Noname manuscript No.
(will be inserted by the editor)

Preference Relation-based Markov Random Fields for Recommender Systems

Shaowu Liu · Gang Li · Truyen Tran · Yuan Jiang

Received: date / Accepted: date

Abstract A *preference relation*-based Top-N recommendation approach is proposed to capture both second-order and higher-order interactions among users and items. Traditionally Top-N recommendation was achieved by predicting the item ratings first, and then inferring the item rankings, based on the assumption of availability of *explicit* feedback such as ratings, and the assumption that optimizing the ratings is equivalent to optimizing the item rankings. Nevertheless, both assumptions are not always true in real world applications. The proposed approach drops these assumptions by exploiting *preference relations*, a more practical user feedback. Furthermore, the proposed approach enjoys the representational power of Markov Random Fields thus side information such as item and user attributes can be easily incorporated. Comparing to related work, the proposed approach has the unique property of modeling both second-order and higher-order interactions among users and items. To the best of our knowledge, this is the first time both types of interactions have been captured in *preference-relation* based methods. Experimental results on public datasets demonstrate that both types of

Shaowu Liu
[1] School of Information Technology, Deakin University, Geelong, Australia
[2] Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
E-mail: s.liu@deakin.edu.au

Gang Li (corresponding author)
School of Information Technology, Deakin University, Geelong, Australia
E-mail: gang.li@deakin.edu.au

Truyen Tran
Pattern Recognition and Data Analytics, Deakin University, Geelong, Australia
E-mail: truyen.tran@deakin.edu.au

Yuan Jiang
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210023, China
E-mail: jiangyuan@nju.edu.cn

interactions have been properly captured, and significantly improved Top-N recommendation performance has been achieved.

Keywords Recommender Systems · Collaborative Filtering · Preference Relation · Pairwise Preference · Markov Random Fields

1 Introduction

Recommender Systems (RecSys) aim to recommend users with some of their potentially interesting items, which can be virtually anything ranging from *movies* to *tourism attractions*. To identify the appropriate items, RecSys attempts to exploit *user preferences* (Koren et al, 2009) and various *side information* including *content* (Balabanović and Shoham, 1997; Basilico and Hofmann, 2004), *temporal dynamics* (Koren, 2010), and *social relations* (Ma et al, 2011). By far, *Collaborative Filtering* (Koren et al, 2009) is one of the most popular RecSys techniques, which exploits user preferences, especially in the form of explicit *absolute ratings*. Nevertheless, relying solely on *absolute ratings* is prone to the *cold-start* problem (Schein et al, 2002) where few ratings are known for *cold* users or items. To alleviate the *cold-start* problem, additional information, which is usually heterogeneous (Basilico and Hofmann, 2004) and implicit (Rendle et al, 2009) must be acquired and exploited.

Recently, a considerable literature (Liu et al, 2009; Rendle et al, 2009; Desarkar et al, 2012; Brun et al, 2010; Shi et al, 2010) has grown up around the theme of *relative* preferences. The underlying motivation is that *relative* preferences are often easier to collect and more reliable as a measure of user preferences. For example, it can be easier for users to tell which item is preferable than expressing the precise degree of liking. Furthermore, studies (Koren and Sill, 2011; Brun et al, 2010) have reported that absolute ratings may not be completely trustworthy. For example, rating 4 out of 5 may in general indicate high quality, but it can mean just average for critics. In fact, users' quantitative judgment can be affected by irrelevant factors such as their *mood* when rating, and this is called misattribution of memory (Schacter and Dodson, 2001).

While users are not good at making consistent quantitative judgments, the *preference relation* (PR), as a kind of relative preference, has been considered as more consistent across like-minded users (Brun et al, 2010; Desarkar et al, 2012, 2010). By measuring the relative order between items, the *PR* is usually less variant to irrelevant factors. For example, a user in a bad *mood* may give lower ratings to all items but the relative orderings between items remain the same. Being a more reliable type of user preference, *PR* is also easier to collect comparing to ratings as it can be inferred from implicit feedback. For example, the *PR* between two items can be inferred by comparing their *ratings*, *page views*, *played counts*, *mouse clicks*, etc. This property is important as not all users are willing to rate their preferences, where collecting feedback implicitly delivers a more user-friendly recommender system. In addition, the output of *RecSys* is often the item ranking, which is also a type of relative preference, and therefore relative preference is more natural input than absolute ratings (Tran et al, 2012; Koren and Sill, 2011).

While *PR* captures user preferences in *pairwise* form, most existing works (Koren and Sill, 2011; Liu et al, 2014) take the *pointwise* approach to exploiting ordinal properties possessed by absolute ratings. To accept the *PR* as input and output item rankings, *pairwise* approaches to RecSys have recently emerged in two forms: memory-based (Brun et al, 2010) and model-based (Liu et al, 2009; Rendle et al, 2009; Desarkar et al, 2012). These studies have shown the feasibility of *PR*-based methods, and demonstrated competitive performance comparing to their underlying models, such as memory-based *K-Nearest Neighbor* (KNN) (Brun et al, 2010) and model-based *Matrix Factorization* (MF) (Desarkar et al, 2012).

However, the limitations of these underlying models have constrained the potential of their *PR* extensions. More specifically, both *KNN* and *MF* based methods can only capture one type of information at a time, while both the *local* and the *global* information are essential in achieving good performance (Tran et al, 2009; Koren, 2008; Liu et al, 2014). We refer to these two types of information as the *local* and the *global* structures of the preferences:

Local Structure The *local structure* (LS) refers to the second-order interactions between similar users (Resnick et al, 1994) or items (Sarwar et al, 2001). This type of information is often used by *neighborhood-based* collaborative filtering, in which the predictions are made by looking at the neighborhood of users (Resnick et al, 1994) or items (Sarwar et al, 2001). *LS*-based approaches ignore the majority of preferences in making predictions, but are effective when the user or item correlations are highly localized.

Global Structure The *global structure* (GS) refers to the weaker but higher-order interactions among all users and items (Koren et al, 2009). This type of information is often used by *latent factor models* such as *Matrix Factorization* (Koren et al, 2009), which aim to discover the latent factor space in the preferences. *GS*-based approaches are often competitive in terms of accuracy and computational efficiency (Koren et al, 2009).

Previous studies have suggested that these two structures are complementary since they address different aspects of the preferences (Tran et al, 2009; Koren, 2008; Liu et al, 2014). However, to the best of our knowledge, there is yet no *PR*-based method that can capture both *LS* and *GS*. Another problem of existing *PR*-based methods is that side information such as item content and user attributes can't be easily incorporated, which is critical in cold-start cases. All the above reasoning lead to a desired model with the following properties: *1*) Accept *PR* as input; *2*) Capture both *LS* and *GS*; *3*) Side information can be easily incorporated; *4*) Output item rankings.

Recent advances in *Markov Random Fields*-based RecSys (Tran et al, 2009; Defazio and Caetano, 2012; Liu et al, 2014) have made it possible to achieve the above objectives. *MRF*-based RecSys was first developed in (Tran et al, 2009) to capture both *LS* and *GS*. Later on, it has been extended in (Liu et al,

2014) to exploit ordinal properties possessed by absolute ratings. Nevertheless, all of these attempts rely on absolute ratings.

This paper aims to push the *MRF*-based RecSys one step further by fitting it into the *PR* framework, namely *Preference Relation-based Markov Random Fields* (PrefMRF) and *Preference Relation-based Conditional Random Fields* (PrefCRF) when side information is incorporated. The remaining parts of this paper are organized as follows. Section 2 introduces the concepts of *PR*-based RecSys and formalizes the problem, followed by a review of related work. Section 3 is devoted to the proposed *PrefMRF* and *PrefCRF* models. Benchmark results on Top-N recommendation are presented in Section 4. Finally, Section 5 concludes this paper by summarizing the main contributions and envisaging future work.

2 Preliminaries and Related Work

Recommender Systems (RecSys) aim at predicting users' future interest in items, and the recommendation task can be considered as a *preference learning* problem, which aims to construct a predictive preference model from observed preference information (Mohri et al, 2012). Existing *preference learning* methods are based on different *learning to rank* approaches (Fürnkranz and Hüllermeier, 2010). Among them, the *pointwise* approach is the choice of most RecSys (Sarwar et al, 2001; Koren, 2008), which exploit absolute ratings, though the *pairwise* approach that exploits *PR* has been largely overlooked until recently. The rest of this section describes the basic concepts and formalizes the *PR*-based RecSys followed by a review of related work.

2.1 Preference Relation

Preference relation has been widely studied in the field of *Information Retrieval* (Cohen et al, 1999; Fürnkranz and Hüllermeier, 2003; Freund et al, 2003; Jin et al, 2002). Nevertheless, *PR*-based RecSys have only emerged recently (Liu et al, 2009; Rendle et al, 2009; Brun et al, 2010; Desarkar et al, 2012). A *preference relation* (PR) encodes user preferences in the form of pairwise ordering between items. This representation is a useful alternative to absolute ratings for three reasons.

Firstly, *PR* is more consistent across like-minded users (Brun et al, 2010; Desarkar et al, 2012) as it is invariant to many irrelevant factors, such as *mood*. Secondly, *PR* is a more natural and direct input for Top-N recommendation, as both the input and the output are relative preferences. Finally, and perhaps most importantly, *PR* can be inferred from implicit feedback such as *page views* and *stayed time*. This property provides an opportunity to utilize the vast amount of implicit data that has already been collected over the years, e.g., *activity logs*. With these potential benefits, we shall take a closer look at the *PR*, and investigate how they can be utilized in RecSys.

We formally define the *PR* as follows. Let $\mathcal{U} = \{u\}^n$ and $\mathcal{I} = \{i\}^m$ denote the set of n users and m items, respectively. The preference of a user $u \in \mathcal{U}$ between items i and j is encoded as π_{uij}, which indicates the strength of user u's *PR* for the ordered item pair (i, j). A higher value of π_{uij} indicates a stronger preference on the first item over the second item.

