Collaborative Evolution for User Profiling in Recommender Systems

Zhongqi Lu[†], Sinno Jialin Pan[‡], Yong Li[♯], Jie Jiang[*], Qiang Yang[†]
[†]Hong Kong University of Science and Technology, Hong Kong
[‡]Nanyang Technological University, Singapore
[♯]VIPSHOP, China [*]Tencent, China
[†]{zluab,qyang}@cse.ust.hk, [‡]sinnopan@ntu.edu.sg,
[♯]pignju@gmail.com, [*]zeus@tencent.com

Abstract

Accurate user profiling is important for an online recommender system to provide proper personalized recommendations to its users. In many real-world scenarios, the user's interests towards the items may change over time. Therefore, a dynamic and evolutionary user profile is needed. In this work, we come up with a novel evolutionary view of user's profile by proposing a Collaborative Evolution (CE) model, which learns the evolution of user's profiles through the sparse historical data in recommender systems and outputs the prospective user profile of the future. To verify the effectiveness of the proposed model, we conduct experiments on a real-world dataset, which is obtained from the online shopping website of Tencent Inc.—*www.51buy.com* and contains more than 1 million users' shopping records in a time span of more than 180 days. Experimental analyses demonstrate that our proposed CE model can be used to make better *future* recommendations compared to several state-of-the-art methods.

1 Introduction

Many real-world recommender systems provide users with recommendations based on their interests. Therefore, to make proper personalized recommendations to a user, an accurate user profile that reflects the user's interests is crucial. Typically, user profiling is either knowledge-based or machine-learning-based [Middleton *et al.*, 2004]. Knowledge-based approaches make use of users' meta information, such as age, gender etc., to design some rules or patterns for recommendations. Machine-learning-based approaches build models to learn users' interests from users' historical behaviours automatically. The majority of current recommender systems adopt the machine-learning-based approaches for user profiling [Adomavicius and Tuzhilin, 2005]. One of the most popular approaches is based on matrix factorization (MF), which learns the *latent* interests of a user as the user's profile by collaboratively factorizing the rating matrix over historically recorded user-item preferences [Koren *et al.*, 2009].

[*]Corresponding author

Though the MF techniques in recommender systems are proven to be promising for user profiling, they have a major assumption that if a user shows interests in some items in the historical records, he/she should be always interested in these types of items in the future. In other words, current MF techniques assume users have constant interests towards the item set. This could only be true for the off-line experiments, in which training data and testing data are drawn uniformly from a static data set. However, in many real-world scenarios, a user's interests towards an item set can change over time [Lathia *et al.*, 2010]. Firstly, a user's interests towards some items can be diminishing. For example, a user may loss the interest in mobile phones if he/she has just purchased one. Secondly, some items may raise as new favorites for a user. For example, a new father may start to look for infant education courses, which he has never thought of before. Thirdly, users' interests on some items may vary upon time. For instance, thick coats may only get interests in the cold winters.

In order to model the interests of users, there are two keys: one is to quantitatively calculate the interests of users at certain time point; the other is to include the changes of interests in future predictions. As suggested by previous researches, users' static latent interests can be learned collaboratively from the sparse dataset in a particular time interval via the MF model. A straightforward solution is to perform an individual MF model to profile users' interests at each time point, and then discover evolutionary patterns by using time series analysis or sequential pattern mining. However, historical data at certain time point can be extremely sparse. In this case, the learned individual MF models may be very inaccurate, which result in the failure of the subsequent discovery of evolutionary patterns. In this paper, we propose a method named Collaborative Evolution (CE) to integrate MF and vector autoregressive [Lütkepohl, 2005] into a unified learning framework, where users' interests at each time point and their evolutions are learned simultaneously and collaboratively.

As an intuitive sketch of the CE model, iteratively we learn the users' latent interests in a collaborative filtering fashion, and then model the evolution of these latent interests to further guide the learning of the users' latent interests in the future. The collaborative learning and the evolutionary learning iteratively affect each other. The CE model outputs the predictions of users' evolutionary interests in the future. Then, based on the predictions of users' evolutionary interests as

user profiles, we make personalized recommendations for the users on certain day of the future. Here is a running example for the CE model to make recommendations: a man is observed to purchase roses in an online shopping system, and the CE model may learn the latent information that this man is possibly in love. Because collaboratively from all men's records, roses are associated to the latent status "in-love". After knowing men's "in-love" status, another information we could get from the training data is statistically what would be their *future* gifts (such as a ring) and when to buy? Then the CE model shall timely recommend a proper item to the man when it comes to the right time.

