Social Temporal Collaborative Ranking for Context Aware Movie Recommendation

NATHAN N. LIU and LUHENG HE, Hong Kong University of Science and Technology
MIN ZHAO, NEC Labs, China

Most existing collaborative filtering models only consider the use of user feedback (e.g., ratings) and meta data (e.g., content, demographics). However, in most real world recommender systems, context information, such as time and social networks, are also very important factors that could be considered in order to produce more accurate recommendations. In this work, we address several challenges for the context aware movie recommendation tasks in CAMRa 2010: (1) how to combine multiple heterogeneous forms of user feedback? (2) how to cope with dynamic user and item characteristics? (3) how to capture and utilize social connections among users? For the first challenge, we propose a novel ranking based matrix factorization model to aggregate explicit and implicit user feedback. For the second challenge, we extend this model to a sequential matrix factorization model to enable time-aware parametrization. Finally, we introduce a network regularization function to constrain user parameters based on social connections. To the best of our knowledge, this is the first study that investigates the collective modeling of social and temporal dynamics. Experiments on the CAMRa 2010 dataset demonstrated clear improvements over many baselines.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Collaborative filtering, recommender systems, user feedback, context awareness

ACM Reference Format:
Liu, N. N., He, L., and Zhao, M. 2013. Social temporal collaborative ranking for context aware movie recommendation. ACM Trans. Intell. Syst. Technol. 4, 1, Article 15 (January 2013), 26 pages.
DOI = 10.1145/2414425.2414440 http://doi.acm.org/10.1145/2414425.2414440

1. INTRODUCTION

The popularization of the World Wide Web has led to an explosive growth of information. Users increasingly rely on effective information retrieval and filtering technologies to cope with information overload. Search systems perform best when the user has explicit and well articulated information needs. In contrast, recommender systems could discover user's unspecified information needs through mining her past behaviors, and automatically suggesting relevant information to users with no explicit inputs required. In recent years, recommender systems have become a critical component in a variety of online services including e-commerce (e.g., Amazon) and online entertainment (e.g., Netflix, Last.FM). Broadly speaking, existing technologies used for recommender systems fall into two categories: *content-based filtering* versus

Author's addresses: N. Liu and L. He, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong; email: nliu@cse.ust.hk; M. Zhao, NEC Labs China, 11th floor, Building A, Innovation Plaza, Tsinghua Science Park, 1 Zhongguancun East Road, Beijing, China.

DOI 10.1145/2414425.2414440 http://doi.acm.org/10.1145/2414425.2414440

collaborative filtering. Content-based filtering analyzes the features associated with the items and users, such as product descriptions and user profiles, in order to match users with items. On the other hand, collaborative filtering (CF) does not require any user or item features. It tries to find different users with correlated preferences based on their past behavior so that one user's unobserved interests could be predicted from the observed interests of other similar minded users.

Most existing collaborative filtering algorithms only work with relatively simple forms of input data, such as a dataset of users' ratings on items, and the recommendations are often generated without considering contextual factors, such as time, location, community, and others. However, in real-world applications, user behaviors are much more complicated. For example, a typical system could not only track users' ratings but also his browsing, search, and click behaviors. Moreover, context information also plays a critical role. Temporal information is very important for recommending highly dynamic contents, such as news articles, and with the popularization of social networking, the effect of social influence among users is also very strong.

The CAMRa 2010 dataset[1] contains significantly richer information than most existing collaborative filtering benchmark datasets, such as Movielens[2] and Netflix.[3] In particular, it enables the study of the following new research problems.

—Unlike traditional datasets, which only provide one type of user feedback (i.e., ratings), the CAMRa 2010 dataset contains multiple types of user feedback (i.e., rating, comment, collection, favorite). While explicit user feedback, such as ratings, could reflect how much a user likes an item based on numerical values, many other types of user actions, such as "collection" and "comment," can be represented only as binary events. How to design models to combine such heterogeneous user feedback is the first problem we study in this work.
—Each rating provided in the CAMRa 2010 dataset is associated with a time stamp and the entire dataset spans a time period of over two years. Moreover, unlike most existing evaluation schemes, such as the Netflix prize competition, which focus on making predictions over a very long period of time (i.e., a year), CAMRa requires movie recommendations for very specific occasions (i.e., Oscar and Christmas) that only cover a very short time period (i.e., a week). How to design time aware recommendation models capable of capturing transcient user and item characteristics is the second problem we study in this work.
—In addition to user feedback and temporal information, CAMRa also provides a friendship-based social network of users. Social networks allow information to propagate along the links and naturally make users influence each other. In most social networks, users are more likely to connect with others with similar characteristics. This effect, often known as *homophily*, has been demonstrated across many different types of social networks. In the context of collaborative filtering, the network structure provides an additional source of information to help us identify similarities among users. Measuring user similarities is highly difficult given very limited amount of user feedback (i.e., user cold start). Therefore, the network-based similarity could effectively complement user feedback under the cold start conditions. How to incorporate this form of social context into collaborative filtering models is the third challenge we study in this work.

To address the first challenge, we develop a novel *collaborative ranking* framework that could effectively aggregate multiple types of explicit and implicit user feedback.

[1]http://www.dai-labor.de/camra2010/datasets/.
[2]http://www.movieslens.org.
[3]http://www.netflixprize.com.

The major difficulty in aggregating explicit and implicit feedback for collaborative filtering lies in coping with the heterogeneous representations of the two forms of user feedback. Explicit feedback is represented as a numeric score within a certain range, such as 1 to 5, whereas implicit feedback is in the form of a binary response indicating whether a user has performed a particular action on an item. In terms of data quality, it is clear that explicit feedback can more precisely capture user preference since its numerical value reflects how much the user likes an item. On the other hand, implicit feedback is more ambiguous and less granular due to its binary representation. But in terms of data availability, implicit feedback is much easier to obtain in real systems and therefore much more abundant. It can be seen that the two types of data have their respective strengths and weaknesses, and naturally complement each other. In this article, we propose to combine heterogeneous user feedback by transforming both explicit and implicit feedback into a unified pairwise preference-based representation, which can then be used as training data for matrix factorization models.

The second part of this article is focused on the study of *time-aware recommendation*. In real-world recommender systems, user feedback is accumulated over time. Each time the user uses the system, some new feedback data could be collected for him. Most existing CF algorithms often ignore the dimension of time and assume that the user and item characteristics are static, whereas in reality temporal dynamics are clearly present in most systems. Item popularity is often subject to seasonality and occasions. User's taste is also likely to change over time. The weekly recommendation track of the CAMRa 2010 challenge aims at evaluating a recommendation algorithm's ability to cope with such temporal dynamics. In particular, it requires making recommendations for two specific weeks corresponding to two occasions, Oscar and Christmas. It is expected that users and items would exhibit certain transcient behaviors in this temporal context. For example, during the Oscar week, movies with award nominations could become more popular than usual, whereas during the Christmas week, movies more suitable for the holiday mood, such as comedy and cartoon, may become more popular than usual. To capture such temporal dynamics that may commonly arise in real world recommender systems, we extend the collaborative ranking model with a time aware parametrization in order to capture nonstationary behaviors. Moreover, we also introduce a novel temporal smoothness regularization term to avoid overfitting due to the increased number of free parameters.

Finally, we further extend the model in order to support *social network aware recommendation* as well. In recent years, social-networking-based services have enjoyed phenomenal success. Many recommender systems include social networking features to enable users with similar interests to connect and interact with each other. Most existing collaborative filtering algorithms are purely based on user feedback, and would therefore fall short under user cold start conditions. How to effectively fuse the knowledge about network structure with the user feedback data is a key challenge in achieving social network awareness. In this article, we design a *network regularization* technique motivated by the homophily effect, which introduces correlation between the parameters of connected users into the matrix factorization model.

The proposed model is validated via extensive experiments on the CAMRa 2010 dataset and it is found that the proposed model significantly outperforms several state of the art collaborative filtering models. We also present detailed discussion about various algorithm design issues and identify several important factors affecting the effectiveness of time and social network awareness including the age, activeness and connectivity of users.

The article is organized as follows. We first give an overview of related existing work on collaborative filtering and context aware recommendation. We then describe the basic framework of the proposed social temporal collaborative ranking approach. In the

following sections, we describe our detailed solution to each of the three key challenges: (a) combining heterogeneous feedback, (b) time aware modeling, and (c) social network aware modeling. Finally, we present experimental results on the CAMRa 2010 dataset and discuss some important findings.

2. RELATED WORK

In this section, we review several directions of existing works on recommender systems: (1) traditional collaborative filtering algorithms using either explicit or implicit user feedback. (2) recommendation algorithms utilizing social network information. (3) recommendation algorithms considering temporal information.

2.1. Collaborative Filtering-Based Recommendation

Collaborative filtering is one of the most widely used and successfully deployed technologies for personalized recommendation. It relies on uncovering the correlated preferences across different users and/or items, so that a user's unobserved preference for an item could be interpolated from the observed preferences associated with other similar users and/or similar items. Generally speaking, there are two common approaches to collaborative filtering: the memory-based approach and the model-based approach.

One particular form of memory-based methods is the user-based model, which estimates the unknown ratings of a target user based on the ratings by a set of neighboring users who tend to rate similarly to the target user. A crucial component of the user-based model is the user-user similarity for determining the set of neighbors. Popular similarity measures include the Pearson Correlation Coefficient (PCC) [Resnick et al. 1994; Herlocker et al. 2002] and the vector similarity (VS) [Breese et al. 1998]. An alternative form of the neighborhood-based approach is the item-based model [Sarwar et al. 2001; Linden et al. 2003]. Here, the item-item similarity is used to select a set of neighboring items that have been rated by the target user and the ratings on the unrated items are predicted based on his ratings on the neighboring items. While memory-based methods are simple and intuitive, they have several drawbacks. Firstly, without any abstraction of the raw data, memory-based methods would suffer from data sparseness. Secondly, memory-based methods often require direct access to the raw data while making predictions, which is computationally expensive. Finally, without an objective function to guide model training, memory-based methods could not be adapted to optimize specific performance metrics associated with particular tasks or application domains.

The model-based approach to CF uses the observed user-item ratings to train a compact model that explains the given data so that ratings could be predicted via the model instead of directly searching in the rating database as the memory-based approach does. Model-based approach would typically involve inducing some latent user and item parameters from the observed user feedback, which could be used to predict user preferences. Commonly used models include matrix factorization [Koren et al. 2009], probabilistic latent semantic analysis [Hofmann 2004] and Bayesian networks [Pennock et al. 2000].

