Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Modeling Users' Dynamic Preference for Personalized Recommendation

Xin Liu
Institute for Infocomm Research (I^2R)
Singapore
liu-x@i2r.a-star.edu.sg

Abstract

Modeling the evolution of users' preference over time is essential for personalized recommendation. Traditional time-aware models like (1) time-window or recency based approaches ignore or deemphasize much potentially useful information, and (2) time-aware collaborative filtering (CF) approaches largely rely on the information of other users, thus failing to precisely and comprehensively profile individual users for personalization. In this paper, for implicit feedback data, we propose a personalized recommendation model to capture users' dynamic preference using Gaussian process. We first apply topic modeling to represent a user's temporal preference in an interaction as a topic distribution. By aggregating such topic distributions of the user's past interactions, we build her profile, where we treat each topic's values at different interactions as a time series. Gaussian process is then applied to predict the user's preference in the next interactions for top-*N* recommendation. Experiments conducted over two real datasets demonstrate that our approach outperforms the state-of-the-art recommendation models by at least 42.46% and 66.14% in terms of precision and Mean Reciprocal Rank respectively.

1 Introduction

In real-world recommendation applications such as Netflix's movie recommendation and Amazon's product recommendation, users' preferences are drifting over time. It is thus essential to capture a user's dynamic preference to provide timely personalized recommendation. In contrast to traditional recommendation scenarios that look at global pattern change, temporal recommendation focuses on local models. Logically, modeling users' dynamic preference requires to address two challenges: (i) precise preference representation and user profile building, (ii) accurate preference evolution inference.

Recently, a set of time-aware recommendation models have been proposed, which give more weight to recent observations [Ding and Li, 2005; Liu *et al.*, 2010], or incorporated temporal effects into latent factor models [Koren, 2009; Xiong *et al.*, 2010; Nguyen *et al.*, 2014]. However, recency based approaches only focus on partial observations thus deemphasizing much signal which might also reflect users' preference (e.g., seasonal/periodic preference) [Aly *et al.*, 2013]. Moreover, most collaborative filtering (CF) models primarily rely on *similar* users/items' information to combat data sparsity, thus failing to build personalized models for individual users who have rich past information [Koren, 2009; Liu and Aberer, 2014; Li *et al.*, 2011]. On the other hand, conventional content-based recommendation models typically utilize concrete contexts such as genres and actors of a movie to build users' profiles, thus suffering from the issue of serendipity (over-specification) [Debnath *et al.*, 2008].

In order to address these issues, we propose a personalized recommendation model that captures users' dynamic preference by applying Gaussian process (GP) [Rasmussen and Williams, 2005]. In contrast to most approaches that handle explicit feedback data (e.g., 5-point scale rating), which is often not available, we focus on implicit feedback data (e.g., listened to a song, clicked an advertisement) that is more pervasive in online applications [Hu *et al.*, 2008]. Our model is constructed on top of several building blocks. Firstly, we assume each item is associated with texts such as its description[1]. A user u may also generate textual contents (e.g., tags) in interactions. Based on such texts, we apply Latent Dirichlet Allocation (LDA) [Blei *et al.*, 2003] to extract a set of topics to represent u's temporal preference (i.e., topic distribution) at the time point when an interaction happened. By aggregating topic distributions and the corresponding timestamp of u's past interactions, we build her profile that records the evolution of her preference. ates the issue of serendipity compared to traditional content based models that use concrete feature vector to represent preference.

Secondly, for user u, we treat the evolution of *each* topic as a time series. We then apply GP, one of the most advanced methods for modeling time series to build a regression model to predict u's future preference, i.e., topic distributions in the next interactions. Once the future preference is inferred, we compute the Jensen-Shannon divergence between u's future topic distribution and the topic distribution of each item candidate to provide top-*N* recommendation.

Thirdly, our GP based model is particularly designed for

[1]This assumption holds in most online applications like e-commerce,review sites, Q&A systems, social media, etc.

users with rich past interactions. In order to handle data sparsity which is common in recommender systems, we propose two heuristic solutions: (1) *profile average*, which predicts a user's future preference by averaging the topic distributions of her past interactions. (2) *user-based CF*, which predicts a user's preference by combining the GP derived preference of other similar users.

The major contribution of this work is to apply GP to capture users' dynamic preference which is represented by a topic distribution. Such a representation not only comprehensively measures users' preference when explicit feedback data is absence, but also encodes the flexibility and hence alleviates the issue of serendipity (compared to the content-based recommendation models that rely on concrete features to represent preference). To the best of our knowledge, this is the first work that applies GP to model users' evolving preference for implicit feedback data to provide top-N recommendation. We conduct experiments using two real datasets to demonstrate the performance of our approach by comparing with the state-of-the-art models.