Definition 1 (Preference Relation) The preference relation is defined as

$$\pi_{uij} = \begin{cases} (\frac{2}{3}, 1] & \text{if } i \succ j \ (u \text{ prefers } i \text{ over } j) \\ [\frac{1}{3}, \frac{2}{3}] & \text{if } i \simeq j \ (i \text{ and } j \text{ are equally preferable to } u) \\ [0, \frac{1}{3}) & \text{if } i \prec j \ (u \text{ prefers } j \text{ over } i) \end{cases} \tag{2.1}$$

where $\pi_{uij} \in [0, 1]$ and $\pi_{uij} = 1 - \pi_{uji}$.

This definition is similar to (Desarkar et al, 2012), however, we allocate an interval for each preference category, i.e., *preferred*, *equally preferred*, and *less preferred*. Indeed, each preference category can be further broken down into more intervals.

Similar to (Brun et al, 2010), the *PR* can be converted into user-wise preferences over items.

Definition 2 (User-wise Preference) The user-wise preference is defined as

$$p_{ui} = \frac{\sum_{j \in \mathcal{I}_u} [\![\pi_{uij} > \frac{2}{3}]\!] - \sum_{j \in \mathcal{I}_u} [\![\pi_{uij} < \frac{1}{3}]\!]}{|\Pi_{ui}|} \tag{2.2}$$

where $[\![\cdot]\!]$ gives 1 for *true* and 0 for *false*, $\mathcal{I}_u$ is the set of items related to user u, and Π_{ui} is the set of user u's PR related to item i.

The user-wise preference p_{ui} falls in the interval $[-1, 1]$, where -1 and 1 indicate that item i is the least or the most preferred item for user u, respectively. The user-wise preference measures the relative position of an item for a particular user, which is different from absolute ratings.

2.2 Problem Statement

Generally, the task of *PR*-based RecSys is to take *PR* as input and output Top-N recommendations. Specifically, let $\pi_{uij} \in \Pi$ encode the *PR* of each user $u \in \mathcal{U}$. Each π_{uij} is defined over an ordered item pair (i, j), denoting $i \prec j$, $i \simeq j$, or $i \succ j$ as described in Eq. 2.1. The goal is to estimate the value of each unknown $\pi_{uij} \in \Pi_{unknown}$, such that $\hat{\pi}_{uij}$ approximates π_{uij}. This can be considered as an optimization task performed directly on the *PR*:

$$\min_{\hat{\Pi}} \sum_{\hat{\pi}_{uij} \in \hat{\Pi}} (\pi_{uij} - \hat{\pi}_{uij})^2 \tag{2.3}$$

However, it can be easier to estimate the $\hat{\pi}_{uij}$ by the difference between the two user-wise preferences p_{ui} and p_{uj}, i.e., $\hat{\pi}_{uij} = \phi(\hat{p}_{ui} - \hat{p}_{uj})$, where $\phi(\cdot)$

is a function that bounds the value into $[0, 1]$ and ensures $\phi(0) = 0.5$. For example, the *inverse-logit* function $\phi(x) = \frac{e^x}{1+e^x}$ can be used when user-wise preferences involve large values. Therefore, the objective of this paper is to solve the following optimization problem:

$$\min_{\hat{\mathbf{p}}_u} \sum_{(\hat{p}_{ui}, \hat{p}_{uj}) \in \hat{\mathbf{p}}_u} (\pi_{uij} - \phi(\hat{p}_{ui} - \hat{p}_{uj}))^2 \tag{2.4}$$

which optimizes the user-wise preferences directly, and Top-N recommendations can be obtained by simply sorting the estimated user-wise preferences.

2.3 Related Work

User preferences can be modeled in three types: *pointwise*, *pairwise*, and *listwise*. Though RecSys is not limited to the *pointwise* absolute ratings, the recommendation task is usually considered as a rating prediction problem. Recently, a considerable literature (Liu et al, 2009; Rendle et al, 2009; Desarkar et al, 2012; Brun et al, 2010; Shi et al, 2010; Rendle et al, 2009; Gantner et al, 2010) has grown up around the theme of *relative* preferences, especially the *pairwise PR*. Meanwhile, the recommendation task is also shifting from rating prediction to item ranking (Weimer et al, 2007; Shi et al, 2010; McNee et al, 2006), in which the ranking itself is also of *relative* preferences.

The use of *relative* preferences has been widely studied in the field of *Information Retrieval* for *learning to rank* tasks. Recently, *PR*-based (Liu et al, 2009; Rendle et al, 2009; Desarkar et al, 2012; Brun et al, 2010) and *listwise*-based (Shi et al, 2010) RecSys have been proposed. Among them, the *PR*-based approach is the most popular, which can be further categorized as *memory-based* methods (Brun et al, 2010) that capture *local structure* and *model-based* methods (Liu et al, 2009; Rendle et al, 2009; Desarkar et al, 2012) that capture *global structure*. To the best of our knowledge, there is yet no *PR*-based method that can capture both *LS* and *GS*.

Advances in *Markov Random Fields* (MRF) and its extension *Conditional Random Fields* (CRF) have made it possible to utilize both *LS* and *GS* by taking advantages of MRF's powerful representation capability. Nevertheless, exploiting the *PR* is not an easy task for *MRF* and *CRF* (Tran et al, 2009; Liu et al, 2014). This observation leads to a natural extension of unifying the MRF models with the *PR*-based models, to complement their strengths. We summarize the capabilities of the existing and our proposed *PrefMRF* and *PrefCRF* models in Table 1.

3 Methodology

In this section, we propose *Preference Relation-based Markov Random Fields* (PrefMRF) to model the *PR* and capture both *LS* and *GS*. When side information is taken into consideration, this model extends to *Preference Relation-based Conditional Random Fields* (PrefCRF). In this work, we exploit *LS* in

Table 1 Capabilities of Different Methods

Method	Input	Output	LS	GS	Side Information
Pointwise Memory-based	Ratings	Ratings	✓		
Pointwise Model-based	Ratings	Ratings		✓	
Pointwise Hybrid	Ratings	Ratings	✓	✓	
Pairwise Memory-based	Preference Relations	Item Rankings	✓		
Pairwise Model-based	Preference Relations	Item Rankings		✓	
PrefMRF	**Preference Relations**	**Item Rankings**	✔	✔	
PrefCRF	**Preference Relations**	**Item Rankings**	✔	✔	✔

terms of the item-item correlations as well as the user-user correlations. The rest of this section introduces the concept of the *Preference Relation-based Matrix Factorization* (PrefNMF) (Desarkar et al, 2012) that will be our underlying model, followed a detailed discussion of the *PrefMRF* and *PrefCRF* on issues including feature design, parameter estimation, and predictions. For ease of reference, notations used throughout this paper are summarized in Table 2. The letters u, v, a, b represent *users*, and the letters i, j, k, l represent *items*.

Table 2 Summary of major notations

Notations	Mathematical Meanings
$\mathcal{U}$	the set of users
$\mathcal{I}$	the set of items
Π	the set of preference relations
p_{ui}	the user-wise preference of user u on item i
$\mathcal{G}$	an undirected graph encodes relations of user-wise preferences
$\mathcal{V}$	the set of vertices each represents a user-wise preference
$\mathcal{E}$	the set of edges each connects two vertices
f_{uv}	the correlation feature between users u and v
f_{ij}	the correlation feature between items i and j
w_{uv}	the weight associated to the user-user correlation feature f_{uv}
w_{ij}	the weight associated to the item-item correlation feature f_{ij}
$Q(p_{ui} \mid u, i)$	the ordinal distribution produced by PrefNMF
$\mathbf{o}$	the side information, e.g., user attributes and item content

3.1 Preference Relation-based Matrix Factorization

Matrix Factorization (MF) (Koren et al, 2009) is a popular approach to *RecSys* that has mainly been applied to absolute ratings. Recently, the *PrefNMF* (Desarkar et al, 2012) model was proposed to adopt *PR* input for *MF* models. Like traditional *MF* models, the *PrefNMF* model discovers the latent factor space shared between users and items, where the latent factors describe both the *taste* of users and the *characteristics* of items. The attractiveness of an item to a user is then measured by the inner product of their latent feature vectors.

3.1.1 Point Estimation

The *PrefNMF* model described in (Desarkar et al, 2012) provides a point estimation to each preference, i.e., a single value. Formally, each user u is associated with a latent feature vector $\mathbf{u}_u \in \mathbb{R}^k$ and each item i is associated with a latent feature vector $\mathbf{v}_i \in \mathbb{R}^k$, where k is the dimension of the latent factor space. The attractiveness of items i and j to the user u are $\mathbf{u}_u^\top \mathbf{v}_i$ and $\mathbf{u}_u^\top \mathbf{v}_j$, respectively. When $\mathbf{u}_u^\top \mathbf{v}_i > \mathbf{u}_u^\top \mathbf{v}_j$ the item i is said to be more preferable to the user u than the item j, i.e., $i \succ j$. The strength of this preference relation π_{uij} can be estimated by $\mathbf{u}_u^\top(\mathbf{v}_i - \mathbf{v}_j)$, and the *inverse-logit* function is applied to ensure $\hat{\pi}_{uij} \in [0, 1]$:

$$\hat{\pi}_{uij} = \frac{e^{\mathbf{u}_u^\top(\mathbf{v}_i - \mathbf{v}_j)}}{1 + e^{\mathbf{u}_u^\top(\mathbf{v}_i - \mathbf{v}_j)}} \tag{3.1}$$

The latent feature vectors $\mathbf{u}_u$ and $\mathbf{v}_i$ are learned by minimizing regularized squared error with respect to the set of all known preference relations $\mathit{\Pi}$:

$$\min_{\mathbf{u}_u, \mathbf{v}_i \in \mathbb{R}^k} \sum_{\pi_{uij} \in \mathit{\Pi} \wedge (i<j)} (\pi_{uij} - \hat{\pi}_{uij})^2 + \lambda(\|\mathbf{u}_u\|^2 + \|\mathbf{v}_i\|^2) \tag{3.2}$$

where λ is the regularization coefficient and $(i < j)$ removes duplicated calculations. The optimization can be done with *Stochastic Gradient Descent* in order to be faster on sparse data, or with *Alternating Least Squares* to enable parallelization on dense data.