2 Preliminary

2.1 Notations and Problem Statement

We first introduce some general notations in this paper. Other specific notations of the proposed method will be further introduced in the subsequent sections. For a fixed time interval t, the browsing history is recorded by an $m \times n$ matrix $\mathbf{R}^t$, where m represents the number of users, n represents the number of items and $\mathbf{R}^t_{i,j}$ is the value of the cell (i, j) of $\mathbf{R}^t$, which can be either a missing value or a rating given by user i on item j. We use the superscript $^\top$ to denote the transport of a matrix. Given a sequence of user browsing behaviors of T time intervals, $\mathbf{R}^1, \mathbf{R}^2, \ldots, \mathbf{R}^T$, we aim to model the users' k-dimensional interests, $\mathbf{U}^t \in \mathbb{R}^{k\times m}$, in each of the time intervals, $t \in \{1, 2, \ldots, T\}$, where k is the parameter of the number of users' implicit interests, and explore their evolutionary processes. We finally make future recommendations to users in time intervals$\{T + 1, T + 2, \ldots\}$ based on their evolutionary interests. Note that each $\mathbf{R}^t$ is much sparser than the aggregation of all the $\{\mathbf{R}^t\}$'s. However, this aggregation makes the evolutionary information discarded.

2.2 Probabilistic Matrix Factorization

Recently, many MF methods have been proposed for CF in recommender systems. In this paper, we start from the probabilistic matrix factorization (PMF) method [Mnih and Salakhutdinov, 2007]. PMF aims to factorize a user-item rating matrix by maximizing the following conditional distribution over the observed ratings at the time interval t,

$$p(\mathbf{R}^t \mid \mathbf{U}^t, \mathbf{V}^t, \sigma^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}\left(\mathcal{N}(\mathbf{R}^t_{ij} \mid \mathbf{U}_i^{t\top}\mathbf{V}^t_j, \sigma^2)\right)^{I^t_{ij}}, \quad (1)$$

where $\mathbf{U}^t \in \mathbb{R}^{k\times m}$ is the latent factor matrix for users (i.e., users' k-dimensional interests) at time interval t, $\mathbf{V}^t \in \mathbb{R}^{k\times n}$ is the latent factor matrix for items (i.e., items' k-dimensional properties) at time interval t, $\mathcal{N}(x \mid \mu, \sigma^2)$ is the probability density function of the Gaussian distribution with the mean μ, and the variance σ^2, and I_{ij} is the indicator function which is equal to 1 if user i browsed item j in the time interval t, and 0 otherwise. Besides, we use a subscript of a matrix to denote the corresponding column of the matrix, e.g., $\mathbf{U}^t_i \in \mathbb{R}^{k\times 1}$ and $\mathbf{V}^t_j \in \mathbb{R}^{k\times 1}$ denote the vector for user i and the vector for item j respectively.

2.3 Model the Changes of User's Interests

A proper recommendation should not only rely on users' current interests, but also need to consider their evolutions. In online shopping scenarios, it is very likely that users change their interests before the next browsing on the shopping website. Therefore, it is crucial for online shopping websites to capture the changes in users' interests.

Intuitively, on one hand, an interest can be changing due to the trend of this particular interest. On the other hand, different interests may be related and affect each other. Therefore, a desirable estimation of an interest should consider both the history of the interest and the influence from the other interests. In light of the above intuition, we propose to model the user's latent interests into the vector autoregressive model [Lütkepohl, 2005].

Based on the vector autoregressive model, the feature vector of a user i in the time interval t can be modeled by the feature vectors of the user in the previous ϕ ($\phi < t$) time intervals, i.e., $t-1, \ldots, t-\phi$:

$$\mathbf{U}^t_i = \mathbf{A}^1_i\mathbf{U}^{t-1}_i + \mathbf{A}^2_i\mathbf{U}^{t-2}_i + \ldots + \mathbf{A}^\phi_i\mathbf{U}^{t-\phi}_i + \boldsymbol{\varepsilon}^t_i, \quad (2)$$