2 Related Work

2.1 Time-aware Recommendation.

Traditional neighborhood-based models tackle temporal effects by assigning more importance to recent observations or focusing on information in specific time windows. For instance, Xiong et al. [Ding and Li, 2005] proposed to utilize a decay function $f(x) = e^{-\alpha t}$ to quantize the effect of users' past ratings, where t is the time at which a rating was given and α is the decay rate which controls the rate of ignoring the effect of a past rating. In [Liu *et al.*, 2010], a similar decay function was used for both similarity computation and rating prediction. An incremental algorithm was proposed to update neighborhood similarity by incorporating new data. In [Lathia *et al.*, 2009], the authors used user-based CF to provide time-aware recommendations by temporally adjusting the size of users' nearest neighbors based on the accuracy measured up to the current time.

For latent factor models, Koren [Koren, 2009] tackled temporal effects by using time functions to model user biases $b_u(t)$, item biases $b_i(t)$, as well as users' latent factors $p_u(t)$ in matrix factorization (MF). The predicted rating is derived as $\hat{r}_{u,i}(t) = \mu + b_u(t) + b_i(t) + q_i^T p_u(t)$, where μ is the global bias and q_i is item i's latent factors. In [Xiong *et al.*, 2010], the authors split time into equal time intervals and added the time dimension to user-item-rating matrix to form a 3-dimension tensor. Alternating least square algorithm is applied to optimize the tensor model. Lu et al. [Lu *et al.*, 2009] proposed a spatio-temporal model where the spatial component measures the correlation across factors of users and items, and the temporal component captures the change of latent factors. Kalman filtering is applied for model estimation. Most CF based models are designed to handle explicit feedback data and predict ratings, while our approach focuses on implicit feedback data and provide top-N recommendation, which is more practical in real-world applications.

Xiang et al. [Xiang *et al.*, 2010] proposed session based temporal graph (STG) to simultaneously model users' long-term and short-term preferences. An extension based on personalized random walk was proposed to model impacts of long-term and short-term preferences for time-aware recommendation. There are some methods like [Li *et al.*, 2012] that apply Hidden Markov Model to learn users' dynamic preference. The effectiveness of such approaches largely depends on proper state definition, which is non-trivial (e.g., rely on side information such as category).

Recently, a series of solutions were proposed to particularly process the highly dynamic data stream in social media like Twitter [Diaz-Aviles *et al.*, 2012; Chen *et al.*, 2012]. However, these approaches cannot be directly applied to more generic scenarios in that they only process recent social updates without considering the longstanding information.

2.2 Gaussian Process based Preference Modeling.

In [Adams *et al.*, 2010], the authors proposed a framework to incorporate side information by coupling multiple probabilistic MF via GP priors, where the latent features are replaced by latent feature functions. Platt et al. [Platt *et al.*, 2002] applied GP for regression to learn a user's preference over music. An algorithm, Kernel Meta-Training (KMT) was proposed to derive a kernel from a set of meta-training functions which share the same function distribution with the final training function. In [Houlsby *et al.*, 2012], GP and CF were combined to learn pair-wise preferences expressed by multiple users. Specifically, the task of learning users' preference was treated as a binary classification with GP where a preference kernel is used. Bonilla et al. [Bonilla *et al.*, 2010] focused on generalizing the knowledge of known users to infer the preference of unknown users. GP prior is applied over users' latent utility functions to learn the similarity of users' preference which can be used to aid in the elicitation process for a new user. However, these approaches fail to capture dynamics when modeling users' preference.

3 Our Approach

3.1 Personalized Recommendation Model.

We denote user set by $\mathcal{U} = \{u_1, u_2, ..., u_m\}$, and item set by $\mathcal{V} = \{v_1, v_2, ..., v_n\}$. By aggregating interactions with items, each user $u \in \mathcal{U}$ maintains a profile ρ_u, recording her experience with the system. We assume each item v is associated with a "bag-of-words", which is constructed from the relevant textual contents such as its description. Moreover, a user may also generate texts summarizing or indicating her experience in the interaction (e.g., reviews in Amazon, tips in Foursquare). We denote user u's past interactions by $\Psi_u = \{\psi_u^1, \psi_u^2, ...\}$. For each interaction $\psi_u^i \in \Psi_u$, we denote the associated texts by T_u^i. The summation of these texts constructs a corpus, which can be used for topic modeling.

User Profiling.