3.1.2 Distribution Estimation

The original *PrefNMF* (Desarkar et al, 2012) computes the attractiveness of an item to a user by the product of their latent feature vectors which results in a scalar value, where the likelihoods of other possible values remain unknown. However, in order to be combined with *MRF* models, we wish to have the distributions over a set of possible values. Therefore the *Random Utility Models* (McFadden, 1980) and *Ordinal Logistic Regression* (McCullagh, 1980) are applied to perform the conversion.

Random Utility Models (McFadden, 1980) assume the existence of a *latent utility* $x_{ui} = \mu_{ui} + \epsilon_{ui}$ that captures how much the user u is interested in the item i, where μ_{ui} captures the interest and ϵ_{ui} is the random noise that follows the logistic distribution as in (Koren and Sill, 2011). The latent utility x_{ui} is then generated from a logistic distribution centered at μ_{ui} with the scale parameter s_{ui} proportional to the standard deviation:

$$x_{ui} \sim \mathit{Logi}(\mu_{ui}, s_{ui}) \tag{3.3}$$

Recall that *PrefNMF* computes the attractiveness of an item to a user by the product of their latent feature vectors, thereby the internal score μ_{ui} can be substituted with the term $\mathbf{u}_u^\top \mathbf{v}_i$:

$$x_{ui} = \mathbf{u}_u^\top \mathbf{v}_i + \epsilon_{ui} \tag{3.4}$$

where $\mathbf{u}_u$ and $\mathbf{v}_i$ are, respectively, the latent feature vectors of the user u and the item i.

Ordinal Logistic Regression (McCullagh, 1980) is then used to convert the user-wise preferences p_{ui} into ordinal values, which assumes that the preference p_{ui} is chosen based on the interval to which the latent utility belongs:

$$p_{ui} = l \text{ if } x_{ui} \in (\theta_{l-1}, \theta_l] \text{ for } l < L \text{ and } p_{ui} = L \text{ if } x_{ui} > \theta_{L-1} \tag{3.5}$$

where L is the number of ordinal levels and θ_l are the threshold values of interest. The probability of receiving a preference l is therefore

$$Q(p_{ui} = l \mid u, i) = \int_{\theta_{l-1}}^{\theta_l} P(x_{ui} \mid \theta)\, \mathrm{d}\theta = F(\theta_l) - F(\theta_{l-1}) \tag{3.6}$$

where $F(\theta_l)$ is the cumulative logistic distribution evaluated at θ_l with standard deviation s_{ui}:

$$F(x_{ui} \leq l \mid \theta_l) = \frac{1}{1 + \exp(-\frac{\theta_{uil} - \mu_{ui}}{s_{ui}})} \tag{3.7}$$

The thresholds θ_l can be parameterized to depend on user or item. This paper employs the user-specific thresholds parameterization described in (Koren and Sill, 2011). Therefore a set of thresholds $\{\theta_{ul}\}_{l=1}^{L}$ is defined for each user u to replace the thresholds θ_{uil} in Eq. 3.7, and is learned from data.

Given the learned ordinal distribution $Q(p_{ui} \mid u, i)$, not only can the preferences be predicted but also the *confidence* for each prediction. The ordinal distribution $Q(p_{ui} \mid u, i)$ captures the *GS* information in a probabilistic form, and will be incorporated into *MRF* and *CRF* in Section 3.4. Note that the user preference is quantized into ordinal values in this process.

3.2 Markov Random Fields

Markov Random Fields (MRF) (Tran et al, 2007; Defazio and Caetano, 2012) model a set of random variables having the Markov property with respect to an undirected graph $\mathcal{G}$. The undirected graph $\mathcal{G}$ consists of a set of vertices $\mathcal{V}$ connected by a set of edges $\mathcal{E}$ without orientation, where two vertices are neighborhoods of each other when connected. Each vertex in $\mathcal{V}$ encodes a random variable, and the Markov property implies that a variable is conditionally independent of others given its neighborhoods.

In this work, we use *MRF* to model user-wise preferences and their interactions with respect to a set of undirected graphs. Specifically for each user u, there is a graph $\mathcal{G}_u$ with a set of vertices $\mathcal{V}_u$ and a set of edges $\mathcal{E}_u$. Each vertex in $\mathcal{V}_u$ represents a preference p_{ui} of user u on the item i. Note that the term *preference* is used instead of *rating* because in the new model the *preference* is not interpolated as absolute ratings but user-wise ordering of items. Each edge in $\mathcal{E}_u$ captures a relation between two preferences by the same user.

Two preferences are connected by an edge if they are given by the same user or on the same item, corresponding to the item-item and user-user correlations, respectively. Modeling these correlations is actually capturing the *LS* information in the preferences. However, it is not easy to model two types of correlation at the same time as it will result in a large graph. Instead, we model the item-item and user-user correlations separately, and merge their predictions. Fig. 1 shows an example of four graphs for user u, user v, item i, and item j. Note that vertices of different graphs are not connected directly, however, the weights are estimated across graphs when the edges correspond to the same correlation. For example, the edge between p_{ui} and p_{uj} and the edge between p_{vi} and p_{vj} are associated with the same item-item correlation ψ_{ij} between items i and j.

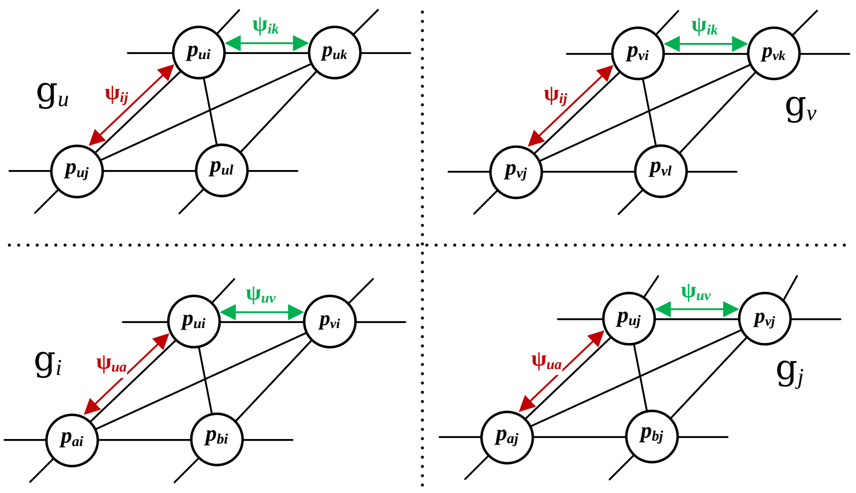


Fig. 1 Example of MRF graphs. u, v, a, and b are users. i, j, l, and k are items.

Formally, let $\mathcal{I}_u$ be the set of all items evaluated by the user u and $\mathcal{U}_i$ be the set of all users who rated the item i. Then denote $\mathbf{p}_u = \{p_{ui} \mid i \in \mathcal{I}_u\}$ to be the joint set of all preferences (the variables) expressed by user u, and $\mathbf{p}_i = \{p_{ui} \mid u \in \mathcal{U}_i\}$ to be the joint set of all preferences (the variables) rated on item i. Under this setting, the *MRF* defines two distributions $P(\mathbf{p}_u)$ and $P(\mathbf{p}_i)$ over the graphs $\mathcal{G}_u$ and $\mathcal{G}_i$, respectively:

$$P(\mathbf{p}_u) = \frac{1}{Z_u}\Psi_u(\mathbf{p}_u) \ , \ P(\mathbf{p}_i) = \frac{1}{Z_i}\Psi_i(\mathbf{p}_i) \tag{3.8}$$

$$\Psi_u(\mathbf{p}_u) = \prod_{(ui,uj)\in\mathcal{E}_u} \psi_{ij}(p_{ui}, p_{uj}) \ , \ \Psi_i(\mathbf{p}_i) = \prod_{(ui,vi)\in\mathcal{E}_i} \psi_{uv}(p_{ui}, p_{vi}) \tag{3.9}$$

where Z_u and Z_i are the normalization terms that ensure $\sum_{\mathbf{p}_u} P(\mathbf{p}_u) = 1$ and $\sum_{\mathbf{p}_i} P(\mathbf{p}_i) = 1$. The term $\psi(\cdot)$ is a positive function known as *potential*.

The potentials $\psi_{ij}(p_{ui}, p_{uj})$ and $\psi_{uv}(p_{ui}, p_{vi})$ capture the correlation between items i and j and correlation between users u and v, respectively:

$$\psi_{ij}(p_{ui}, p_{uj}) = \exp\{w_{ij} f_{ij}(p_{ui}, p_{uj})\} \tag{3.10}$$

$$\psi_{uv}(p_{ui}, p_{vi}) = \exp\{w_{uv} f_{uv}(p_{ui}, p_{vi})\} \tag{3.11}$$

where $f_{ij}(\cdot)$ and $f_{uv}(\cdot)$ are the feature functions to be designed shortly in Section 3.4.1, and w_{ij} and w_{uv} are the corresponding weights.

These correlation features capture the *LS* information, while the weights realize the importance of each correlation feature. With the weights estimated from data, the unknown preference p_{ui} can be predicted using item-item correlations:

$$\hat{p}_{ui} = \underset{p_{ui} \in [-1,1]}{\arg\max}\ P(p_{ui} \mid \mathbf{p}_u) \tag{3.12}$$

or using user-user correlations:

$$\hat{p}_{ui} = \underset{p_{ui} \in [-1,1]}{\arg\max}\ P(p_{ui} \mid \mathbf{p}_i) \tag{3.13}$$

where the confidences of the predictions can be measured by $P(p_{ui} \mid \mathbf{p}_u)$ and $P(p_{ui} \mid \mathbf{p}_i)$.

3.3 Conditional Random Fields

Despite user preferences, various side information including *content* (Balabanović and Shoham, 1997; Basilico and Hofmann, 2004), *temporal dynamics* (Koren, 2010), and *social relations* (Ma et al, 2011) are also important in making quality recommendations. While there exist methods to incorporate side information, there is yet no *PR*-based method that can achieve this.