where $\boldsymbol{\varepsilon}^t_i$ is an uncorrelated Gaussian noise with a zero-mean and a covariance matrix $\mathbf{D}_i \in \mathbb{R}^{k\times k}$ to be estimated, and $\{\mathbf{A}^j_i \in \mathbb{R}^{k\times k}\}^\phi_{j=1}$ is the set of coefficient matrices to be learned, each matrix of which represents the correlation of features in each of the previous ϕ time intervals. After $\{\mathbf{A}^j_i \in \mathbb{R}^{k\times k}\}^\phi_{j=1}$ and $\mathbf{D}_i$ are learned, one can use them and the observed time series of the user' interests $\{\mathbf{U}^t_i\}^T_{t=1}$ to predict the user's future interests $\{\mathbf{U}^t_i\}_{t=T+1}$, recursively. In the following, we present how to estimate the parameters in (2) for a particular user.

Least Square Estimation with Fixed ϕ

Suppose that for a particular user i, a series of k-dimensional feature vectors of T time intervals $\{\mathbf{U}^t_i\}^T_{t=1}$ is available. The parameters $\{\mathbf{A}^t_i\}^\phi_{t=1}$ and $\mathbf{D}_i$ are to be estimated. For convenience in presentation, we introduce the operator $vec(\cdot)$, which transforms a matrix into a vectors by stacking the columns. Other notations used for parameter estimation are in Table 1.

Table 1: Notations

Symbol	Definition	Deminsion
$\mathbf{Y}_i$	$\left[\mathbf{U}^1_i, \ldots, \mathbf{U}^T_i\right]$	$k \times T$
$\mathbf{B}_i$	$\left[\mathbf{A}^1_i, \ldots, \mathbf{A}^\phi_i\right]$	$k \times k\phi$
$\mathbf{Z}^t_i$	$\left[\mathbf{U}_i^{t\top}, \ldots, \mathbf{U}_i^{t-\phi\top}\right]^\top$	$(k\phi) \times 1$
$\mathbf{Z}_i$	$\left[\mathbf{Z}^1_i, \ldots, \mathbf{Z}^T_i\right]$	$(k\phi) \times T$
$\boldsymbol{\delta}_i$	$\left[\boldsymbol{\varepsilon}^1_i, \ldots, \boldsymbol{\varepsilon}^T_i\right]$	$k \times T$
$\boldsymbol{y}_i$	$vec(\mathbf{Y}_i)$	$(kT) \times 1$
$\boldsymbol{\beta}_i$	$vec(\mathbf{B}_i)$	$(k^2\phi) \times 1$

To calculate $\{\mathbf{A}^t_i\}^\phi_{t=1}$, following the mathematical derivation in [Neumaier and Schneider, 2001], we obtain

$$\boldsymbol{\beta}_i = ((\mathbf{Z}_i\mathbf{Z}_i^\top)^{-1}\mathbf{Z}_i \otimes \mathbf{I}_k)\boldsymbol{y}_i, \quad (3)$$

where $\mathbf{I}_k$ is the $k \times k$ identity matrix, $\otimes$ denotes the Kronecker product. Based on the definition of $\boldsymbol{\beta}_i$, one can recover $\mathbf{B}_i$ and thus obtain $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$.

To calculate the covariance matrix $\mathbf{D}_i$,

$$\mathbf{D}_i = \frac{1}{T-\phi-1} \sum_{t=\phi+1}^{T} \boldsymbol{\eta}_i^t \boldsymbol{\eta}_i^{t\top}, \tag{4}$$

where $\boldsymbol{\eta}_i^t = \mathbf{U}_i^t - \sum_{l=1}^{\phi} \mathbf{A}_i^l \mathbf{U}_i^{t-l}$.

3 Collaborative Evolution

A primary problem in recommender systems is to know how likely a user is interested in an item on a future day. This likelihood could be measured by the probability that the user would browse the item on the particular day. Given historical records, e.g., browsing log, a desirable recommender system makes future recommendations based on the predictions of the user's interests on the item set. Note that the historical records can only reveal the users' historical interests. To make proper item recommendations in a future time, we need to infer the users' interests at the time when the recommendations are being made. In this section, we describe our proposed method for making recommendations based on users' evolutionary interests in detail.

