynTranPF vs FixTranPF, the red curve (DynTranPF vs ST-PFUV) also shows that the temporal performance of RSs can be significantly improved by using the personalized and item-wise temporal dynamics. This set of fully tailored dynamical systems not only finely models the temporal information and dynamic structures of user and item latent factors across time frames, but also functions as a set of useful components by providing the tracking and prediction procedures with the meaningful and specific inertia of user behaviors and item properties over these time frames where users and items are inactive.

4.6.4 Computational time

As a sequential approach, the proposed methods dynamically learn the model parameters and track user and item latent factors. This approach does not change the computational complexity of prediction. Thus, the total running times at each time frame (including both training or retraining time and prediction time) are averaged to compare the efficiency of these methods. The algorithms are implemented in Matlab and run on a 4-core machine with 2.50G Hz CPU and 32G memory. The running times of these methods under temporal recall metric on the Hetrec and VideoGames datasets are listed in the first and third lines in Table 4.4, respectively. Their running times under other temporal metrics have the similar effects and they are ignored for clarity.

As mentioned before, it is also straightforward to speed up DynTranPF by parallelizing the Monte Carlo method and learning algorithm. The running times of these parallelized versions (with 4 threads) are listed in the second and fourth lines in Table 4.4. A parallelized version of PMF is not implemented. The key observation in the table is that DynTranPF can be greatly speeded up by parallelization. Therefore, it is reasonable to expect the running time of DynTranPF is comparable with baseline methods when it is deployed on more sophisticated machines. Note that PMF has been among the fastest state-of-the-art CF methods.

Method	PMF	ST-PFUV	DynTranPF-User	DynTranPF
Hetrec time	0.0	*0.0*	0.0	0.0
Hetrec parallel	-	*41.8*	115.3	191.8
VideoGames time	0.0	*0.0*	0.0	0.0
VideoGames parallel	-	*148.9*	0.0	259.8

Table 4.4: The average running times on the Hetrec and VideoGames datasets (the unit is second and the best performance in italic font).

4.6.5 Cold Start Transition Problem

This subsection focuses on the study of the effectiveness of the proposed method to tackle the *cold start transition* problem. Due to the properties of video games, the number

of users and items with such a problem are quite stable over time in the VideoGames dataset. Therefore, 5% of users are randomly marked as suffering from the *cold start transition* problem in the VideoGames dataset. Meanwhile, items in the corresponding tuples (user, item and the rating) become the candidates for items with the *cold start transition* problem, which leads to 1.05% items having the *cold start transition* problem. Table 4.5 summarizes the number of users and items that do not suffer from the *cold start transition* problem during the testing period in those three datasets adopted in the experiments.

object	MovieLens	Hetrec	VideoGames
user	530	1206	13175
item	1493	7497	17485

Table 4.5: The number of users and items that do not suffer from the *cold start transition* problem during the testing period.

Table 4.6, Table 4.7 and Table 4.8 demonstrate the performance of RandColdTran and LearnColdTran on the 1-st time frame during the testing period. All the improvement brought by LearnColdTran is statistically significant for both paired and unpaired t-tests [115] with $p = 0.05$. The results verify that these users and items with the *cold start transition* problem indeed have similar temporal patterns as their "like-minded" users and similar items without such a problem, and LearnColdTran does benefit from collaborative tendencies and collaborative observation models. They also verify that LearnColdTran can generate accurate initial model parameters for these users and items suffering from the *cold start transition* problem. For example, LearnColdTran improves the temporal accuracy measurement over that of RandColdTran by 7.61%, 16.19% and 9.24% on the VideoGames dataset, respectively. Recall that all the parameters are tuned under the temporal recall metric.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
LearnColdTran	*0.0709±0.0048*	*0.1502±0.0173*	*0.1124±0.0327*
RandColdTran	0.0180±0.0016	0.0698±0.0039	0.0823±0.0050

Table 4.6: Results of methods on the MovieLens dataset for the 1-st time frame (mean±standard deviation). The best performance is in italic font for each metric.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
LearnColdTran	*0.0247±0.0015*	*0.0936±0.0103*	*0.2505±0.0138*
RandColdTran	0.0178±0.0010	0.0788±0.0062	0.2141±0.0158

Table 4.7: Results of methods on the Hetrec dataset for the 1-st time frame (mean±standard deviation). The best performance is in italic font for each metric.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
LearnColdTran	*0.0099±0.0002*	*0.0603±0.0028*	*0.2389±0.0077*
RandColdTran	0.0092±0.0002	0.0519±0.0015	0.2187±0.0047

Table 4.8: Results for temporal metrics on the VideoGames dataset at the 1-st frame (mean±standard deviation). The best performance is in italic font for each metric).

It is also useful to verify the benefits of a good initial transition for dynamically updating transition models over time. Hence, the comparisons will be conducted on the temporal performance of two versions of DynTranPF that are separately initialized by RandColdTran and LearnColdTran. These methods are named as DynTranRand and DynTranLearn, respectively. Table 4.9, Table 4.10 and Table 4.11 illustrate their performance.

Compared with results in those three tables with those results in Table 4.6, Table 4.7 and Table 4.8, even though the improvement has been reduced, a good initialization generated by LearnColdTran still leads to a superior temporal performance. This reduction is not unexpected. As the time goes, users are providing more feedback while items are receiving more ratings. Therefore, these users and items suffering from the *cold start transition* problem will be gradually getting over this problem. This is especially true for the VideoGames dataset, where users and items with such a problem are randomly introduced as a small portion of the total population. However, it is desirable for RSs to provide these users suffering from the *cold start transition* problem with some reasonably satisfied recommendations when little feedback is available. Otherwise, these users may lose their trust in the systems and depress the usage of them. Furthermore, if the problems of cold start users or items are considered at the same time, the *cold start transition* problem may exist in each time frame. Therefore, it is rational to expect that DynTranRand will fail to generate appealing recommendations over time.

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
DynTranLearn	*0.1098±0.0031*	*0.2319±0.0071*	*0.1971±0.0083*
DynTranRand	0.1029±0.0028	0.2290±0.0097	0.1922 ±0.0083

Table 4.9: Results for temporal metrics on the MovieLens 100K dataset over all the time frames (mean±standard deviation). The best performance is in italic font for each metric).

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
DynTranLearn	*0.0216±0.0005*	*0.1152±0.0029*	*0.2685±0.0069*
DynTranRand	0.0193±0.0006	0.1053±0.0041	0.2614±0.0037

Table 4.10: Results for temporal metrics on the Hetrec dataset over all the time frames (mean±standard deviation). The best performance is in italic font for each metric).

Method	$prec_t(k)$	$recall_t(k)$	$top_t(k)$
DynTranLearn	*0.0202±0.0002*	*0.1269±0.0010*	*0.3180±0.0017*
DynTranRand	0.0196±0.0008	0.1220±0.0012	0.3026±0.0067

Table 4.11: Results for temporal metrics on the VideoGames dataset over all the time frames (mean±standard deviation). The best performance is in italic font for each metric).

4.7 Summary

A novel probabilistic personalized and item-wise model has been presented to tackle the *cold start transition* problem in learning the tendency of user preferences for RSs. To fully adapt to the rich and diverse dynamical systems for various users and items, an online adaptive ELM is developed to model the temporal and dynamic information intrinsic in users' historical feedback. By exploiting the historical feedback and temporal interactions from "like-minded" users and similar items, a new inference and learning algorithm, considering the model uncertainties for emulating the interested tendency, is also developed for the Top-N recommendation task over time. The algorithm tracks latent factors representing user preferences and item attractiveness and adaptively updates model parameters to put more emphasis on the current trend. The *cold start transition* problem, which is particularly outstanding in dynamic RSs, is solved by learning from collaborative tendencies.

The proposed methods are evaluated on three real-world benchmark datasets under the temporal extensions of accuracy metrics. The experimental results demonstrate that those

methods significantly improve the recommendation performance over a variety of state-of-the-art algorithms. The results confirm that those methods can effectively model the temporal dynamics of rich and diverse user preferences and item preferences. For users and items that are inactive for some time frames, the experiments also verify that it is more reasonable to assume the tendency follows the inertia of the modeled dynamics instead of just randomly searching the latent space. The learning algorithm that dynamically updates model parameters is shown to be able to handle the introduced model uncertainties and efficiently guide the propagation of latent factors. The experiments also illustrate the advantages of temporal dynamic model over static ones and the benefits of tracking both user preferences and item attractiveness instead of tracking merely user preferences. The experimental results relating to the *cold start transition* problem also illustrates that the proposed learning algorithm can accurately learn the missed transition systems by utilizing knowledge from similar users and items.

Up to the current discussion, the research has focused on the modeling of temporal dynamics of latent factors, such as their transitional relations, to capture the tendency of user preferences and item attractiveness. Nevertheless, the temporal dynamics of the variations of latent factors also convey rich and helpful information to reflect the tendency of user preferences and item attractiveness in RSs, which will be the emphasis of the next chapter.

Chapter 5

Bayesian Wishart Matrix Factorization

Existing methods to exploit temporal dynamics in RSs, such as the discussed methods in Chapter 3 and Chapter 4, usually focus on the modeling of transitional relations of user preferences and item attractiveness across time intervals. The temporal dynamics of variations of user preferences and item attractiveness are largely neglected. However, the fluctuation of dynamics also conveys meaningful information and should be worth exploring. By modeling the temporal dynamics of those variations, many sudden changes or local temporal effects in the tendency of user preferences and item attractiveness could be well modeled and properly controlled. Meanwhile, it is more interesting and challenging to predict the preferences of those users that do have diverse preference patterns. Those diverse behaviors of those users are also taken into account by modeling the temporal dynamics of variations. Moreover, while modeling the temporal variations in a hierarchical way, the temporal effects can also be shared across time frames, which should be beneficial to mitigate the problem of data sparsity in RSs. In order to achieve the desired properties described above, a Bayesian MF-based CF method is developed to model the temporal dynamics of variations among user preferences and item attractiveness from a totally different algorithmic perspective.

5.1 Introduction

Methods developed in Chapter 3 and Chapter 4 are based on the state space based approach, whose state space consists of latent vectors representing user preferences and item attractiveness. This approach is a popular technique that exploits temporal information to boost the performance of CF in RSs [80, 158]. Through the usage of state space and the dynamic systems built on it, this approach not only has a solid theoretical ground but also seamlessly exploits the temporal dynamics and the collaboration among entities over time. The first order Gaussian random walk is usually adopted as the de facto dynamic system in RSs due to its simplicity and a lack of the prior knowledge about user preference and item attractiveness. Methods of this approach demonstrate a promising performance according to the reported experiments.

Along other direction, some extra latent vectors have been used to take a specific care of temporal and dynamic information in user feedback. Those methods enhance the capability of traditional models that only considers user preferences and item preferences in a static way. Some particular constraints or conditions depicting the tendency of user preferences and item attractiveness are then constructed in terms of the expected behaviors of those temporal latent vectors. For example, it is straightforward to extend the static models with a temporal constraint that requires user latent factors vary smoothly across two consecutive time frames. By optimizing the objective function of latent factors with those temporal constraints, user preferences over items at various time frames can be predicted.

It is a fundamental problem in data mining and machine learning to model the dependencies among random variables or vectors. Therefore, the modeling of covariance matrices receives many attentions because it is the simplest measure of dependency. Nevertheless, this is not the case in existing RSs, which usually assume that covariance matrices are nearly constant over time for simplicity. Under temporal context, those matrices are traditionally either empirically tuned or predefined in dynamic models [80, 158, 160] or simply learned to fit some localized distributions that do not explore and share the temporal infor-

mation across time frames. More importantly, it is more useful and challenging for RSs to generate recommendations for those users that have diverse feedback patterns. However, by imposing the universal and static assumption on those dependencies, existing methods usually do not emphasize on modeling the temporal behaviors of such users. In summary, the aspects relating to temporal dynamics of covariances have been largely neglected in RSs.

In this chapter, the proposed method aims to fill out this gap by imposing priors to model the temporal dynamics of variations of user preferences and item attractiveness in order to improve the performance of RSs. The work introduces a novel and orthogonal dimension that concerns the dynamic covariance matrices so that the performance of RSs can be improved by relying on the modeling over this largely neglected direction, especially for modeling the temporal behaviors of those users that do have diverse feedback patterns. In addition to imposing the priors on the temporal dynamics of covariance matrices, it also seems possible to have priors imposed on the transitional relations of user preferences and item attractiveness at the same time. However, this approach may make the final model too complex to apply to the real world scenarios for RSs. Therefore, this approach is not considered in this chapter.

In the proposed method, the generalized Wishart process (GWP) [236] will be used to model the priors to enable the static MF method [197] to handle the temporal and dynamic aspects of the evolving data. It also finely models the temporal dynamics of user preferences for those users having diverse feedback pattern. The GWP process is a stochastic process that defines a prior for covariance matrices over time. It alleviates the problems of existing multivariate volatility models [28], such as poor scalability on the dimension of covariances and a lack of generality of learning and inference procedures. This stochastic process is a collection of positive semi-definite random matrices indexed by any arbitrary input variable, and it is developed to model the index varying or dynamic covariances. For simplicity, the index is referred to be time index. To the best of the authors' knowledge, the proposed method is the first work that integrates the priors over dynamic covariance matrices into MF and then applies to the problem of dynamic recommendations.

Specifically, the proposed method uses two sets of latent vectors to represent compactly user preferences and item attractiveness at each time frame. The initial settings of those latent vectors are learned by a probabilistic matrix factorization (PMF) method. Then, the proposed system employs the GWP process as the priors for the dynamic covariance matrices of user latent vectors and item latent factors, respectively. The changing of user preferences and item popularity is thus modeled and controlled by temporal dynamics of those covariances of latent vectors[1]. The temporal behaviors of those users that do have diverse feedback patterns are also taken care of by the proposed method. By modeling the temporal dynamics of those dynamic covariance matrices, the influences of temporal fluctuation and smoothness constraints, which are usually complicated and conflicted among diverse users and items, can be flexibly modeled and efficiently reconciled. Due to the symmetry in the developed method, the temporal behaviors of those items that do have received the diverse feedback are also taken into account. For the personalized recommendation, a personalized list of predicted ratings on unseen items for each user is generated based on the learned user and item latent vectors.

The main contributions of this chapter are [159]:

1) a novel Bayesian Wishart matrix factorization method is developed for RSs to model the tendency of user preferences and item attractiveness effectively via the direct controlling of temporal dynamics of covariances of user and item latent vectors. The developed method takes into account the temporal behaviors of those users and items that have or receive diverse feedback;

2) an effective and efficient inference algorithm is developed for the proposed Bayesian model by combining the collapsed Gibbs sampling method [21] and elliptical slice sampling method [177];

3) it is experimentally shown that the proposed model and learning algorithm lead to the improvement in the temporal performance of RSs for personalized rating prediction on public benchmark datasets.

[1]Latent factors and latent vectors are used interchangeably in this chapter.

This chapter is organized as follows. A brief discussion of the related work is presented in Section 5.2. Section 5.3 briefly describes Bayesian MF method as the preliminary. The proposed model is discussed in detail in Section 5.4. In Section 5.5, the inference algorithm of the proposed model is covered. The performance of proposed method is compared with a variety of baseline methods in Section 5.6. The summary is presented in Section 5.7.

5.2 Related Work

Before the award-winning method timeSVD++ [127, 128] is proposed, the temporal information in user feedback has been largely ignored in the development of CF. After bucketizing the user feedback into time frames, timeSVD++ captures only the *local* changes in user preferences over time. Compared with PMF, timeSVD++ introduces extra latent vectors that are interpreted in a way to capture temporal behaviors within the specific time frame. Temporal constraints are also added into the objective function for the optimization problem. Unlike the proposed method, this method is not a Bayesian treatment and faces complicated parameter tuning. Meanwhile, the prediction for users' interests in timeSVD++ is actually *post hoc* about what interests *would have been* in the past, rather than what interests would be in the future.

Bayesian probabilistic tensor factorization (BPTF) [241] extends Bayesian MF (BMF) [196] by introducing latent vectors on the time dimension to take care of the temporal information. Unlike the proposed method that models both the temporal and dynamic behaviors for each user and item, BPTF focuses on the overall effects of temporal information that are shared across all users and items. Meanwhile, like the popular timeSVD++ method, BPTF is also a binning-based approach [160]. Both of those methods predict user interests at a time instance using all the collected data. That is, the prediction is actually *post hoc* about what interests *would have been* in the past rather than in future, which conflicts with real-world scenarios.

While retaining the normal-Wishart priors over latent factors as adopted in BMF, user

latent factors are collapsed into user group latent factors in dynamic Bayesian probabilistic matrix factorization [56] at every time frame. A hierarchical Dirichlet process (HDP) [223] has been imposed over the grouping of users in BMF to share information across user latent factors over various time intervals. The HDP prior then determines the membership of every user at each time step. Nevertheless, the temporal dynamics of latent factors are not modeled across time intervals in [56]. In contrast, the GWP processes employed in the developed method explicitly models the temporal dynamics of variations of latent factors across time intervals. In addition to only considering the local effects captured by user group latent factors, item attractiveness in [56] is assumed to be slowly changed and item latent factors are thus assumed to be static and capture global effects. Because dynamic Bayesian probabilistic matrix factorization [56] is not a personalized approach to recommendations and focuses mainly on the modeling of dynamic memberships of users, this method is not compared in the experiments.