Probabilistic topic models such as LDA have been applied to extract and represent users' preference in different application scenarios, e.g., Web search and recommendation [Harvey *et al.*, 2013; Agarwal and Chen, 2010]. In this work, we

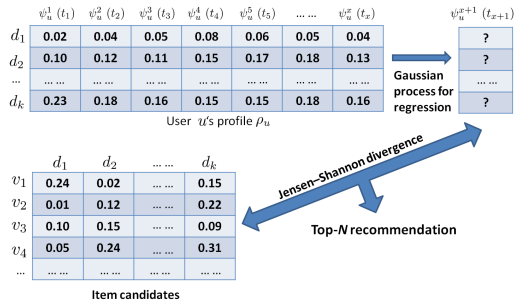


Figure 1: Users' dynamic preference modeling and recommendation based on GP. The upper left table gives an example of user u's profile ρ_u, which records the topic distributions of her past interactions and the corresponding timestamp.

follow this trend to profile users by applying LDA[2].

Every interaction ψ_u^v between user u and item v is associated with textual content T_u^v. We treat each such textual information as a document, and the aggregation of textual representations of all users' interactions form a text corpus $\mathcal{T}$, based on which, we perform LDA to extract k topics $D = \{d_1, ..., d_k\}$ such that each word w has a probability $\phi_{w,d}$ of being assigned to topic d, and each document (e.g., interaction T_u^v) is represented by a topic distribution θ_u^v.

By leveraging topic representation, we build user u's profile ρ_u as follows: The interactions (with timestamp) are sorted in chronological order. Each interaction between user u and item v is represented by a topic distribution θ_u^v. For each topic, we model its distribution with time as a time series. The basic idea of our approach is to predict the value of each topic at a given time by applying GP, and then recommend the items whose topic distributions are the most similar to the predicted topic distribution. Fig. 1 (upper left table) shows an example of user u's profile.

Dynamic Preference Modeling.
Given user u's profile ρ_u, we apply GP to predict her future preference, i.e., topic distributions of the items that user u is likely to interact with in the next interactions. For each topic $d_j \in D$, a series of its values $\mathfrak{D}_j = \{\mathfrak{d}_{j,1}, \mathfrak{d}_{j,2}, ..., \mathfrak{d}_{j,x}\}$ with the corresponding timestamp $\mathfrak{T} = \{t_1, t_2, ..., t_x\}$ (in chronological order) forms the training data. We define the function f that maps the timestamp $\mathfrak{T}$ to the values of topics $\mathfrak{D}$: $f : \mathfrak{T} \rightarrow \mathfrak{D}$. To take into account the noise that is common in practice (e.g., an item's textual description may not completely reflect its underlying properties), we assume additive independent identically distributed Gaussian noise ϵ with variance σ^2: $\mathfrak{D}_j = f(\mathfrak{T}) + \epsilon$. The purpose of GP regression is to infer the distribution of function f to predict the values $\mathfrak{D}'_j = \{\mathfrak{d}_{j,x+1}, \mathfrak{d}_{j,x+2}, ...\}$ of d_j at later time points $\mathfrak{T}' = \{t_{x+1}, t_{x+2}, ...\}$. The joint distribution of the observed values and the test values to be predicted is obtained by:

$$\begin{bmatrix} f \\ f' \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \mu(\mathfrak{T}) \\ \mu(\mathfrak{T}') \end{bmatrix}, \begin{bmatrix} K(\mathfrak{T},\mathfrak{T}) + \sigma^2 \mathrm{I} & K(\mathfrak{T},\mathfrak{T}') \\ K(\mathfrak{T}',\mathfrak{T}) & K(\mathfrak{T}',\mathfrak{T}') \end{bmatrix}), \quad (1)$$

[2]Remind that our model is designed for implicit feedback data, so explicit feedback like 5-point rating is not considered for preference representation and profile building.

where I is the identity matrix, $\mu(.)$ is the mean function and K is the covariance matrix where the (r, c)th element of K represents the covariance evaluated at time point t_r and t_c, i.e., $k(t_r, t_c)$. Accordingly, the elements of $K(\mathfrak{T},\mathfrak{T})$, $K(\mathfrak{T},\mathfrak{T}')$ and $K(\mathfrak{T}',\mathfrak{T}')$ represent the covariances evaluated at all time point pairs from $\mathfrak{T}$, $\mathfrak{T}$ and $\mathfrak{T}'$, and $\mathfrak{T}'$ respectively.

By conditioning the joint Gaussian prior distribution on observations, we derive the probability of f', which follows Gaussian distribution:

$$f'|f, \mathfrak{T}, \mathfrak{T}' \sim \mathcal{N}(\mu(\mathfrak{T}') + K(\mathfrak{T}',\mathfrak{T})(K(\mathfrak{T},\mathfrak{T}) + \sigma^2 I)^{-1}(f - \mu(\mathfrak{T})), K(\mathfrak{T}',\mathfrak{T}') - K(\mathfrak{T}',\mathfrak{T})(K(\mathfrak{T},\mathfrak{T}) + \sigma^2 I)^{-1}K(\mathfrak{T},\mathfrak{T}')). \quad (2)$$

Therefore, the best estimate of f' is the mean of its distribution:

$$\bar{f}' = \mu(\mathfrak{T}') + K(\mathfrak{T}',\mathfrak{T})(K(\mathfrak{T},\mathfrak{T}) + \sigma^2 I)^{-1}(f - \mu(\mathfrak{T})), \quad (3)$$

and the corresponding variance can be used to measure the confidence of the prediction.