3.1 Bootstrap

Before applying the vector autoregressive method to model the evolutionary process of users' interests, one needs to obtain an initial sequence of users' latent feature vectors. Given a series of user-item rating matrices of T time intervals, $\{\mathbf{R}^t\}_{t=1}^{T}$, we propose use the first T_0 ($T_0 \leq T$) rating matrices $\{\mathbf{R}^t\}_{t=1}^{T_0}$ to learn the initial sequence of factor matrices, $\{\mathbf{U}^t\}_{t=1}^{T_0}$, for users. Instead of learning the factor matrices independently, we further propose to learn them jointly by enforcing the factor matrices for items to be shared over different time intervals, i.e., $\mathbf{V}^1 = \ldots = \mathbf{V}^{T_0}$. In the sequel, we use $\mathbf{V}^0$ to denote the shared factor matrix for items over time. The motivations behind this are two-folds: 1) though users' interests are changing over time, the properties of items are relatively stable, and 2) since rating matrices in each time interval can be very spare, jointly factorizing all the matrices can obtain more precise factor matrices for both users and items. Therefore, in the bootstrap step, we aim to learn the initial factor matrices, $\{\mathbf{U}^t\}_{t=1}^{T_0}$ and $\mathbf{V}^0$ by solving the following optimization problem,

$$\max_{\{\mathbf{U}^t\}_{t=1}^{T_0}, \mathbf{V}^0} \sum_{t=1}^{T_0} \prod_{i=1}^{m} \prod_{j=1}^{n} \left(\mathcal{N}\left(\mathbf{R}_{ij}^t \mid \mathbf{U}_i^{t\top} \mathbf{V}_j^0, \sigma^2 \right) \right)^{I_{ij}^t},$$

where we place zero-mean spherical Gaussian priors [Dueck and Frey, 2004; Mnih and Salakhutdinov, 2007] on $\mathbf{V}^0$ and each $\mathbf{U}^t$.

We conduct theoretical study on how many time intervals should be used in the bootstrap step, i.e., how to set the value of T_0. In general, without considering any additional information, e.g., rating matrices in time intervals from T_0 to T, one can apply the Gaussian elimination approach to solve (2) with the initial sequence of user factor matrices, $\{\mathbf{U}^t\}_{t=1}^{T_0}$. However, to use Gaussian elimination accurately, the length of the initial sequence, T_0, is required to satisfy some constraint, which is shown in the following proposition.

Proposition 1. *Suppose the evolution is of order ϕ and the dimensionality of a user's feature vector is k, then the minimum number of the time intervals to be used in the bootstrap step is $k\phi$.*

Proof. According to (3), in order to avoid for underestimation when using Gaussian elimination, the number of rows in $\boldsymbol{y}$ should be greater than the number of rows in $\boldsymbol{\beta}$. Note that the number of rows in $\boldsymbol{y}$ is kT_0, where T_0 is the number of time intervals in the bootstrap step, and the number of rows in $\boldsymbol{\beta}$ is $k^2\phi$. $kT_0 \geq k^2\phi$ implies $T_0 \geq k\phi$. That is, the minimum number of time intervals in the bootstrap step is $k\phi$. The proof is completed. □

Above all, the Proposition 1 suggests that T_0 is required to satisfy $T_0 \geq k\phi$.

3.2 Collaborative Evolution

After the bootstrap step, we obtain the factor matrix for items, $\mathbf{V}^0$, and for each user i, we have an initial sequence of factor matrices $\{\mathbf{U}_i^t\}_{t=1}^{T_0}$ and an initial sequence of coefficient matrices, $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$. All that remains is the collaborative evolution (CE) step, which is to learn the factor matrices $\mathbf{V}^t$ and $\mathbf{U}^t$ in each time interval in $\{T_0+1, \ldots, T\}$, and update $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$ for each user i by performing PMF on the rating matrices, $\{\mathbf{R}^t\}_{t=T_0+1}^{T}$, and applying the vector autoregressive model on $\{\mathbf{U}_i^t\}_{t=T_0+1}^{T}$ recursively and simultaneously. The overall optimization problem to be solved can be written as follows,

$$\max_{\substack{\{\mathbf{A}_i^t\}_{t=1}^{\phi}, \\ \{\mathbf{U}^t, \mathbf{V}^t\}_{t=T_0+1}^{T}}} \sum_{t=T_0+1}^{T} \prod_{i=1}^{m} \prod_{j=1}^{n} \left(\mathcal{N}\left(\mathbf{R}_{ij}^t \mid \mathbf{U}_i^{t\top} \mathbf{V}_j^t, \sigma^2 \right) \right)^{I_{ij}^t}, \tag{5}$$