One advantage of *Markov Random Fields* is its extensibility, thus side information can be easily incorporated by extending the *MRF* to *Conditional Random Fields* (CRF). In *MRF*, the item-item and user-user correlations are modeled by a set of graphs, where each graph has a set of vertices representing the preferences. To incorporate side information, the *MRF* is extended to *CRF* by conditioning each vertex on a set of global observations $\mathbf{o}$, i.e., the side information in our context. Specifically, each user u is associated with a set of attributes $\{\mathbf{o}_u\}$ such as *gender* and *occupation*. Similarly, each item i is associated with a set of attributes $\{\mathbf{o}_i\}$ such as *genres* of movie. This side information is encoded as the set of global observations $\mathbf{o} = \{\{\mathbf{o}_u\}, \{\mathbf{o}_i\}\}$. The graphs for item-item and user-user correlations conditioned on global observations are illustrated in Fig. 2.

Using the same settings as *MRF* in Section 3.2, the *CRF* models the conditional distributions $P(\mathbf{p}_u \mid \mathbf{o})$ and $P(\mathbf{p}_i \mid \mathbf{o})$ over the graphs $\mathcal{G}_u$ and $\mathcal{G}_i$, respectively:

$$P(\mathbf{p}_u \mid \mathbf{o}) = \frac{1}{Z_u(\mathbf{o})} \Psi_u(\mathbf{p}_u, \mathbf{o})\ ,\ P(\mathbf{p}_i \mid \mathbf{o}) = \frac{1}{Z_i(\mathbf{o})} \Psi_i(\mathbf{p}_i, \mathbf{o}) \tag{3.14}$$

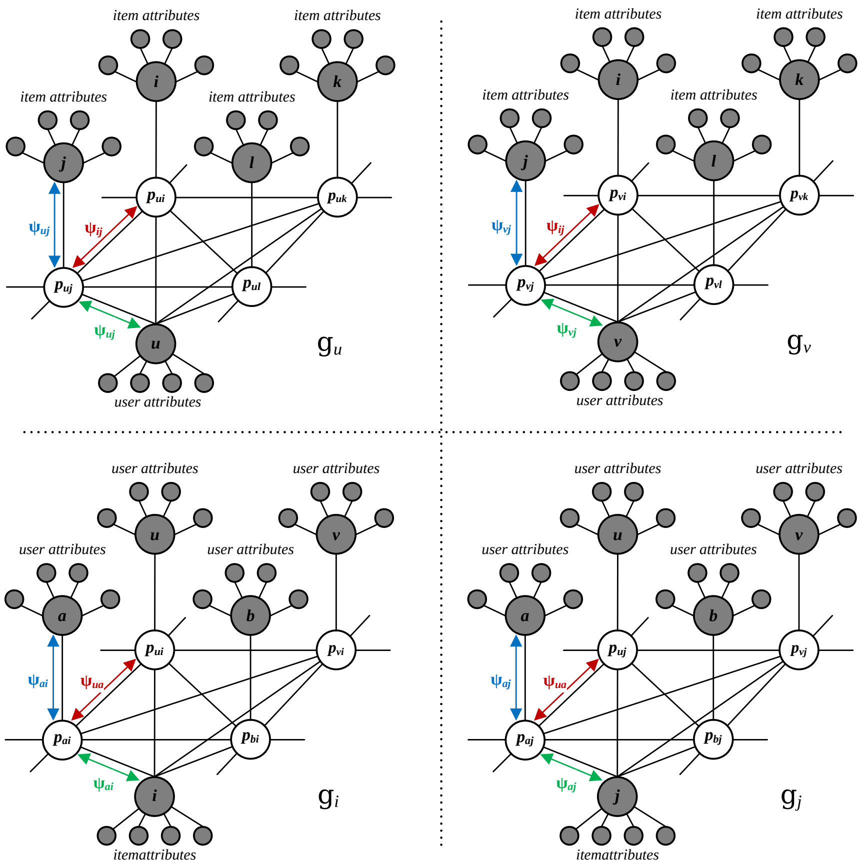


Fig. 2 Example of CRF graphs. u, v, a, and b are users. i, j, l, and k are items.

$$\Psi_u(\mathbf{p}_u, \mathbf{o}) = \prod_{(ui)\in\mathcal{V}_u} \psi_{ui}(p_{ui}, \mathbf{o}) \prod_{(ui,uj)\in\mathcal{E}_u} \psi_{ij}(p_{ui}, p_{uj}) \tag{3.15}$$

$$\Psi_i(\mathbf{p}_i, \mathbf{o}) = \prod_{(ui)\in\mathcal{V}_i} \psi_{ui}(p_{ui}, \mathbf{o}) \prod_{(ui,vi)\in\mathcal{E}_i} \psi_{uv}(p_{ui}, p_{vi}) \tag{3.16}$$

where $Z_u(\mathbf{o})$ and $Z_i(\mathbf{o})$ are the normalization terms that ensure $\sum_{\mathbf{p}_u} P(\mathbf{p}_u \mid \mathbf{o}) = 1$ and $\sum_{\mathbf{p}_i} P(\mathbf{p}_i \mid \mathbf{o}) = 1$.

The potentials $\psi_{ij}(\cdot)$ and $\psi_{uv}(\cdot)$ capture the item-item and user-user correlations in the same way as in Markov Random Fields, i.e., Eq. 3.10 and Eq. 3.11. The potential $\psi_{ui}(\cdot)$ captures the global observations associated with the user u and the item i:

$$\psi_{ui}(p_{ui}, \mathbf{o}) = \exp\{\mathbf{w}_u^\top \mathbf{f}_u(p_{ui}, \mathbf{o}_i) + \mathbf{w}_i^\top \mathbf{f}_i(p_{ui}, \mathbf{o}_u))\} \tag{3.17}$$

where $\mathbf{f}_u$ and $\mathbf{f}_i$ are the features to be designed shortly in Section 3.4.1, and $\mathbf{w}_u$ and $\mathbf{w}_i$ are the corresponding weights realize the importance of each feature.

With the weights estimated from data, the unknown preference p_{ui} can be predicted using item-item correlations:

$$\hat{p}_{ui} = \operatorname*{arg\,max}_{p_{ui} \in [-1,1]} P(p_{ui} \mid \mathbf{p}_u, \mathbf{o}) \tag{3.18}$$

or using user-user correlations:

$$\hat{p}_{ui} = \operatorname*{arg\,max}_{p_{ui} \in [-1,1]} P(p_{ui} \mid \mathbf{p}_i, \mathbf{o}) \tag{3.19}$$

where $P(p_{ui} \mid \mathbf{p}_u, \mathbf{o})$ and $P(p_{ui} \mid \mathbf{p}_i, \mathbf{o})$ give the confidence of the predictions.

3.4 PrefCRF: Unifying PrefNMF and CRF

The *CRF* model captures the *LS* information by modeling item-item and user-user correlations under the framework of probabilistic graphical models. However, it employs the log-linear modeling as shown in Eq. 3.10, Eq. 3.10, and Eq. 3.17, and therefore does not enable a simple treatment of *PR*. The *PrefNMF* model, on the other hand, can nicely model the *PR* but is weak in capturing the *LS* and side information. The complementarity between these two techniques enables the unified *PrefCRF* model to utilise the advantages of both models.

Essentially, the proposed *PrefCRF* model promotes the agreement between the *GS* discovered by the *PrefNMF*, the *LS* discovered by the *MRF*, and the side information discovered by the *CRF*. More specifically, the *PrefCRF* model combines the item-item and user-user correlations (Eq. 3.15 and Eq. 3.16) and the ordinal distributions $Q(p_{ui} \mid u, i)$ over user-wise preferences obtained from Eq. 3.6:

$$P(\mathbf{p}_u \mid \mathbf{o}) \propto \Psi_u(\mathbf{p}_u, \mathbf{o}) \prod_{p_{ui} \in \mathbf{p}_u} Q(p_{ui} \mid u, i) \tag{3.20}$$

$$P(\mathbf{p}_i \mid \mathbf{o}) \propto \Psi_i(\mathbf{p}_i, \mathbf{o}) \prod_{p_{ui} \in \mathbf{p}_i} Q(p_{ui} \mid u, i) \tag{3.21}$$

where Ψ_u is the potential function capturing the interaction among items evaluated by user u, and Ψ_i is the potential function capturing the interaction among users who rated item i. Put together, the joint distribution $P(\mathbf{p}_u)$ for each user u can be modeled as:

$$P(\mathbf{p}_u) \propto \exp\left(\sum_{p_{ui}, p_{uj} \in \mathbf{p}_u} w_{ij} f_{ij}(p_{ui}, p_{uj}) + \sum_{(ui) \in \mathcal{V}_u} \psi_{ui}(p_{ui}, \mathbf{o}) \right) \prod_{p_{ui} \in \mathbf{p}_u} Q(p_{ui} \mid u, i) \tag{3.22}$$

and the joint distribution $P(\mathbf{p}_i)$ for each item i can be modeled as:

$$P(\mathbf{p}_i) \propto \exp\left(\sum_{p_{ui}, p_{vi} \in \mathbf{p}_i} w_{uv} f_{uv}(p_{ui}, p_{vi}) + \sum_{(ui) \in \mathcal{V}_i} \psi_{ui}(p_{ui}, \mathbf{o}) \right) \prod_{p_{ui} \in \mathbf{p}_i} Q(p_{ui} \mid u, i) \tag{3.23}$$

where there is a graph for each user or item but the weights are optimized by all users or all items.

3.4.1 Feature Design

A feature is essentially a function f of $n > 1$ arguments that maps the n-dimensional input into the unit interval $f : \mathbb{R}^n \to [0, 1]$. We design the following kinds of features:

Correlation Features The item-item correlation is captured by the feature:

$$f_{ij}(p_{ui}, p_{uj}) = g(|(p_{ui} - \bar{p}_i) - (p_{uj} - \bar{p}_j)|) \tag{3.24}$$

and the user-user correlation is captured by the feature:

$$f_{uv}(p_{ui}, p_{vi}) = g(|(p_{ui} - \bar{p}_u) - (p_{vi} - \bar{p}_v)|) \tag{3.25}$$

where $g(\alpha) = 1 - \alpha/(L-1)$ normalizes feature values and α plays the role of deviation, where L is the number of ordinal levels described in Section 3.1.2. The terms $\bar{p}_i$, $\bar{p}_j$, $\bar{p}_i$, and $\bar{p}_j$ are the item or user averages. The item-item correlation feature captures the intuition that correlated items should be ranked similarly by the same user after offsetting the goodness of each item. Similarly, the user-user correlation feature captures the intuition that correlated users should rate the same item similarly.