In addition, almost all of the existing methods that integrate temporal information into MF impose an implausible assumption of constant variance or covariances among latent vectors. This assumption neglects diverse dependencies among latent factors at each time frame. Therefore, it in turns ignores the diverse patterns of mutual influences among different characteristics of user preferences and item attractiveness. Moreover, this assumption ignores the trends of those dependencies across time frames, which only imposes some static and constant smooth constraints of the dynamics of those factors. It may prevent the methods from handling the sudden and dramatic changes in user preferences and item attractiveness over time. The proposed method releases this assumption by explicitly modeling the temporal dynamics of variations of user and item latent vectors. Furthermore, almost all of those methods require the exploitation of conjugate priors to facilitate the inference procedure in the developed models. Although this requirement reduces the computational complexity of inference, it may limit the applicability and flexibility of adopted priors. However, by exploiting more sophisticated sampling approach, the proposed method does not have this kind of limitation, which will be discussed in detail in Section 5.5.

There have been some other studies on exploiting temporal dynamics to improve the performance of RSs. However, those studies [158, 80, 160, 55] are mainly based on a state space approach, which uses the dynamic system to capture the transitional relations of latent vectors. For example, Kalman filtering method is used in [158, 80] only to track the trends of user latent factors in RSs. Different from modeling the explicit feedback as in the proposed method and BPTF, the Poisson distribution is exploited in dynamic Poisson factorization [55] to model the binary (implicit) feedback of the observation function at each time interval. Meanwhile, the method imposes the independent first order Gaussian random walk over *each* factor of latent factors and it focuses on the modeling of transitions of latent factors. The method in [160] releases the assumptions of linear and Gaussian dynamic system for user preferences and static item attractiveness in [158, 80, 55]. It designs an observation function that explicitly considers that user feedback is not missing at random. Because those methods do not aim to extend the applicability of Bayesian-based MF methods to temporal scenarios on explicit user feedback or they focus on the modeling of transitions of latent factors, the performance of those methods is not compared with the proposed method in the experiments in this chapter.

5.3 Preliminaries

Before discussing the proposed method, Bayesian matrix factorization (BMF) [196] will be briefly described as the preliminaries. BMF introduces a Bayesian treatment of probabilistic matrix factorization (PMF) [197], which has been widely used in RSs as a model-based CF method. PMF suffers from the high model control complexity and only comes up with a point estimation. Moreover, PMF may have a small degree of freedom to capture meaningful prior knowledge for latent vectors. This problem is particularly evident in RSs, where a large number of users and items makes user preferences and item characteristics very diverse. Therefore, BMF imposes the priors over both the means and covariance matrices of those latent vectors to overcome these problems. It also exploits the Gaussian-Wishart distribution as the prior to make its learning and inference stages

tractable.

5.3.1 Model

Assuming N users and M items, let $R \in \mathcal{R}^{N \times M}$ be a user-item preference matrix with an entry $r_{u,i}$ representing the rating given by user u to item i. Rating $r_{u,i}$ is assumed to be generated by a Gaussian distribution $P(r_{u,i}|U_u, V_i)$ conditioned on K-dimensional vectors U_u and V_i, which are the u-th row and the i-th row from corresponding user and item latent matrices $U \in \mathcal{R}^{N \times K}$ and $V \in \mathcal{R}^{M \times K}$. Furthermore, those latent vectors are assumed to be marginally independent while any rating $r_{u,i}$ is assumed to be conditionally independent of the presence of latent vectors U_u and V_i for user u and item i [196]. In addition to the latent vectors and ratings that are represented in PMF, $\Theta_U = \{\mu_U, \Lambda_U\}$ and $\Theta_V = \{\mu_V, \Lambda_V\}$ are used to denote the means and covariances of user latent vectors and item latent vectors, respectively.

Let ν_0 denote the degrees of freedom and W_0 be the $K \times K$ dimensional scale matrix. The Wishart distribution $\mathcal{W}$ is defined as $\mathcal{W}(\Lambda|W_0, \nu_0) = \frac{1}{C}|\lambda_0|^{(\nu_0 - D - 1)/2} exp(-\frac{1}{2} Tr({W_0}^{-1}\Lambda))$, where C is the normalization constant for this distribution and D is the number of the dimensions of Λ. For convenience, let Θ_0 denote the hyperparameters $\{W_0, \nu_0, \mu_0\}$. Thus, the Gaussian-Wishart priors on the user latent factors are defined as $P(\Theta_U|\Theta_0) = P(\mu_U|\Lambda_U)P(\Lambda_U) = \mathcal{N}(\mu_U|\mu_0, (\beta_0\Lambda_U)^{-1}) * \mathcal{W}(\Lambda_U|W_0, \nu_0)$. Without any prior knowledge, these parameters are usually predefined in the model [196]. For example, the degree of freedom ν_0 is usually set to $K+1$, the hyperparameter μ_0 is usually set to be a K-dimensional zero vector and W_0 is set to be a $K \times K$ identity matrix.

5.3.2 Inference

The MAP estimation of user and item latent vectors is traditionally used to train PMF with user feedback as input. This inference procedure results in a point estimation and is prone to overfitting. In order to overcome these issues, BMF integrates over all the possible

values of model parameter values and hyperparameters and utilizes the Gibbs sampling method [36] to learn those parameters and hyperparameters. By using conjugate priors, these conditional posteriors can be analytically derived and thus easily sampled.

5.4 Dynamic Matrix Factorization with Generalized Wishart Processes

In this section, a dynamic MF method that integrates the GWP processes into MF is discussed in detail. This integration enables the static method to exploit the temporal and dynamic information on user feedback from a perspective that is usually neglected in CF. For clarity and easy exposition, the explicit feedback from users is taken as input in the following discussion. As shown in Section 5.5, the inference procedure for the proposed model is very general because of no requirement of the usage of conjugate priors. Therefore, it is straightforward to apply this model to other types of user feedback by replacing the observation function in the model.

Unlike the traditional dynamic MF methods that impose the priors over the transitional relations among latent vectors across various time frames, the proposed method exploits temporal and dynamic information in RSs by modeling the temporal dynamics of covariance matrices $\Sigma_U(t)$ and $\Sigma_V(t)$ for user latent vectors U_t and item latent vectors V_t over time t. The proposed method finely models the relatively abrupt changes in temporal behaviors within time frames and the relatively stable trends in user preferences and item attractiveness in user feedback. This is because the method not only models the diverse and various dependencies among latent factors within each time frame, but also imposes flexible constraints over the dynamics of variations of latent vectors.

5.4.1 The Model

Let $\{u_l | u_l \in \mathcal{R}^K, l = 1, \ldots, \nu\}$ be a set of ν independent and identical distributed (i.i.d.) samples from a K-dimensional multivariate normal distribution $\mathcal{N}(0, W_0)$ with covariance W_0. A Wishart distribution $\mathcal{W}$ with a $K \times K$ dimensional scale matrix W_0 and ν degrees of freedom can be constructed as the summation of the outer products of u_l as $\mathcal{W}_K(W_0, \nu) = \sum_{l=1}^{\nu} u_l u_l^T$.

Inspired by the above procedure to construct the Wishart distribution, the GWP process for user latent factors can be constructed from some Gaussian processes. A Gaussian process is a stochastic process where any finite collection of random variables sampled from the process complies with a joint multivariate normal distribution. Let GP denote a Gaussian process. The distribution of the function $P(t)$, which works as the fundamental component to construct the GWP process, is assumed to have a Gaussian process prior as $P(t) \sim GP(m(t), k(t, t'))$. The function $m(t)$ is a mean function over input t, and $k(t, t') = cov(u(t), u(t'))$ is the kernel function that models the covariance $cov(\cdot, \cdot)$ of $P(t)$ between time t and time t'. Let $\{P_{l,d}(t) | l = 1, \ldots, \nu, d = 1, \ldots, K\}$ denote νK independent Gaussian process functions sampled from the Gaussian process $GP(0, k(t, t'))$. Due to the property of independence, it is straightforward to have the vector of $P_{l,d}(t)$ comply with a multivariate normal distribution as $(P_{l,d}(t_1), \ldots, P_{l,d}(t_N)) \sim \mathcal{N}(0, K)$, where K is $N \times N$ matrix with $K_{p,q} = k(t_p, t_q)$.

Let $\hat{P}_{U,l}(t) = (P_{U,l,1}(t), P_{U,l,2}(t), \ldots, P_{U,l,K}(t))^T$ denote a vector of samples at time t of the $l-$th degree of freedom for user latent factors. Let $\mathcal{V}$ denote a $K \times K$-dimensional scale matrix. For the GWP process, the means of the underlying Gaussian processes are restricted to $m(t) = 0$. With an additional requirement that $k(t, t) = 1$, a GWP process $\mathcal{GWP}_U(\nu, \mathcal{V}, k(t, t'))$ for user latent factors U_t at time t is constructed as follows,

$$\Sigma_U(t) = \sum_{l=1}^{\nu} L_U \hat{P}_{U,l}(t) \hat{P}_{U,l}(t)^T L_U^T \sim \mathcal{W}_K(\mathcal{V}, \nu), \tag{5.1}$$

where L_U represents the lower Cholesky decomposition of the scale matrix $\mathcal{V}$ for user latent factors such that $L_U L_U^T = \mathcal{V}$. The above construction ensures that the covariance matrix

$\Sigma_U(t)$ of the GWP process for user latent factors has a Wishart marginal distribution $\mathcal{W}_K(\mathcal{V}, \nu)$ at every time t [236]. The parameters in the GWP process are also easily interpretable. Intuitively, the scale matrix mainly works as the shape parameter while the kernel function focuses on controlling the dynamics of how covariance matrix $\Sigma_U(t)$ varies over time. This kind of parametrization clearly separates the contributions between the shape parameters and temporal dynamic parameters. In particular, the parameter L_U describes the expectation of $\Sigma_U(t)$ for all t, the kernel parameter $\theta_{\mathcal{P}_U}$ in $k(t, t')$ controls how the variation behaves over time and the degrees of freedom ν expresses how flexible the prior is allowed around the expectation of covariance matrix $\Sigma_U(t)$.

The graphical model of Bayesian Wishart MF method (BWMF) is depicted in Figure 5.1. For clarity, only a slice of the graphical model at time t and $t+1$ is illustrated. Similar to the assumption made in BMF, user and item latent vectors are also assumed to be marginally independent. Any rating $r_t^{u,i}$ in the t-th time frame is assumed to be conditionally independent, given the latent vectors U_t^u and V_t^i for user u and item i at time t. Inspired by BMF, user latent factors at time t are modeled as multivariate normal distribution as follows,

$$P(U_t|\mu_{U,t}, \Sigma_U(t)) = \mathcal{N}(U_t|\mu_{U,t}, \Sigma_U(t)) = \prod_{u=1}^{N} \mathcal{N}(U_t^u|\mu_{U,t}^u, \Sigma_U(t)), \tag{5.2}$$

where $\mu_{U,t}$ is mean vectors for user latent factors in the t-th time frame.

For user latent vectors, the covariance matrix $\Sigma_U(t)$ in the t-th time frame is modeled as the marginal distribution of the GWP process $\mathcal{GWP}_U(\nu, \mathcal{V}, k(t, t'))$,

$$P(\Sigma_U(t)|L_U, \nu, \hat{P}(t)) = \delta(\Sigma_U(t) - \sum_{l=1}^{\nu} L_U \hat{P}_{U,l}(t) \hat{P}_{U,l}(t)^T L_U^T), \tag{5.3}$$

where δ is the Dirac function. Without any prior knowledge, the prior distribution for L_U is defined as the spherical Gaussian prior as $P(L_U) = \mathcal{N}(L_U|0, \sigma_L \mathcal{I})$, where $\sigma_L \in R^+$, and $\mathcal{I}$ is a $K \times K$ dimensional identity matrix. In the experiments, it is set $\sigma_L = 1$ to let it behave as a vague prior.

Let $\mathcal{P}_U = \{\hat{P}_{U,l}(t)|l = 1, \ldots, \nu, t = 1, \ldots, T\}$ be the union of all the independent Gaussian process function values in the GWP process $\mathcal{GWP}_U(0, \mathcal{V}, k(t, t'))$ for user latent factors.

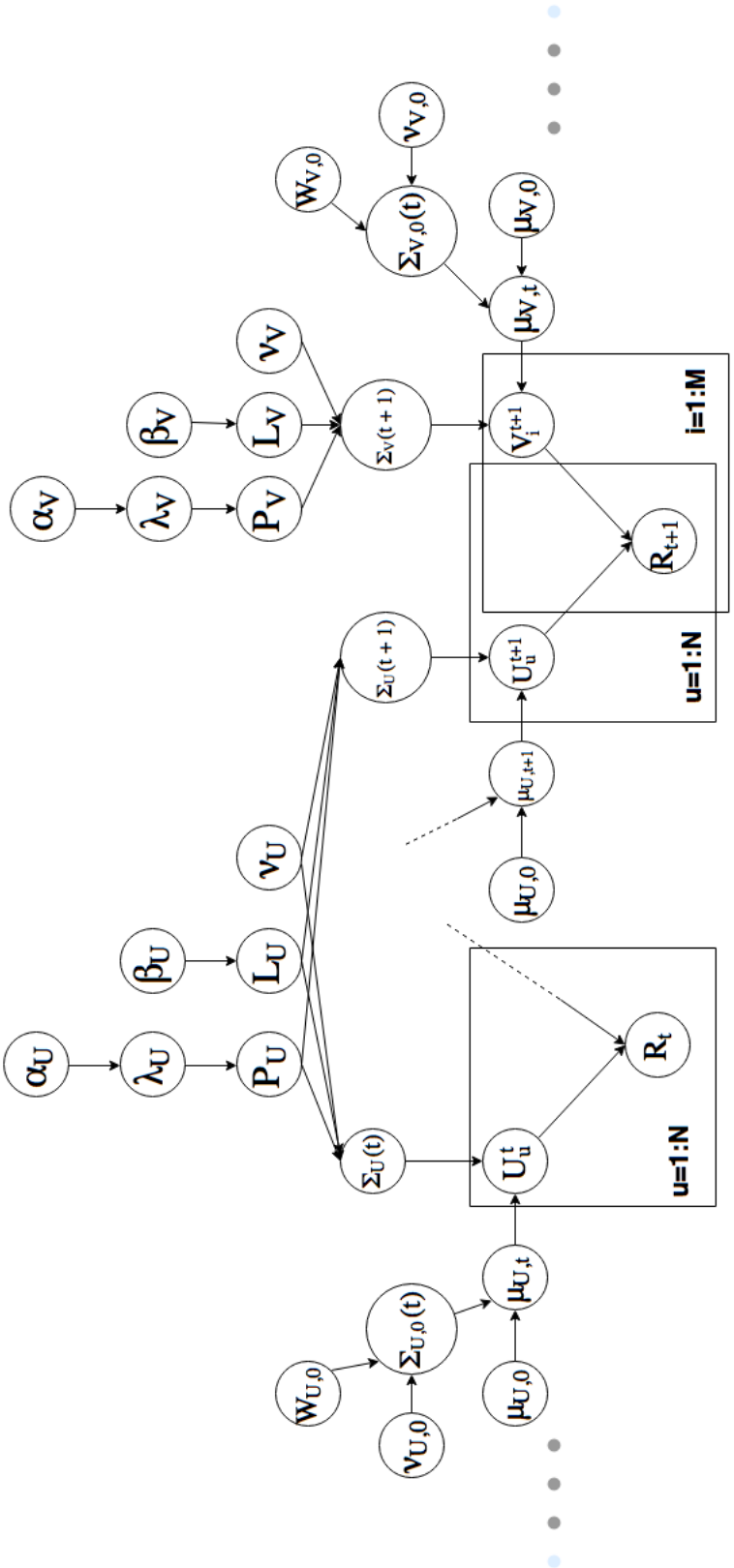


Figure 5.1: The graphical model for Bayesian Wishart matrix factorization method.

Following the approach developed in [236], the prior distribution over the Gaussian process function values $\mathcal{P}_U$ can be defined after reordering the entries in $\mathcal{P}_U$. Let $P_{l,d,t}$ denote one entry in the $TK\nu$ dimensional vector $\mathcal{P}_U$. Specifically, after fixing the dimensions along the degree of freedom and dimensionality, the entries of $\mathcal{P}_U$ are ordered by running the time steps t from 1 to T. Then, the dimension index d is increased from 1 to K. Finally, the index l for degrees of freedom ν is increased from 1 to ν. Let $P_{l,d,t}$ denote one entry in the $TK\nu$ dimensional vector $\mathcal{P}_U$. The pseudo code of the reordering procedure is expressed in Algorithm 2. Then, the prior for $\mathcal{P}_U$ can be defined by a multivariate normal distribution

```
for l:=1 to ν do
    for d:=1 to K do
        for t:=1 to T do
            P_{l,d,t} := P_{l,d}(t)
        end
    end
end
```

Algorithm 2: Reordering entries in $\mathcal{P}_U$ to define the prior over $\hat{P}_l(t)$.

as follows,

$$P(\mathcal{P}_U|\theta_{\mathcal{P}}) = \mathcal{N}(\mathcal{P}_U|0, K_{U,B}), \tag{5.4}$$

where $K_{U,B}$ is a $TK\nu \times TK\nu$-dimensional block diagonal covariance matrix. The $K\nu$ block matrices in $K_{U,B}$ are formed by the replication of the T-dimensional covariance matrix K with $K_{t,t'} = k(t, t')$.