where the following two priors are placed on $\mathbf{U}^t$ and $\mathbf{V}^t$ respectively,

$$p(\mathbf{V}^t \mid \mathbf{V}^0, \sigma_v^2) = \prod_{j=1}^{n} \mathcal{N}(\mathbf{V}_j^t \mid \mathbf{V}^0, \sigma_v^2 \mathbf{I}_k), \tag{6}$$

$$p(\mathbf{U}^t \mid \mathbf{W}_i^t, \sigma_u^2) = \prod_{i=1}^{m} \mathcal{N}(\mathbf{U}_i^t \mid \mathbf{W}_i^t, \sigma_u^2 \mathbf{I}_k), \tag{7}$$

and $\mathbf{W}_i^t$ is recursively updated via

$$\mathbf{W}_i^t = \mathbf{A}_i^1 \mathbf{U}_i^{t-1} + \mathbf{A}_i^2 \mathbf{U}_i^{t-2} + \ldots + \mathbf{A}_i^{\phi} \mathbf{U}_i^{t-\phi}. \tag{8}$$

The prior in (6) is to constrain the new factorized latent matrix $\mathbf{V}^t$ for items not to be very different from $\mathbf{V}^0$, since the properties of items are assumed to be stable over time. The prior in (7) aims to use the evolutionary process to affect the shape of the new factorized latent matrix $\mathbf{U}^t$ for users.

As shown in [Mnih and Salakhutdinov, 2007], maximizing the log-posterior is equivalent to minimizing the sum-of-squared errors with quadratic regularization terms, the optimization problem in (5) can be rewritten into the following

Algorithm 1 Collaborative Evolution

Input: Users' browsing records $\{\mathbf{R}^t\}_{t=1}^T$, number of latent features k, number of evolution coefficients ϕ
Bootstrap:
for $t = 1$ to T_0 **do**
  $\mathbf{R}_{ij}^t \sim \mathbf{U}_i^{t\top}\mathbf{V}_j^0$
end for
Initialize $\{\mathbf{A}^t\}_{t=1}^{\phi}$ via (3).
Collaborative Evolution:
for $t = T_0 + 1$ to T **do**
  Update $\mathbf{W}_i$ via (8).
  while $\mathbf{U}^t$ and $\mathbf{V}^t$ are not converge **do**
    Update $\mathbf{U}^t$ and $\mathbf{V}^t$ via (10), respectively
  end while
  Update $\{\mathbf{A}^t\}_{t=1}^{\phi}$ via (3).
end for
Output: The evolution coefficients $\{\mathbf{A}^t\}_{t=1}^{\phi}$ and the feature vectors $\{\mathbf{U}^t\}_{t=1}^T$ for each user, and the shared factor matrices $\{\mathbf{V}^0\}$ for items.

equivalent form,

$$\min_{\substack{\{\mathbf{A}_i^t\}_{t=1}^{\phi},\\ \{\mathbf{U}^t,\mathbf{V}^t\}_{t=T_0+1}^{T}}} J = \sum_{t=T_0+1}^{T}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(I_{ij}^t\left(\mathbf{R}_{ij}^t - \mathbf{U}_i^{t\top}\mathbf{V}_j^t\right)^2 + \frac{\sigma^2}{\sigma_u^2}\left\|\mathbf{U}_i^t - \mathbf{W}_i^t\right\|^2 + \frac{\sigma^2}{\sigma_v^2}\left\|\mathbf{V}_j^t - \mathbf{V}_j^0\right\|^2\right) \quad (9)$$

Note that there are three types of parameters in CE, i.e., $\{\mathbf{U}^t\}$'s, $\{\mathbf{V}^t\}$'s and $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$. We propose to optimize the objective in (9) by alternatively updating $\{\mathbf{U}^t\}$'s, $\{\mathbf{V}^t\}$'s, and $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$. The updating rule for $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$ is given by Least Square Estimation in the previous section, and the updating rules for $\mathbf{U}^t$ and $\mathbf{V}^t$ in each time interval t are shown as follows respectively,