Attribute Features Each user u and item i has a set of attributes $\mathbf{o}_u$ and $\mathbf{o}_i$, respectively. These attributes are mapped to preferences by the following features:

$$\begin{aligned} \mathbf{f}_i(p_{ui}) &= \mathbf{o}_u g(|(p_{ui} - \bar{p}_i)|) \\ \mathbf{f}_u(p_{ui}) &= \mathbf{o}_i g(|(p_{ui} - \bar{p}_u)|) \end{aligned} \tag{3.26}$$

where $\mathbf{f}_i$ models which users like the item i and $\mathbf{f}_u$ models which classes of items the user u likes.

Since one correlation feature exists for each possible pair of co-rated items, the number of correlation features can be large which makes the estimation slow to converge and less robust. Therefore we only keep the correlation features if strong item-item correlation or user-user correlation exists. Specifically, the *strong correlation features* $\mathbf{f}_{\text{strong}}$ are extracted based on the *Pearson Correlation* and a user-specified *minimum correlation threshold*. Note that the correlation is calculated based on the user-wise preferences generated from *PR* thus the rule of using *PR* as input is not violated.

3.4.2 Parameter Estimation

In general, *MRF*-based models cannot be determined by standard maximum likelihood approaches, instead, approximation techniques are often used in practice such as *Pseudo-likelihood* (Besag, 1974) and *Contrastive Divergence* (CD) (Hinton, 2002). The *Pseudo-likelihood* leads to exact computation of the loss function and its gradient with respect to parameters, and thus is faster.

The CD-based methods may, on the other hand, lead to better estimation given enough time. As the experiments involve different settings and large numbers of features, this study employs the *Pseudo-likelihood* technique to perform efficient parameter estimation by maximizing the regularized sum of log local likelihoods:

$$log\mathcal{L}(\mathbf{w}) = \sum_{p_{ui}\in\Pi} \log P(p_{ui} \mid \mathbf{p}_u, \mathbf{o}) - \frac{1}{2\sigma^2}\mathbf{w}^\top\mathbf{w} \tag{3.27}$$

where $\mathbf{w}$ are the weights and $1/2\sigma^2$ controls the regularization. To make the notation uncluttered, we write $\mathbf{p}_u$ instead of explicitly as $\mathbf{p}_u \backslash p_{ui}$. In this section we describe the parameter estimation of item-item correlations, where the user-user correlations can be estimated in the same way by replacing items with users.

The local likelihood in Eq. 3.27 is defined as:

$$P(p_{ui} \mid \mathbf{p}_u, \mathbf{o}) = \frac{1}{Z_{ui}(\mathbf{o})} Q(p_{ui} \mid u, i)\psi_{ui}(p_{ui}, \mathbf{o}) \prod_{p_{uj}\in\mathbf{p}_u} \psi_{ij}(p_{ui}, p_{uj}) \tag{3.28}$$

where $Z_{ui}(\mathbf{o})$ is the normalization term:

$$Z_{ui} = \sum_{p_{ui}=l_{min}}^{l_{max}} Q(p_{ui} \mid u, i)\psi_{ui}(p_{ui}, \mathbf{o}) \prod_{p_{uj}\in\mathbf{p}_u} \psi_{ij}(p_{ui}, p_{uj}) \tag{3.29}$$

where l_{min} is the first and l_{max} is the last interval, i.e., 1 and 3 in our settings.

To optimize the parameters, we use the stochastic gradient ascent procedure that updates the parameters by passing through the set of ratings of each user:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta\nabla log\mathcal{L}(\mathbf{w}) \tag{3.30}$$

where η is the learning rate. More specifically, for each p_{ui} we update the attribute weights $\mathbf{w}_o = \{\mathbf{w}_u, \mathbf{w}_i\}$ and correlation weight w_{ij} for each neighbor $p_{uj} \in \mathbf{p}_u$ using the gradient of the log pseudo-likelihood

$$\frac{\partial log\mathcal{L}}{\partial \mathbf{w}_o} = \mathbf{f}_o(p_{ui}, \mathbf{o}) - \sum_{p_{ui}=l_{min}}^{l_{max}} P(p_{ui} \mid \mathbf{p}_u, \mathbf{o})\mathbf{f}_o(p_{ui}, \mathbf{o}) - \frac{w_i}{\sigma^2} \tag{3.31}$$

$$\frac{\partial log\mathcal{L}}{\partial w_{ij}} = f_{ij}(p_{ui}, p_{uj}) - \sum_{p_{ui}=l_{min}}^{l_{max}} P(p_{ui} \mid \mathbf{p}_u, \mathbf{o})f_{ij}(p_{ui}, p_{uj}) - \frac{w_{ij}}{\sigma^2} \tag{3.32}$$

3.4.3 Item Recommendation

The ultimate goal of *RecSys* is often to rank the items and recommend the Top-N items to the user. To obtain the item rankings, the *PrefCRF* produces distributions over the user-wise preferences, which can be converted into point estimate:

Most Likely Preference The preference can be determined by selecting the preference with the greatest mass in local likelihood:

$$\hat{p}_{ui} = \arg\max_{p_{ui}} P(p_{ui} \mid \mathbf{p}_u, \mathbf{o}) \tag{3.33}$$

where the local likelihood is given by Eq. 3.28. The local likelihood serves as a *confidence* measure.

Smoothed Expectation When the prediction is not restricted to discrete values, the expectation can be used instead:

$$\hat{p}_{ui} = \sum_{p_{ui}=l_{min}}^{l_{max}} p_{ui} P(p_{ui} \mid \mathbf{p}_u, \mathbf{o}) \tag{3.34}$$

where l refers to the intervals of user-wise preferences: from least to most preferred. Note that l is limited to the simplest case of three intervals in our settings, but more intervals are possible.

The predictions by item-item correlation and user-user correlations can be merged by taking the mean value, and then items can be sorted and ranked accordingly. Finally, Alg. 1 summarizes the learning and prediction procedures for the *PrefCRF*.

3.4.4 Computational Complexity

We perform the computational complexity analysis on the *PrefMRF* and its underlying *PrefNMF* algorithms. Given n users and m items each with d_u and d_i preferences, respectively. Let us temporarily ignore the user-specified latent factors. Then the complexity of both *PrefNMF* and *PrefMRF* is $O(nd_u^2)$. However, in practice few items co-rated by the same user are strong neighbors of each other due to the correlation threshold defined in Section 3.4.1. As a result, the computation time of *PrefMRF* tends to be $O(nd_uc)$ where c is a factor of the correlation threshold. It should be noted that Eq. 3.32 in the most inner loop of Alg. 1 is not scaled with d_u as the normalization term Z_{ui} of the local likelihood is computed in the outer loop and reused for inner loop.

4 Experiment and Analysis

To study the performance of the proposed *PrefMRF* and *PrefCRF* models, comparisons were done with the following representative algorithms: *a*) *K-Nearest Neighbors* (KNN) (Resnick et al, 1994), which represents the methods

Algorithm 1 *PrefCRF* Algorithm

```
Input: Explicit or implicit preferences.
Step 1: Infer $PR$ from preferences.
Step 2: Predict user-wise preferences $\hat{p}_{ui}$ using Eq. 2.2.
Step 3: Predict distribution for each $\hat{p}_{ui}$ using Eq. 3.6.
Step 4: Repeat
for each $u \in \mathcal{U}$ do
    for each $p_{ui} \in \mathbf{p}_u$ do
        Compute normalization term $Z_{ui}$ using Eq. 3.29
        Compute local likelihood using Eq. 3.28
        Compute attribute feature $\mathbf{f}_i$ and $\mathbf{f}_u$ using Eq. 3.26
        Compute gradients for attribute features $\mathbf{f}_o$ using Eq. 3.31
        Update $\mathbf{w}_o$ with the gradient using Eq. 3.30
        for each $p_{uj} \in \mathbf{p}_u$, $i \neq j \wedge f_{ij} \in \mathbf{f}_{\text{strong}}$ do
            Get correlation feature $f_{ij}$ and $f_{uv}$ using Eq. 3.24 and Eq. 3.25
            Get gradient for correlation feature $f_{ij}$ using Eq. 3.32
            Update $w_{ij}$ with the gradient using Eq. 3.30
        end for
    end for
end for
Until stopping criteria met
Predictions:
* Predict user-wise preferences using Eq. 3.34 or Eq. 3.33.
* Select Top-N items according to estimated preferences.
```

exploiting the *LS* from absolute ratings; *b*) *Non-negative Matrix Factorization* (NMF) (Koren et al, 2009), which represents the methods exploiting the *GS* from absolute ratings; *c*) *Preference Relation-based KNN* (PrefKNN) (Brun et al, 2010), which exploits the *LS* from *PR*; *d*) *Preference Relation-based NMF* (PrefNMF) (Desarkar et al, 2012), which exploits the *GS* from *PR*.

4.1 Experimental Settings

4.1.1 Datasets

Ideally, the experiments should be conducted on datasets that contain user preferences in two forms: *PR* and *absolute ratings*. Unfortunately no such dataset is publicly available at the moment, therefore we choose to compile the rating-based datasets into the form of *PR*. We use the same conversion method as in (Desarkar et al, 2012) by comparing the ratings of each ordered pair of items co-rated by the same user. For example, 1 is assigned to the *PR* π_{uij} if $p_{ui} > p_{uj}$; 0 is assigned if $p_{ui} < p_{uj}$, and 0.5 is assigned if $p_{ui} = p_{uj}$. Though this experiment converts ratings into PR, it should be noted that the proposed PR-based models are designed to handle implicit preferences such as *page views*, *mouse clicks*, and *time stayed*, which can be converted into PR. In such cases, the pointwise models are not applicable.

Experiments were conducted on two datasets: the *MovieLens*-1M [1] and the *EachMovie* [2] datasets. The *MovieLens*-1M dataset contains more than 1 million ratings by $6,040$ users on $3,900$ movies. The *EachMovie* dataset contains 2.8 million ratings by $72,916$ users on $1,628$ movies. The minimum rating is 1 and we cap the maximum at 5 for both datasets. The impact of side information is studied on the *MovieLens*-1M dataset which provides *gender*, *age*, and *occupation* information about users and *genres* of movies.