Traditional memory or model-based CF methods focus on predicting the absolute ratings a user would assign to an item therefore treating CF as a regression problem. Several recent works [Liu and Yang 2008; Weimer et al. 2008; Liu et al. 2009] have suggested that it would be more appropriate to treat CF as a ranking problem to focus on modeling the relative ordering of items rather than the absolute ratings since regression errors can be misleading. To focus on ensuring correct item ordering, the memory-based method EigenRank [Liu and Yang 2008] transforms rating data into pairwise preferences, which are then used to measure user similarities based on rank consistency and to order items based on rank aggregations. There are also several

recent model-based methods [Liu et al. 2009; Weimer et al. 2008; Rendle et al. 2009] that focus on ranking items correctly. These include the adaptation of matrix factorization to optimize ranking performance metric, such as NDCG [Weimer et al. 2008], and probabilistic mixture models of pairwise preferences [Liu et al. 2009]. While the previous two methods are specifically designed for handling explicit feedback in the form of numeric ratings, [Rendle et al. 2009] suggested a way to derive pairwise preferences from implicit feedback, which is used to design preference-based loss functions for training matrix factorization model.

2.2. Social Network Aware Recommendation

Online social networking services have enjoyed enormous growth in recent years. As a result, social network-based recommendation is becoming an increasingly popular topic in the research community [Liu et al. 2010; Ma et al. 2009, 2011; Jamali and Ester 2009, 2010]. While traditional collaborative filtering algorithms rely on the observed rating behavior to uncover the implicit correlations between the preferences of different users, social networks allow users to explicitly identify other users who either are trustworthy or share similar tastes. This additional knowledge about explicit connections between users can be very useful when the correlations between users could not be reliably inferred from limited amount of ratings data.

TrustWalker [Jamali and Ester 2009] uses a random walk model to combine rating behaviors and trust relations. Under this model, the probability of using the rating on an item by another user is determined by how likely the target user would visit this rating when randomly following user-item edges defined by ratings and user-user edges defined by trust relations. There are also several variations of the matrix factorization model for incorporating social network information. The social trust ensemble (STE) model [Ma et al. 2009] uses a linear combination of a user's own latent factors and his neighbors' latent factors when making predictions. More recently, several works have independently proposed social-network-based regularization methods that constrain the latent factors of connected users to be close to each other. In particular, Jamali and Ester [2010] adopt a regularization function that penalizes the difference between a user's latent factors and the average of the latent factors of his neighbors. The social-network-based regularization function suggested in Liu et al. [2010] penalizes the sum of differences between a user's latent factors with each individual neighbor. Ma et al. [2011] compare both forms of regularization and finds that both work more effectively than the earlier STE model.

2.3. Time Aware Recommendation

Temporal dynamics is of great importance in most real world recommender systems where item popularity and user preferences could be highly nonstationary. Thanks to the availability of large scale datasets, such as Netflix, which contain data collected over a substantial period of time, there has been growing interest in modeling temporal dynamics in recommender systems in recent years. [Koren 2009] extends standard matrix factorization models by introducing time dependent user/item biases and user latent factors to cope with changing user and item characteristics. [Xiong et al. 2010] and [Karatzoglou et al. 2010] model time stamped user-item ratings as a three dimensional tensor so that a user's rating on an item within a particular time step depends not only on the user and item factors but also the latent factors associated with the time step. To cope with temporal dynamics in implicit user feedback data, [Xiang et al. 2010] proposes a random walk model where the user-item edges are weighted according to recency so the user's more recent preferences would be more strongly emphasized during the random walk.

2.4. Our Contributions

In relation to these three directions of existing research, the technical contributions of the current work are twofold: Firstly, explicit and implicit feedback have mostly been studied separately in the past, our model is the first to aggregate multiple heterogeneous forms of user feedback using a ranking-based model. Secondly, we propose the first model to integrate temporal and social context within a unified matrix factorization framework and examine the interactions of the two types of contexts.

3. OVERVIEW OF THE PROBLEM

The core task of a recommendation algorithm is to predict which items a user may like or dislike based on his observed feedback. In most applications, the end goal is to rank items by their predicted relevance to a target user in order to generate a recommendation list. User feedback is often expressed either explicitly by providing a numerical rating, or implicitly by performing a particular action (e.g., click, favorite) on an item. One interesting aspect of the CAMRa challenge is that there are multiple types of explicit as well as implicit user feedback. In addition to the commonly used ratings data, there are also several types of implicit user feedback, such as which user specified which movies as favorite, commented on which movies and collected which movies into his lists. How to effectively combine these heterogeneous forms of user feedback represents a novel challenge that has not been well studied before. In addition to providing heterogeneous user feedback, another unique aspect of the CAMRa challenge is that it provides both temporal and social network context, enabling us to design and study algorithms that simultaneously exploit both types of information. While the use of temporal and social network contexts in recommender systems have been studied separately in the past, there are few existing models that utilize both types of context. In this section, we present an overview of the notations and the matrix factorization framework that we rely on to tackle the various challenges posed by CAMRa 2011.

3.1. Notations

We use $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ to denote the set of m users and $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ to denote the set of n items in the system. Both explicit and implicit feedback are observable interactions between $\mathcal{U}$ and $\mathcal{I}$ that could reflect a user's preferences on items.

Explicit feedback is usually represented in the form of numeric ratings assigned to items by users, the magnitude of which indicates the degree to which a user likes a particular item. We use $x_{ui} \in \mathbb{R}$ to denote user u's rating on item i and let $S^* \subseteq \mathcal{U} \times \mathcal{I}$ denote the set of user-item pairs for which explicit feedback are available. We use $X \in \mathbb{R}^{m \times n}$ to denote the matrix of observed ratings. For convenience, we use the $\mathcal{I}_{u*}$ to refer to the set of items i on which user u's explicit feedback is observed (i.e., $(u, i) \in S^*$). Similarly, $\mathcal{U}_{i*}$ denotes the set of users, whose explicit feedback on item i is observed.

For implicit feedback, x_{ui}'s are usually binary values and are assumed to be known for all entries in the matrix X. For example, $x_{ui} = +1$ may indicate a user has watched the movie while $x_{ui} = -1$ may indicate he has not watched the movie. We use $\mathcal{S}_+ \subseteq \mathcal{U} \times \mathcal{I}$ and $\mathcal{S}_- \subseteq \mathcal{U} \times \mathcal{I}$ to denote the set of indices of the positive and negative entries in the matrix X respectively. We also use $\mathcal{I}_{u+}$ and $\mathcal{I}_{u-}$ to denote the sets of items for which $x_{ui} = +1$ and -1 respectively. Symmetrically, we define user sets $\mathcal{U}_{i+}$ and $\mathcal{U}_{i-}$.

3.2. Preliminaries

Suppose there are Z types of user feedback, each of which is represented as a particular rating matrix $X^z \in \mathbb{R}^{m \times n}$. Let $\widehat{x}_{ui}$ denote the predicted rating on item i by user u and $\widehat{X}$ denote the matrix containing all the predicted ratings. We want to design a model to predict $\widehat{x}_{ui}$ using a function with parameters Θ, which are often learnt through solving

an optimization problem of the following general form:

$$\min_{\Theta} \sum_{z} \mathcal{L}(X^z, \widehat{X}) + \mathcal{R}(\Theta). \tag{1}$$

Here $\mathcal{L}(X^z, \widehat{X})$ is a loss function measuring the discrepancy between the observed user feedback in the rating matrix X^z and the predicted ratings in $\widehat{X}$. The regularization function $\mathcal{R}(\Theta)$ penalizes overly complicated model in order to suppress overfitting.

In recent years, matrix factorization [Koren et al. 2009] models have gained popularity due to their widely publicized success in the Netflix Challenge.[4] In addition to modeling ratings data, their effectiveness has also been demonstrated on implicit feedback datasets [Pan and Scholz 2009; Hu et al. 2008]. In most existing matrix factorization models, Θ consists of two matrices $W \in \mathbb{R}^{k \times m}$ and $H \in \mathbb{R}^{k \times n}$, which correspond to the user and the item factor matrix, respectively. Matrix factorization models produce the predicted rating matrix $\widehat{X}$ as the product of the two factors (i.e., $\widehat{X} = W^T H$), which leads to the following parametric form of $\widehat{x}_{ui}$:

$$\widehat{x}_{ui} = \mathbf{w}_u^T \mathbf{h}_i, \tag{2}$$

where $\mathbf{w}_u$ and $\mathbf{h}_i$ denote the u-th and i-th column in W and H respectively.

For explicit feedback datasets, the goal is often to make the predicted values $\widehat{x}_{ui}$ as close to the observed values x_{ui} as possible. This is often achieved by adopting the commonly used squared error loss form of the loss function:

$$\mathcal{L}(X, \widehat{X}) = \sum_{(u,i) \in S^*} (x_{ui} - \widehat{x}_{ui})^2. \tag{3}$$

Note that the squared loss is only computed over the nonmissing entries in S^*. Other forms of loss functions, such as hinge loss, that are more suitable for the ordinal ratings have also been considered [Rennie and Srebro 2005]. More recently, Liu et al. [2009] and Weimer et al. [2008] also proposed a new class of loss functions that tries to ensure the consistency of the ordering of items based on X and $\widehat{X}$ for each user instead of enforcing that the corresponding entries in X and $\widehat{X}$ are close to each other.

For the regularization function $\mathcal{R}(\Theta)$, a commonly used regularization term is the Frobenius norm of the parameter matrices, which is adopted by collaborative filtering algorithms [Koren et al. 2009; Rennie and Srebro 2005] due to its smooth differentiable property. In case of ordinary matrix factorization models where $\Theta = \{W, H\}$, this is defined as:

$$\mathcal{R}(\Theta) = C \cdot \left(\|W\|_F^2 + \|H\|_F^2 \right) = C \cdot \left(\sum_{i=1}^{m} \sum_{f=1}^{k} w_{if}^2 + \sum_{u=1}^{n} \sum_{f=1}^{k} h_{uf}^2 \right), \tag{4}$$

where the parameter $C \geq 0$ controls the strength of regularization, which allows us to balance between training error and model complexity.