Without any prior knowledge, a vague gamma prior is placed on the parameter $\theta_{\mathcal{P}_U}$ in kernel function $k(t, t')$ as $P(\theta_{\mathcal{P}_U}|\alpha_{\mathcal{P}}, \beta_{\mathcal{P}}) = Gamma(\theta_{\mathcal{P}_U}|\alpha_{\mathcal{P}}, \beta_{\mathcal{P}})$, where $\alpha_{\mathcal{P}}$ and $\beta_{\mathcal{P}}$ are parameters for the Gamma distribution $Gamma(\theta_{\mathcal{P}_U})$.

At time t, a multivariate normal distribution with a Wishart prior is used as the prior of

the mean vector $\mu_{U,t}$ to enable a fully Bayesian treatment as follows,

$$P(\mu_{U,t}|\mu_{U,0}, \Sigma_{U,0}(t)) = \mathcal{N}(\mu_{U,t}|\mu_{U,0}, \Sigma_{U,0}(t)), \tag{5.5}$$

$$P(\Sigma_{U,0}(t)|\nu_{U,0}, W_{U,0}) = \mathcal{W}(\Sigma_{U,0}(t)|\nu_{U,0}, W_{U,0}). \tag{5.6}$$

These distributions do not explicitly consider the temporal information across time frames. Intuitively, they are designated to emphasize on the local effects of latent vectors, which in turn capture the local effects of user preferences and item attractiveness. Although these priors and hyper priors will introduce some extra degrees of freedom in the model, the fully Bayesian modeling and the sampling approach used in the inference could alleviate the severity of overfitting. Following the setup in [196, 145], the means and covariances are accordingly set as $\mu_{U,0} = 0$ and $W_{U,0} = \mathcal{I}$ in the experiments.

Instead of using the prior as defined above, the common practice tends to use the distribution of $\Sigma_U(t)$ in Eq (5.1) as the prior for the covariance of $\mu_{U,t}$. From the perspective of the graphical model, this alternative configuration mimics the Gaussian-Wishart prior in Bayesian MF method. However, due to the different generative processes for the covariance in those two models, if the alternative prior was used, the posterior distribution of $\mu_{U,t}$ in the inference procedure cannot be properly approximated. In this regard, the proposed model sticks to the multivariate normal distribution with a Wishart distribution prior. According to the graphical model in Figure 5.1, the probability distributions and priors relating to item latent factors can be derived in a symmetric way.

5.5 Inference

BWMF will predict user preferences over those unrated items at time t. Hence, the goal of the inference is to obtain some proper estimations of user and item latent factors U_t and V_t at time t for the prediction. As a Bayesian method, those latent vectors can be estimated from the posterior distribution $P(U_{1:t}, V_{1:t}|R_{1:t})$ given all the available feedback from the 1-st to the current t-th time frames. Let model parameters $\Theta_{\mathcal{GWP}} = \{L_U, L_V, \nu, \theta_{\mathcal{P}_U}, \theta_{\mathcal{P}_V}, \mu_{U,1:t}, \mu_{V,1:t}\}$ and $\Theta = \{\Sigma_{U,0}(t), \Sigma_{V,0}(t)|t \in [1, \ldots, T]\}$. Ac-

cording to the construction of GWP process, the dynamic covariance matrices $\Sigma_U(t)$ can be marginalized into the composition of parameters $\Theta_{\mathcal{GWP}}$. Therefore, the posterior distributions for user and item latent vectors at various time frames can be marginally obtained from the posterior distribution $P(U_{1:t}, V_{1:t}, \Theta_{\mathcal{GWP}}, \Theta|R_{1:t})$. As shown in the graphical model and the definitions in Section 5.4.1, the GWP processes introduce the non-conjugate priors to the distributions of those latent factors. Hence, it is infeasible to compute this posterior distribution analytically. The inference is thus approximated by applying the collapsed Gibbs sampling method to the posterior distribution $P(U_{1:t}, V_{1:t}, \Theta_{\mathcal{GWP}}, \Theta|R_{1:t})$.

Without resorting to conjugate priors, BWMF also enjoys the advantages of GWP process on easily modeling user feedback that has diverse inherent natures. To be adapted to various real-world scenarios, BWMF only needs to change the kernel function used in the GWP process. The inference procedure is almost identical, apart from slightly modifying the concrete form of the kernel function and adjusting the specific parameters to be sampled. In contrast, if the dynamic system, such as the Gaussian random walk used in BTPF [241], cannot accurately reflect the inherent nature of user feedback, it is not a trivial task to come up with a proper stochastic process to capture the changed scenarios and develop an inference algorithm to learn its parameters.

5.5.1 The Overall Procedure

After initializing $U_{1:t}$, $V_{1:t}$, $\Theta_{\mathcal{GWP}}$, and Θ, the sampling procedure cycles through the following conditional posterior distributions by using collapsed Gibbs sampling. For each posterior, elliptical slice sampling is applied when the conjugate prior does not exist. The pseudo code for the overall procedure is shown in Algorithm 3, and its details will be

discussed later.

$$P(\mathcal{P}_U|R,\Theta,U,V,\Theta_{\mathcal{GWP}}) \propto P(U|\mathcal{P}_U,\Theta,\nu,L_U)P(\mathcal{P}_U|\theta_{\mathcal{P}_U},\nu), \tag{5.7}$$

$$P(\theta_{\mathcal{P}_U}|R,\theta_{\mathcal{P}_V},U,V,\Theta,L_U,L_V,\nu,\mu_U,\mu_V,\mathcal{P}_U,\mathcal{P}_V) \propto P(\mathcal{P}_U|\theta_{\mathcal{P}_U})P(\theta_{\mathcal{P}_U}), \tag{5.8}$$

$$P(L_U|R,\nu,\Theta,U,V,L_U,L_V,\mu_U,\mu_V,\theta_{\mathcal{P}_U},\theta_{\mathcal{P}_V},\mathcal{P}_U,\mathcal{P}_V) \propto P(U|\mathcal{P}_U,\Theta,\nu,L_U)P(L_U), \tag{5.9}$$

$$\begin{aligned} &P(\mu_U|R,\mu_V,L_U,L_V,\nu,\Theta,U,V,\theta_{\mathcal{P}_U},\theta_{\mathcal{P}_V},\mathcal{P}_U,\mathcal{P}_V) \\ &\qquad \propto P(U|\mu_U,\mathcal{P}_U,\nu,L_U)P(\mu_U|\mu_{U,0},\Sigma_{U,0}), \end{aligned} \tag{5.10}$$

$$\begin{aligned} &P(\nu|R,L_U,L_V,\mu_U,\mu_V,\Theta,U,V,\mathcal{P}_U,\mathcal{P}_V,\theta_{\mathcal{P}_U},\theta_{\mathcal{P}_V}) \\ &\qquad \propto P(U|\mathcal{P}_U,\Theta,\nu,L_U)P(V|\mathcal{P}_V,\Theta,\nu,L_V)P(\nu), \end{aligned} \tag{5.11}$$

$$P(\Sigma_{U,0}|R,\Theta,\Theta_{\mathcal{GWP}}) \propto P(\mu_U|\mu_{U,0},\Sigma_{U,0})P(\Sigma_{U,0}|\mu_{U,0},W_{U,0}), \tag{5.12}$$

$$P(U|R,\Theta,\nu,L_U,L_V,\mu_U,\mu_V,V,\mathcal{P}_U,\mathcal{P}_V,\theta_{\mathcal{P}_U},\theta_{\mathcal{P}_V}) \propto P(R|U,V)P(U|\mathcal{P}_U,\mu_U,\Theta,\nu,L_U). \tag{5.13}$$

For clarity, the subscripts about time are omitted in the above equations. The left sides of the proportion in the above formulas are derived by using Bayes' rule and the conditional independence from the graphical model shown in Figure 5.1. It is possible to define a prior over the degrees of freedom ν and the reversible-jump MCMC method [91] could be used to obtain its posterior distribution. In the following inference procedure, it is fixed to be $\nu = K + 1$ to let the prior as flexible as possible. This simplicity also reduces the computational complexity of the inference procedure for the developed model.

5.5.2 Sampling the Gaussian Process Function Values $\mathcal{P}_U$

The posterior distribution of $\mathcal{P}_U$ is proportional to two distributions. The first one is the likelihood distribution $P(U|\mathcal{P}_U,\Theta,\nu,L_U)$ and the other is the prior $P(\mathcal{P}_U|\theta_{\mathcal{P}_U},\nu)$. As shown in Section 5.4.1, the prior $P(\mathcal{P}_U|\theta_{\mathcal{P}_U},\nu)$ models a random vector that has a $TK\nu$-dimensional multivariate normal vector with zero mean. This highly correlated prior makes it difficult to sample from the posterior distribution $P(\mathcal{P}_U|R,\Theta,U,V,\Theta_{\mathcal{GWP}})$. Following the approach in [236], the elliptical slice sampling method [177], which is specially designed to sample from the posterior with highly correlated Gaussian priors, is used to update

Let U, V initialized by PMF, $\nu = K + 1$, $\mu_U = \mu_V = 0$, $L_U = L_V = \mathcal{I}$ and $\Sigma_{U,0} = \Sigma_{V,0} = \mathcal{I}$

for $s \in$ *#iterations in the Gibbs sampling* **do**

  for $t \in 1 \dots T$ **do**

    $\mathcal{P}_{U,s} \sim P(\mathcal{P}_U | R, \Theta, U, V, \Theta_{\mathcal{GWP}})$ using Algorithm 4

    $L_{U,s} \sim P(L_U | R, \nu, \Theta, U, V, L_U, L_V, \mu_U, \mu_V, \theta_{\mathcal{P}_U}, \theta_{\mathcal{P}_V}, \mathcal{P}_U, \mathcal{P}_V)$ based on the modifications of Algorithm 4 as in Section 5.5.5

    sampling $L_{V,s}$ similar to the above step

    $\mu_{U,s} \sim P(\mu_U | R, \mu_V, L_U, L_V, \nu, \Theta, U, V, \theta_{\mathcal{P}_U}, \theta_{\mathcal{P}_V}, \mathcal{P}_U, \mathcal{P}_V)$

    sampling $\mu_{V,s}$ similar to the above step

    $\nu_s \sim P(\nu | R, L_U, L_V, \mu_U, \mu_V, \Theta, U, V, \mathcal{P}_U, \mathcal{P}_V, \theta_{\mathcal{P}_U}, \theta_{\mathcal{P}_V})$

    $\Sigma_{U,0,S} \sim P(\Sigma_{U,0} | R, \Theta, \Theta_{\mathcal{GWP}}) \propto P(\mu_U | \mu_{U,0}, \Sigma_{U,0}) P(\Sigma_{U,0} | \mu_{U,0}, W_{U,0})$

    $U_s \sim P(U | R, \Theta, \nu, L_U, L_V, \mu_U, \mu_V, V, \mathcal{P}_U, \mathcal{P}_V, \theta_{\mathcal{P}_U}, \theta_{\mathcal{P}_V})$

    sampling V_s similar to the above step

  end

end

Generate prediction of $R_{t+1}^{u,i}$ using Eq 5.19

Algorithm 3: The sampling procedure of BWMF. The subscripts about time are omitted for clarity.

jointly every element of $\mathcal{P}_U$. This sampling method also has no free parameter, which reduces the burden to the model complexity control.

The likelihood distribution $P(U | \mathcal{P}_U, \Theta, \nu, L_U)$ for user latent factors over time can be

derived as follows,

$$
\begin{aligned}
P(U|\mathcal{P}_U, \Theta, \nu, L_U) &= \prod_{t=1}^{T} \int P(U_t|\mu_{U,t}, \Sigma_t) P(\mu_{U,t}|\mu_{U,0}, \Sigma_U) P(\Sigma_t|L_U, \mathcal{P}_U, \nu, \theta_{\mathcal{P}_U}) d\mu_{U,t} d\Sigma_t \\
&= \prod_{t=1}^{T} \int P(U_t|\mu_{U,t}, \Sigma^*) P(\mu_{U,t}|\mu_{U,0}, \Sigma_U) d\mu_{U,t} \\
&= \prod_{t=1}^{T} |\Sigma^*(t)|^{-\frac{N}{2}} |\Sigma_U|^{-\frac{1}{2}} |\Sigma_-(t)|^{\frac{1}{2}} \cdot \\
&\quad exp(-\frac{1}{2} \sum_i {U_t^{(i)}}^T \Sigma^*(t)^{-1} U_t^{(i)} - \frac{1}{2} \mu_{U,0}^T \Sigma_U^{-1} \mu_{U,0} + \frac{1}{2} \mu_-^T \Sigma_-^T \mu_-), \qquad (5.14)
\end{aligned}
$$

where $\Sigma^*(t) = \sum_{l=1}^{\nu} L_U \hat{P}_l(t) \hat{P}_l(t)^T L_U^T$, $\Sigma_-^{-1}(t) = N\Sigma^*(t)^{-1} + \Sigma_U^{-1}$, $\mu_- = \Sigma_-(t)(\Sigma_U^{-1}\mu_{U,0} + N\Sigma^*(t)^{-1}\bar{U}_t)$ and $\bar{U}_t = \frac{1}{N}\sum_{u=1}^{N} U_t^u$. The latent vector of user u at time t is denoted as U_t^u. The second equation in Eq (5.14) is derived by exploiting the construction of GWP process, i.e., $P(\Sigma|L_U, P, \nu, \theta_{\mathcal{P}_U}) = \delta(\Sigma - \Sigma^*(t))$. In order to facilitate the numeric computation involved, the logarithms of the likelihood and the prior are used in the elliptical slice sampling in the implementation. As it is straightforward to obtain these logarithms, they are omitted here for clarity.

The pseudo code for sampling the posterior distribution of $\mathcal{P}_U$ is listed in Algorithm 4. The ESS in the algorithm represents the elliptical slice sampling function, which can be treated as an oracle machine here. The ESS function takes as inputs the candidate sample x_s and the likelihood function $F(\cdot)$. The constant S_{ESS} is set to be the number of samples obtained from the sampling procedure. The pseudo code for sampling other posteriors is omitted in the following discussion because they share the same template as shown in Algorithm 4.

let $F(U) = log(P(U|\mathcal{P}_U, \Theta, \nu, L_U))$, $P_1 \sim \mathcal{N}(x_s|0, K_{U,B})$

for *s:= 2 to* S_{ESS} **do**
| $x_s \sim \mathcal{N}(x|0, K_{U,B})$ $P_{s+1} \sim ESS(P_{s-1}, x_s, F(\cdot))$

end

Algorithm 4: The sampling procedure of the Gaussian process function values $\mathcal{P}_U$.

5.5.3 Sampling the Posteriors of $\Sigma_{U,0}(t)$ and $\mu_{U,t}$

As a multivariate normal distribution is imposed on $\mu_{U,t}$ as the prior, the posterior $P(\Sigma_{U,0}(t)|R, \Theta, \Theta_{\mathcal{GWP}})$ can be analytically obtained,

$$P(\Sigma_{U,0}(t)|R, \Theta, \Theta_{\mathcal{GWP}}) = \mathcal{W}(\Sigma_{U,0}(t)|W_0^*, \nu_0^*), \tag{5.15}$$

where $(W_0^*)^{-1} = W_{U,0}^{-1} + (\mu_{U,t} - \mu_{U,0})(\mu_{U,t} - \mu_{U,0})^T$ and $\nu_0^* = \nu_{U,0} + 1$.

By utilizing the conjugate relation between the likelihood function $P(U|\mu_{U,t}, \Sigma^*)$ and the conditional distribution $P(\mu_{U,t}|\mu_{U,0}, \Sigma_{U,0}(t))$, the posterior distribution $P(\mu_{U,t}|R, \theta_{L_U}, L_U, \nu, \mu_{U,t}, U, V, \mathcal{P}_U)$ can be approximated as follows,

$$\begin{aligned} P(\mu_{U,t}|R, \theta_{\mathcal{P}_U}, L_U, \nu, U, \mathcal{P}_U) &\propto P(U_t|\mu_{U,t}, \Sigma^*(t))P(\mu_{U,t}|\mu_{U,0}, \Sigma_{U,0}(t)), \\ &\propto \mathcal{N}(U_t|\mu_{U,t}, \Sigma^*(t))\mathcal{N}(\mu_{U,t}|\mu_{U,0}, \Sigma_{U,0}(t)) \\ &= \mathcal{N}(\mu_{U,t}|\mu_t^*, \Sigma_+^*), \end{aligned} \tag{5.16}$$

where $\mu_t^* = \Sigma_+^*(t)(\Sigma_{U,0}^{-1}(t)\mu_{U,0} + N\Sigma^*(t)^{-1}\bar{U}_t)$ and $\Sigma_+^*(t) = (\Sigma_{U,0}^{-1}(t) + N\Sigma^*(t)^{-1})^{-1}$.