In order to predict the next value of topic d_j, it is important to define the proper mean function $\mu(.)$ and covariance function $k(.,.)$. For mean function, we use a simple linear function which is parameterized by the weight a and the normalization constant b: $\mu(t) = at + b$.

Covariance function is the core of GP. By considering the complexity of the topic time-series (e.g., the smoothness varies with inputs), we choose a well known non-stationary function called *Neural Network Covariance Function*:

$$k(t, t') = h^2 \sin^{-1}\left(\frac{(1 + tt')/\lambda^2}{\sqrt{(1 + (1 + t^2)/\lambda^2)(1 + (1 + t'^2)/\lambda^2)}}\right) \quad (4)$$

where h^2 and λ are hyperparameters that control the scale (or variance) of the function output and the length-scale of the input time respectively.

Once the mean function and the covariance function are determined, we use Eq. 3 to calculate values of all topics D (i.e., user u's temporal preference) at a future time point t' that is of interest: $\theta_{u,t'} = \{\mathfrak{d}_{1,t'}, \mathfrak{d}_{2,t'}, ..., \mathfrak{d}_{k,t'}\}$. Note that the final topic distribution is obtained by normalizing the predicted topic values such that $\sum_{i=1}^{k} \mathfrak{d}_{i,t'} = 1$.

Making recommendation. To recommend items to user u, we calculate the (symmetric) Jensen-Shannon divergence between user u's temporal preference $\theta_{u,t'}$ and the topic distribution θ_v of each item candidate v:

$$D_{JS}(\theta_{u,t'} \parallel \theta_v) = \frac{1}{2} D_{KL}(\theta_{u,t'} \parallel \bar{\theta}) + \frac{1}{2} D_{KL}(\theta_v \parallel \bar{\theta}), \quad (5)$$

where $\bar{\theta} = \frac{(\theta_{u,t'} + \theta_v)}{2}$ and $D_{KL}(p_1 \parallel p_2)$ is the Kullback-Leibler divergence:

$$D_{KL}(p_1 \parallel p_2) = \sum_i \ln\left(\frac{p_1(i)}{p_2(i)}\right) p_1(i). \quad (6)$$

Top-N recommendation is generated by sorting items in ascending order of the derived Jensen-Shannon divergence between their topic representations and the user's inferred preference. That is, the items that are most consistent with user u's temporal preference are recommended. The entire process is depicted in Fig. 1.

Model Fitting.
GP regression works in practice under the assumption that the associated hyperparameters are properly chosen. In our personalized recommendation model, the hyperparameters include the parameters of mean function, covariance function, as well as the encoded noise: $\Theta = \{a, b, h, \lambda, \sigma\}$. In this subsection, we discuss how to estimate these hyperparameters in the light of the training data.

According to the definition of GP, the distribution of the data follows multivariate normal distribution:

$$p(\mathfrak{D}|\mathfrak{T}, \Theta) = (2\pi)^{-\frac{x}{2}} |K|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathfrak{D}-\mu)^T K^{-1}(\mathfrak{D}-\mu)}, \quad (7)$$

where x is the number of training cases. Then the (log) marginal likelihood is obtained by marginalizing over the latent function f:

$$\begin{aligned} L = \log p(\mathfrak{D}|\mathfrak{T}, \Theta) = -\frac{x}{2}\log 2\pi - \frac{1}{2}\log|K| - \\ \frac{1}{2}(\mathfrak{D}-\mu)^T K^{-1}(\mathfrak{D}-\mu). \end{aligned} \quad (8)$$

To find the hyperparameters, we maximize the marginal likelihood function L by seeking the partial derivatives of the marginal likelihood with respect to the corresponding hyperparameters $\Theta = \{a, b, h, \lambda, \sigma\}$ (see Eq. 9 and 10). Gradient based optimizer (e.g., conjugate gradient method) is then applied to find the best hyperparameters.

$$\frac{\partial L}{\partial \Theta_\mu} = -(\mathfrak{D}-\mu)^T K^{-1} \frac{\partial \mu}{\partial \Theta_\mu}, \quad (9)$$

where Θ_μ indicates the hyperparameters of mean function, i.e., a and b.

$$\begin{aligned} \frac{\partial L}{\partial \Theta_K} = \frac{1}{2}(\mathfrak{D}-\mu)^T K^{-1} \frac{\partial K}{\partial \Theta_K} K^{-1}(\mathfrak{D}-\mu) \\ -\frac{1}{2} tr(K^{-1}\frac{\partial K}{\partial \Theta_K}), \end{aligned} \quad (10)$$

where $tr(A)$ returns the trace of matrix A and Θ_K indicates hyperparameters of covariance function, i.e., h, λ, σ.