In the following sections, we will show how to extend this very general framework to model heterogeneous user feedback as well as to incorporate time and social network awareness when designing appropriate parametrization of the predictor $\widehat{x}_{ui}$, loss function $\mathcal{L}(\cdot)$ and regularization function $\mathcal{R}(\cdot)$.

4. COMBINING HETEROGENEOUS USER FEEDBACK VIA COLLABORATIVE RANKING

The vast majority of existing collaborative filtering models have solely considered the use of explicit feedback. Only in the recent years have researchers began to develop

[4] http://www.netflixprize.com/.

algorithms specifically designed for implicit feedback. However, to the best of our knowledge, very few models can utilize explicit and implicit feedback simultaneously. In this section, we develop the *collaborative ranking* model that is based on deriving pairwise preferences from both explicit and implicit feedback.

4.1. Explicit versus Implicit Feedback

Unlike explicit feedback, implicit feedback is more ambiguous and less granular in indicating how much a user likes a particular item. For example, a user may not really like a song or a news story he just accidently played or read. Similarly, it would be possible for him to actually like a song or a news story he has not played or read because he is not aware of such items. Without further information, it would be very hard to discern such ambiguities in implicit feedback and most existing works [Hu et al. 2008; Pan and Scholz 2009] simply treat positive and negative feedback as numerical values 1 and 0 and minimize the squared error over the entire matrix. Sometimes, additional attribute information can also be observed in conjunction with implicit feedback. For example, for clicks, we may also observe the number of times (i.e., frequency) a user clicks on pages about an item, whereas for page views, the user's dwell time would reflect how useful she finds the page. In this work, we focus on the simpler binary forms of implicit feedback, which could sufficiently capture all types of user behavior in the CAMRa 2010 dataset. We would address the issue of handling implicit feedback with numeric attributes in future research.

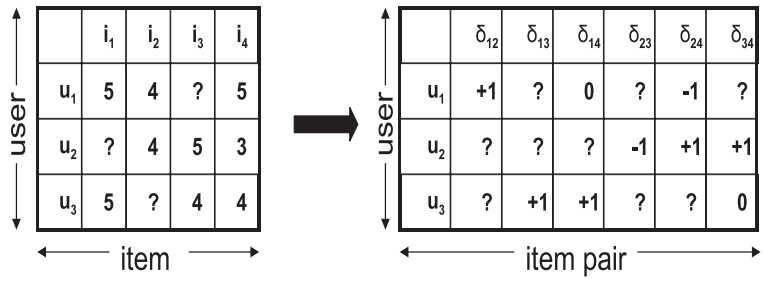
Although some positive results were observed for such regression oriented models on implicit feedback datasets, it is unclear how to apply regression models to combine multiple types of explicit and implicit feedback in order to predict scores $\widehat{x}_{ui}$ that would be consistent with user preferences expressed in both explicit and implicit feedback. The main difficulty is that the different numeric feedback representations have different scale and granularity, which makes it hard to apply regression models under this setting.

While most existing CF models are based on building a regression model to predict rating scores, a new ranking oriented approach advocating the modeling of pairwise preferences was proposed by Liu and Yang [2008] and successfully applied to model both explicit feedback [Liu et al. 2009] and implicit feedback [Rendle et al. 2009]. The key idea is to design appropriate learning criteria to correctly rank the items in the order induced by the ratings rather than to produce the exact rating scores. Similarly, the Bayesian Personalized Ranking(BPR) model [Rendle et al. 2009] adopts the same line of thinking for implicit feedback and proposes a training criterion that is based on ranking item in I_u^+ higher than those in I_u^- for each user u. A key step in collaborative ranking based on both explicit and implicit feedback is the transformation of the observed user dependent item ratings x_{ui} to user dependent item pairwise preferences δ_{uij}.

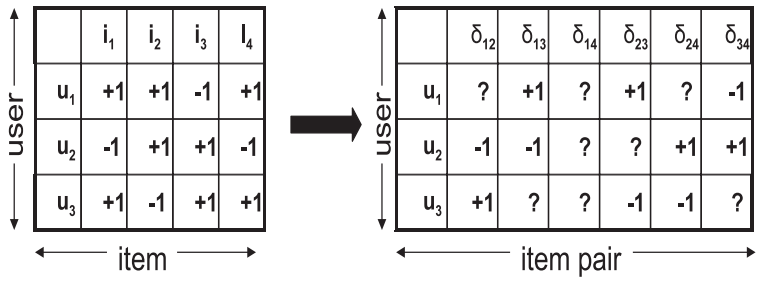
In particular, for explicit feedback, we can derive δ_{uij} based on x_{ui} and x_{uj} using the following rule if i and j have both been rated by u:

$$\delta_{uij} = \begin{cases} +1 & \text{if } x_{ui} > x_{uj} \\ 0 & \text{if } x_{ui} = x_{uj} \\ -1 & \text{if } x_{ui} < x_{uj}. \end{cases}$$

So δ_{uij} is equal to +1 if u likes i more than j and is equal to -1 if the opposite is true. A special case is when δ_{uij} equals 0, which means that u has no preference over the two items i and j. Figure 1(a) shows an example of transforming a rating matrix containing explicit feedback into the corresponding pairwise preference-based representation. The special case of $\delta_{uij} = 0$ is known as tied preference [Zhou et al. 2008]. As most recommender systems only solicit ratings on a coarse scale with limited



(a) Transforming explicit feedback to pairwise preferences



(b) Transforming implicit feedback to pairwise preferences

Fig. 1. Unifying explicit and implicit feedback with pairwise preference-based representation.

number of preference levels (e.g., 1 to 5), it is very common for a user to assign the same score to many different items. When learning ranking functions with this kind of ordinal labeling, explicitly modeling tied rankings has been shown to be helpful [Zhou et al. 2008].

For implicit feedback, δ_{uij} is derived based on the following rules:

$$\delta_{uij} = \begin{cases} +1 \text{ if } i \in \mathcal{I}_{u+} \text{ and } j \in \mathcal{I}_{u-} \\ -1 \text{ if } i \in \mathcal{I}_{u-} \text{ and } j \in \mathcal{I}_{u+}, \end{cases}$$

which stipulates that an item with positive feedback is more preferred over an item with negative feedback, whereas there are no pairwise preferences between two items that both have positive or negative feedback. Figure 1(b) shows an example of transforming a binary rating matrix containing implicit feedback into a pairwise preference-based representation. For implicit feedback, we also do not consider the case of tied rankings due to two reasons: (1) implicit feedback is abundant and we can extract sufficient pairwise preferences for first two cases, and (2) the x_{ui} themselves are ambiguous and not as informative as numeric ratings so it is unreasonable to assume that a user u likes an item i as much as he likes j only because x_{ui} and x_{uj} are both equal to $+1$ or -1.

Note that by definition, $\delta_{uij} = -\delta_{uji}$. Thus, when extracting the pairwise preferences, it would suffice to just consider those (u, i, j) triples, for which $\delta_{uij} = +1$. Following the

aforementioned criteria, we can derive from either explicit feedback or implicit feedback datasets the following two collection of pairwise preferences: (1) $\mathcal{D}_0 = \{(u, i, j) : x_{ui} = x_{uj}\}$, the set of tied preferences extracted from explicit feedback data (2) $\mathcal{D}_1 = \{(u, i, j) : x_{ui} > x_{uj}\}$, the set of nontied preferences extracted from either implicit or explicit feedback data.

4.2. Bradley-Terry Model for Pairwise Preferences

Bradley-Terry is a widely used probability model for outcomes of paired comparisons [Marden 1995], and it has been successfully used in learning ranking functions for information retrieval applications [Burges et al. 2005; Zhou et al. 2008]. A nice property of the Bradley-Terry model is that it only requires knowledge about how each pair of items should be relatively ordered rather than the absolute scores of each item. Given the heterogeneous forms of user feedback, it is not clear how to derive absolute relevance scores for each item. But as we show, both explicit and implicit feedback can be represented in the form of pairwise preferences as required by the Bradley-Terry model, which provides an effective means to cope with the heterogeneity of user feedback.

Under the Bradley-Terry model, each user is associated with n parameters $\widehat{x}_{u1}, \ldots, \widehat{x}_{un}$ reflecting the utility of each item to each user such that the higher the value of $\widehat{x}_{ui}$ compared to $\widehat{x}_{uj}$, the more likely δ_{uij} would be $+1$ (i.e., user u likes item i more than j). Let $\widehat{\delta}_{uij} = \widehat{x}_{ui} - \widehat{x}_{uj}$. The probabilities of the various outcomes of a pairwise preference δ_{uij} are defined as follows:

$$\begin{aligned} P(\delta_{uij} = +1) &= \sigma_\gamma(\widehat{\delta}_{uij}) \\ P(\delta_{uij} = -1) &= \sigma_\gamma(-\widehat{\delta}_{uij}) \\ P(\delta_{uij} = 0) &= 1 - \sigma_\gamma(\widehat{\delta}_{uij}) - \sigma_\gamma(-\widehat{\delta}_{uij}), \end{aligned}$$

where $\sigma_\gamma(\cdot)$ is the logistic sigmoid with an additional parameter γ:

$$\sigma_\gamma(x) = \frac{1}{1 + \gamma e^{-x}}. \tag{5}$$

The parameter $\gamma \geq 1$ controls the probability of ties. If $\gamma = 1$, the model does not consider tied preferences, and $P(\delta_{uij} = +1) = 1 - P(\delta_{uij} = -1)$. A γ value greater than 1 would account for tied rankings. Given the Bradley-Terry model for each user, personalized recommendations can be easily obtained by ranking the items according to $\widehat{x}_{ui}$, which would naturally correspond to the maximum likelihood-based ranking under the probability distribution in Bradley-Terry Model [Liu et al. 2009].