5.5.4 Sampling User Latent Factors

The posterior distribution of user latent factors is proportional to the products of the likelihood distribution $P(R|U, V)$ and the conditional distribution $P(U|\mathcal{P}_U, \mu_{U,t}, \Theta_{L_U}, \Theta, \nu, L_U)$. The likelihood distribution is analogously defined as the observation function in PMF, where every rating $r_t^{u,i}$ given by user u to item i at time t follows a Gaussian distribution with mean $(U^u)^T V^i$ and a predefined standard deviation σ_R.

The distribution $P(U|\mathcal{P}_U, \mu_U, \Theta_{L_U}, \Theta, \nu, L_U)$ can be approximated as follows,

$$\begin{aligned} P(U|\mathcal{P}_U, \mu_U, \Theta_{L_U}, \Theta, \nu, L_U) &\propto \prod_{t=1}^{T}\prod_{u=1}^{N} P(U_t^u|\mu_{U,t}, \Sigma_U(t))P(\Sigma_U(t)|\mathcal{P}_U, \nu, L_U), \\ &\propto \prod_{t=1}^{T}\prod_{u=1}^{N} P(U_t^u|\mu_{U,t}, \Sigma_U^{*,t}), \end{aligned} \tag{5.17}$$

where $\Sigma_U^{*,t} = \sum_{l=1}^{\nu} L_U \hat{P}_{U,l}(t) \hat{P}_{U,l}(t)^T L_U^T$. According to the second proportion in the above derivation, the conditional distribution can be finally approximated by a multivariate normal distribution.

By exploiting the conjugate properties between the likelihood distribution and the approximated conditional distribution in Eq (5.17), the sampling of the posterior can be further simplified. This simplification further reduces the computational complexity and speed up the sampling procedure.

$$P(U_t^u|R, \theta_{L_U}, \nu, L_U, \mu_{U,t}, \Theta, V, \mathcal{P}_U) \propto P(R|U,V) P(U_t^u|\mu_{U,t}, \Sigma_U^{*,t}) = \mathcal{N}(U_t^u|\mu^*(t), \Sigma^*(t)), \tag{5.18}$$

where $\mu^*(t) = [\Sigma^*(t)](\sigma_R \sum_{i=1}^{M} I_{u,i}(V_t^i r_t^{u,i}) + \Sigma_U^{*,t^{-1}} \mu_{U,t})$, $\Sigma^*(t) = \Sigma_U^{*,t^{-1}} + \sigma_R \sum_{i=1}^{M} I_{u,i}(V_t^i V_t^{i^T})$. The indicator function $I_{u,i}$ equals to 1 when user u has rated item i, and 0 otherwise.

5.5.5 Sampling Other Parameters

The procedure to sample the posterior of the free parameter L_U is similar to the sampling procedure of the Gaussian process function values $\mathcal{P}_U$. Because a vague prior is imposed on L_U, the elliptical slicing sampling conducts a random searching in the parameter space.

In the experiments, the Gaussian kernel $k(t,t') = exp(-(t-t')/\alpha^2)$ is used for the kernel function k. Hence, $\theta_{L_U} = \alpha_U$. This kernel models the effects that the temporal influences should reduce as the temporal distance increases. Note that it is possible to have a kernel function $k(t,t')$ that changes from row to row to represent the different length-scale for various dimensions in the latent space. However, due to the conditional independence in the model, the likelihood function in the posterior $P(\theta_{L_U}|R, L_U, \nu, \Theta, U, V, \mathcal{P}_U, \mu_{U,t})$ only depends on $\mathcal{P}_U$. Therefore, the kernel function $k(t,t')$ in the model does not consider such flexibility to avoid overfitting during the inference. Because the kernel function $k(t,t')$ only employs one scalar parameter to control its scale, the Metropolis-Hasting MCMC method [21] is used to obtain its samples from the posterior distribution.

5.5.6 Prediction

The predictive distribution $P(R^{u,i}_{*,t+1}|R_{1:t},\Theta)$ of the rating $R^{u,i}_{*,t+1}$ for user u on its unrated item i at the $(t+1)$-th time frame is obtained by integrating out the model parameters and hyperparameters in BWMF. This predictive distribution is also too complex to be analytically derived. After obtained the sets of sampled user and item latent factors, $P(R^{u,i}_{*,t+1}|R_{1:t},\Theta)$ can be approximated as follows,

$$\begin{aligned} P(R^{u,i}_{*,t+1}|R_{1:t},\Theta) = \int & P(R^{u,i}_{*,t+1}|U^u_{t+1},V^i_{t+1})P(U^u_{t+1},V^i_{t+1}|U^u_t,V^i_t)P(U^u_t,V^i_t|R_{1:t},\Theta_{\mathcal{GWP}},\Theta) \\ & P(\Sigma_U(t),\Sigma_V(t)|L_U,\mu_U,\nu,\Theta)P(\Theta_U,\Theta_V|\Theta)d\Sigma_U(t)d\Sigma_V(t)dU^u_{t+1}dV^i_{t+1} \\ & dU^u_t dV^i_t d\Theta_{\mathcal{GWP}} \\ & \sim \frac{1}{S}\sum_{s=1}^{S} P(R^{u,i}_{*,t+1}|U^{u,(s)}_t,V^{i,(s)}_t) \end{aligned} \tag{5.19}$$

where S is the number of samples obtained from the Gibbs sampling. Meanwhile, to reduce the computational complexity of prediction in BWMF, the above approximation adopts $P(U^u_{t+1},V^i_{t+1}|U^u_t,V^i_t)=\delta(U^u_{t+1}-U^u_t)\delta(V^v_{t+1}-V^v_t)$. The estimated prediction $R^{u,i}_{*,t+1}$ can be thus approximated as $\frac{1}{S}\sum_{s=1}^{S}((U^{u,(s)}_t)^T V^{v,(s)}_t)$.

5.5.7 Computational Complexity

Recall that latent factors are $U_t \in \mathcal{R}^{N\times K}$ and $V_t \in \mathcal{R}^{M\times K}$ with $K \ll min(N,M)$. The complexity of *personalized* prediction in BWMF at each time frame in the recommendation task keeps unchanged as $O(KM)$ in PMF.

The complexity of computing $P(U|\mathcal{P}_U,\theta,\nu,L_U)$ is $O(TN\nu K^3)$, which is dominated by the determinant operations ($O(K^3)$) and inversion operations ($O(K^3)$). By pre-computing the inversion of $\Sigma^*(t)$ in Eq (5.14), the complexity can be reduced to $O(T(\nu K^3+NK^2))$. Similarly, the complexities of computing $P(\mu_{U,t}|R,\theta_{\mathcal{P}_U},L_U,\nu,\mathcal{P}_U)$ and $P(U^{(i)}_t|R,$ $\theta_{L_U},\nu,L_U,\mu_{U,t},\Theta,V,\mathcal{P}_U)$ are $O(K^3)$ and $O(\nu K^2+K^3+|\bar{R}_U|K^2)$, where $|\bar{R}_U|$ represents the average number of ratings for users. The inversion operations also dominate the computation. In order to sample other parameters shown in Section 5.5.5, the complexity

depends on the number of instances to be sampled and the complexity of computing the acceptance ratio. Because the kernel parameter is a scalar, the number of samples (including burning period) is set to be comparable to K in Metropolis-Hasting MCMC method. Therefore, the complexity is $O(CK^3)$, where C is a tiny constant. As the complexity of elliptical slice sampling is linear to the product of the number of samples and the complexity of the input likelihood function, the overall complexity of the inference method for BWMF is $O(S_{ESS}T(\nu K^3+(N+M)K^2))$, which is linear in terms of the number of users and items in the system. In the experiments, S_{ESS} is set to 2, which still complies with the stochastic climbing of the input likelihood function in elliptical slice sampling. The complexity of BWMF is thus $O(T(\nu K^3 + (N + M)K^2))$.

5.6 Experiments

BWMF is tested on public benchmark datasets including Movielens [1], Hetrec [3] and Netflix [108]. MovieLens spans 32 weeks with integer rating from 1 to 5 while HetRec spans 12 years with half mark rating from 1 to 5. These two datasets are selected to study the performance of the proposed method for short and long periods of time. Netflix is adopted to verify the performance of the proposed method on a reasonably large dataset.

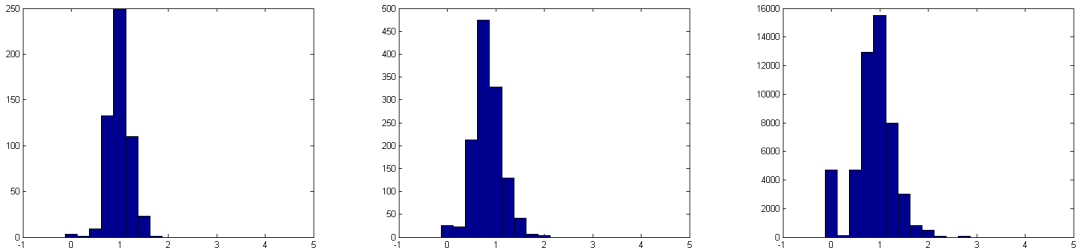
Protocol Ratings are grouped based on the time frame to which a rating's timestamp belongs. Ratings before a predefined time instance are used as the training data, while ratings after it are used as the test data. This setting is preferred over a random split of all the data. As in a real-world deployment, it is infeasible to generate predictions using any information in the future. The training periods for MovieLens, HetRec and Netflix datasets are Sep. $\sim$ Dec. 1997, Sep. 1997 $\sim$ Dec. 2007 and 1-st $\sim$ 15-th months, respectively. Their testing periods are 1-st $\sim$ 16-th weeks in 1998, Jan. 2008 $\sim$ Dec. 2008 and 16-th $\sim$ 27-th months, respectively. The different time units are selected to ensure that ratings for each user in a time slot are not too sparse. Notice that all test periods are after associated training periods mimicking a real-world scenario, i.e. predicting the future.

Based on this setup, MovieLens contains 530 users and 1493 items (93.3% overall sparsity and 99.96% sparsity in the last time frame). HetRec contains 1775 users and 9228 items (95.1% overall sparsity and 99.98% sparsity in the last time frame). Netflix has 480,189 users and 17,770 items, which has the overall sparsity of 98.84%. Following the protocol adopted in [145], 4% of random samples of Netflix is used. The sampled dataset contains 20% users and 20% movies that were randomly selected from the whole pool. This leads to a dataset with the overall sparsity of 98.6% and the sparsity of 99.98% in the last time frame.

Metric The root mean square error (RMSE) [196, 158] is selected as the metric to assess the performance of rating prediction. The RMSE metric is defined as $RMSE = \sqrt{\frac{1}{NM}\sum_{u=1}^{N}\sum_{i=1}^{M}(r_{u,i} - \hat{r}_{u,i})^2}$, where $\hat{r}_{u,i}$ is the predicted rating for user u on item i.

As mentioned before, it is more challenging to model those users in RSs that do manifest a diverse rating pattern over items. Figure 5.2a, 5.2b and 5.2c demonstrate the histograms of the standard deviations of users' ratings in the training data from MovieLens, Hetrec and Netflix datasets, respectively. As shown in those figures, most of the users do not tend to give diverse rating values across items. Therefore, it should be more interesting to investigate the temporal behaviors of RSs by only considering those users that do have diverse feedback patterns. Hence, RMSE is restricted to those users that have large rating variances in the training data such that their variances are among the top N% of the whole users. This refined version of RMSE is denoted as TOP_RMSE. In the experiments, N is set to be 20, because the rating covariance is found to be roughly around 1.4 in all of these three datasets under this setting. In order to measure the temporal performance of RSs, the temporal extensions [160] of these two metrics are used, which are values of those metrics measured at every time frame in the testing period.

Baseline methods The proposed method models the historical feedback from users to predict the user preferences over items in the successive time frames, which extends the Bayesian treatment of MF for CF method. In order to test the performance of BWMF, the following methods are adopted as the baseline methods: Bayesian probabilistic tensor decomposition (BPTF) [145] and Bayesian matrix factorization method (BMF) [196].



(a) MovieLens dataset. (b) Hetrec dataset. (c) Netflix dataset.

Figure 5.2: The histogram of standard deviations of users' ratings from training data.

However, BMF is a static CF method by design. By retraining it at each time frame with all the data upto the time step, which is a common practice in the real-world deployment, this method is enhanced by exploiting temporal information to make the comparison fair. The dynamic extension of this method is named as dynamic BMF thereafter.

BMF is a state-of-the-art method for CF under the static assumption. Its dynamic extension represents the common practice in the real-world deployment. BPTF is the state-of-the-art method for CF that explicitly takes into account the contribution of temporal information during the modeling of the overall interactions between user preferences and item attractiveness. By comparing with those two baseline methods, it is attempted to answer the following questions regarding the proposed method:

1) Is systematically utilizing temporal priors more efficient at modeling temporal dynamics in RSs than the commonly dynamic retraining of a static method?

2) Is it beneficial to impose the GWP process as the priors over the temporal dynamics of variations of latent vectors during the process of MF, especially for those users that have diverse rating patterns?

3) Is the improvement introduced by the proposed method significant?

4) Is it still beneficial to directly impose the priors on the temporal dynamics of variations of user and item latent vectors instead of on the transitional relations of those latent vectors across consecutive time frames as the existing approaches do? How about those users that have diverse rating patterns?

5.6.1 Experimental Results

All the compared methods in the experiments are repeated 10 times and the means and standard deviations of the results are reported. PMF with moderate regularization, which inspired by the setup used in [196, 145], is used to initialize all of these methods. Note that this setup will not influence the conclusions made in the experiments. PMF is also run 10 times with the identical training data and settings, and the run with the best performance under temporal RMSE metric is used to fulfill the task of initialization. The temporal behaviors of the compared methods are also studied and analyzed.

MovieLens dataset For user and item latent vectors, it is set to $K = 4$ for all the compared methods. The standard deviation σ_R for rating is 0.5 for all those methods.

Results The first row in Table 5.1 shows the results of the compared methods under temporal RMSE_TOP and RMSE metrics. Among these results, BWMF has the best performance. Its temporal RMSE is 0.9682 and temporal RMSE_TOP 0.9988. For users that have diverse rating patterns, the performance of BWMF improves that of dynamic BMF by 2.07%. Note that it is very challenging to improve the performance of RSs on personalized rating prediction from the recent state-of-the-art methods. The 1% improvement is usually regarded as a significant improvement and is qualified to win the Netflix Prize [125]. Meanwhile, for the measurement over all users, BWMF still performs better than other baseline methods do. All of the improvement introduced by BWMF under both metrics, except for the temporal RMSE over dynamic BMF, is statistically significant under both paired and unpaired t tests with $p = 0.05$.

This improvement can be ascribed to the fine modeling and learning of temporal dynamics of variations of user and item latent vectors, which in turn reflects the tendency of user preferences and item attractiveness. The results also show that BWMF is more effective to model the user's diverse preferences by controlling the fluctuation of latent factors via dynamic covariance matrices. Nevertheless, both dynamic BMF and BPTF do not exploit the dynamics in this direction.

Method	RMSE	RMSE_TOP	Dataset
Dynamic BMF	0.9693 ± 0.0013	1.0199 ± 0.0041	MovieLens
BPTF	0.9901 ± 0.0078	1.0394 ± 0.0160	
BWMF	*0.9682* ± 0.0018	*0.9988* ± 0.0110	
Dynamic BMF	0.8123 ± 0.0005	1.0221 ± 0.0017	Hetrec
BPTF	0.8130 ± 0.0011	1.0242 ± 0.0034	
BWMF	*0.8099* ± 0.0006	*1.0151* ± 0.0012	
Dynamic BMF	0.8995 ± 0.0002	1.0688 ± 0.0003	Netflix
BPTF	0.8950 ± 0.0006	1.0650 ± 0.0008	
BWMF	*0.8876* ± 0.0007	*1.0552* ± 0.0009	

Table 5.1: Comparative results of methods under RMSE and RMSE_TOP metrics. The best performance is in italic font.

Temporal behaviors To further evaluate temporal behaviors of compared methods, the average of accumulated improvement (AAI) over time is adopted [160]. Let the performance of any two methods under RMSE in month t be $RMSE_1(t)$ and $RMSE_2(t)$, respectively. The AAI in month t_1 is $\frac{1}{t_1}\sum_{t=1}^{t_1}(RMSE_1(t) - RMSE_2(t))$ [2]. The AAI metric for RMSE_TOP can be defined similarly.

Figure 5.3 plots the AAI among dynamic BMF, BPTF and BWMF for users that have diverse rating patterns. Figure 5.4 plots the AAI among those methods for all the users. Except in the 1-st week of the blue (dash-dotted) curve (BWMF vs dynamic BMF) in Figure 5.4, all the curves relating to BWMF in both of Figure 5.4 and Figure 5.3 are below zero. These curves show that BWMF constantly outperforms baseline methods by utilizing the GWP process as the priors to finely model the temporal variations of latent vectors, which reflect the temporal dynamics in user preferences and item popularity.

The improvement on BPTF can be partially explained by the priors that it adopts. The priors designated to represent the transitional relations of latent vectors on temporal dimension may be not flexible enough to cooperate with user and item latent vectors to capture the diverse and dynamic user preferences. Compared with dynamic BMF in Figure 5.3, the tendency of the yellow (stared) curve shows that BWMF is more effective

[2]In contrary to common situations, the smaller the RMSE value is, the better the method performs. Therefore, it is actually expecting that the AAI curve is below x-axis if the first method in AAI metric is expected to perform better.