3.2 Handling Data Sparsity

Our personalized recommendation model is particularly designed for users with sufficient historical information such that a reliable GP regression model can be built. However, in real-world recommender systems, a large fraction of users may have limited past information (i.e., data sparsity) thus cannot benefit from our model. To handle this issue, we propose two heuristic solutions as follows.

Profile Average. Although user u has only a couple of interactions that are not sufficient for GP, we can simply calculate the average of topics distributions of her past interactions to represent her current preference:

$$\theta_u = \frac{\sum_{i=1}^{n_u} \theta_{T_i}}{n_u}, \quad (11)$$

where n_u is the number of u's past interactions. We then use θ_u to calculate the Jensen-Shannon divergence between u' preference and the topic representation θ_v of an item candidate v, based on which we provide top-*N* items that have the lowest Jensen-Shannon divergence.

Collaborative Filtering. Another way to handle data sparsity is to apply CF by leveraging the information of other similar users. We divide user base $\mathcal{U}$ into two parts: the users (denoted by $\mathcal{U}_C$) with complete profile for GP[3] and users (denoted by $\mathcal{U}_I$) whose profiles are incomplete for GP.

We first calculate the average of each user's topic distributions of her past interactions (see Eq. 11). Then for each user $u \in \mathcal{U}_I$, we calculate the similarity between user u and every user $u' \in \mathcal{U}_C$ based on the Jensen-Shannon divergence between their averaged preference:

$$s_{u,u'} = 1 - D_{JS}(\theta_u \parallel \theta_{u'}) \quad (12)$$

We apply user-based CF to derive u's temporal preference at time point t'. We assume that based on our personalized GP based model, we have derived the temporal preference $\theta_{u',t'}$ of each user $u' \in \mathcal{U}_C$ (see Eq. 3). The temporal preference of user u at time point t' is then calculated as: $\theta_{u,t'} = \frac{\sum_{i=1}^{N_u} s_{u,u_i}\theta_{u_i,t'}}{\sum_{i=1}^{N_u} s_{u,u_i}}$, where N_u is the number of selected users ($u_i \in \mathcal{U}_C$) that are the most similar to user u. The similarity is calculated using Eq. 12.

We will compare the performance of these two heuristic solutions: profile average and user-based CF in Section 4.

3.3 Discussion.

It is worth noting that we are not going to propose a novel topic model but rely on existing approaches. From this perspective, our model is flexible in that more sophisticated models such as dynamic topic models [Blei and Lafferty, 2006] and the ones for short texts [Yan *et al.*, 2013] can be applied. Note that besides timestamp, other temporal features such as hour-of-the-day and day-of-the-week may also help to infer users' temporal preference. Such features can be easily incorporated into GP by adding additional dimensions to the input $\mathfrak{T}$ (i.e., multi-dimension input for GP).

The time complexity of our model is mainly determined by the inversion of a matrix where the standard methods require time $\mathcal{O}(n^3)$ for a $n \times n$ matrix. This can be improved by applying faster matrix multiplication method such as Coppersmith-Winograd algorithm. Alternatively, approximation techniques like variational Bayesian inference might be applied to accelerate the learning process. It is worth mentioning that our approach builds recommendation models for individual users separately[4], i.e., there is no need to collect other users' information to train a model like factorizing a rating matrix. So our approach can be completely and safely parallelized to cater to large-scale datasets.

We also want to emphasize that in contrast to traditional content-based models which suffer from serendipity issue, by profiling users using probabilistic topic modeling, our approach alleviates the serendipity by taking into account the flexibility for preference representation, i.e., instead of concrete meta information, e.g., genres and actors of a movie, we use topic distribution, which is more flexible.

[3]The completeness of a user's profile is determined by multiple factors, e.g., a user has at least 10 interactions, or the variance of the inferred posterior distribution is smaller than a threshold.

[4]This statement holds true if profile average method (see Section 3.2) is applied to handle data sparsity issue.

4 Evaluation

4.1 Experimental Settings

Datasets

The first dataset we used was collected from Delicious (http://www.delicious.com). The data [Cantador *et al.*, 2011] contains 1,867 users and 69,226 URLs with 104,799 unique (user , URL) pairs. When bookmarking a URL, a user may also assign tags to the URL. There are 53,388 unique tags and 437,593 tag assignments (with timestamp). In the experiments, to conduct topic modeling, we treat each URL as a document, the associated tags as terms of the document. The purpose of a recommendation model is to recommend the URLs that the target user is likely to bookmark next time.