For a reliable and fair comparison, each dataset is split into train and test sets, and the following settings are aligned to related work (Weimer et al, 2007). As the sparsity levels differ between the *MovieLens*-1M and the *EachMovie* datasets, different numbers of ratings are reserved for training and the rest for testing. Specifically, for each user in the *MovieLens*-1M we randomly select N = 30, 40, 50, 60 ratings for training, and put the rest for testing. Some users do not have enough ratings thus were excluded from experiments. The *EachMovie* has less items but much more users than *MovieLens*-1M, therefore it is safe to remove some less active users and we set N = 70, 80, 90, 100 to investigate the performance on a dense dataset.

4.1.2 Evaluation Metrics

Traditional recommender systems aim to optimize *RMSE* or *MAE* which emphasizes on absolute ratings. However, the ultimate goal of recommender systems is usually to obtain the ranking of items (Koren and Sill, 2011), where good performance on *RMSE* or *MAE* may not be translated into good ranking results (Koren and Sill, 2011). Therefore, we employ two evaluation metrics: *Normalized Cumulative Discounted Gain@T* (NDCG@T) (Järvelin and Kekäläinen, 2002) which is popular in academia, and *Mean Average Precision@T* (MAP@T) (Chapelle et al, 2009) which is popular in contests [3]. Among them, the NDCG@T metric is defined as

$$\text{NDCG@}T = \frac{1}{K(T)} \sum_{t=1}^{T} \frac{2^{r_t} - 1}{\log_2(t+1)} \tag{4.1}$$

where r_t is the relevance judgment of the item at position t, and $K(T)$ is a normalization constant. The MAP@T metric is defined as

$$\text{MAP@}T = \frac{1}{|\mathcal{U}_{test}|} \sum_{u \in \mathcal{U}_{test}} \sum_{t=1}^{T} \frac{P_u(t)}{min(m_u, t)} \tag{4.2}$$

where m_u is the number of relevant items to user u, and $P_u(t)$ is user u's precision at position t. Both metrics are normalized to $[0, 1]$, and a higher value indicates better performance.

[1] http://grouplens.org/datasets/movielens
[2] http://grouplens.org/datasets/eachmovie
[3] KDD Cup 2012 and Facebook Recruiting Competition

These metrics, together with other ranking-based metrics, require a set of relevant items to be defined in the test set such that the predicted rankings can be evaluated. The relevant items can be defined in different ways. In this paper, we follow the same selection criteria used in the related work (Koren, 2008; Brun et al, 2010) to consider items with the highest ratings as relevant.

4.1.3 Parameter Setting

For a fair comparison, we fix the number of latent factors to 50 for all algorithms, the same as in related work (Cremonesi et al, 2010). The number of neighbors for *KNN* algorithms is set to 50. We vary the minimum correlation threshold to examine the performance with different numbers of features. Different values of the regularization coefficient are also tested.

4.2 Results and Analysis

We first compare the performance of the proposed *PrefMRF* and *PrefCRF* models with four related models: *KNN*, *NMF*, *PrefKNN*, and *PrefNMF*, where the *PrefNMF* is the targeted model, and then investigate the impact of parameter settings.

4.2.1 Comparison on Top-N Recommendation

Comparison of these algorithms is conducted by measuring the *NDCG* and the *MAP* metrics on Top-N recommendation tasks. Each experiment is repeated ten times with different random seeds and we report the mean results with standard deviations on the *MovieLens*-1M dataset in Table 3 and the *EachMovie* dataset in Table 4. Note that only the *MovieLens*-1M dataset has side information which is used by the *PrefCRF* model. The *PrefMRF* as well as other models are based on only preferences data. We also report the *NDCG* and *MAP* values by varying the position T (i.e., how many items to recommend) in Fig. 3 for the *MovieLens*-1M dataset and in Fig. 4 for the *EachMovie*-1M dataset. The following observations can be made based on the results.

Firstly, the *KNN* and the *PrefKNN* methods didn't perform well on *MovieLens*-1M comparing with *Matrix Factorization* based methods. One possible reason is that predictions are made based only on the neighbors, and as a result too much information has been ignored especially when the dataset is large. However, the performance of *KNN*-based methods has improved on the *EachMovie* dataset as we reserved more ratings for training, i.e., better neighbors can be found for prediction.

Secondly, *PrefNMF* outperforms *NMF* on the *MovieLens*-1M dataset which is consistent with the results reported in (Desarkar et al, 2012). However, *PreNMF* does not perform well on *EachMovie* where its performance is only

Table 3 Mean results and standard deviation over ten runs on *MovieLens*-1M dataset.

	Given 30			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.3969 ± 0.0020	0.4081 ± 0.0029	0.2793 ± 0.0021	0.2744 ± 0.0025
NMF	0.5232 ± 0.0057	0.5195 ± 0.0040	0.3866 ± 0.0055	0.3549 ± 0.0037
PrefKNN	0.3910 ± 0.0044	0.4048 ± 0.0038	0.2745 ± 0.0043	0.2720 ± 0.0037
PrefNMF	0.5729 ± 0.0049	0.5680 ± 0.0041	0.4387 ± 0.0046	0.3992 ± 0.0033
PrefMRF	**0.6020 ± 0.0050**	**0.5934 ± 0.0039**	**0.4721 ± 0.0050**	**0.4244 ± 0.0036**
PrefCRF	**0.6316 ± 0.0076**	**0.5966 ± 0.0028**	**0.6254 ± 0.0073**	**0.4245 ± 0.0028**
	Given 40			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.4108 ± 0.0040	0.4252 ± 0.0036	0.2936 ± 0.0036	0.2877 ± 0.0034
NMF	0.5323 ± 0.0050	0.5291 ± 0.0034	0.3976 ± 0.0045	0.3631 ± 0.0035
PrefKNN	0.4122 ± 0.0024	0.4283 ± 0.0024	0.2944 ± 0.0023	0.2904 ± 0.0023
PrefNMF	0.5773 ± 0.0037	0.5732 ± 0.0028	0.4437 ± 0.0041	0.4019 ± 0.0032
PrefMRF	**0.6215 ± 0.0029**	**0.6140 ± 0.0023**	**0.4844 ± 0.0025**	**0.4420 ± 0.0020**
PrefCRF	**0.6435 ± 0.0064**	**0.6092 ± 0.0023**	**0.6420 ± 0.0062**	**0.4392 ± 0.0021**
	Given 50			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.4273 ± 0.0040	0.4424 ± 0.0027	0.3078 ± 0.0038	0.3015 ± 0.0026
NMF	0.5360 ± 0.0041	0.5326 ± 0.0036	0.4010 ± 0.0040	0.3669 ± 0.0025
PrefKNN	0.4326 ± 0.0027	0.4483 ± 0.0030	0.3125 ± 0.0024	0.3070 ± 0.0022
PrefNMF	0.5761 ± 0.0067	0.5745 ± 0.0035	0.4424 ± 0.0064	0.4019 ± 0.0033
PrefMRF	**0.6248 ± 0.0053**	**0.6172 ± 0.0032**	**0.4896 ± 0.0053**	**0.4460 ± 0.0027**
PrefCRF	**0.6648 ± 0.0055**	**0.6158 ± 0.0018**	**0.6580 ± 0.0059**	**0.4471 ± 0.0024**
	Given 60			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.4480 ± 0.0044	0.4622 ± 0.0035	0.3266 ± 0.0036	0.3163 ± 0.0027
NMF	0.5462 ± 0.0068	0.5409 ± 0.0063	0.4109 ± 0.0069	0.3734 ± 0.0055
PrefKNN	0.4526 ± 0.0062	0.4689 ± 0.0039	0.3301 ± 0.0051	0.3223 ± 0.0033
PrefNMF	0.5756 ± 0.0062	0.5733 ± 0.0048	0.4409 ± 0.0059	0.4007 ± 0.0037
PrefMRF	**0.6422 ± 0.0037**	**0.6301 ± 0.0037**	**0.5112 ± 0.0035**	**0.4600 ± 0.0026**
PrefCRF	**0.6772 ± 0.0074**	**0.6242 ± 0.0018**	**0.6715 ± 0.0072**	**0.4536 ± 0.0016**

slightly better than user-based *KNN*. The reason behind could be that *EachMovie* is much denser than the *MovieLens*-1M dataset, which makes the number of *PR* huge and difficult to tune optimal parameters. Besides, we observe that *PrefNMF* in general only achieves a slight improvement with more training data and even drops a bit with *Given* 60. Similarly for the *EachMovie* dataset. With these observations, it appears that for a given number of users, the *PrefNMF* can be trained reasonably well with fewer data.

Finally, the proposed *PrefMRF* and *PrefCRF* have made further improvements over *PrefNMF* on both datasets through capturing both *LS* and *GS*, as well as exploiting side information. From Fig. 3 we can see that the algorithms stabilized around position 10 and *PrefMRF* and *PrefCRF* consistently deliver better performance than others. It should be noted that the performance of *PrefMRF* and *PrefCRF* rely on their underlying model that captures the *GS*. In other words, the performance may vary when the *PrefNMF* is replaced with other alternative methods such as (Liu et al, 2009).

Table 4 Mean results and standard deviation over ten runs on *EachMovie* dataset withouut side information.