Given m users and n items in a recommender system, the Bradley-Terry model would involve $m \times n$ parameters $\widehat{x}_{ui}$, which can be represented by the matrix $\widehat{X}$. A natural approach to learning $\widehat{X}$ is using the maximum likelihood principle, which leads to the following Bradley-Terry model-based loss function applicable to both explicit feedback and implicit feedback:

$$\mathcal{L}_{bt}(X, \widehat{X}) = \sum_{(u,i,j)\in\mathcal{D}_0} \ln(1 - \sigma_\gamma(\widehat{\delta}_{uij}) - \sigma_\gamma(-\widehat{\delta}_{uij})) + \sum_{(u,i,j)\in\mathcal{D}_1} \ln \sigma_\gamma(\widehat{\delta}_{uij}). \tag{6}$$

In Liu et al. [2009], a probabilistic mixture model known as probabilistic latent preference analysis (PLPA) was developed for decomposing $\widehat{X}$ via a mixture of Bradley-Terry models, reducing the number of parameters from mn to $(m + n)k$. However, the expectation-maximization algorithm for fitting the PLPA model required to explicitly construct the entire set of pairwise preferences, which could lead to quadratic increase in terms of complexity and was therefore not scalable enough for massive datasets.

Here, we take the matrix factorization-based formulation as in Rendle et al. [2009], and develop a stochastic gradient descent algorithm with bootstrap sampling for fitting the model, which randomly samples pairwise preferences on the fly and is therefore highly scalable.

Combining with Frobenius norm-based regularization in Equation (4), we can define the overall objective function of the collaborative ranking model for combining multiple heterogeneous forms of user feedback:

$$\arg\min_{\Theta} \sum_{z} \mathcal{L}_{bt}(X^z, \widehat{X}) + \mathcal{R}(\Theta), \tag{7}$$

where we replaced the regression loss over a single rating matrix in Equation (3) with the sum of ranking losses over multiple matrices.

4.3. Learning the Collaborative Ranking Model

In the previous section, we have defined a general pairwise preference-based loss function Equation (6) applicable to both explicit and implicit feedback as well as the the coranking model Equation (7) for aggregating explicit and implicit feedback. As both the loss function and the regularization functions are smooth and differentiable, it is tempting to directly apply standard gradient descent style algorithms to minimize the objective function. However, note that in order to compute $\mathcal{L}_{bt}$ in each iteration, one has to sum over all the triples (u, i, j) in the sets $\mathcal{D}_0$ and $\mathcal{D}_1$, whose size is on the order of $O(mn^2)$. In both cases, we can see that the number of training instances will be quadratic to the number of training instances for the rating-based representations. Thus computing the gradient over all triples in the sets $\mathcal{D}_0$ and $\mathcal{D}_1$ would be computationally infeasible.

Here we take the stochastic gradient descent approach, which has been successfully used for learning matrix factorization models on the large scale Netflix challenge dataset [Koren et al. 2009]. The idea is to randomly sample a triple (u, i, j) and update the model parameters based on the loss over the pairwise preference δ_{uij}. Applying the chain rule, we may obtain the following general gradient descent-based update rule for parameters Θ when $\delta_{uij} = 1$:

$$\Theta \leftarrow \Theta - \alpha \cdot \left(\left(1 - \sigma_\alpha(\widehat{\delta}_{uij})\right) \cdot \frac{\partial}{\partial \Theta} \widehat{\delta}_{uij} + \frac{\partial}{\partial \Theta} \mathcal{R}(\Theta) \right), \tag{8}$$

where α is the parameter that controls the size of each gradient descent step.

In case of a tied ranking, that is, $\delta_{uij} = 0$, the update rules will be as follows:

$$\Theta \leftarrow \Theta - \alpha \cdot \left(\left(\sigma_\alpha(\widehat{\delta}_{uij}) - \sigma_\alpha(-\widehat{\delta}_{uij})\right) \cdot \frac{\partial}{\partial \Theta} \widehat{\delta}_{uij} + \frac{\partial}{\partial \Theta} \mathcal{R}(\Theta) \right). \tag{9}$$

By plugging in the appropriate $\frac{\partial}{\partial \Theta} \widehat{\delta}_{uij}$ for different parameterizations of the predictor $\widehat{x}_{ui}$, we can easily obtain stochastic gradient update rules for different models. For example, for the basic matrix factorization models $\widehat{X} = W^T H$, the gradients with respect to $\mathbf{w}_u$ and $\mathbf{h}_i$ are:

$$\frac{\partial}{\partial \mathbf{w}_u} \widehat{\delta}_{uij} = \mathbf{h}_i - \mathbf{h}_j \ , \quad \frac{\partial}{\partial \mathbf{h}_i} \widehat{\delta}_{uij} = \mathbf{w}_u.$$

The optimization procedure is shown in Algorithm 1. The algorithm takes as input a set of Z matrices $\{X^1, X^2, \ldots, X^Z\}$, each of which contains a particular type of either explicit or implicit user feedback. In each iteration, the algorithm first sweeps through randomly sampled triples (u, i, j) from explicit feedback. Notice that rather than explicitly constructing the set of triples D^*, whose size would be $O(mn^2)$, we instead

directly generate each triple (u, i, j) based on the ratings set S^* with the operation *bootstrap_explicit()*. This is done by firstly randomly selecting a user u and two items rated by the users from I_{u*}, which is a constant time operation involving the generation of three random numbers.

After processing explicit feedback, the algorithm then processes implicit feedback. To sample a triple (u, i, j), we need to randomly select i from I_{u+} and j from I_{u-}. However, it would require $O(mn)$ space to store both S^+ and S^-, which is unfeasible. We therefore use the following rejective sampling approach to randomly generate (u, i, j) only based on S^+. Firstly, a user u is randomly selected and an item i is sampled from I_{u+}. Secondly, an item j in I_{u-} is randomly selected via a rejective sampling procedure, which repeatedly generate random number j between 1 and m and returns j only if $j \notin I_{u+}$. Checking whether $j \notin I_{u+}$ can be efficiently performed in constant time by building a hash set for each I_{u+}. Notice that the probability of rejection is equal to $\frac{|I_{u+}|}{|I|}$, which is very small as most users would have positive feedback on a very small proportion of items. So the bootstrapping-based sampling operation *bootstrap_implicit()* is also a constant time operation.

ALGORITHM 1: Stochastic Gradient Descent for the Collaborative Ranking Model

Input: A set of Z matrices $\{X^1, X^2, \ldots, X^Z\}$ containing different types of user feedback
Output: Model parameters Θ
Initialize Θ with random values;
for *iter* $\leftarrow 1$ **to** *MAX_IT ER* **do**
  foreach $X \in \{X^1, X^2, \ldots, X^Z\}$ **do**
    for $t = 1$ *to* T **do**
      if X^z *contains explicit feedback* **then**
        $(u, i, j) \leftarrow$ *bootstrap_explicit*(X);
      else
        $(u, i, j) \leftarrow$ *bootstrap_implicit*(X);
      end
      if $\delta_{uij} \neq 0$ **then**
        $\Theta \leftarrow \Theta - \alpha \cdot \left(\left(1 - \sigma_\alpha(\widehat{\delta}_{uij})\right) \cdot \frac{\partial}{\partial \Theta} \widehat{\delta}_{uij} + \frac{\partial}{\partial \Theta} \mathcal{R}(\Theta) \right)$;
      else
        $\Theta \leftarrow \Theta - \alpha \cdot \left(\left(\sigma_\alpha(\widehat{\delta}_{uij}) - \sigma_\alpha(-\widehat{\delta}_{uij})\right) \cdot \frac{\partial}{\partial \Theta} \widehat{\delta}_{uij} + \frac{\partial}{\partial \Theta} \mathcal{R}(\Theta) \right)$;
      end
    end
  end
end

5. TIME AWARE COLLABORATIVE RANKING

In most real-world recommender systems, both users and items could exhibit time dependent behaviors. For example, users tastes would naturally change over time and the their preferences may be inconsistent under different circumstances. For example, during holidays people may become more interested in movies that are suitable for watching with family. On the other hand, item popularity may also be rather time dependent. For example, a movie's popularity would suddenly rise if it was nominated for awards. In this section, we extend the basic collaborative ranking described in previous section with time-dependent parameter in order to capture such temporal dynamics. In particular, we take a discrete time modeling approach, in which we let the model change at regularly spaced time steps, where a parameter d determines the

number of days between consecutive time steps. Given d, we could then divide the entire time span into T time steps, $t_1, \ldots t_T$. Under a discrete time model, $\Theta = \{\Theta_0, \Theta_1, \ldots, \Theta_T\}$ where Θ_0 represents time independent model parameter while Θ_1 to Θ_t represent parameters specific to each time step t. The parameter d controls the granularity of the temporal dynamics being captured by the model. With large d values, we have less time steps and the model would mostly capture long term behaviors of users or items. By reducing d, the model becomes more and more flexible such that very granular temporal events can also be accounted for by the model.

Let t_{ui} denote the date when the rating x_{ui} was produced. We can also split the set of observed ratings $\mathcal{D}$ into $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T$ such that $(u, i) \in \mathcal{D}_k$ if and only if $t_{k-1} \leq t_{ui} \leq t_k$. At each time step t, we would have some parameters specific to this time step in order to capture time dependent properties. Note that for implicit feedback datasets, t_{ui} is only known for entries that correspond to positive feedback (i.e., $(u, i) \in \mathcal{S}_+$).

The first step in adapting the collaborative ranking model is to derive time dependent user preferences $\delta_{uij}(t)$. To derive the time dependent preferences $\delta_{uij}(t)$, we followed similar rules as defined in section 4.1 with the only difference being that we now only consider entries in $\mathcal{D}_t$ as being rated while all other entries are considered unrated. To be more specific, for explicit feedback, $\delta_{uij}(t) = +1$ if u rated both i and j within time step t and that $x_{ui} > x_{uj}$. For implicit feedback, temporal information is only available for positive ratings but not negative ratings, we thus derive a time dependent pairwise preference $\delta_{uij}(t) = +1$ if x_{ui} is a positive rating within time step t while x_{uj} is a negative rating.

5.1. Sequential Matrix Factorization Models

Our goal is to design a time varying predictor $\widehat{x}_{ui}(t)$ in order to capture the time dependent preference of a user for an item. In this work, we designed and compared several forms of parametrization by adapting the basic matrix factorization model $\widehat{x}_{ui} = \mathbf{w}_u^T \cdot \mathbf{h}_i$. The general idea is to designate a factor matrix for each time step, so the set of factor matrices would naturally form a temporally ordered sequence. Thus, we refer to this class of models as sequential matrix factorization.