$$\mathbf{U}_i^t \leftarrow \mathbf{U}_i^t - \frac{\partial \mathcal{J}}{\partial \mathbf{U}_i^t}, \text{ and } \mathbf{V}_j^t \leftarrow \mathbf{V}_j^t - \frac{\partial \mathcal{J}}{\partial \mathbf{V}_j^t}, \quad (10)$$

where

$$\frac{\partial \mathcal{J}}{\partial \mathbf{U}_i^t} = -2\sum_{j=1}^{n}(I_{ij}^t\mathbf{V}_j^t(\mathbf{R}_{ij}^t - \mathbf{U}_i^{t\top}\mathbf{V}_j^t)) + 2\frac{\sigma^2}{\sigma_u^2}(\mathbf{U}_i^t - \mathbf{W}_i^t),$$
$$\frac{\partial \mathcal{J}}{\partial \mathbf{V}_j^t} = -2\sum_{i=1}^{m}(I_{ij}^t\mathbf{U}_i^t(\mathbf{R}_{ij}^t - \mathbf{U}_i^{t\top}\mathbf{V}_j^t)) + 2\frac{\sigma^2}{\sigma_v^2}(\mathbf{V}_j^t - \mathbf{V}_j^0).$$

The overall algorithm of CE is summarized in Algorithm 1.

3.3 Item Recommendations in Future time

With the evolution coefficients matrices $\{\mathbf{A}^t\}_{t=1}^{\phi}$, the shared factor matrix $\mathbf{V}^0$ for items, and the feature vectors $\{\mathbf{U}^t\}_{t=1}^T$ for each user learned in the previous section, to provide recommendations to users at a particular time interval $T+\lambda$, we first compute $\mathbf{W}^{T+\lambda}$ by recursively applying (8), and then make predictions on which items the user would be interested by computing the following ratings for each item,

$$\mathbf{R}_{ij}^{T+\lambda} = \mathbf{W}_i^{T+\lambda^\top}\mathbf{V}_j^0. \quad (11)$$

3.4 Analysis of the Evolution Coefficients

We use data from the first T_0 time intervals to estimate initial parameters in the bootstrap step, and data of the last $N = T - T_0$ time intervals to estimate the parameters of the CE model. Besides, we suggest the minimum value of T_0 to estimate accurate initial parameters. Now, we study the reliability of the evolution coefficients $\{\mathbf{A}^t\}_{t=1}^{\phi}$ in terms of N. Specifically, we fix all other parameters, i.e., $\{\mathbf{U}^t\}_t$ and $\{\mathbf{V}^t\}_t$, and analyze $\{\mathbf{A}_i^t\}_{t=1}^{\phi}$ for a particular user i. For the ease of notations, we omit the subscript i in the following analysis. To start with, we first define the covariance matrix estimate for $vec(\hat{\mathbf{B}})$ as,

$$\hat{\Sigma}_B = (\sum_{t=1}^{T}\mathbf{U}^t(\mathbf{U}^t)^\top)^{-1} \otimes \mathbf{D}, \quad (12)$$

where $\mathbf{D}$ is defined in (4). For convenience in presentation, we define the operator $[\cdot]_j$ as the j-th element in a vector, and $[\cdot]_{ij}$ as the (i,j) element of a matrix.

Following [Neumaier and Schneider, 2001], a confidence interval for $\psi = [vec(\mathbf{B})]_j$ of the parameter matrix $\mathbf{B}$ can be constructed from the distribution of the t-ratio, $t = \frac{\hat{\psi}-\psi}{\hat{\sigma}_\psi}$, where $\hat{\psi} = [vec(\hat{\mathbf{B}})]_j$, and $\hat{\sigma}_\psi^2 = [\hat{\Sigma}_B]_{jj}$.

Assume that the t-ratio follows Student's t-distribution, $t(\cdot)$, with N degrees of freedom. Then with the $(\alpha \times 100)\%$ confidence, the margin of error for a parameter estimation $\hat{\psi}$ is defined as

$$\hat{\psi}_\pm = \hat{\psi} - \psi = t\left(N, \frac{1+\alpha}{2}\right)\hat{\sigma}_\psi \quad (13)$$

where $\alpha \in [0,\ 1]$ and $t(d,\theta)$ is the θ-quantile of a t-distribution with d degrees of freedom.