	Given 70			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.7088 ± 0.0020	0.7115 ± 0.0015	0.6012 ± 0.0027	0.5767 ± 0.0017
NMF	0.7581 ± 0.0022	0.7577 ± 0.0017	0.6524 ± 0.0026	0.6225 ± 0.0020
PrefKNN	0.7260 ± 0.0022	0.7307 ± 0.0018	0.6197 ± 0.0020	0.5990 ± 0.0016
PrefNMF	0.7408 ± 0.0033	0.7348 ± 0.0039	0.6330 ± 0.0035	0.5800 ± 0.0038
PrefMRF	$\mathbf{0.8317 \pm 0.0032}$	$\mathbf{0.8245 \pm 0.0029}$	$\mathbf{0.7512 \pm 0.0039}$	$\mathbf{0.6921 \pm 0.0034}$
	Given 80			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.7146 ± 0.0018	0.7168 ± 0.0017	0.6070 ± 0.0021	0.5825 ± 0.0019
NMF	0.7636 ± 0.0021	0.7638 ± 0.0018	0.6583 ± 0.0025	0.6286 ± 0.0018
PrefKNN	0.7337 ± 0.0028	0.7377 ± 0.0018	0.6271 ± 0.0029	0.6057 ± 0.0021
PrefNMF	0.7422 ± 0.0036	0.7319 ± 0.0040	0.6329 ± 0.0039	0.5774 ± 0.0033
PrefMRF	$\mathbf{0.8364 \pm 0.0036}$	$\mathbf{0.8232 \pm 0.0030}$	$\mathbf{0.7553 \pm 0.0038}$	$\mathbf{0.6991 \pm 0.0032}$
	Given 90			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.7191 ± 0.0022	0.7279 ± 0.0028	0.6120 ± 0.0021	0.5933 ± 0.0013
NMF	0.7712 ± 0.0039	0.7692 ± 0.0033	0.6663 ± 0.0043	0.6431 ± 0.0034
PrefKNN	0.7418 ± 0.0028	0.7421 ± 0.0015	0.6357 ± 0.0030	0.6192 ± 0.0020
PrefNMF	0.7456 ± 0.0031	0.7358 ± 0.0038	0.6357 ± 0.0040	0.5819 ± 0.0036
PrefMRF	$\mathbf{0.8394 \pm 0.0035}$	$\mathbf{0.8249 \pm 0.0032}$	$\mathbf{0.7474 \pm 0.0037}$	$\mathbf{0.7046 \pm 0.0032}$
	Given 100			
Algorithm	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.7279 ± 0.0028	0.7277 ± 0.0015	0.6238 ± 0.0032	0.5973 ± 0.0021
NMF	0.7741 ± 0.0030	0.7717 ± 0.0028	0.6719 ± 0.0034	0.6411 ± 0.0030
PrefKNN	0.7505 ± 0.0019	0.7511 ± 0.0012	0.6478 ± 0.0020	0.6231 ± 0.0014
PrefNMF	0.7391 ± 0.0033	0.7298 ± 0.0034	0.6318 ± 0.0039	0.5761 ± 0.0039
PrefMRF	$\mathbf{0.8418 \pm 0.0031}$	$\mathbf{0.8277 \pm 0.0030}$	$\mathbf{0.7546 \pm 0.0038}$	$\mathbf{0.7063 \pm 0.0036}$

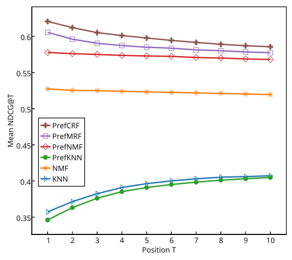
Table 5 Paired *t*-test for *PrefMRF* and *PrefNMF*.

Settings			*t*-test statistics		
Dataset	Sparsity	Metric	df	t	*p*-value
MovieLens	*Given* 60	*NDCG*@10	9	16.6218	< 0.00001
MovieLens	*Given* 60	*MAP*@10	9	23.5517	< 0.00001
EachMovie	*Given* 100	*NDCG*@10	9	72.4189	< 0.00001
EachMovie	*Given* 100	*MAP*@10	9	72.1346	< 0.00001

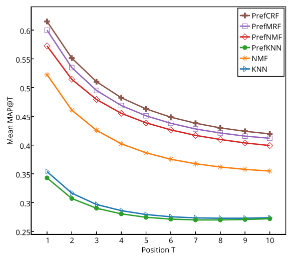
To confirm the improvements, a paired *t*-test (two-tailed) with a significance level of 95% has been applied to the best *PrefMRF* and the second best *PrefNMF*. Results shown in Table 5 confirm that the performance of models with and without capturing the *LS* is statistically significant.

4.2.2 Performance on Various Data Sparsity Levels

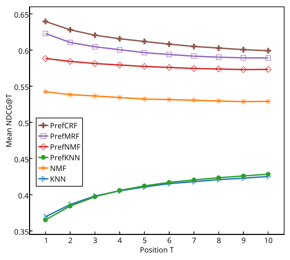
To thoroughly examine the performance of these algorithms, we compare their performance under different settings of density levels by varying the training set sizes: from *Given* 30 to *Given* 60 on *MovieLens*-1M dataset, and from *Given*



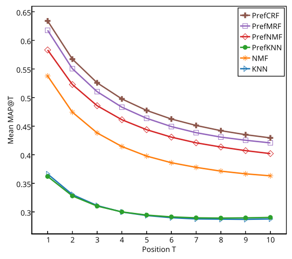
(a) NDCG@T (Given 30)



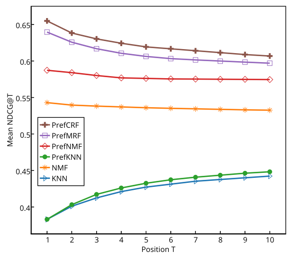
(b) MAP@T (Given 30)



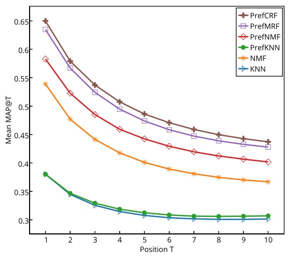
(c) NDCG@T (Given 40)



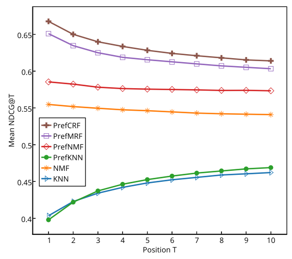
(d) MAP@T (Given 40)



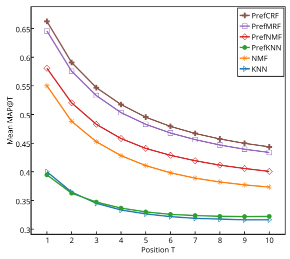
(e) NDCG@T (Given 50)



(f) MAP@T (Given 50)



(g) NDCG@T (Given 60)



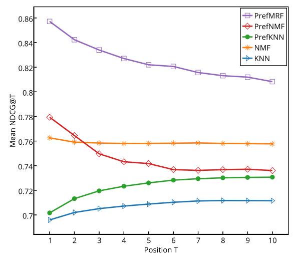
(h) MAP@T (Given 60)

Fig. 3 Performance of different position T on MovieLens-1M dataset.

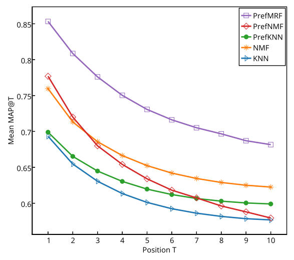
70 to *Given* 100 on *EachMovie* dataset. Results are plotted in Fig. 5. It can be observed that in general more training data results in better performance, and this is particularly true for *LS*-based models. However, *PrefNMF* does not gain much benefit from more data and even performs slightly worse in *Given* 60. The *PrefMRF* on the other hand consistently gains performance from more data as the *LS* information can be better captured.

4.2.3 Impact of Minimum Correlation Threshold

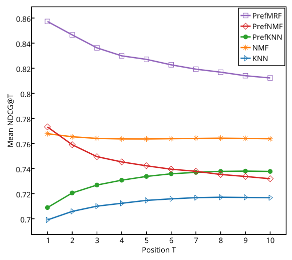
As described in Section 3.4.1, a *minimum correlation threshold* is required to control the number of features in the *PrefMRF* model. By default, each pair of co-rated items has a feature which results in a large number of features. However, many of these features are useless if the item-item correlations are



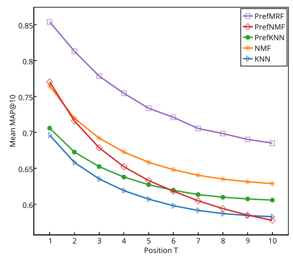
(a) NDCG@T (Given 70)



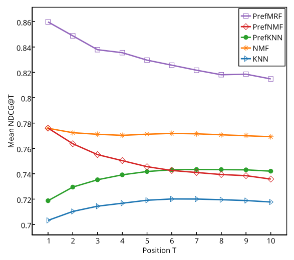
(b) MAP@T (Given 70)



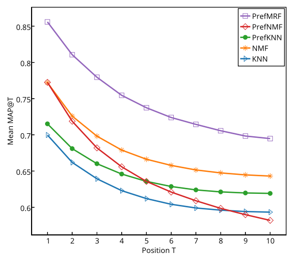
(c) NDCG@T (Given 80)



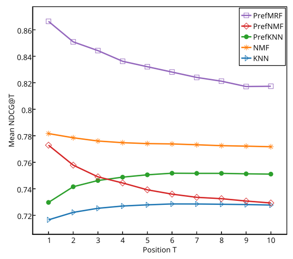
(d) MAP@T (Given 80)



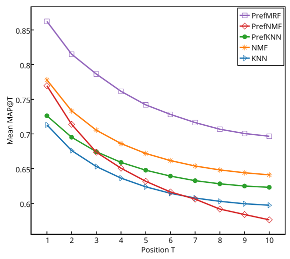
(e) NDCG@T (Given 90)



(f) MAP@T (Given 90)



(g) NDCG@T (Given 100)



(h) MAP@T (Given 100)

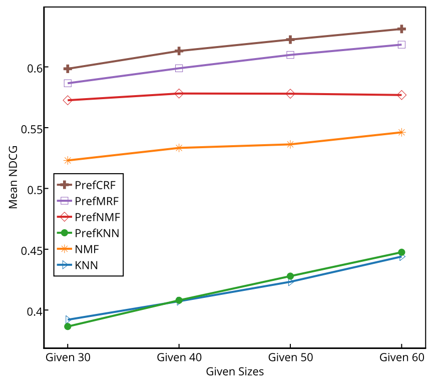
Fig. 4 Performance of different position T on EachMovie dataset.

weak. To make the model more robust and with faster convergence, a *minimum correlation threshold* is applied to remove weak features. Specifically, the feature is removed if two items has a correlation measured by *Pearson* correlation less than the threshold. Results are plotted in Fig. 6(a).