5.1.1. User-Based Models. Our first extension, the user-based model, is designed to track time dependent user interests. Similar to the item models, we use a sequence of user factor matrices $\{W_1, \ldots, W_T\}$ to enable time dependent user representations. In this model, we make the item factor H static, so the full set of parameters is $\Theta = \{W_1, \ldots, W_T, H\}$. The predictor is defined as follows:

$$\widehat{x}_{ui}(t) = \mathbf{w}_{u,t}^T \cdot \mathbf{h}_i \tag{10}$$

where $\mathbf{w}_{u,t}$ denotes the u-th column in W_t. The total number of parameters is $mkT + nk$.

In order to apply the collaborative ranking algorithm to this model, we only need to derive the gradients of the predictor $\widehat{\delta}_{uij}(t) = \widehat{x}_{ui}(t) - \widehat{x}_{uj}(t)$ with respect to $\mathbf{w}_{u,t}$ and $\mathbf{h}_i$:

$$\frac{\partial}{\partial \mathbf{w}_{u,t}} \widehat{\delta}_{uij}(t) = \mathbf{h}_i - \mathbf{h}_j \ , \qquad \frac{\partial}{\partial \mathbf{h}_i} \widehat{\delta}_{uij}(t) = \mathbf{w}_{u,t} \tag{11}$$

5.1.2. Item-Based Model. Our second extension, the item-based model, is based on the hypothesis that item characteristics such as popularity would change over time whereas the user's tastes are relatively stable. Therefore, for each time step, we use a time dependent factor matrix H_t to capture each item's representation at each time step t while the same user factor W is applied at every time step. Therefore, the parameter collection is $\Theta = \{W, H_1, \ldots, H_T\}$. This leads to the following form of the time dependent predictor:

$$\widehat{x}_{ui}(t) = \mathbf{w}_u^T \mathbf{h}_{i,t} \tag{12}$$

where $\mathbf{h}_{i,t}$ denotes the i-th column in H_t. Under this model, the total number of parameters is $mk + nkT$.

Similarly, the gradients of the predictor $\widehat{\delta}_{uij}(t) = \widehat{x}_{ui}(t) - \widehat{x}_{uj}(t)$ with respect to $\mathbf{w}_u$ and $\mathbf{h}_{i,t}$ are computed by:

$$\frac{\partial}{\partial \mathbf{w}_u}\widehat{\delta}_{uij}(t) = \mathbf{h}_{i,t} - \mathbf{h}_{j,t}\ , \quad \frac{\partial}{\partial \mathbf{h}_{i,t}}\widehat{\delta}_{uij}(t) = \mathbf{w}_u. \tag{13}$$

5.1.3. Discussion. In addition to only letting one of user or item factor matrices change over time, we also tried a more elaborate version in which both user and item factor matrices are time dependent. However, we did not obtain any further improvement from this more flexible model. While both user and item models can significantly outperform static models that ignore temporal information, we nevertheless found that the item-based model consistently led to better performance than the user-based model, which indicates that the items in the CAMRa dataset tend to exhibit highly nonstationary properties whereas the users appeared to have relatively more stable preferences over time. Note that our finding is in contrast with the results on the Netflix datasets [Koren 2009], where time-dependent user factors were found to work best. We believe that a major reason that led to this result is the different time span in the two datasets. The Netflix data spans over 7 years of time whereas the CAMRa dataset spans less than 2 years, so it would be much less likely for a user to have drastic changes in his preference over a much shorter period of time.

5.2. Temporal Smoothness Regularization

The capability of time-dependent modeling comes at the expense of high parametric complexity. In particular, the sequential matrix factorization model would contain up to T times as many parameters as the basic matrix factorization models. With such an expressive model, it is important to impose certain constraints based on the problem structure in order to alleviate overfitting. One particular assumption we make about the underlying temporal dynamics in the datasets is that the user and item characteristics are not likely to change drastically in consecutive time steps. To reflect this in our sequential matrix factorization, we only need to ensure that the time dependent model parameters Θ_t and Θ_{t+1} in two consecutive time periods do not deviate from each other drastically. Mathematically, this can be translated into the following temporal smoothness regularization function on the collection of parameters $\Theta = \{\Theta_0, \Theta_1, \ldots, \Theta_T\}$:

$$\mathcal{R}(\Theta) = C_1 \cdot \sum_{t=0}^{T} \|\Theta_t\|_F^2 + C_2 \cdot \left(\sum_{t=2}^{T} \|\Theta_t - \Theta_{t-1}\|_F^2\right), \tag{14}$$

where we augmented the Frobenius norm-based regularization term with another term that penalizes the Frobenius norm of the difference $\Theta_t - \Theta_{t-1}$. It is easy to derive the gradients of this regularization function due to the smooth differentiability of the Frobenius norm.

$$\frac{\partial}{\partial \Theta_t}\mathcal{R}(\Theta) \propto C_1 \cdot \Theta_t + C_2 \cdot (2 \cdot \Theta_t - \Theta_{t-1} - \Theta_{t+1}). \tag{15}$$

6. SOCIAL NETWORK AWARE COLLABORATIVE RANKING

Within social networks, people with similar social-demographic or behavioral characteristics are more likely to connect to each other. This is often known as the homophily principle: similarity breeds connection. The homophily effect has been demonstrated across a variety of online social networks. The most important implication of this theory for social network aware recommendation is that people that are connected to each

other are expected to have similar tastes. As a result, the given social network structure can be viewed as a form of prior knowledge regarding similarity between users. The social network provided in the CAMRa dataset is in the form of an undirected graph. To incorporate such knowledge, we design a cost function in the following form:

$$\sum_{(u,v)\in\mathcal{E}} s_{uv} \cdot \|\Theta_u - \Theta_v\|_F^2, \tag{16}$$

where $\mathcal{E}$ denotes the set of observed edges between users, s_{uv} denotes the weight associated with an edge. $\Theta_u = \{\Theta_{u,0}, \Theta_{u,1}, \ldots, \Theta_{u,T}\}$ denote the collection of time independent and time dependent parameters associated with user u. The intuition behind this cost function is to penalize the factors of two closely related users that deviate from each other. We refer to this form of regularization as *social regularization*.

Combining this with the Frobenius norm regularization and temporal smoothness regularization, we have the complete regularization function for the social temporal collaborative ranking model:

$$\mathcal{R}(\Theta) = C_1 \cdot \sum_{t=0}^{T} \|\Theta_t\|_F^2 + C_2 \cdot \sum_{t=2}^{T} \|\Theta_t - \Theta_{t-1}\|_F^2 + C_3 \cdot \sum_{(u,v)\in\mathcal{E}} s_{uv} \cdot \|\Theta_u - \Theta_v\|_F^2, \tag{17}$$

where the parameters C_1, C_2, C_3 controls the strength of the three types of regularization. The network regularization term only has effect on user parameters. The gradient of the regularization function with respect to the time independent user parameters does not involve the temporal smoothness regularization and is as follows:

$$\frac{\partial}{\partial \Theta_{u,0}} \mathcal{R}(\Theta) \propto C_1 \cdot \Theta_{u,0} + C_3 \cdot \sum_{v:(u,v)\in\mathcal{E}} s_{uv} \cdot (\Theta_{u,0} - \Theta_{v,0}). \tag{18}$$

On the other hand, the user's time dependent parameters are affected by both temporal smoothness and social regularization. For the two special cases when $t = 1$ and T, the gradients are:

$$\frac{\partial}{\partial \Theta_{u,1}} \mathcal{R}(\Theta) \propto C_1 \cdot \Theta_{u,t} + C_2 \cdot (\Theta_{u,1} - \Theta_{u,2}) + C_3 \cdot \sum_{v:(u,v)\in\mathcal{E}} s_{uv} \cdot (\Theta_{u,1} - \Theta_{v,1}) \tag{19}$$

$$\frac{\partial}{\partial \Theta_{u,T}} \mathcal{R}(\Theta) \propto C_1 \cdot \Theta_{u,t} + C_2 \cdot (\Theta_{u,T} - \Theta_{u,T-1}) + C_3 \cdot \sum_{v:(u,v)\in\mathcal{E}} s_{uv} \cdot (\Theta_{u,T} - \Theta_{v,T}). \tag{20}$$

Finally, the gradient with respect to time dependent user parameters for any time steps between 1 and T is:

$$\frac{\partial}{\partial \Theta_{u,t}} \mathcal{R}(\Theta) \propto C_1 \cdot \Theta_{u,t} + C_2 \cdot (2 \cdot \Theta_{u,t} - \Theta_{u,t-1} - \Theta_{u,t+1}) + C_3 \cdot \sum_{v:(u,v)\in\mathcal{E}} s_{uv} \cdot (\Theta_{u,t} - \Theta_{v,t}). \tag{21}$$

6.1. Edge Weighting Strategies

Given a social network G, one way to define s_{uv} is to make it equal to 1 if the two users are connected, and 0 if otherwise. This method assumes that a user should be equally influenced by each of his neighbors, which is not a reasonable assumption since in reality social ties among people often have nonuniform strengths. To refine this model, for every pair of connected user u and v, we also assign a real valued weight to s_{uv} to reflect the degree to which the parameters of user u and v should be correlated to each other. Note that the weights are only computed for pairs of users that are connected to each other, whereas for those $(u, v) \notin \mathcal{E}$, s_{uv} would always be 0. In the following, we

describe two methods for calculating the weight s_{uv} based on the observed user ratings as well as the temporal ordering of their actions.

6.1.1. Similarity-Based Weighting. One way to detect if a social connection implies homophily is to measure the consistency of the two users' observed ratings. Intuitively, if two users tend to have very different opinions regarding the items, then the observed social connection between them is probably a spurious edge that should not be considered in the regularization. A commonly used measure for such purpose is the Pearson Correlation Coefficient between two users' ratings:

$$s_{uv} = \frac{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (x_{ui} - \overline{x}_u)(x_{vi} - \overline{x}_v)}{\sqrt{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (x_{ui} - \overline{x}_u)^2} \sqrt{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} (x_{vi} - \overline{x}_v)^2}}. \tag{22}$$

6.1.2. Influence-Based Weighting. The similarity-based weighting in the previous section only tells the correlation between two users' tastes. Moreover, it is also a symmetric measure such that $s_{uv} = s_{vu}$. A problem with this measure is that it does not take into account the time when those actions occurred. However, the temporal order of the user actions is essential for analyzing social influences between users [Anagnostopoulos et al. 2008]. Social influence plays an important role in dictating users' behavior. In particular, a user u has influence on another user v if and only if v is observed to take an action after u has taken the same action. This implies that in order to determine if user u has a strong influence on another user v, we not only need to check if they tend to take similar actions, but also examine if user u is the one who always does them first. This leads to the following asymmetric influential strength measure s_{uv}, which reflects the strength of influence u has on v:

$$s_{uv} = \sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} \exp\left(-\frac{t_{ui} - t_i^0}{\bar{t}_i}\right) - \exp\left(-\frac{t_{vi} - t_i^0}{\bar{t}_i}\right), \tag{23}$$

where t_i^0 denotes the time when i received its first rating in the dataset and $\bar{t}_i$ is the average time of ratings on i. This formulation is based on the following reasoning: u has a strong influence on v if they have rated many common items and that u often rates such items before v.