We also used the data collected from Last.fm (http://www.lastfm.com). The data [Cantador *et al.*, 2011] records the information of 1,892 users' listening history with 17,632 music artists. There are 92,834 unique (user , artist) pairs. When a user listen to an artist, she may assign tags to the artists. There are 11,946 unique tags and 186,479 tag assignments (with timestamp). Similar to Delicious data, we treat each artist as a document and the associated tags as terms for topic modeling. The purpose of a recommendation model is to recommend artists that the target user is likely to listen to.

For both datasets, we rank each user's interactions in chronological order. The first x interactions are used for training and the rest are for testing. We will demonstrate how the value of x influences the performance of our model.

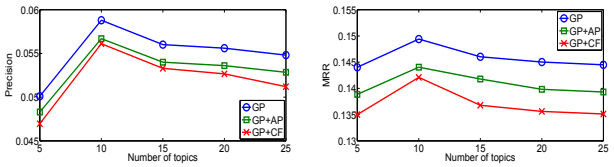
Comparison

We compare the proposed personalized recommendation model with several representative baselines: (1) *UCF-LDA*. This is an extension of a user-based CF where the similarity is measured based on the Jensen-Shannon divergence between two users' topic represented preference. We adopted the incremental algorithm proposed in [Liu *et al.*, 2010] to timely update neighborhood (with size of 50) similarity. (2) *BaseMF*. This is the basic MF that predicts a user's preference on an item. Top-*N* recommendation is provided by sorting the candidate items in descending order of the predicted preference. (3) *WRMF*. This is a representative top-N recommendation model for implicit feedback data [Hu *et al.*, 2008]. Count data is modeled as the confidence of preference inference in MF. (4) *timeSVD++*. This is a representative time-aware model incorporating dynamics proposed in [Koren, 2009]. Specifically, user bias, item bias and user latent factors are modeled as functions of time.

The parameters of MF based models such as latent factor vector dimensionality, learning rate, regularization terms are determined by 5-fold cross validation. For each model, we ran experiments 10 times and show the averaged results. We also conducted t-tests to confirm that all results are statistically significant (two-tailed, paired t-test, p-values < 0.001).

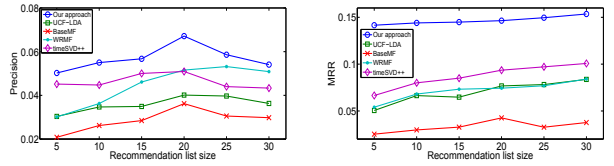
Metrics

We measure the performance of recommendation models using precision@N, which is the ratio of the successfully predicted test items to the top-N recommendation. We also use Mean Reciprocal Rank (MRR), a popular ranking metric to measure recommendation quality by finding out how far from the top of the list the first successfully predicted item is (averaged over all test cases): $MRR = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{R_i}$, where R_i is the position of the first successfully predicted item in the recommendation list returned for the ith user.

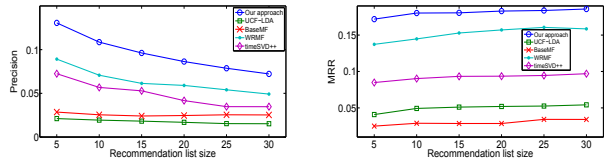


(a) Precision@10. (b) MRR@10.

Figure 2: Influence of topic number (Delicious data).



(a) Precision (Delicious). (b) MRR (Delicious).



(c) Precision (Last.fm). (d) MRR (Last.fm).

Figure 3: Influence of different recommendation list size.

4.2 Experimental Results

Design Validation.

We use GP based approach to learn dynamic preference of users who have at least 10 interactions. The hyperparameters h, λ and σ are initialized to 1. Fig. 2 shows the performance of our approach when Delicious data[5] is used. For active users, our approach produces remarkably high precision and MRR, demonstrating the advantage of GP based model. When data sparsity handling strategies are adapted to cope with inactive users whose number of past interactions is less than 10, the overall performance decreases. An interesting phenomenon is that the simple GP+AP (i.e., profile average) outperforms the more complicated GP+CF (i.e., user-based CF). This is probably due to the characteristics of the data: users do not significantly change their preference (although may vary within a certain scope), so the simple GP+AP works well while GP+CF may introduce noises from other users. However, we still want to keep GP+CF because it can help to alleviate over-specification in our content based model, thus might be more suitable in other scenarios. In the following experiments where data sparsity handling strategy is needed, we use GP+AP for our approach.