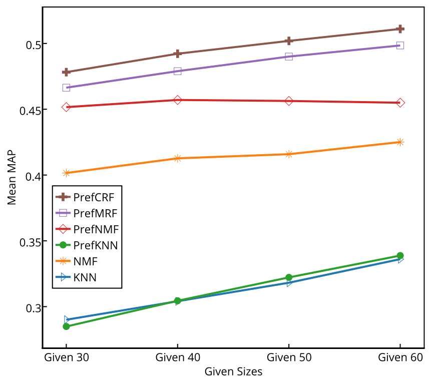
It can be observed that a smaller correlation threshold delivers better performance, however, the number of features will also increase. To balance the performance and computation time, it is wise to select a moderate level of threshold depending on the dataset.

4.2.4 Impact of Regularization Coefficient

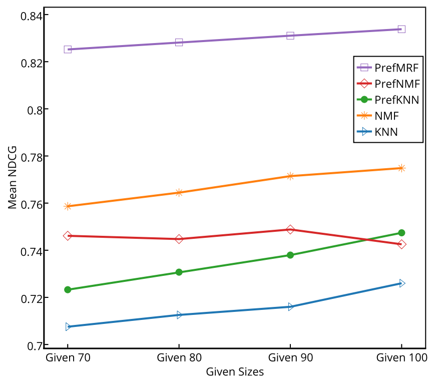
As the number of features in *PrefMRF* can be large, the model might be prone to over-fitting. Therefore, we investigate the impact of regularization settings as plotted in Fig. 6(b).



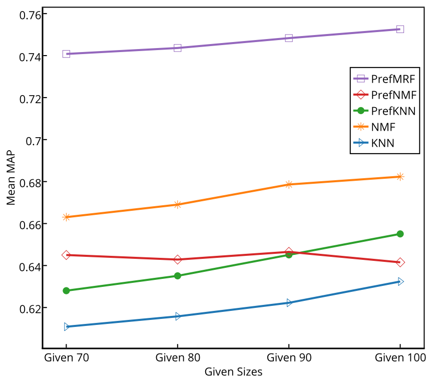
(a) Mean NDCG by training set sizes on MovieLens-1M dataset.



(b) Mean MAP by training set sizes on MovieLens-1M dataset.



(c) Mean NDCG by training set sizes on EachMovie dataset.



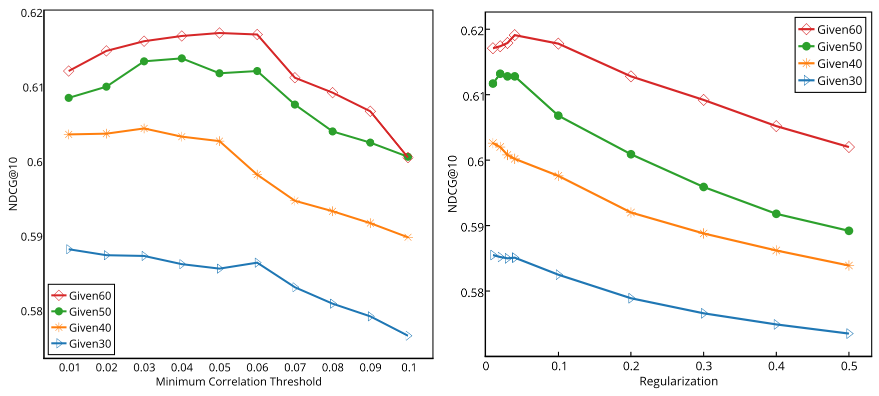
(d) Mean MAP by training set sizes on EachMovie dataset.

Fig. 5 Impact of Sparsity Levels.

We observe that the performance is better when a small regularization penalty applies. In other words, the *PrefMRF* can generalize reasonably well without too much regularization. This can be explained as the weights of item-item correlations are not user-specific but shared by all users, thus they cannot over-fit every user perfectly.

5 Conclusions and Future Work

In this paper we presented the *PrefMRF* model, which takes advantage of both the representational power of the *MRF* and the ease of modeling preference relations by the *PrefNMF*. To the best of our knowledge, there was no *PR*-based method that can capture both *LS* and *GS*, until the *PrefMRF* model was proposed in this work. In addition, side information can be easily



(a) NDCG@10 by Threshold (b) NDCG@10 by Regularization

Fig. 6 Impact of Parameters (MovieLens-1M)

incorporated by extending the *PrefMRF* model to the *PrefCRF* model. Experimental results on public datasets demonstrate that both types of interactions have been properly captured by *PrefMRF*, and significantly improved Top-N recommendation performance has been achieved. In addition, the *PrefMRF* model provides a generic interface for unifying various models other than the *PrefNMF* used in this paper. In other words, a model that captures the *GS* should be able to take advantage of *PrefMRF* to capture *LS* as well by substituting their predictions into the formulas in Section 3.1.2.

For future work, we would like to work on two directions. First, the computation efficiency of *PR*-based matrix factorization needs to be improved given that the number of preference relations is usually much larger than absolute ratings. This is feasible as each user has his/her own set of preference relations, thus the learning process can be parallelized. Secondly, it would be interesting to see how *PR*-based methods perform on real implicit preferences dataset, such as *page views* and *mouse clicks*.

Acknowledgements This work was partially supported by the National Science Foundation of China (61673201), the Collaborative Innovation Center of Novel Software Technology and Industrialization, and Guangxi Key Laboratory of Trusted Software (No. KX201528).

References

Balabanović M, Shoham Y (1997) Fab: content-based, collaborative recommendation. Commun ACM 40(3):66–72

Basilico J, Hofmann T (2004) Unifying collaborative and content-based filtering. In: ICML'04, ACM, pp 65–72

Besag J (1974) Spatial interaction and the statistical analysis of lattice systems. Journal of the Royal Statistical Society, Series B 36(2):192–236

Brun A, Hamad A, Buffet O, Boyer A (2010) Towards preference relations in recommender systems. In: Preference Learning (PL 2010) ECML/PKDD

Chapelle O, Metlzer D, Zhang Y, Grinspan P (2009) Expected reciprocal rank for graded relevance. In: CIKM'09, ACM, pp 621–630

Cohen WW, Schapire RE, Singer Y (1999) Learning to order things. J Artif Int Res 10(1):243–270

Cremonesi P, Koren Y, Turrin R (2010) Performance of recommender algorithms on top-n recommendation tasks. In: RecSys'10, ACM, pp 39–46

Defazio A, Caetano T (2012) A graphical model formulation of collaborative filtering neighbourhood methods with fast maximum entropy training. In: ICML'12, pp 265–272

Desarkar MS, Sarkar S, Mitra P (2010) Aggregating preference graphs for collaborative rating prediction. In: RecSys'10, ACM, pp 21–28

Desarkar MS, Saxena R, Sarkar S (2012) Preference relation based matrix factorization for recommender systems. In: UMAP'12, Springer, pp 63–75

Freund Y, Iyer R, Schapire RE, Singer Y (2003) An efficient boosting algorithm for combining preferences. Journal of machine learning research 4:933–969

Fürnkranz J, Hüllermeier E (2003) Pairwise preference learning and ranking. In: Machine Learning: ECML'03, Springer, pp 145–156

Fürnkranz J, Hüllermeier E (2010) Preference learning. Springer

Gantner Z, Drumond L, Freudenthaler C, Rendle S, Schmidt-Thieme L (2010) Learning attribute-to-feature mappings for cold-start recommendations. In: Data Mining (ICDM), 2010 IEEE 10th International Conference on, IEEE, pp 176–185

Hinton G (2002) Training products of experts by minimizing contrastive divergence. Neural Computation 14(8):1771–1800

Järvelin K, Kekäläinen J (2002) Cumulated gain-based evaluation of ir techniques. ACM TOIS 20(4):422–446

Jin R, Si L, Zhai C (2002) Preference-based graphic models for collaborative filtering. In: UAI'02, Morgan Kaufmann Publishers Inc., pp 329–336

Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD'08, ACM, pp 426–434

Koren Y (2010) Collaborative filtering with temporal dynamics. Commun ACM 53(4):89–97

Koren Y, Sill J (2011) Ordrec: an ordinal model for predicting personalized item rating distributions. In: RecSys'11, ACM, pp 117–124

Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. IEEE Computer 42(8):30–37

Liu NN, Zhao M, Yang Q (2009) Probabilistic latent preference analysis for collaborative filtering. In: CIKM'09, ACM, pp 759–766

Liu S, Tran T, Li G, Jiang Y (2014) Ordinal random fields for recommender systems. In: ACML'14, JMLR: workshop and conference proceedings, pp 283–298

Ma H, Zhou D, Liu C, Lyu MR, King I (2011) Recommender systems with social regularization. In: WSDM'11, ACM, pp 287–296

McCullagh P (1980) Regression models for ordinal data. Journal of the Royal Statistical Society, Series B 42(2):109–142

McFadden D (1980) Econometric models for probabilistic choice among products. Journal of Business 53(3):S13–S29

McNee SM, Riedl J, Konstan JA (2006) Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: CHI EA'06, ACM, pp 1097–1101

Mohri M, Rostamizadeh A, Talwalkar A (2012) Foundations of machine learning. MIT press

Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, AUAI Press, pp 452–461

Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) An open architecture for collaborative filtering of netnews. In: CSCW'94, ACM, pp 175–186

Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: WWW'10, ACM, pp 285–295

Schacter DL, Dodson CS (2001) Misattribution, false recognition and the sins of memory. Philosophical Transactions of the Royal Society B: Biological Sciences 356(1413):1385–1393

Schein AI, Popescul A, Ungar LH, Pennock DM (2002) Methods and metrics for cold-start recommendations. In: SIGIR'02, ACM, pp 253–260

Shi Y, Larson M, Hanjalic A (2010) List-wise learning to rank with matrix factorization for collaborative filtering. In: RecSys'10, ACM, pp 269–272

Tran T, Phung DQ, Venkatesh S (2007) Preference networks: Probabilistic models for recommendation systems. In: AusDM'07, ACS, pp 195–202

Tran T, Phung DQ, Venkatesh S (2009) Ordinal boltzmann machines for collaborative filtering. In: UAI'09, AUAI Press, pp 548–556

Tran T, Phung DQ, Venkatesh S (2012) Learning from ordered sets and applications in collaborative ranking. In: ACML'12, JMLR: workshop and conference proceedings, pp 427–442

Weimer M, Karatzoglou A, Le QV, Smola A (2007) Maximum margin matrix factorization for collaborative ranking. In: NIPS'07, pp 1593–1600