7. EXPERIMENTS

7.1. Data Description

We conducted our experiments based on the Filmtipset dataset provided to participants of CAMRa 2010 [Said et al. 2010], which is one of the first large scale dataset to contain heterogeneous user feedback, temporal information as well as social network structure. The Filmtipset dataset contains around 6 million ratings on 54,977 movies by 35,351 users, which were collected between 2008 and 2010. In addition to explicit feedback in the form of ratings, the dataset also provides several types of implicit user feedback including which user marked which movies as favorite, which user added which movies to his personal collections and which user commented on which movies.

The CAMRa contest aims at evaluating movie recommendation quality during two specific time periods corresponding to two special occasions, the Oscar week, which spans from 2010/02/27 to 2010/03/07, and Christmas week, which spans from 2009/12/21 to 2009/12/27. We therefore constructed two sets of training and test data for evaluating recommendation quality in the two weeks respectively. For each target week, we use all the ratings dated before the beginning of the target week as the training data and the ratings within that week as the test data. The number of ratings in the training and test set for the Oscar week are 5,862,452 and 33,548 respectively,

whereas there are 5,374,000 and 23,393 ratings in the training and test set for the Christmas week. We use this kind of chronological order-based data split in order to ensure that training data precedes test data. We believe this is a more reasonable evaluation scheme compared with randomly splitting the data into training and test sets without considering temporal ordering, since real systems have to predict users' future preferences based on past behavior.

7.2. Evaluation Metric

Traditionally, the performance of recommendation algorithms is measured by the accuracy of the algorithm's predictions on a set of held out ratings, so only items, ratings on which are observed are considered for evaluation. However, it has been shown that the availability of observed ratings is not a random phenomenon but is in fact strongly correlated with user's preferences, since most users are much more likely to rate movies they like over movies they dislike [Marlin and Zemel 2007]. In real world recommender systems, the recommended items need to be selected from the pool of items not rated by the user yet. So it is more appropriate to evaluate algorithms by ranking both rated and unrated items and examine the positions of relevant items (i.e., items with high ratings), which has been suggested as a new evaluation scheme [Steck 2010; Koren 2010]. Intuitively, it is desirable for a good recommendation algorithm to rank relevant items higher than both irrelevant (i.e., low rated) and unrated items. A major difficulty with this evaluation scheme is that it is unknown whether an unrated item is relevant or irrelevant, so it is unclear how to determine the quality of a ranked list consisting of both rated and unrated items. Fortunately, under the assumption that ratings on relevant items are *missing at random* [Steck 2010], the values of any ranking metric computed by treating unrated items as irrelevant would be proportional to the the value computed if all ratings on relevant items were observed. In our experiments, we follow this scheme and compute several commonly used ranking performance metric. It should be noted that since the potentially relevant unrated items are treated as irrelevant under this scheme, the resulted performance metric values would *under estimate* the actual performance, but could effectively serve for the purpose of comparing the relative performances of different algorithms.

The ratings in the Filmtipset dataset range from 1 to 5. We treat items with ratings higher than or equal to 4 as being relevant to a user whereas items not rated or rated lower than 4 are treated as irrelevant. Different algorithms are then used to generate ranked lists of items not rated by the user. Using the aforementioned rule, we then use the highly rated items in the test set to identify the relevant results in the ranked list and measure the quality of the ranked list of items using several binary relevance-based information retrieval performance metrics including Precision at K ($Pre@K$), Mean Average Precision (MAP) and Area under Curve (AUC) [Herlocker et al. 2004].

Precision at K is a widely used performance metric that is defined as the fraction of relevant results among the top K results.

$$Pre@K = \frac{N_K^+}{N_K^+ + N_K^-}, \tag{24}$$

where N_K^+ and N_K^- are the number of good and bad recommendations among the top K results so we have $N_K^+ + N_K^- = K$. In our experiments, the $Pre@K$ values are first computed for each user independently and then the average is computed as the final performance.

Rather than considering the precision at only the K-th position, the average precision (AP) metric considers all rank positions with relevant results and is defined as the

average of the precisions at these positions:

$$AP = \frac{\sum_{k=1}^{N} Pre@k \times \delta^{+}(k)}{N^{+}}, \tag{25}$$

where N^{+} denotes the total number of relevant items, $Pre@k$ is the precision at the k-th position as defined in Equation (24) and $\delta^{+}(k)$ is a binary indicator function that returns 1 if the result ranked at the k-th position is relevant and 0 if otherwise. The mean average precision (MAP) is simply the mean of the average precisions computed for individual users.

Area under Curve (AUC) is defined as the area under the receiver operating characteristic (ROC) curve, which is mathematically equivalent to the proportion of correctly ordered items in the ranked list. A pair of items is discordant if the item at the higher rank position is irrelevant while the result at the lower rank position is relevant:

$$AUC = 1 - \frac{N^{\mp}}{N^{+} \times N^{-}}, \tag{26}$$

where $N^{\mp}$ denotes the number of discordant item pairs while N^{+} and N^{-} denotes the total number of relevant and irrelevant items. Given a set of users, we first compute AUC for each user's ranked list and then take the average across different users.

All of the three performance metrics (i.e., MAP,AUC and $PRE@K$) are score metrics, for which a value closer to 1.0 indicates better performance.

7.3. Comparison with Baseline Methods

We compared the proposed social temporal collaborative ranking (ST-CoR) model with the following models.

—The SVD++ model [Koren 2010], which uses implicitly rated items for each user to augment his latent factors in a matrix factorization model. This model is one of the state of the art techniques for collaborative filtering and forms a key technique in the winning solution to the Netflix prize competition.
—The Bayesian Personalized Ranking (BPR) [Rendle et al. 2009] model is a ranking-oriented matrix factorization model designed for handling implicit feedback. It infers that a user u prefers item i over j if positive feedback is observed for (u, i) but not for (u, j).
—The implicit feedback-based matrix factorization (IFMF) model [Hu et al. 2008] is a regression-oriented matrix factorization model for collaborative filtering that treats implicit feedback as 0 and 1 values and then learns a matrix factorization model by optimizing the least squared error over this binary matrix.
—The basic collaborative ranking model without time and social network awareness (CoR). A comparison with this model allows us to separate the effect of combining heterogeneous user feedback from the effect of considering temporal and social context.

For the ST-CoR model, we set the number of days for each time step equal to 20 days and make C_1 and C_2 equal to 0.002 and 0.0015. We let parameter C_3 equal to 0.0008 and 0 for the Oscar week and Christmas week dataset respectively. These parameters are chosen via validation on a small sub set of the ratings within the target weeks that are excluded from the final set of test ratings.

The results of all different algorithms are summarized in Tables I and II, where the values inside the brackets indicating standard deviations. From the results, we can make the following observations.

Table I. Results on Oscar Week Dataset

Algorithm	AUC	MAP	Pre@10	Pre@20
SVD++	0.8362(±0.0063)	0.0166(±0.0023)	0.0278(±0.0049)	0.0299(±0.0037)
IFMF	0.9184(±0.0058)	0.0333(±0.0018)	0.0406(±0.0066)	0.0352(±0.0032)
BPR	0.9285(±0.0068)	0.0267(±0.0035)	0.0368(±0.0043)	0.0332(±0.0028)
CoR	0.9310(±0.0042)	0.0287(±0.0027)	0.0387(±0.0058)	0.0366(±0.0027)
ST-CoR	**0.9410**(±0.0059)	**0.0840**(±0.0030)	**0.0684**(±0.0029)	**0.0494**(±0.0022)

Table II. Results on Christmas Week Dataset

Algorithm	AUC	MAP	Pre@10	Pre@20
SVD++	0.8446(±0.0053)	0.0178(±0.0019)	0.0196(±0.0037)	0.0144(±0.0041)
IFMF	0.9215(±0.0047)	0.0299(±0.0015)	0.0316(±0.0044)	0.0267(±0.0030)
BPR	0.9283(±0.0056)	0.0224(±0.0029)	0.0249(±0.0039)	0.0220(±0.0025)
CoR	0.9309(±0.0042)	0.0265(±0.0023)	0.0329(±0.0041)	0.0277(±0.0022)
ST-CoR	**0.9428**(±0.0044)	**0.0921**(±0.0024)	**0.0587**(±0.0022)	**0.0392**(±0.0019)

(1) The proposed ST-CoR model outperformed all the baselines generally by more than two standard deviations.
(2) The implicit feedback-based models(IFMF, BPR) appeared to be much more effective than explicit feedback-based model SVD++. This is mostly because SVD++'s learning criterion is strongly geared toward optimizing the root mean squared error, which is very different from the ranking performance metrics used in the CAMRa contest.
(3) Combining different forms of feedback via CoR indeed led to much better performances than those models (i.e., IFMF, BPR) that only consider implicit feedback.

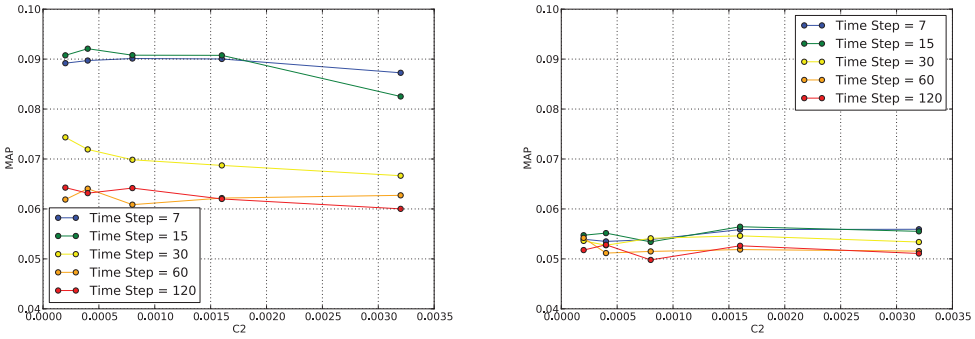
It can be seen that the obtained MAP and $Pre@K$ scores are quite low, which is the result of pessimistically treating unrated items as irrelevant. As we have argued previously, the results should not be taken as a absolute measure of recommendation quality.