[5]The Last.fm data based experiments, which show the similar results, are not presented due to space limit.

Table 1: Influence of the volume of training data (top-10 recommendation)

training vol.	Delicious data			Last.fm data		
	Precision	MRR	Coverage	Precision	MRR	Coverage
5	0.0561	0.1399	95.07%	0.1542	0.3234	60.62%
10	0.0588	0.1494	91.16%	0.1635	0.3410	44.56%
15	0.0599	0.1502	88.70%	0.1675	0.3502	36.79%
20	0.0615	0.1534	85.75%	0.1689	0.3577	31.45%

Table 2: Performance comparison (averaged over different recommendation list sizes)

	Delicious data		Last.fm data	
	Precision	MRR	Precision	MRR
Our approach	**0.0573**	**0.1495**	**0.0985**	**0.1851**
BaseMF	0.0287	0.0367	0.0256	0.0299
timeSVD++	0.0464	0.0872	0.0489	0.0922
WRMF	0.0447	0.0718	0.0627	0.1492
UCF-LDA	0.0360	0.0701	0.0177	0.0501

Fig. 2 also shows the performance of our approach and its variants with different number of topics for preference representation. The general trends are both precision and MRR first increase, when arriving at a certain point, they start decreasing with the increasing number of topics. Specifically, for both datasets, 10 topics achieve the highest precision and MRR. In the following experiments, we use 10 topics for all top modeling based models.

Next, we study the influence of the volume of training data on the performance of our GP based recommendation model. Table 1 summarizes precision@10, MRR@10 and coverage (fraction of users that can be served by GP model) of our approach when different number of past interactions is used for training (ranging from 5 to 20 with 5 as the increment). Obviously, the more the training data, the higher the precision and MRR are. But on the other hand, more training data prevents more users from benefiting from the GP based model. For this set of experiments, we believe 5–10 past interactions is a reasonable choice. In the comparison studies (see the next section), for our approach, we use 10 as the threshold to distinguish active and inactive users that will be handled by GP based model and average profile strategy respectively.

Finally, we validate the choice of our covariance function, the core of GP. We compare the neural network (NN) covariance function with other well known ones: squared exponential (SE) covariance function and periodic (PD) covariance function. Experiments over Last.fm data show that NN improves SE by 5.98% and 10.60% in terms of precision and MRR respectively. Similar results are obtained using Delicious data. This is mainly because SE is infinitely differentiable (too smooth), which might be unrealistic in practice. PD performs worst among the three functions, which implies that users do not demonstrate evident periodic behavior. Note that we are not claiming that NN covariance function is always the best one, and the choice of covariance function should depend on characteristics of the data, but in our experiments, this is a judicious choice.

Comparison Study.

We compare the performance of our approach (GP+AP) with that of the state-of-the-art (see Section 4.1). Fig. 3 shows the precision and MRR of different recommendation models with different recommendation list sizes (range from 5 to 30 with 5 as the increment). With dynamic user similarity, UCL-LDA consistently outperforms baseMF when Delicious data is used. For Last.fm data, baseMF generates higher precision but lower MRR than UCL-LDA does. This proves that although baseMF has been widely used for rating prediction, it is not optimized for ranking task. By considering temporal effects, timeSVD++ evidently outperforms baseMF and UCF-LDA, demonstrating the importance of modeling users' dynamic preference. As the representative method for implicit feedback data, even though no temporal information is considered, WRMF outperforms timeSVD++ for Last.fm data, but was beaten for Delicious data. In all cases, our approach significantly outperforms other models due to three important designs: (1) precisely represent users' dynamic preference using topic modeling; (2) model users' dynamic preference using GP to accurately infer users' preference evolution for time-aware recommendation; (3) design effective topic model based data sparsity handling strategies to bootstrap users who have insufficient interaction data for GP.

We also summarize the performance of all recommendation models by averaging their precision and MRR when different recommendation list sizes are applied (see Tab. 2). Similar to previous results, UCF-LDA outperforms baseMF except for the case that is measured by precision for Last.fm data. WRMF and timeSVD++ perform similarly for Delicious data but WRMF is more advantageous when Last.fm data is used. In summary, by averaging the performance when different recommendation list sizes and different datasets are applied, our approach improves baseMF, timeSVD++, WRMF and UCF-LDA by 192.21%, 62.46%, 42.46% and 257.83% in terms of precision; 413.21%, 86.10%, 66.14% and 191.36% in terms of MRR.

5 Conclusion

In this paper, we model users' dynamic preference for recommendation by (1) applying LDA to represent a user's temporal preference; (2) treating the evolution of the user's preference (i.e., topic distribution) as time series which are learned and inferred using GP regression. In order to handle data sparsity issue, we propose two heuristics to derive the preference of users who has sparse interaction information. Experiments conducted over two real datasets show that our approach significantly outperforms the state-of-the-art models in terms of precision and MRR. As for the future work, we intend to in-

corporate other temporal features to better model users' preference evolution.