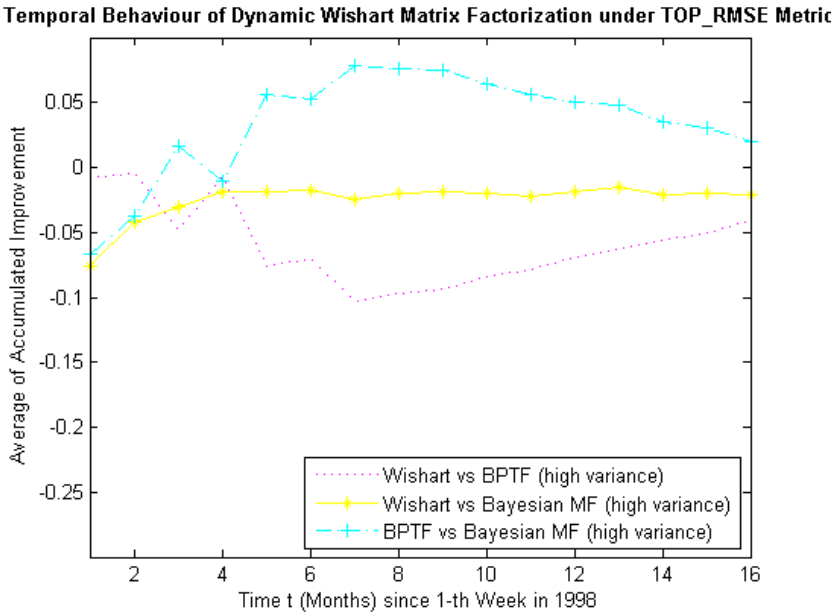


Figure 5.3: The average of accumulated improvement over time for the compared methods on the MovieLens dataset since the 1-st week in 1998. The time unit is week and the metric is temporal TOP_RMSE.

at exploiting the underlying temporal variations. This observation is further consolidated by the blue (dash-dotted) curve in Figure 5.4. Meanwhile, the magenta (dotted) curve in Figure 5.3, which is a comparison between BWMF and BPTF, shows that BWMF also constantly outperforms the Bayesian method with priors to guide the transitions of temporal latent factors. In addition, BPFT only focuses on the global temporal effects across all users and items. Unlike BWMF, this modeling approach makes BPTF not able to catch up with the latest trend as time goes. This phenomenon is evident in Figure 5.3, where crossings exist between the curves relating to dynamic BMF and BPTF and the horizontal axis.

Hetrec dataset Similar to the experiments on MovieLens, the dimensions of user and item latent vectors are also set to $K = 4$ for all the methods under investigation. However, the standard deviation σ_R for rating is set to 1 for all those methods. According to the experiments, this setting will lead all the compared methods to have a better performance compared with the setting with $\sigma_R = 0.5$ that is used in MovieLens.

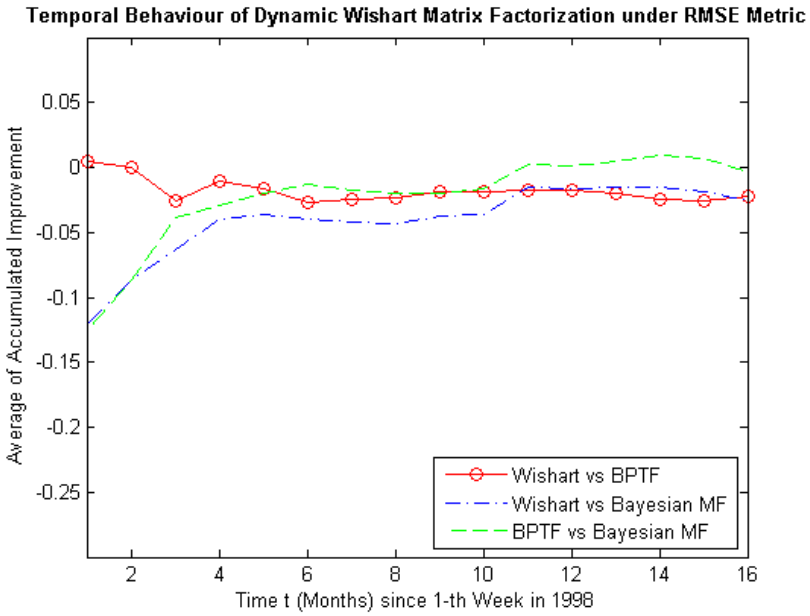


Figure 5.4: The average of accumulated improvement over time for the compared methods on the MovieLens dataset since the 1-st week in 1998. The time unit is week and the metric is temporal RMSE.

Results The second row in Table 5.1 lists the results of the compared methods under temporal RMSE_TOP and RMSE metrics. Among these results, BWMF has the best performance. Its RMSE is 0.8099 and RMSE_TOP 1.0151. For those users that do not tend to provide the constant feedback, BWMF still performs much better than dynamic BMF. For example, for the TOP_RMSE metric, BWMF outperforms dynamic BMF and BPTF by 0.68% and 0.89%, respectively. All of the improvement introduced by BWMF is statistically significant under both paired and unpaired t tests with $p = 0.05$.

Compared with experiments in the previous section, the improvement of BWMF over other baseline methods on Hetrec is still significant but not as outstanding as the improvement achieved on MovieLens. After doing some exploratory analysis, it is found that this phenomenon may be ascribed to the fact that different rating scales are adopted in Hetrec and MovieLens. Hetrec allows half marks given by users while MovieLens only allows integer ratings. This half-mark rating system, to some extents, reduces the magnitude of the errors when the predicted ratings do not comply with their ground truth.

Temporal behaviors Similar to the previous experiments on MovieLens, the AAI metric is used to take a closer inspection on the temporal behaviors for the compared methods on Hetrec. Figure 5.5 plot the AAI among dynamic BMF, BPTF and the proposed BWMF methods for users that have diverse rating patterns. Figure 5.6 plot the AAI among those methods for all the users. Except for the 2-nd month at the beginning of the magenta (dotted) curve, both of the (magenta dotted and yellow starred) curves representing the similar comparison for TOP_RMSE in Figure 5.5 are also below zero. Similarly, both of the (red circled and blue dash-dotted) curves representing the comparison between BWMF and other baseline methods are below zero in Figure 5.6. Those figures illustrate that the developed method constantly outperforms the baseline methods under a long period. Comparing Figure 5.6 to Figure 5.5, it is also shown that BWMF constantly performs much better than other baseline methods for users that tend to have diverse rating patterns. Compared with those in Figure 5.6, the curves in Figure 5.5 demonstrates much larger fluctuation, which is an inherent nature of the larger variances from those users that have diverse rating patterns [3].

Although dynamic BMF performs better than BPTF as shown in Table 5.1, it is not easy to distinguish their performance over time as shown by the green (dash) curve in Figure 5.6. Moreover, except for the initial time frames in Figure 5.5, dynamic BMF is more capable of coping with users that have diverse rating patterns. The rationale behind this observation can be ascribed to the modeling approach taken by BPTF. Even though transitional relations are modeled in this Bayesian method, it only captures the temporal effects across all users.

Netflix dataset Netflix contains much more users and items than MovieLens and Hetrec. The dimensions of user and item latent vectors are set to $K = 10$ for all the compared methods to provide those models adequate degrees of freedom, which is tuned based on the performance of PMF that is trained and tested with temporal RMSE metric using the

[3] According to the exploratory analysis, the standard deviation for the top 20% users is about 1.44, which is the largest among the three datasets adopted in the experiments.

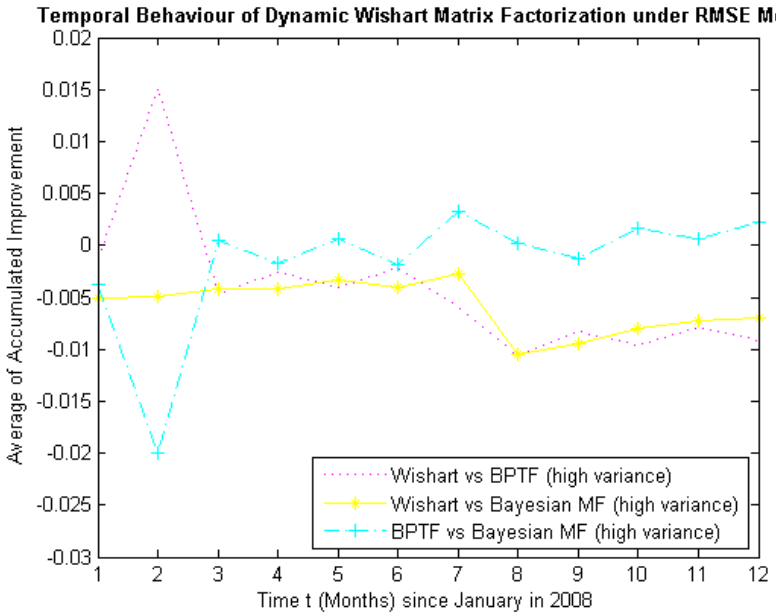


Figure 5.5: The average of accumulated improvement over time for the compared methods on the Hetrec dataset since January 2008. The time unit is month and the metric is temporal TOP_RMSE.

exact training data and test data as the compared methods. The standard deviation α for ratings is also set to 1 for all those methods according to the trial experiments.

Results The last row in Table 5.1 shows the results of the methods under temporal RMSE_TOP and RMSE metrics. Among these results, BWMF has the best performance. For example, its RMSE is 0.8876 and its RMSE_TOP is 1.0552. Similar to previous experiments, BWMF still outperforms other methods. For example, BWMF improves the dynamic BMF by 1.27% for users that tend to have diverse rating patterns. This result shows BWMF works successfully on a reasonably large and sparse real-world dataset. All of the improvement introduced by BWMF is statistically significant under both paired and unpaired t test with $p = 0.05$. Note that 10-dimensional timeSVD++ has a performance of 0.8971 under RMSE and 20-dimensional timeSVD++ has 0.8891 under RMSE (quoted from the paper), which is worse than the proposed BWMF method. Even though the performance of timeSVD++ may be improved as its dimensions is growing, the experimental results of timeSVD++ are again obtained on predictions for users' *post hoc*

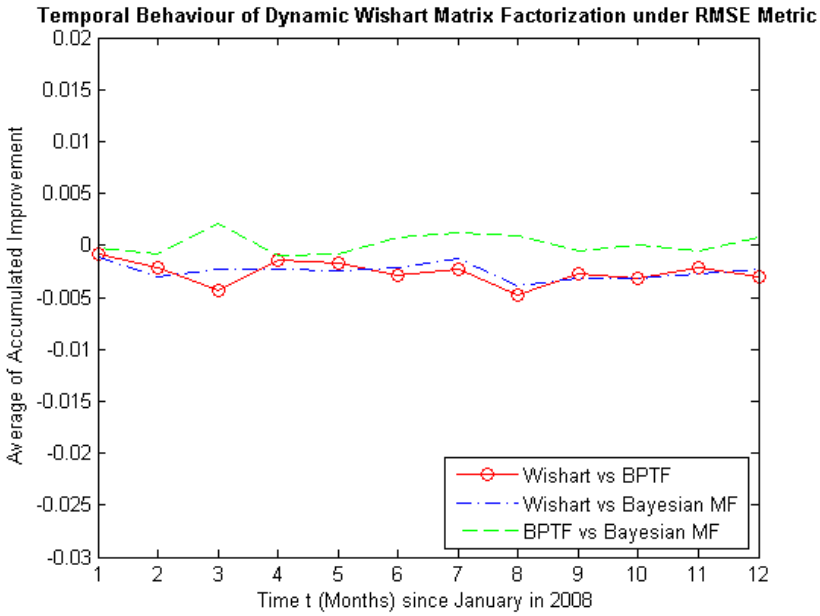


Figure 5.6: The average of accumulated improvement over time for the compared methods on the Hetrec dataset since January 2008. The time unit is month and the metric is temporal RMSE.

interests about what interests *would have been* in the past. Those experimental results are much less convincing for modeling dynamics than experimental results obtained in this chapter which are about what interests would be in the future.

Temporal behaviors Similar to previous experiments, the AAI metric is used to make a closer inspection on temporal behaviors for the compared methods. Figure 5.7 plots the AAI among dynamic BMF, BPTF and BWMF methods for users that have diverse rating patterns. Figure 5.8 plots the AAI among those methods for all the users. All the curves depicted in the figures are below zero and have smooth and steady traces, showing that BWMF constantly outperforms the baseline methods under a relatively large dataset. Compared with the corresponding figures in previous experiments, the curves in Figure 5.7 demonstrate fewer variations. This phenomenon in Figure 5.7 is just an inherent nature of measurement metric, which measures the average performance, under a large number of users.

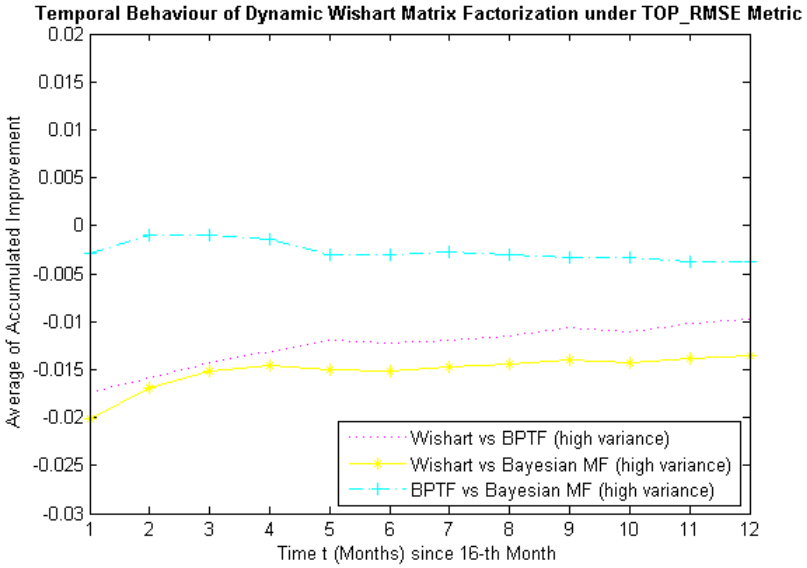


Figure 5.7: The average of accumulated improvement over time for the compared methods on the Netflix dataset since the 16-th month. The time unit is month and the metric is temporal TOP_RMSE.

Discussion By comparing the results obtained using temporal RMSE_TOP and RMSE metrics on those three datasets, it is clear that BWMF can effectively model user preferences over time, especially for those users that demonstrate more complicated patterns of rating values. This conclusion is not unexpected to reach. In general, both dynamic BMF and BPTF do not model the temporal dynamics of variations of latent vectors. While modeling the interaction between user preferences and item attractiveness, the contributions from the covariances and temporal variations do not receive any special emphasis in those two methods. BPTF does consider the temporal dynamics, but it only focuses on modeling transitional relations of latent vectors. Although the priors are only imposed on dynamic covariance matrices of latent vectors in BWMF, the inference procedure developed in the proposed method will pass the influences of these priors to the means of latent vectors as shown in the inference procedure.

According to the above experimental results, BWMF is more efficient at modeling temporal and dynamic information for RSs, compared with the retraining approach conducted by dynamic BMF. This result verifies that it is beneficial to impose the GWP processes as

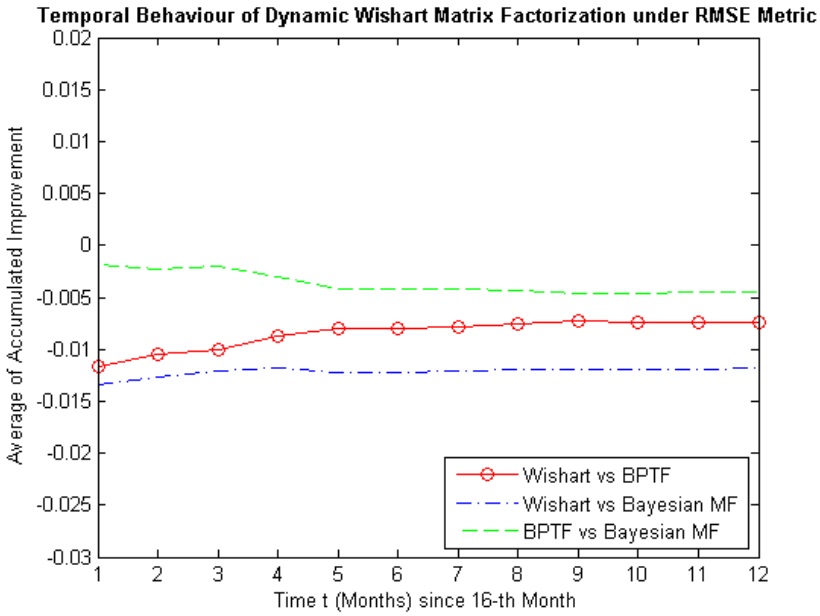


Figure 5.8: The average of accumulated improvement over time for the compared methods on the Netflix dataset since the 16-th month. The time unit is month and the metric is temporal RMSE.

the priors on the temporal dynamics of variations of user and item latent vectors during the process of MF.