To gain a deeper understanding of how temporal context and social context affect the performance of the proposed algorithm, we conduct more detailed result analysis in the following sections. Since the different performance metrics are generally consistent with each other, we would only report results measured using MAP.

7.4. Time-Aware Collaborative Ranking Results

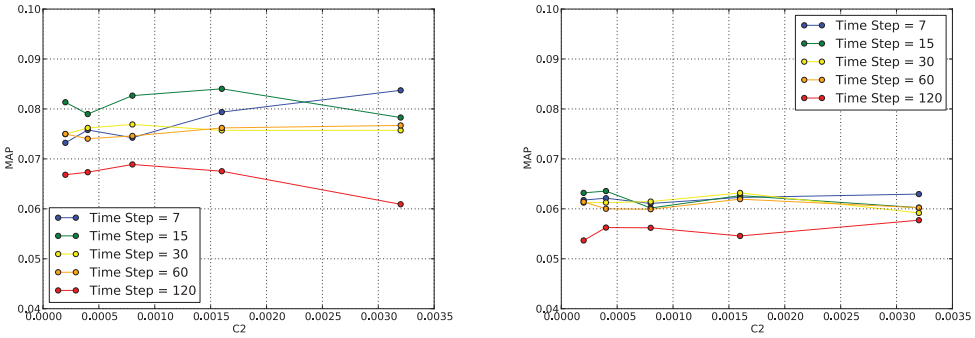
7.4.1. Effect of Time Granularity and Temporal Smoothness Regularization. In this set of experiments, we would like to study the effect of the time granularity in the sequential matrix factorization model. The time granularity is controlled by the length (i.e., number of days) of each time step. With larger time steps, the model only captures long term behaviors of users and items. Whereas with smaller time steps, the model will be able to capture more short term behaviors, such as the sudden surge of popularity of the Oscar nominated films in the weeks before the award ceremony.

In Figures 2 and 3, we plot the performances of both user- and item-based sequential matrix factorization models with time step ranging from 7 days to 120 days. For each model, we gradually increased the value of C_2 from 0 to 0.0035, corresponding to increasing strengths of the temporal smoothness-based regularization. The value of C_1 had been set to 0.002, which was found to perform best when there was no temporal smoothness regularization (i.e., $C_2 = 0$). From the results, we can make the following observations.



(a) Item-Oriented Model (b) User-Oriented Model

Fig. 2. Performances of time aware collaborative ranking during the Christmas week.



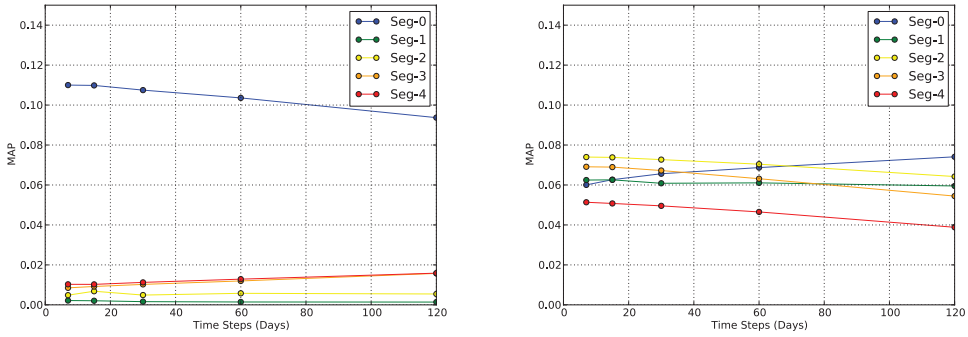
(a) Item-Oriented Model (b) User-Oriented Model

Fig. 3. Performances of time aware collaborative ranking during Oscar week.

(1) Very small time steps will lead to overfitting and poor performance. On the other hand, too large time steps also perform badly due to their inability to capture short term temporal dynamics.
(2) The temporal smoothness regularizer plays an important role in alleviating overfitting. For example, the most granular model with time step equal to 7 days tends to perform badly when C_2 is small, but as C_2 increases, its performance consistently improves and could outperform the less granular models.
(3) The item-oriented model is much more effective than the user-oriented model, which indicates that the effect of temporal dynamics of the items is much stronger than that of the users.

7.4.2. Detailed Analysis. In this set of experiments, we would like to carefully examine what kind of users and items are most likely to benefit from time aware modeling. In particular, we hypothesize that there are two important attributes of a user or item that would be correlated with the effectiveness of time aware modeling.

—*Longevity*. We define the longevity of a user or item as the amount of time between the current time and the-time of the earliest observed rating associated with the user or item.
—*Activeness*. We measure the activeness of a user or item as the total number of ratings associated with the user and item.



(a) Item segments defined by longevity (b) User segments defined by longevity

Fig. 4. Performances of time aware collaborative ranking for different item and user segments defined by longevity in Oscar week.

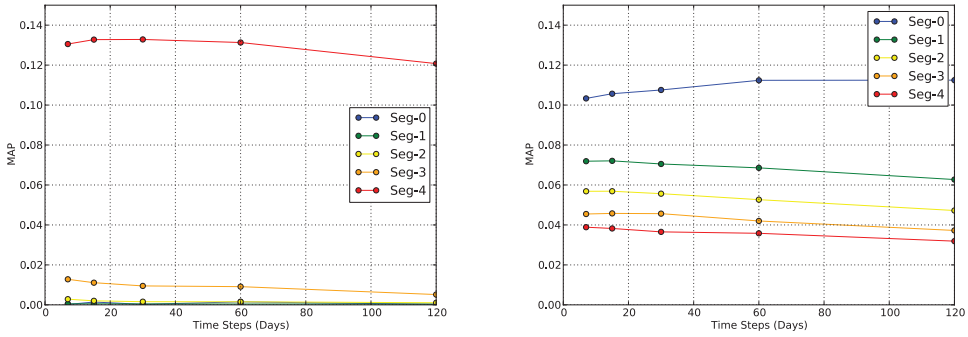
After computing longevity and activeness for every user and item, we rank the users and items by longevity and activeness and partition users and items into 5 equal-size segments based on longevity and activeness rank respectively. Segment 0 corresponds to the 20% users or items with the lowest longevity/activeness and segment 4 corresponds to the 20% users or items with the highest longevity/activeness. We then measure the performances within each user/item segment separately.

Figure 4(a) and 4(b) show the results obtained for different user and item segments defined based on longevity, which reveal the following findings.

—The performance on segment 0 containing the newest items is generally much better than the other segments consisting of older items. We think this is because users are typically more likely to rate more recent movies.
—Newer items tend to be more accurately recommended with granular models (i.e., models with small time steps) as can be seen in Figure 4(a). We believe this is due to the granular model's better capability of capturing short term behaviors, such as the steep increase in popularity right after a movie's release.
—More granular models tend to work more effectively for older users whereas less granular models are more effective for newer users as can be seen from Figure 4(b). This is because old users are more likely to have their preferences changed over time whereas new users' tastes would be relatively stable over the short time span they are in the system.

Similarly, Figure 5(a) and 5(b) show the results obtained for different user and item segments defined based on activeness, which reveal the following findings.

—The performance on segment 0 containing the most frequently rated items is generally much better than the other segments consisting of less popular items. We believe this is could be attributed to the long tail nature of the movie domain where the majority of the ratings are given to the most popular items. With more observed ratings in the training data, the most frequently rated items included in item segment 0 could be more accurately modeled, leading to the higher performance on this segment.
—Models with smaller time steps perform consistently better on the most active item segments (i.e., items that are most frequently rated). However, introducing time awareness leads to poorer performance on the least active items (i.e., long tail items), which should be due to overfitting resulted from the increased model complexity.



(a) Item segments defined by activeness (b) User segments defined by activeness

Fig. 5. Performances of time aware collaborative ranking for item and user segments defined by activeness in Oscar week.

—Granular models with small time steps tend to perform better on the active users in segment 1 to 4 whereas larger time steps are more suitable for the least active users in segment 0, as can be seen in Figure 5(b). We think it is because inactive users tend to suffer more from overfitting when time steps are small as there would be very little user feedback from such users in each time step. For example, with time step equal to 7 days, each user would have on average only 1.36 ratings within each time step.

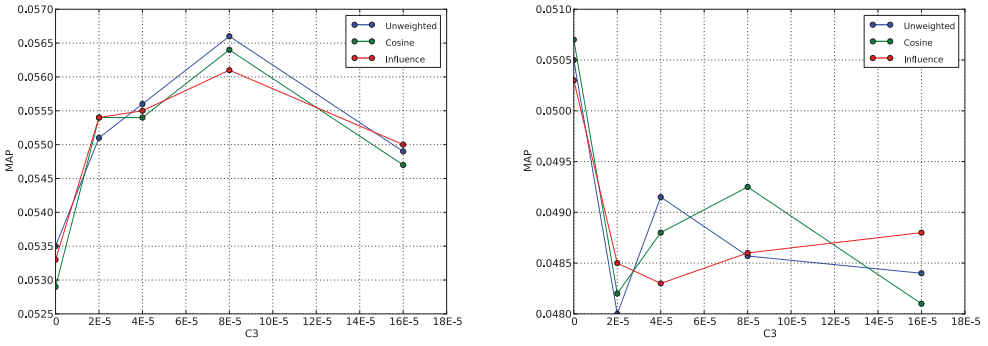
7.5. Social Network Aware Collaborative Ranking Results

7.5.1. Comparing Different Social Network Weighting Techniques. In this set of experiments, we attempt to answer two questions: (1) can social network regularization technique improve the recommendation quality and, (2) which edge weighting technique is most effective. We compare the performances of the social network regularized matrix factorization models with three different edge weighting techniques: (a) unweighted edges; (b) cosine weighted edges; (c) influence weighted edges. For all three models, we gradually increase the value of the parameter C_3 from 0 to 0.00032, thus imposing increasingly stronger social network regularization.