References

[Adams *et al.*, 2010] Ryan P. Adams, George E. Dahl, and Iain Murray. Incorporating side information into probabilistic matrix factorization using Gaussian processes. In *Proceedings of the 26th UAI*, 2010.

[Agarwal and Chen, 2010] Deepak Agarwal and Bee-Chung Chen. flda: matrix factorization through latent dirichlet allocation. In *Proceedings of the 3rd WSDM*, 2010.

[Aly *et al.*, 2013] Mohamed Aly, Sandeep Pandey, Vanja Josifovski, and Kunal Punera. Towards a robust modeling of temporal interest change patterns for behavioral targeting. In *Proceedings of the 22nd WWW*, 2013.

[Blei and Lafferty, 2006] David M. Blei and John D. Lafferty. Dynamic topic models. In *Proceedings of the 23rd ICML*, 2006.

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[Bonilla *et al.*, 2010] Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. Gaussian process preference elicitation. In *Proceedings of NIPS*, 2010.

[Cantador *et al.*, 2011] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems. In *Proceedings of the 5th RecSys*, 2011.

[Chen *et al.*, 2012] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th SIGIR*, 2012.

[Debnath *et al.*, 2008] Souvik Debnath, Niloy Ganguly, and Pabitra Mitra. Feature weighting in content based recommendation system using social network analysis. In *Proceedings of the 17th WWW*, 2008.

[Diaz-Aviles *et al.*, 2012] Ernesto Diaz-Aviles, Lucas Drumond, Lars Schmidt-Thieme, and Wolfgang Nejdl. Real-time top-n recommendation in social streams. In *Proceedings of the 6th RecSys*, 2012.

[Ding and Li, 2005] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th CIKM*, 2005.

[Harvey *et al.*, 2013] Morgan Harvey, Fabio Crestani, and Mark J. Carman. Building user profiles from topic models for personalised search. In *Proceedings of the 22nd CIKM*, 2013.

[Houlsby *et al.*, 2012] Neil Houlsby, Jose Miguel Hernandez-Lobato, Ferenc Huszar, and Zoubin Ghahramani. Collaborative gaussian processes for preference learning. In *Proceedings of NIPS*, 2012.

[Hu *et al.*, 2008] Yifan Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th ICDM*, 2008.

[Koren, 2009] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th SIGKDD*, 2009.

[Lathia *et al.*, 2009] Neal Lathia, Stephen Hailes, and Licia Capra. Temporal collaborative filtering with adaptive neighbourhoods. In *Proceedings of the 32nd SIGIR*, 2009.

[Li *et al.*, 2011] Ruijiang Li, Bin Li, Cheng Jin, Xiangyang Xue, and Xingquan Zhu. Tracking user-preference varying speed in collaborative filtering. In *Proceedings of the 25th AAAI*, 2011.

[Li *et al.*, 2012] Jialing Li, Li Li, Yuheng Wu, and Shangxiong Chen. An improved recommender based on hidden markov model. In *Proceedings of PRICAI*, 2012.

[Liu and Aberer, 2014] Xin Liu and Karl Aberer. Towards a dynamic top-n recommendation framework. In *Proceedings of the 8th RecSys*, 2014.

[Liu *et al.*, 2010] Nathan N. Liu, Min Zhao, Evan Xiang, and Qiang Yang. Online evolutionary collaborative filtering. In *Proceedings of the 4th RecSys*, 2010.

[Lu *et al.*, 2009] Zhengdong Lu, Deepak Agarwal, and Inderjit S. Dhillon. A spatio-temporal approach to collaborative filtering. In *Proceedings of the Third ACM Conference on Recommender Systems*, 2009.

[Nguyen *et al.*, 2014] Trung V. Nguyen, Alexandros Karatzoglou, and Linas Baltrunas. Gaussian process factorization machines for context-aware recommendations. In *Proceedings of the 37th ACM SIGIR*, 2014.

[Platt *et al.*, 2002] John C. Platt, Christopher J.C. Burges, Steven Swenson, Christopher Weare, and Alice Zheng. Learning a gaussian process prior for automatically generating music playlists. In *Proceedings of NIPS*, 2002.

[Rasmussen and Williams, 2005] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[Xiang *et al.*, 2010] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. Temporal recommendation on graphs via long- and short-term preference fusion. In *Proceedings of the 16th SIGKDD*, 2010.

[Xiong *et al.*, 2010] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of SDM*, 2010.

[Yan *et al.*, 2013] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd WWW*, 2013.