Meanwhile, the comparisons of experimental results between BTPF and BWMF demonstrate that BWMF performs much better than the state-of-the-art method specially developed to extend temporal dynamics for model-based CF. Note that the performance of BTPF is also worse than that of dynamic BMF on MovieLens and Hetrec. The results show that BTPF does not perform well under the scenarios where the user feedback does not tend to have periodic properties. Regarding Bayesian filtering, BTPF resembles more to a smoothing method rather than a prediction method, which focuses more on what interests *would have been* in the past rather than in future. The results also confirm that it is feasible for CF to model directly temporal dynamics of the variations of user and item latent vectors instead of modeling the transitional relations of those latent vectors between consecutive time frames.

Also, unlike BMF and BTPF, BWMF does not barely rely on global latent parameters.

Method	Bayesian MF	Bayesian PTMF	BWMF
Hetrec	276.6	*150.6*	897.5
Hetrec parallel	127.6	*41.1*	551.9
Netflix	1040.8	*667.6*	6072.3
Netflix parallel	626.2	*445.8*	4635.4

Table 5.2: Average running times (implemented in Matlab and run on a 4-core machine with 3.3G Hz CPU and 8G memory). The unit is second and the best performance is in italic font.

The improvement of the performance of RSs over time can also benefit from the localization of latent vectors within each time frame. Meanwhile, as shown in the graphical model of BWMF, this design of localization does not prevent latent factors from sharing information via the GWP process priors, which helps to mitigate the problem of data sparsity in RSs.

Computational time The running time at each time frame (including both training, retraining and prediction time) is averaged to compare the efficiency of the compared methods. All of the compared methods are running on a standalone machine with 3 GHz CPU and 8 GB memory. Bayesian BTPF is based on the mixed programming of Matlab and C. The implementations of other two methods are based on pure Matlab without optimization. The running times of these methods under RMSE on Hetrec and Netflix are listed in the first and third lines in Table 5.2. Their running times under Movielens have the similar effects, and they are ignored for clarity. It is straightforward to speed up BWMF by parallelizing the sampling procedure in terms of users and items. The running times of these parallelized versions (with four threads) are listed in the second and fourth lines in Table 5.2. The key observation is that BWMF can be speeded up by parallelization. It is thus expected that the running time of the proposed method is comparable with that of baseline methods on more sophisticated deployments. Meanwhile, the implementation can be also speeded up by using C to optimize the concrete computation.

5.7 Summary

A novel Bayesian Wishart matrix factorization model is developed to improve the performance of RSs over time, especially for those users that do have diverse feedback patterns. The developed model exploits the generalized Wishart process to identify and control the temporal dynamics of variations of user and item latent factors, which in turn controls the trend and fluctuation of user preferences and item attractiveness over time. The temporal behaviors of those users with diverse feedback patterns are transparently taken care of by the developed method. Meanwhile, a new learning and inference algorithm, which combines the collapsed Gibbs sampling method and the elliptical slice sampling method, is also developed for the model. The developed learning and inference algorithm does not require the usage of conjugate priors.

The proposed model is evaluated on three real-world public benchmark datasets under the temporal extensions of some widely used metrics for personalized rating prediction. The experimental results illustrate that the proposed method not only outperforms a variety of state-of-the-art methods, but also significantly improves the recommendation performance over compared methods when it models the preferences of those users with diverse rating patterns. The results also confirm that it is feasible to model the temporal dynamics of variations of user and item latent vectors to handle the temporal and dynamic information on user feedback.

Currently, the proposed method is developed to handle explicit user feedback. The implicit feedback in RSs demonstrates its own characteristics and requires specific treatment. Therefore, it is worth designing observation functions in BWMF that take into account the implicit feedback in future. Meanwhile, inversion operations of matrices, which has the major contribution to time complexity of BWMF, should also be optimized. For example, the techniques involved in online updating, which are widely used in methods of online updating approach in Chapter 2, could be exploited to achieve this kind of optimization.

Chapter 6

Conclusion and Future Work

The goal of this research is to improve the performance of recommender systems (RSs) under temporal context. The research work presented in this thesis aims to overcome the outstanding problems of exploiting temporal dynamics in RSs. In order to solve those identified problems in this research, methods have been developed and empirically studied on a variety of public available benchmark datasets for RSs with significant performance improvement, showing the contribution of this research.

6.1 Summaries

The summaries of the research in this thesis are highlighted in this section.

6.1.1 Tracking the Tendency of User Preferences and Item Attractiveness

The developed methods in Chapter 3 aim to solve the first three problems in existing methods when exploiting temporal dynamics in RSs. In particular, these problems in existing methods are listed as follows,

- the tendency of user preferences is usually assumed to be linear and Gaussian;
- item characteristics or attractiveness is usually assumed to be static;
- the generative process for user feedback is usually assumed to be Gaussian distributed.

Those problems significantly reduce the performance of existing RSs.

A novel probabilistic temporal bilinear model is developed to solve the above problems and improve the performance of RSs over time. The developed model exploits the temporal and dynamic information in users' historical feedback to model the tendency of both user preferences and item attractiveness more finely. It simultaneously tracks latent factors representing user preferences and item attractiveness for the Top-N recommendation. In order to better capture the generative process of user preferences on items over time, an observation function is also developed, which is proper for the Top-N recommendation under temporal context. The designed observation function does not assume that user feedback has to be Gaussian distributed. Meanwhile, this model enforces the temporal interactions between user preferences and item attractiveness and dynamically adjusts significance on different dimensions of user and item latent factors.

In addition to the three problems listed above, to simultaneously solve the problems of data sparsity and scalability under temporal context, a novel two-phase self-training mechanism is developed to construct a small but delicate set of observations from missing data dynamically. A new learning and inference algorithm combining a sequential Monte Carlo method and the expectation maximization algorithm is also developed to take advantages of temporal information and dynamic structure in the feedback.

The proposed method is evaluated on three real-world datasets, MovieLens 100K, Hetrec and Netflix datasets, under the temporal extensions of accuracy metrics. The experimental results demonstrate that the developed methods significantly improve the recommendation performance over a variety of state-of-the-art methods, which confirms that the proposed methods can effectively and efficiently learn and track both latent factors and model

parameters over time. The experiments also illustrate the advantages of the temporal dynamic model over static ones and the benefits of tracking both user preferences and item attractiveness instead of tracking merely user preferences.

6.1.2 On Learning the Dynamics in Temporal Recommender Systems

The developed methods in Chapter 4 aim to solve the fourth and fifth problems in existing methods when exploiting temporal dynamics in RSs. In particular, these problems are listed as follows,

- the model structure to capture the tendency of user preferences and item attractiveness is either predefined or learned from a linear system;
- rather than using personalized and item-wise dynamic systems, a universal dynamic model is commonly adopted across all users and items in the system.

A novel probabilistic personalized and item-wise model, namely DynTranPF, has been presented in Chapter 4 to tackle the *cold start transition* problem in learning the tendency of user preferences for RSs. In order to fully adapt to the rich and diverse dynamical systems for various users and items, an online adaptive extreme learning machine is developed to model the temporal and dynamic information intrinsic in users' historical feedback. For the developed method, There are no constraints or assumptions imposed on the structure of the temporal interactions between user preferences and item attractiveness. Meanwhile, by learning and dynamically updating the personalized and item-wise dynamic systems, the tendency of user preferences and item attractiveness can be finely modeled and flexibly tailored to fit the individual scenario with respect to each user and item. By exploiting the historical feedback and temporal interactions from "like-minded" users and similar items, a new inference and learning algorithm, considering the model uncertainties for emulating the interested tendency, is also developed for Top-N recommendation over time. The algorithm tracks latent factors representing user preferences and item attractiveness and adaptively updates model parameters to emphasize on the current trend. The *cold start*

transition problem, which is particularly outstanding in exploiting temporal dynamics in RSs, is solved by learning from collaborative tendencies.

In summary, the improvement achieved by the developed method includes the follows,

1) the dynamic adaptation of the transition models for each user and item over time is capable of finely modeling the tendency of user preferences and item attractiveness;

2) when data are sparse, user preferences and item popularity represented by latent factors can be better guided by the inertia of learned transitions in the DynTranPF method than simply randomly searching in some other existing methods;

3) because DynTranPF does not impose fewer assumptions on the structures of the tracked tendency, the personalized and item-wise transition models can be fully tailored to the diverse scenarios of the modeled dynamics. Finally, the uncertainty in the model has been effectively considered by the developed inference and learning algorithm in DynTranPF. Instead of treating as predefined priors, model parameters are adapted in accordance with the current observations along with latent factors.

The presented methods in Chapter 4 are evaluated on three real-world datasets, MovieLens 100K, Hetrec and Amazon Video Games datasets, under the temporal extensions of accuracy metrics. The experimental results demonstrate that our methods significantly improve the recommendation performance over a variety of state-of-the-art algorithms. The results confirm that the proposed methods can effectively model the temporal dynamics of rich and diverse user preferences and item preferences. For users and items that are inactive for some time frames, the experiments also verify that it is more appropriate to assume the tendency follows the inertia of the modeled dynamics instead of just randomly searching the latent space. The learning algorithm that dynamically updates model parameters is shown to be able to handle the introduced model uncertainties and efficiently guide the propagation of latent factors. The experiments also illustrate the advantages of the temporal dynamic model over static ones and the benefits of tracking both user preferences and item attractiveness instead of tracking merely user preferences.

The experimental results relating to the *cold start transition* problem also illustrates that the proposed learning algorithm can accurately learn the missed transition systems by utilizing knowledge from similar users and items.

6.1.3 Bayesian Wishart Matrix Factorization

The developed methods in Chapter 5 aim to solve the sixth or the last problem in existing methods when exploiting temporal dynamics in RSs. In particular, this problem is,

- the temporal dynamics of variation of user preferences and item attractiveness are largely neglected. Almost all of existing work on RSs focuses on the modeling of temporal dynamics of average behaviors of user preference and item attractiveness.

A novel Bayesian matrix factorization model has been presented in Chapter 5 to model the temporal dynamics of variations of user preferences and item attractiveness and improve the performance of RSs over time, especially for those users that do have diverse feedback patterns. The developed model exploits the generalized Wishart process to identify and control the temporal dynamics of variations of user and item latent factors, which in turn controls the trend and fluctuation of user preferences and item attractiveness over time. The temporal behaviors of those users that do have diverse feedback patterns are transparently taken care of by the developed method. Meanwhile, a new learning and inference algorithm, which combines the collapsed Gibbs sampling method and the elliptical slice sampling method, is also developed for the model.

The proposed model is then evaluated on three real-world datasets, MovieLens 100K, Hetrec and Netflix datasets, under the temporal extensions of some widely used metrics for rating prediction. Not only the experimental results illustrate that Bayesian Wishart matrix factorization method improves the recommendation performance over a variety of state-of-the-art methods, but also the proposed Bayesian Wishart model significantly improves the recommendation performance over those methods when it models the preferences of those users that tend to have diverse rating patterns. The results also confirm

that it is feasible to model the temporal dynamics of covariance matrices of user and item latent vectors in order to handle the temporal and dynamic information on user feedback.

6.2 Future Work

Based on the research work presented and discussed in previous chapters, several possible directions are worth exploring in the future.

6.2.1 The Dynamic Systems

First, more sophisticated techniques could be exploited to represent and learn the dynamic systems of user preferences and item characteristics. For example, non-parametric Bayesian approach, such as, Gaussian process is an immediate candidate for modeling the dynamic systems of user preferences and item attractiveness. Although it is simple to utilize the Gaussian process for scalar output, how to develop a Gaussian process with multivariate output is still a novel field and there are only a few studies [20, 121, 122] on it. Meanwhile, for the non-parametric Bayesian approach, it is necessary to retain all the historical data in memory. It is also challenging to reduce both the time and space complexity of the adopted non-parametric Bayesian process when applying it to exploit temporal dynamics in RSs, considering the huge number of users and items in the real-world deployment.

6.2.2 The Observation Models

It is also worth investigating the temporal behaviors of RSs for multiple aspects of items. Instead of only generating an overall rating, many reviews rate items based on various criteria. For example, the popular Zagat survey rates the performance of restaurants based on four components: food, decor, services and cost. Therefore, the information in various components of a rating could be exploited to design some proper observation functions

for recommendation tasks that consider multiple recommendation criteria. Moreover, the developed methods in Chapter 3, 4 and 5 can also be extended with observation functions that explicitly take into account the objective functions or metrics used in learning to ranking. For example, the NDCG metric can be directly used in Eq (3.5) to make the observation functions capture user preferences for items in a ranking-oriented approach. Note that the sampling mechanism in particle filtering does not require that the observation function should be differentiable. However, considering the number of samples used in particle filtering and the time complexity of NDCG metric applied to RSs, it may be still necessary to properly approximate this metric to reduce the computational complexity of RSs incurred by this kind of observation function.

6.2.3 The Cold Start Problems

Meanwhile, the problems of cold start user and cold start item are not taken into account in the proposed methods. Another direction of future work could extend the developed models with the ability to handle with those cold start problems [13]. For example, techniques utilizing social networking in RSs, which are discussed in Chapter 2, could be integrated into the proposed methods to provide them with the initial personalized recommendations for users.

6.2.4 Exploiting Temporal Priors

Furthermore, it is expected that the computational complexity is extremely unaffordable for RSs, when considering the priors over both temporal dynamics of variations of latent factors and the transitional relations of latent factors. Nevertheless, this issue also provides the researchers with the opportunity to develop exciting algorithms that exploit the advantages of both priors.

Bibliography

[1] *MovieLens 100K dataset, http://www.grouplens.org/data/* (2003).

[2] *Netflix Prize Update* (2010).

[3] *HetRec MovieLens dataset, http://www.grouplens.org/node/462* (2011).

[4] *Amazon Video Games dataset, https://snap.stanford.edu/data/web-Amazon.html* (2013).

[5] Adamidis, K. Theory & methods: An em algorithm for estimating negative binomial parameters. *Australian & New Zealand Journal of Statistics*, 2 (1999), 213–221.

[6] Adomavicius, G., and Kwon, Y. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems*, 3 (May 2007), 48–55.

[7] Adomavicius, G., and Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 6 (June 2005), 734–749.

[8] Agarwal, D., and Chen, B.-C. flda: Matrix factorization through latent dirichlet allocation. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining* (2010), pp. 91–100.

[9] Agarwal, D., Chen, B.-C., and Elango, P. Fast online learning through offline initialization for time-sensitive recommendation. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), pp. 703–712.

[10] Agarwal, D., Chen, B.-C., Elango, P., and Ramakrishnan, R. Content recommendation on web portals. *Communications of the ACM*, 6 (June 2013), 92–101.

[11] Agarwal, D., Chen, B.-C., Elango, P., and Wang, X. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2011), pp. 132–140.

[12] Agarwal, D., Chen, B.-C., Elango, P., and Wang, X. Personalized click shaping through lagrangian duality for online recommendation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2012), pp. 485–494.

[13] Agarwal, D., Chen, B.-C., and Long, B. Localized factor models for multi-context recommendation. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (2011), pp. 609–617.

[14] Aharon, M., Aizenberg, N., Bortnikov, E., Lempel, R., Adadi, R., Benyamini, T., Levin, L., Roth, R., and Serfaty, O. Off-set: One-pass factorization of feature sets for online recommendation in persistent cold start settings. In *Proceedings of the 7th ACM Conference on Recommender Systems* (2013), pp. 375–378.

[15] Aharon, M., Elad, M., and Bruckstein, A. Svdd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 11 (Nov. 2006), 4311–4322.

[16] Aharon, M., Kagian, A., Kaplan, Y., Nissim, R., and Somekh, O. Serving ads to "yahoo answers" occasional visitors. In *Proceedings of the 24th International Conference on World Wide Web* (2015), pp. 1257–1262.

[17] Aizenberg, N., Koren, Y., and Somekh, O. Build your own music recommender by modeling internet radio streams. In *Proceedings of the 21st International Conference on World Wide Web* (2012), pp. 1–10.

[18] Albadvi, A., and Shahbazi, M. A hybrid recommendation technique based on product category attributes. *Expert Systems with Applications*, 9 (2009), 11480 – 11488.

[19] Altman, A., and Tennenholtz, M. Ranking systems: The pagerank axioms. In *Proceedings of the 6th ACM Conference on Electronic Commerce* (2005), pp. 1–8.

[20] Alvarez, M., and Lawrence, N. D. Sparse convolved gaussian processes for multi-output regression. In *Advances in Neural Information Processing Systems* (2009), Curran Associates, Inc., pp. 57–64.

[21] Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. An introduction to mcmc for machine learning. *Machine Learning*, 1-2 (2003), 5–43.

[22] Arthur, D., Manthey, B., and Röglin, H. Smoothed analysis of the k-means method. *Journal of the ACM*, 5 (Oct. 2011), 19:1–19:31.

[23] Babu, R. V., Suresh, S., and Makur, A. Online adaptive radial basis function networks for robust object tracking. *Computer Vision and Image Understanding*, 3 (2010), 297–310.

[24] Balakrishnan, S., and Chopra, S. Collaborative ranking. In *Proceedings of the 5th ACM international conference on Web Search and Data Mining* (2012), pp. 143–152.