Since the social network regularization function only works on the user parameters, we choose the user-based sequential matrix factorization with each time step equal to 15 days as the underlying model, on which the effect of regularizing user parameters should be most significant. The parameter C_1 and C_2 are set to 0.002 and 0.0015 respectively, which is a setting found to work well according to the results in the previous section.

The detailed results were plotted in Figure 6(a) and 6(b), which reveal the following findings.

—Social network regularization had completely different effects on the two time periods. For the Oscar week, the performance significantly improved by nearly 6% compared with the version without social regularization, whereas in the Christmas week introducing social regularization had only worsened the performance. Such contrary findings suggest that the strength of social influence appears to be occasion dependent. In particular, Oscar and Christmas are two occasions of rather distinct nature. Oscar is a pure movie related event, so it is of no surprise that the member of a movie related social network would actively interact with and influence each other during this period of time. On the other hand, Christmas is a holiday event when people would spend most of their time with family members so their choice in



(a) Oscar Week (b) Christmas Week

Fig. 6. Comparison of different edge weighting techniques for social network regularization.

movies would depend less on their personal preferences and the social network but depend more on the preferences of their family members or the mood of the movie (e.g., a comedy may be preferred over a horror movie during Christmas).

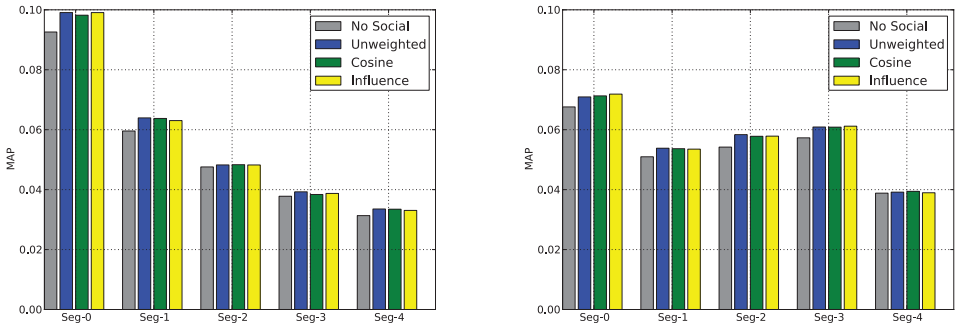
—Edge weighting did not appear to significantly affect the performance of the social network regularized matrix factorization model as the three lines corresponding to different weighting methods were very close to each other. The Filmtipset social network consists mostly of movie fans, rather than being a general social network like Facebook in which people would connect to each other for diverse reasons. This special nature of the Filmtipset social network may explain why it did not help much to assign weights to the edges in order to filter out spurious links.

7.5.2. Detailed Analysis. In order to further study for what kind of users is social network regularization most useful, we again try to partition users into different segments based on the following two criteria.

—*Activeness*. We measure the activeness of a user as the total number of ratings she has in the dataset.
—*Connectivity*. We measure the connectivity of a user as the number of friends she has in the social network.

We then rank users according to their activeness and connectivity respectively to form 5 equal sized segments, with segment 0 containing the least active/connected 20% users and segment 4 containing the most active/connected 20% users. Here we only present the analysis results obtained for the Oscar week dataset, on which social regularization was successful. The detailed results on different users segments shown in Figure 7 reveal the following interesting findings.

—In Figure 7(a), we can see that social network regularization is most helpful on the least active user segments (0-2). Since the number of ratings for these users is very limited, the knowledge about social network structure is therefore particularly useful for modeling these users' preference by correlating the parameters of connected users. On the other hand, for the very active users in segments 3 and 4 (i.e., users with many ratings), it appears that the users' own ratings are already quite sufficient to model his preference so incorporating social network information does not lead to any further improvement.
—From Figure 7(b), an interesting finding is that the social network regularization does not lead to much improvement for segment 4, which correspond to users with



(a) User Segments Defined by Activeness (b) User Segments Defined by Connectivity

Fig. 7. Performances of different social regularization methods on different user segments.

the highest connectivity (i.e., largest number of friends), whereas predictions for users with fewer friends tend to benefit more from social network regularization. This finding indicates that the fewer friends a user has in the Filmtipset network the stronger his preferences are correlated with these friends.

8. CONCLUSION AND FUTURE WORK

In this article, we described our solution to the context aware movie recommendation challenge (CAMRa 2010). We developed a social temporal collaborative ranking model that can simultaneously achieve three objectives: (1) combine both explicit and implicit user feedback. (2) support time awareness using an expressive sequential matrix factorization model and a temporal smoothness regularization function to tackle overfitting, and (3) support social network awareness by incorporating a network regularization term. Experiments on the CAMRa 2010 Filmtipset dataset demonstrated clear improvement over baseline methods. We have also conducted detailed analysis on the experimental results and obtained some interesting insights about context aware recommendation: (1) more granular temporal models would work best for old users and new items as well as for users and items with large number of ratings. (2) the effectiveness of social network regularization tends to be occasion dependent and tends to be more useful for users with small number of connections.

The current model has a number of parameters that had to be carefully tuned to achieve optimal performance. In the future, we will try to design more adaptive algorithms that could automatically select these parameters, for example, via hierarchical Bayesian models. Moreover, our analysis on the performance obtained for different user and item segments revealed that different parameters and model design choices are most effective for particular user and item segments. So in order to achieve the optimal performances across different user and item segments, we should not rely on a single model but on a collection of heterogeneous models with different combinations of features (i.e., time awareness, social network awareness) and parameter settings, and carefully choose the most appropriate subset of models to make predictions for a particular user/item under a certain context. In the future, we would like to formally realize this idea using a stacking approach, where the predictions of different models are combined with a supervised learning model. In particular, the input features to the stacking model will not only consist of model predictions but also of a set of meta-features describing the user and item (e.g., popularity, age, activeness). Such a stacking model would then be able to make informed decisions about which models to choose given the different instances described by the meta features.

REFERENCES

ANAGNOSTOPOULOS, A., KUMAR, R., AND MAHDIAN, M. 2008. Influence and correlation in social networks. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (KDD'08)*. ACM Press, 7–15.

BREESE, J. S., HECKERMAN, D., AND KADIE, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*. 43–52.

BURGES, C., SHAKED, T., RENSHAW, E., LAZIER, A., DEEDS, M., HAMILTON, N., AND HULLENDER, G. 2005. Learning to rank using gradient descent. In *Proceedings of 22nd International Conference on Machine Learning (ICML'05)*. 89–96.

HERLOCKER, J., KONSTAN, J., TERVEEN, L., AND RIEDL, J. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22*, 5–53.

HERLOCKER, J., KONSTAN, J., AND RIEDL, J. 2002. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retrieval 5*, 4, 287–310.

HOFMANN, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst. 22*, 1, 89–115.

HU, Y., KOREN, Y., AND VOLINSKY, C. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*. 263–272.

JAMALI, M. AND ESTER, M. 2009. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. ACM, New York, NY, 397–406.

JAMALI, M. AND ESTER, M. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys'10)*. ACM, New York, NY, 135–142.

KARATZOGLOU, A., AMATRIAIN, X., BALTRUNAS, L., AND OLIVER, N. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys'10)*. ACM, New York, NY, USA, 135–142.

KOREN, Y. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. ACM, New York, NY, 447–456

KOREN, Y. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data 4*, 1, 1–24.

KOREN, Y., BELL, R. M., AND VOLINSKY, C. 2009. Matrix factorization techniques for recommender systems. *IEEE Comput. 42*, 8, 30–37.

LINDEN, G., SMITH, B., AND YORK, J. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput. 7*, 1, 76–80.

LIU, N. N., CAO, B., ZHAO, M., AND YANG, Q. 2010. Adapting neighborhood and matrix factorization models for context aware recommendation. In *Proceedings of the Workshop on Context-Aware Movie Recommendation (CAMRa'10)*. ACM, New York, NY, 7–13.

LIU, N. N. AND YANG, Q. 2008. Eigenrank: a ranking-oriented approach to collaborative filtering. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 83–90.

LIU, N. N., ZHAO, M., AND YANG, Q. 2009. Probabilistic latent preference analysis for collaborative filtering. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'09)*. ACM, New York, NY, 759–766.

MA, H., KING, I., AND LYU, M. R. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*. ACM, New York, NY, 203–210.

MA, H., ZHOU, D., LIU, C., LYU, M. R., AND KING, I. 2011. Recommender systems with social regularization. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining (WSDM'11)*. ACM, New York, NY, 287–296.

MARDEN, J. I. 1995. *Analyzing and Modeling Rank Data*. Chapman & Hall, New York.

MARLIN, B. M. AND ZEMEL, R. S. 2007. Collaborative filtering and the missing at random assumption. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*.

PAN, R. AND SCHOLZ, M. 2009. Mind the gaps: Weighting the unknown in large-scale one-class collaborative filtering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. ACM, New York, NY, 667–676.

PENNOCK, D. M., HORVITZ, E., LAWRENCE, S., AND GILES, C. L. 2000. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'00)*. 473–480.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. 2009. L.s.: Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*.

Rennie, J. D. M. and Srebro, N. 2005. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*. ACM, New York, NY, 713–719.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'94)*. 175–186.

Said, A., Berkovsky, S., and De Luca, E. W. 2010. Putting things in context: Challenge on context-aware movie recommendation. In *Proceedings of the Workshop on Context-Aware Movie Recommendation (CAMRa'10)*. ACM, New York, NY, 2–6.

Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'01)*. 285–295.

Steck, H. 2010. 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 (KDD'10)*. ACM, New York, NY, 713–722.

Weimer, M., Karatzoglou, A., Le, Q. V., and Smola, A. 2008. Cofi rank maximum margin matrix factorization for collaborative ranking. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., and Sun, J. 2010. Temporal recommendation on graphs via long- and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*. ACM, New York, NY, USA, 723–732.

Xiong, L., Chen, X., Huang, T.-K., Schneider, J., and Carbonell, J. G. 2010. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In *Proceedings of the SIAM International Conference on Data Mining (SDM'10)*.

Zhou, K., Xue, G.-R., Zha, H., and Yu, Y. 2008. Learning to rank with ties. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. ACM, New York, NY, 275–282.

Received December 2010; revised April 2011, August 2011, September 2011; accepted September 2011