[25] Baltrunas, L., and Ricci, F. Dynamic item weighting and selection for collaborative filtering. *Web mining*, 0 (2007).

[26] Barber, D., Cemgil, A. T., and Chiappa, S., Eds. *Bayesian Time Series Models*. Cambridge University Press, 2011.

[27] Baumol, W., and Blinder, A. *Microeconomics: Principles and Policy*. Cengage Learning, 2011.

[28] Bauwens, L., Laurent, S., and Rombouts, J. V. K. Multivariate garch models: a survey. *Journal of Applied Econometrics*, 1 (2006), 79–109.

[29] Bell, R. M., Koren, Y., and Volinsky, C. The bellkor solution to the netflix grand prize, 2009.

[30] Bergstra, J., and Bengio, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* (Feb. 2012), 281–305.

[31] Bernardes, D., Diaby, M., Fournier, R., FogelmanSoulié, F., and Viennet, E. A social formalism and survey for recommender systems. *ACM SIGKDD Explorations Newsletter*, 2 (Dec. 2015), 20–37.

[32] Bertsekas, D. *Nonlinear Programming*. Athena Scientific, 1995.

[33] Bhatia, N., and Vandana. Survey of nearest neighbor techniques. *Computer Research Repository* (2010).

[34] Bian, J., Long, B., Li, L., Moon, T., Dong, A., and Chang, Y. Exploiting user preference for online learning in web content optimization systems. *ACM Transactions on Intelligent Systems and Technology*, 2 (Apr. 2014), 33:1–33:23.

[35] Bilmes, J. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models, 1997.

[36] Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[37] Blei, D. M. Probabilistic topic models. *Communications of the ACM*, 4 (Apr. 2012), 77–84.

[38] Bobadilla, J., Ortega, F., Hernando, A., and Bernal, J. A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems* (Feb. 2012), 225 – 238.

[39] Bobadilla, J., Ortega, F., Hernando, A., and Gutirrez, A. Recommender systems survey. *Knowledge-Based Systems* (2013), 109 – 132.

[40] Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of the 19th International Conference on Computational Statistics* (2010), pp. 177–187.

[41] Bottou, L. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*. Springer Berlin Heidelberg, 2012, pp. 421–436.

[42] Brenner, A., Pradel, B., Usunier, N., and Gallinari, P. Predicting most rated items in weekly recommendation with temporal regression. In *Proceedings of the Workshop on Context-Aware Movie Recommendation* (2010), pp. 24–27.

[43] Briers, M., Doucet, A., and Maskell, S. Smoothing algorithms for statespace models. *Annals of the Institute of Statistical Mathematics*, 1 (2010), 61–89.

[44] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning* (2005), pp. 89–96.

[45] Burges, C. J. C., Ragno, R., and Le, Q. V. Learning to Rank with Nonsmooth Cost Functions. In *Advances in Neural Information Processing Systems* (2006), pp. 193–200.

[46] Burke, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 4 (Nov. 2002), 331–370.

[47] Burke, R. Evaluating the dynamic properties of recommendation algorithms. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (2010), pp. 225–228.

[48] Cacheda, F., Carneiro, V., Fernández, D., and Formoso, V. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web*, 1 (Feb. 2011), 2:1–2:33.

[49] Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y. S., Compton, P., and Mahidadia, A. Learning collaborative filtering and its application to people to people recommendation in social networks. In *Proceedings of IEEE 10th International Conference on Data Mining* (2010), pp. 743–748.

[50] Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y. S., Compton, P., and Mahidadia, A. Collaborative filtering for people to people recommendation in social networks. In *Proceedings of AI 2010: Advances in Artificial Intelligence: 23rd Australasian Joint Conference* (2011), pp. 476–485.

[51] Campos, P., Dez, F., and Cantador, I. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 1-2 (2014), 67–119.

[52] Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., and Li, H. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning* (2007), pp. 129–136.

[53] Chai, T., and Draxler, R. R. Root mean square error (rmse) or mean absolute error (mae)? arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 3 (2014), 1247–1250.

[54] Chapelle, O., and Keerthi, S. S. Efficient algorithms for ranking with svms. *Information Retrieval*, 3 (June 2010), 201–215.

[55] Charlin, L., Ranganath, R., McInerney, J., and Blei, D. M. Dynamic poisson factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems* (2015), pp. 155–162.

[56] Chatzis, S. Dynamic bayesian probabilistic matrix factorization. In *AAAI Conference on Artificial Intelligence* (2014).

[57] Chen, A. Context-aware collaborative filtering system: Predicting the users preference in the ubiquitous computing environment. In *Proceedings of the First International Conference on Location- and Context-Awareness* (2005), pp. 244–253.

[58] Chen, C., Yin, H., Yao, J., and Cui, B. Terec: A temporal recommender system over tweet stream. *Proceedings of the VLDB Endowment*, 12 (Aug. 2013), 1254–1257.

[59] Chen, G., Wang, F., and Zhang, C. Collaborative filtering using orthogonal nonnegative matrix tri-factorization. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops* (2007), pp. 303–308.

[60] Chen, Z. Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond. Tech. rep., McMaster University, 2003.

[61] Chu, W., and Park, S. T. Personalized recommendation on dynamic content using predictive bilinear models. In *Proceedings of the 18th international conference on World wide web* (2009), pp. 691–700.

[62] Chung, T. S., Rust, R. T., and Wedel, M. My mobile music: An adaptive personalization system for digital audio players. *Market Science*, 1 (2009), 52–68.

[63] Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S.-i. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley Publishing, 2009.

[64] Coles, S. *An Introduction to Statistical Modeling of Extreme Values*. Springer, 2001.

[65] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms, Third Edition*. The MIT Press, 2009.

[66] Cremonesi, P., Koren, Y., and Turrin, R. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems* (2010), pp. 39–46.

[67] Dao, T. H., Jeong, S. R., and Ahn, H. A novel recommendation model of location-based advertising: Context-aware collaborative filtering using ga approach. *Expert Systems with Applications*, 3 (2012), 3731 – 3739.

[68] Das, A. S., Datar, M., Garg, A., and Rajaram, S. Google news personalization: Scalable online collaborative filtering. In *Proceedings of the 16th International Conference on World Wide Web* (2007), pp. 271–280.

[69] Das, K. The laplacian spectrum of a graph. *Computers & Mathematics with Applications*, 56 (2004), 715 – 724.

[70] de Campos, L. M., Fernndez-Luna, J. M., Huete, J. F., and Rueda-Morales, M. A. Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks. *International Journal of Approximate Reasoning*, 7 (2010), 785 – 799.

[71] Diaz-Aviles, E., Drumond, L., Gantner, Z., Schmidt-Thieme, L., and Nejdl, W. What is happening right now ... that interests me?: Online topic discovery and recommendation in twitter. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (2012), pp. 1592–1596.

[72] Diaz-Aviles, E., Drumond, L., Schmidt-Thieme, L., and Nejdl, W. Real-time top-n recommendation in social streams. In *Proceedings of the Sixth ACM Conference on Recommender Systems* (2012), pp. 59–66.

[73] Ding, Y., and Li, X. Time weight collaborative filtering. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management* (2005), pp. 485–492.

[74] Djuric, N., Grbovic, M., Radosavljevic, V., Bhamidipati, N., and Vucetic, S. Non-linear label ranking for large-scale prediction of long-term user interests. In *AAAI Conference on Artificial Intelligence* (2014).

[75] Doucet, A., de Freitas, N., and Gordon, N. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo Methods in Practice*. Springer New York, 2001, pp. 3–14.

[76] Ekstrand, M. D., Riedl, J. T., and Konstan, J. A. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 2 (Feb. 2011), 81–173.

[77] Faragher, R. Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]. *IEEE Signal Processing Magazine*, 5 (sep 2012), 128–132.

[78] Fesenmaier, D. R., Werthner, H., and Wober, K. W. *Travel Destination Recommendation Systems: Behavioural Foundations and Applications (Cabi Publishing)*. CAB International, 2006.

[79] Fletcher, R. *Quadratic Programming*. John Wiley & Sons, Ltd, 2000, pp. 229–258.

[80] Freddy Chong Tat Chua, Richard J. Oentaryo, E.-P. L. Modeling temporal adoptions using dynamic matrix factorization. In *Proceedings of the 11th IEEE International Conference on Data Mining* (2013), pp. 91–100.

[81] Gama, J. a., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM Computing Surveys*, 4 (Apr. 2014), 44:1–44:37.

[82] Gantner, Z., Drumond, L., Freudenthaler, C., and Schmidt-Thieme, L. Personalized ranking for non-uniformly sampled items. In *KDD Cup* (2012), pp. 231–247.

[83] Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. *Bayesian Data Analysis, Third Edition*. Taylor & Francis, 2013.

[84] Gelman, A., Vehtari, A., Jylnki, P., Robert, C., Chopin, N., and Cunningham, J. P. Expectation propagation as a way of life. *arxiv.org* (2014).

[85] Gemulla, R., Nijkamp, E., Haas, P. J., and Sismanis, Y. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2011), pp. 69–77.

[86] George Casella, E. I. G. Explaining the gibbs sampler. *The American Statistician*, 3 (1992), 167–174.

[87] Golbeck, J. Generating predictive movie recommendations from trust in social networks. In *Trust Management*. Springer Berlin Heidelberg, 2006, pp. 93–104.

[88] Goodwin, G. C., and Aguero, J. C. Approximate em algorithms for parameter and state estimation in nonlinear stochastic models. In *Proceedings of SIAM Data Mining* (2005).

[89] Gopalan, P., Hofman, J. M., and Blei, D. M. Scalable recommendation with poisson factorization. *arXiv.org* (2013).

[90] Gopaluni, B. Particle filter approach to nonlinear system identification under missing observations with a real application. In *15th IFAC Symposium on System Identification* (2009).

[91] GREEN, P. J. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 4 (1995), 711–732.

[92] Grimes, D. B., Shon, A. P., and Rao, R. P. N. Probabilistic bilinear models for appearance-based vision. In *Proceedings of the ninth IEEE International Conference on Computer Vision* (2003), pp. 1478–1485.

[93] Groh, G., and Ehmig, C. Recommendations in taste related domains: Collaborative filtering vs. social filtering. In *Proceedings of the 2007 International ACM Conference on Supporting Group Work* (2007), pp. 127–136.

[94] Gunawardana, A., and Meek, C. A unified approach to building hybrid recommender systems. In *Proceedings of the Third ACM Conference on Recommender Systems* (2009), pp. 117–124.

[95] Guo, L., Ma, J., Jiang, H.-R., Chen, Z.-M., and Xing, C.-M. Social trust aware item recommendation for implicit feedback. *Journal of Computer Science and Technology*, 5 (2015), 1039–1053.

[96] Guy, I., Zwerdling, N., Carmel, D., Ronen, I., Uziel, E., Yogev, S., and Ofek-Koifman, S. Personalized recommendation of social software items based on social relations. In *Proceedings of the Third ACM Conference on Recommender Systems* (2009), pp. 53–60.

[97] Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research* (Mar. 2003), 1157–1182.

[98] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer New York Inc., 2001.

[99] He, H., and Garcia, E. A. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 9 (Sept. 2009), 1263–1284.

[100] Hirschberg, D. S. Algorithms for the longest common subsequence problem. *Journal of the ACM*, 4 (Oct. 1977), 664–675.

[101] Hofmann, T. Probabilistic latent semantic indexing. In *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (1999), pp. 50–57.

[102] Hong, W., Li, L., and Li, T. Product recommendation with temporal dynamics. *Expert Systems with Applications*, 16 (2012), 12398 – 12406.

[103] Hoyer, P. O. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research* (Dec. 2004), 1457–1469.

[104] Huang, G.-B., e. a. Extreme learning machines: a survey. *International Journal of Machine Learning and Cybernetics*, 2 (2011), 107–122.

[105] Hüllermeier, E., Fürnkranz, J., Cheng, W., and Brinker, K. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 16-17 (Nov. 2008), 1897–1916.

[106] Hunter, D. R. Mm algorithms for generalized bradley-terry models. *The Annals of Statistics*, 1 (feb 2004), 384–406.

[107] Jahrer, M., and Tscher, A. Collaborative filtering ensemble for ranking. In *KDD Cup* (2012), pp. 153–167.

[108] James, B., and Stan, L. The netflix prize, 2007.

[109] Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., and Stumme, G. Tag recommendations in social bookmarking systems. *AI Communications*, 4 (Dec. 2008), 231–247.

[110] Jaynes, E., and Bretthorst, G. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

[111] Jeong, B., Lee, J., and Cho, H. An iterative semi-explicit rating method for building collaborative recommender systems. *Expert Systems with Applications*, 3, Part 2 (2009), 6181 – 6186.

[112] Jeong, B., Lee, J., and Cho, H. An iterative semi-explicit rating method for building collaborative recommender systems. *Expert Systems with Applications*, 3 (Apr. 2009), 6181–6186.

[113] Jia, Z., Yang, Y., Gao, W., and Chen, X. User-based collaborative filtering for tourist attraction recommendations. In *Proceedings of 2015 IEEE International Conference on Computational Intelligence Communication Technology* (2015), pp. 22–25.

[114] Jiang, M., Cui, P., Wang, F., Xu, X., Zhu, W., and Yang, S. Fema: Flexible evolutionary multi-faceted analysis for dynamic behavioral pattern discovery. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014), pp. 1186–1195.

[115] Kantor, P. B. *Recommender systems handbook*. Springer, 2009.

[116] Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., and Wu, A. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7 (jul 2002), 881–892.

[117] Karatzoglou, A. Collaborative temporal order modeling. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (2011), pp. 313–316.

[118] Kim, H., Takaya, N., and Sawada, H. Tracking temporal dynamics of purchase decisions via hierarchical time-rescaling model. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (2014), pp. 1389–1398.

[119] Kim, H. K., Kim, J. K., and Ryu, Y. Personalized recommendation over a customer network for ubiquitous shopping. *IEEE Transactions on Services Computing*, 2 (apr 2009), 140–151.

[120] Kline, M., and Berardi, L. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing and Applications*, 4 (Dec. 2005), 310–318.

[121] Ko, J., and Fox, D. Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (2008), pp. 3471–3476.

[122] Ko, J., and Fox, D. Learning gp-bayesfilters via gaussian process latent variable models. *Autonomous Robots*, 1 (2010), 3–23.

[123] Koenigstein, N., Dror, G., and Koren, Y. Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (2011), pp. 165–172.

[124] Kohavi, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (1995), pp. 1137–1143.

[125] Konstan, J., and Riedl, J. Deconstructing recommender systems. http://spectrum.ieee.org/computing/software/deconstructing-recommender-systems/, 2012.

[126] Konstas, I., Stathopoulos, V., and Jose, J. M. On social networks and collaborative recommendation. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2009), pp. 195–202.

[127] 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* (2008), pp. 447–456.

[128] Koren, Y. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (2009), pp. 447–456.

[129] Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 8 (aug 2009), 30–37.

[130] Kowald, D., Seitlinger, P., Kopeinik, S., Ley, T., and Trattner, C. Forgetting the words but remembering the meaning: Modeling forgetting in a verbal and semantic tag recommender. In *Mining, Modeling, and Recommending 'Things' in Social Media*. Springer International Publishing, 2015, pp. 75–95.

[131] Kowald, D., Seitlinger, P., Trattner, C., and Ley, T. Long time no see: The probability of reusing tags as a function of frequency and recency. In *Proceedings of the 23rd International Conference on World Wide Web* (2014), pp. 463–468.

[132] Krzywicki, A., Wobcke, W., Kim, Y., Cai, X., Bain, M., Mahidadia, A., and Compton, P. Collaborative filtering for people-to-people recommendation in online dating: Data analysis and user trial. *International Journal of Human-Computer Studies* (2015), 50 – 66.

[133] Kuleshov, V., Chaganty, A. T., and Liang, P. Tensor factorization via matrix factorization. *Computer Research Repository* (2015).

[134] Lafferty, J. D., McCallum, A., and Pereira, F. C. N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning* (2001), pp. 282–289.

[135] Larrain, S., Trattner, C., Parra, D., Graells-Garrido, E., and Nørvåg, K. Good times bad times: A study on recency effects in collaborative filtering for social tagging. In *Proceedings of the 9th ACM Conference on Recommender Systems* (2015), pp. 269–272.

[136] Lathia, N., Hailes, S., Capra, L., and Amatriain, X. Temporal diversity in recommender systems. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2010), pp. 210–217.

[137] Lee, D. D., and Seung, H. S. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems* (2001), pp. 556–562.

[138] Li, B., Yang, Q., and Xue, X. Can movies and books collaborate?: Cross-domain collaborative filtering for sparsity reduction. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence* (2009), pp. 2052–2057.

[139] Li, B., Zhu, X., Li, R., Zhang, C., Xue, X., and Wu, X. Cross-domain collaborative filtering over time. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence* (2011), pp. 2293–2298.

[140] Li, H. A short introduction to learning to rank. *IEICE Transactions on Information and Systems*, 10 (2011), 1854–1862.

[141] Li, M., Dias, B. M., Jarman, I., El-Deredy, W., and Lisboa, P. J. Grocery shopping recommendations based on basket-sensitive random walk. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009), pp. 1215–1224.

[142] Li, W., Wang, X., Zhang, R., Cui, Y., Mao, J., and Jin, R. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2010), pp. 27–36.

[143] Li, Y., and Tang, J. Expertise search in a time-varying social network. In *Proceedings of the Ninth International Conference on Web-Age Information Management* (2008), pp. 293–300.

[144] Lian, D., and Xie, X. Mining check-in history for personalized location naming. *ACM Transactions on Intelligent Systems and Technology*, 2 (Apr. 2014), 32:1–32:25.

[145] Liang, X., Xi, C., Tzu-Kuo, H., Jeff, S., and Jaime, G. C. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of SIAM Data Mining* (2010), pp. 211–222.

[146] Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. Facing the cold start problem in recommender systems. *Expert Systems with Applications* (2014), 2065 – 2073.

[147] Lin, Y.-R., Sundaram, H., De Choudhury, M., and Kelliher, A. Temporal patterns in social media streams: Theme discovery and evolution using joint analysis of content and context. In *Proceedings of IEEE International Conference on Multimedia and Expo* (2009), pp. 1456–1459.

[148] Linden, G., Smith, B., and York, J. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 1 (jan 2003), 76–80.

[149] Ling, G., Yang, H., King, I., and Lyu, M. Online learning for collaborative filtering. In *Proceedings of the 2012 International Joint Conference on Neural Networks* (2012), pp. 1–8.

[150] Liu, D. C., and Nocedal, J. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 3 (Dec. 1989), 503–528.

[151] Liu, F., and Lee, H. J. Use of social network information to enhance collaborative filtering performance. *Expert Systems with Applications*, 7 (2010), 4772 – 4778.

[152] Liu, J. S. The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 427 (1994), 958–966.

[153] Liu, N. N., He, L., and Zhao, M. Social temporal collaborative ranking for context aware movie recommendation. *ACM Transactions on Intelligent Systems and Technology*, 1 (Feb. 2013), 15:1–15:26.

[154] Liu, N. N., Zhao, M., Xiang, E., and Yang, Q. Online evolutionary collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (2010), pp. 95–102.

[155] Liu, X., and Aberer, K. Towards a dynamic top-n recommendation framework. In *Proceedings of the 8th ACM Conference on Recommender Systems* (2014), pp. 217–224.

[156] Lowe, D. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2 (2004), 91–110.

[157] Lu, Q., Chen, T., Zhang, W., Yang, D., and Yu, Y. Serendipitous personalized ranking for top-n recommendation. In *Proceedings of 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology* (2012), pp. 258–265.

[158] Lu, Z., Agarwal, D., and Dhillon, I. S. A spatio-temporal approach to collaborative filtering. In *Proceedings of the Third ACM conference on Recommender systems* (2009), pp. 13–20.

[159] Luo, C., and Cai, X. Bayesian wishart matrix factorization. *The journal track of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD)* (2016). Under review.

[160] Luo, C., Cai, X., and Chowdhury, N. Self-training temporal dynamic collaborative filtering. In *Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2014), pp. 461–472.

[161] Luo, C., Cai, X., and Chowdhury, N. Probabilistic temporal bilinear model for temporal dynamic recommender systems. In *Proceedings of 2015 International Joint Conference on Neural Networks* (2015), pp. 1–8.

[162] Luo, C., Cai, X., and Chowdhury, N. Towards solving the cold start transition problem in dynamic recommender systems. In *2015 IEEE 12th International Conference on e-Business Engineering* (2015), pp. 95–100.

[163] Luo, Z.-Q., and Yu, W. An introduction to convex optimization for communications and signal processing. *IEEE Journal on Selected Areas in Communications*, 8 (aug 2006), 1426–1438.

[164] MacKay, D. J. C. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, 2002.

[165] Magnus, J. R. On the concept of matrix derivative. *Journal of Multivariate Analysis*, 9 (2010), 2200–2206.

[166] Mahmood, T., and Ricci, F. Towards learning user-adaptive state models in a conversational recommender system. In *LWA* (2007), pp. 373–378.

[167] Maksai, A., Garcin, F., and Faltings, B. Predicting online performance of news recommender systems through richer evaluation metrics. In *Proceedings of the 9th ACM Conference on Recommender Systems* (2015), pp. 179–186.

[168] Marinho, L., Nanopoulos, A., Schmidt-Thieme, L., Jschke, R., Hotho, A., Stumme, G., and Symeonidis, P. Social tagging recommender systems. In *Recommender Systems Handbook*. Springer US, 2011, pp. 615–644.

[169] McAuley, J., and Leskovec, J. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems* (2013), pp. 165–172.

[170] McAuley, J. J., and Leskovec, J. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22Nd International Conference on World Wide Web* (2013), pp. 897–908.

[171] Menon, A., Mehrotra, K., Mohan, C. K., and Ranka, S. Characterization of a class of sigmoid functions with applications to neural networks. *Neural Networks*, 5 (1996), 819 – 835.

[172] Merris, R. Laplacian matrices of graphs: a survey. *Linear Algebra and its Applications* (1994), 143 – 176.

[173] Meyer, A. d. S., Garcia, A. A. F., Souza, A. P. d., and Souza Jr., C. L. d. Comparison of similarity coefficients used for cluster analysis with dominant markers in maize (Zea mays L). *Genetics and Molecular Biology* (2004), 83 – 91.

[174] Minka, T. P. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence* (2001), pp. 362–369.

[175] Mohammadiha, N., Smaragdis, P., Panahandeh, G., and Doclo, S. A state-space approach to dynamic nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 4 (feb 2015), 949–959.

[176] Montgomery, D., Peck, E., and Vining, G. *Introduction to Linear Regression Analysis*. Wiley, 2012.

[177] Murray, I., Adams, R. P., and MacKay, D. J. C. Elliptical slice sampling. In *AISTATS* (2010), pp. 541–548.

[178] Nakatsuji, M., Fujiwara, Y., Uchiyama, T., and Toda, H. Collaborative filtering by analyzing dynamic user interests modeled by taxonomy. In *Proceedinsg of the 11th International Semantic Web Conference* (2012), pp. 361–377.

[179] Ostuni, V. C., Di Noia, T., Di Sciascio, E., and Mirizzi, R. Top-n recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM Conference on Recommender Systems* (2013), pp. 85–92.

[180] Osugi, T., Kim, D., and Scott, S. Balancing exploration and exploration: a new algorithm for active machine learning. In *Proceedings of the Fifth IEEE International Conference on Data Mining* (2005), pp. 8 pp.–.

[181] Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, nov 1999.

[182] Pan, J., Ma, Z., Pang, Y., and Yuan, Y. Robust probabilistic tensor analysis for time-variant collaborative filtering. *Neurocomputing* (2013), 139 – 143.

[183] Pan, J.-Y., Yang, H.-J., Faloutsos, C., and Duygulu, P. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2004), pp. 653–658.

[184] Poriya, A., Bhagat, T., Patel, N., and Sharma, R. Article: Non-personalized recommender systems and user-based collaborative recommender systems. *International Journal of Applied Information Systems*, 9 (mar 2014), 22–27.

[185] Porteous, I., Bart, E., and Welling, M. Multi-hdp: A non parametric bayesian model for tensor factorization. In *Proceedings of the 23rd National Conference on Artificial Intelligence* (2008), pp. 1487–1490.

[186] Porteous, I., Newman, D., Ihler, A., Asuncion, A., Smyth, P., and Welling, M. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2008), pp. 569–577.

[187] Ravn, M. O., and Uhlig, H. On adjusting the Hodrick-Prescott filter for the frequency of observations. *The Review of Economics and Statistics*, 2 (May 2002), 371–375.

[188] Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence* (2009), pp. 452–461.

[189] Rendle, S., Freudenthaler, C., and Schmidt-Thieme, L. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web* (2010), pp. 811–820.

[190] Resnick, P., and Varian, H. R. Recommender systems. *Communications of the ACM*, 3 (Mar. 1997), 56–58.

[191] Rong, P., Yunhong, Z., Bin, C., Liu, N. N., Lukose, R., Scholz, M., and Qiang, Y. One-class collaborative filtering. In *Proceedings of the Eighth IEEE International Conference on Data Mining* (2008), pp. 502–511.

[192] Rue, H., and Held, L. *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall, 2005.

[193] Saha, A., and Sindhwani, V. Learning evolving and emerging topics in social media: A dynamic nmf approach with temporal regularization. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining* (2012), pp. 693–702.

[194] Sahoo, N., Krishnan, R., Duncan, G., and Callan, J. Research note—the halo effect in multicomponent ratings and its implications for recommender systems: The case of yahoo! movies. *Information Systems Research*, 1 (Mar. 2012), 231–246.

[195] Sahoo, N., Singh, P. V., and Mukhopadhyay, T. A hidden markov model for collaborative filtering. *MIS Quarterly*, 4 (2012), 1329–1356.

[196] Salakhutdinov, R., and Mnih, A. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the International Conference on Machine Learning* (2008), pp. 880–887.

[197] Salakhutdinov, R., and Mnih, A. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems* (2008).

[198] Salter, J., and Antonopoulos, N. Cinemascreen recommender agent: Combining collaborative and content-based filtering. *IEEE Intelligent Systems*, 1 (Jan. 2006), 35–41.

[199] Samuel Nowakowski, A. B. Automatic tracking and control for web recommendation new approaches for web recommendation. *International Journal On Advances in Intelligent Systems, IARIA* (2013).

[200] Sanjeev Arulampalam, M., Maskell, S., Gordon, N., and Clapp, T. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 2 (2002), 174–188.

[201] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (2001), pp. 285–295.

[202] Saveski, M., and Mantrach, A. Item cold-start recommendations: Learning local collective embeddings. In *Proceedings of the 8th ACM Conference on Recommender Systems* (2014), pp. 89–96.

[203] Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (2002), pp. 253–260.

[204] Semeraro, G., Lops, P., Basile, P., and de Gemmis, M. Knowledge infusion into content-based recommender systems. In *Proceedings of the Third ACM Conference on Recommender Systems* (2009), pp. 301–304.

[205] Shi, Y., Larson, M., and Hanjalic, A. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (2010), pp. 269–272.

[206] Shi, Y., Larson, M., and Hanjalic, A. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the 4th ACM conference on Recommender Systems* (2010), pp. 269–272.

[207] Sindhwani, V., Bucak, S. S., Hu, J., and Mojsilovic, A. One-class matrix completion with low-density factorizations. In *Proceedings of the Eighth IEEE International Conference on Data Mining* (2010), pp. 1055–1060.

[208] Smyth, B. Case-based recommendation. In *The Adaptive Web*. Springer Berlin Heidelberg, 2007, pp. 342–376.

[209] Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J. Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 12 (2008), 4629–4640.

[210] Srkk, S. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[211] Stamp, M. A revealing introduction to hidden markov models, 2004.

[212] Steck, H. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), pp. 713–722.

[213] Stern, D. H., Herbrich, R., and Graepel, T. Matchbox: Large scale online bayesian recommendations. In *Proceedings of the 18th International Conference on World Wide Web* (2009), pp. 111–120.

[214] Su, P., and Ye, H. An item based collaborative filtering recommendation algorithm using rough set prediction. In *Proceedings of International Joint Conference on Artificial Intelligence* (2009), pp. 308–311.

[215] Su, X., and Khoshgoftaar, T. M. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* (Jan. 2009), 4:2–4:2.

[216] Sun, J., Ma, J., Liu, Z., and Miao, Y. Leveraging content and connections for scientific article recommendation in social computing contexts. *The Computer Journal*, 9 (2014), 1331–1342.

[217] Sun, J., Parthasarathy, D., and Varshney, K. Collaborative kalman filtering for dynamic matrix factorization. *IEEE Transactions on Signal Processing*, 14 (jul 2014), 3499–3509.

[218] Sun, J. Z., Varshney, K. R., and Subbian, K. Dynamic matrix factorization: A state space approach. *Computer Research Repository* (2011).

[219] Sundhararajan, S., Pahwa, A., and Krishnaswami, P. A comparative analysis of genetic algorithms and directed grid search for parametric optimization. *Engineering with Computers*, 3 (1998), 197–205.

[220] Symeonidis, P., Nanopoulos, A., and Manolopoulos, Y. Providing justifications in recommender systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 6 (nov 2008), 1262–1272.

[221] Takács, G., and Tikk, D. Alternating least squares for personalized ranking. In *Proceedings of the Sixth ACM Conference on Recommender Systems* (2012), pp. 83–90.

[222] Takács, G., and Tikk, D. Alternating least squares for personalized ranking. In *Proceedings of the Sixth ACM Conference on Recommender Systems* (2012), pp. 83–90.

[223] Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 476 (2006), 1566–1581.

[224] Vargas, S., and Castells, P. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (2011), pp. 109–116.

[225] Vembu, S., and Grtner, T. Label ranking algorithms: A survey. In *Preference Learning*. Springer Berlin Heidelberg, 2011, pp. 45–64.

[226] Vitter, J. S. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 1 (Mar. 1985), 37–57.

[227] Wainwright, M. J., and Jordan, M. I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1-2 (Jan. 2008), 1–305.

[228] Wang, C., and 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* (2011), pp. 448–456.

[229] Wang, J., Sarwar, B., and Sundaresan, N. Utilizing related products for post-purchase recommendation in e-commerce. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (2011), pp. 329–332.

[230] Wang, J., and Zhang, Y. Opportunity model for e-commerce recommendation: Right product; right time. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2013), pp. 303–312.

[231] Wang, K., Zhang, R., Liu, X., Guo, X., Sun, H., and Huai, J. Time-aware travel attraction recommendation. In *Web Information Systems Engineering* (2013), pp. 175–188.

[232] Wang, Z., Djuric, N., Crammer, K., and Vucetic, S. Trading representability for scalability: Adaptive multi-hyperplane machine for nonlinear classification. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2011), pp. 24–32.

[233] Wei, X., Sun, J., and Wang, X. Dynamic mixture models for multiple time series. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence* (2007), pp. 2909–2914.

[234] Weimer, M., Karatzoglou, A., and Smola, A. Improving maximum margin matrix factorization. In *Machine Learning and Knowledge Discovery in Databases* (2008), pp. 14–14.

[235] Wettschereck, D., Aha, D., and Mohri, T. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 1-5 (1997), 273–314.

[236] Wilson, A. G., and Ghahramani, Z. Generalised wishart processes. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence* (2011), pp. 736–744.

[237] Winn, J. M., and Bishop, C. M. Variational message passing. *Journal of Machine Learning Research* (2005), 661–694.

[238] Xiang, L., and Yang, Q. Time-dependent models in collaborative filtering based recommender system. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies* (2009), pp. 450–457.

[239] Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., and 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* (2010), pp. 723–732.

[240] Xiao, L. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research* (Dec. 2010), 2543–2596.

[241] Xiong, L., Chen, X., Huang, T.-K., Schneider, J., and Carbonell, J. G. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of SIAM Data Mining* (2010), pp. 211–222.

[242] Yang, H., King, I., and Lyu, M. R. Online learning for multi-task feature selection. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (2010), pp. 1693–1696.

[243] Ye, M., Liu, X., and 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* (2012), pp. 671–680.

[244] Yin, B., Yang, Y., and Liu, W. Exploring social activeness and dynamic interest in community-based recommender system. In *Proceedings of the 23rd International Conference on World Wide Web* (2014), pp. 771–776.

[245] Yin, D., Hong, L., and Davison, B. D. Exploiting session-like behaviors in tag prediction. In *Proceedings of the 20th International Conference Companion on World Wide Web* (2011), pp. 167–168.

[246] Yin, D., Hong, L., Xue, Z., and 0001, B. D. D. Temporal dynamics of user interests in tagging systems. In *AAAI Conference on Artificial Intelligence* (2011).

[247] Yuan, Q., Cong, G., Ma, Z., Sun, A., and Thalmann, N. M. Time-aware point-of-interest recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2013), pp. 363–372.

[248] Zhang, L., Tang, J., and Zhang, M. Integrating temporal usage pattern into personalized tag prediction. In *Web Technologies and Applications* (2012), pp. 354–365.

[249] Zhang, W., Chen, T., Wang, J., and Yu, Y. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2013), pp. 785–788.

[250] Zhang, Y., Ahmed, A., Josifovski, V., and Smola, A. Taxonomy discovery for personalized recommendation. In *Proceedings of the Seventh ACM International Conference on Web Search and Data Mining* (2014), pp. 243–252.

[251] Zhao, G., Lee, M. L., Hsu, W., and Chen, W. Increasing temporal diversity with purchase intervals. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2012), pp. 165–174.

[252] Zhao, P., Jin, R., Yang, T., and Hoi, S. C. Online auc maximization. In *Proceedings of the 28th International Conference on Machine Learning* (2011), pp. 233–240.

[253] Zheng, N., and Li, Q. A recommender system based on tag and time information for social tagging systems. *Expert Systems with Applications*, 4 (2011), 4575 – 4587.

[254] Zhu, X. Semi-supervised learning literature survey. Tech. rep., Computer Sciences, University of Wisconsin-Madison, 2006.

[255] Zia, A., Kirubarajan, T., Reilly, J. P., Derek, Y., Punithakumar, K., and Shirani, S. An em algorithm for nonlinear state estimation with model uncertainties. *IEEE Transactions on Signal Processing*, 3 (2008), 921–936.