Proposition 2. *The reliability of the CE Model is improved monotonously when T increases.*

Proof. Based on (13), $\hat{\psi}_\pm$ is defined by a t-distribution with $N = T - T_0$ degrees of freedom. When the number of time intervals T increases, the degrees of freedom N also increases. Moreover, a fixed $\alpha \times 100\%$ confidence interval for a t-distribution with larger degrees of freedom leads to smaller values of the margin of error of the parameter estimate. Therefore, the CE model is improved monotonously by decreasing the margin of error of the parameter estimate, when the number of time intervals for training is increasing. This completes the proof. □

4 Experiments

In this section, we conduct experiments on a real-world online shopping dataset to verify the effectiveness of CE in the recommender system.

4.1 The Dataset

The real-world dataset used for experiments is from a Chinese e-commerce website, *www.51buy.com*. The dataset covers all users' browsing records from April 2013 to September 2013. The long time span of the data makes it suitable to evaluate

the effect of changing in users' interests. For further evaluations and future researches, the full dataset and the code in the experiments are released at `http://zhongqi.me`.

4.2 Experimental Settings

A primary problem in the recommender system is to know how likely a user is interested in an item on a future day. This likelihood could be measured by the probability that the user tends to browse the item on a particular day. In our experiments, we make the duration of each time interval to be one day. From the browsing log, we first construct an user-item matrix for each day. The cell value of the matrix is the count of a user's daily browsing an item. We further normalize each user-item matrix such that for each user, the sum of the item browsing, if is not 0, is equal to 1. With these normalized user-item matrices, we apply our proposed CE model to make predictions for a particular future day.

For evaluation, we use the "Root Mean Square Error" (RMSE). The RMSE is widely adopted as the evaluation matric in the evaluation of recommender systems, such as the Netflix Prize [Bennett and Lanning, 2007]. In our settings, the RMSE indicates the difference between the true and predicted likelihood that a user browsing an item.

4.3 Performance Comparison

The comparison experiments are conducted on the dataset, which contains about 1 million users and more than $270,000$ items in a time span of 180 days. To show the effectiveness of our proposed method, we compare CE with the following baseline methods:

- *PMF* [Mnih and Salakhutdinov, 2007]. PMF is an effective MF method for missing value prediction. In order to demonstrate the effect of the changes in users' interests in the training data, we trained PMF with different training sets: "PMF_rcnt60d", "PMF_rcnt30d" and "PMF_rcnt15d", which use the most recent 60, 30 and 15 days' records in the training set respectively.
- *BPMF* [Salakhutdinov and Mnih, 2008]. BPMF is a fully Baysian treatment of the PMF models. Although the BPMF approach is computationally expensive than PMF, it usually leads to better performance.
- *timeSVD++* [Koren, 2009]. timeSVD++ tracks the changes throughout the life span of the data. This method is reported to achieve excellent performance on the Netflix dataset [Bennett and Lanning, 2007] by considering temporal dynamics.

We present the results of these baselines with fine-tuned parameters. Figure 1 shows the performance comparison.

Firstly, we aim to verify the claim that the historical data and the *future* data have different distributions. For the three PMF baselines, i.e. "PMF_rcnt15d", "PMF_rcnt30d", "PMF_rcnt60d", which use the recent 15, 30, 60 days of historical data for training respectively, we can observe that using the recent 30 days of historical data achieves best performance among the three. Empirically on this dataset, using 30 days of data happens to be a balance between the training data insufficiency (15 days of data) and the change of users' interests (60 days of data).

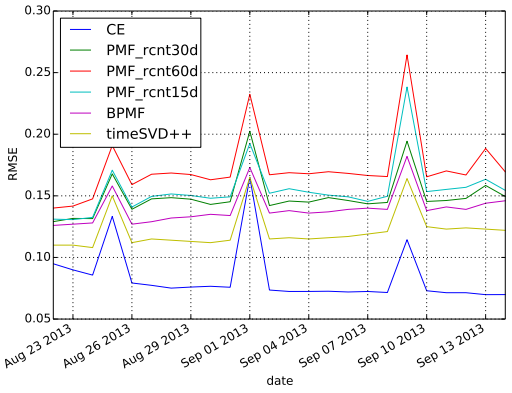

Figure 1: Performance Comparison. The training set contains browsing history of more than $270,000$ items from March 1, 2013 to August 21, 2013.

Secondly, our CE method achieves the best performance comparing with both BPMF and timeSVD++. Besides, we observe that both timeSVD++ and our CE methods demonstrate advantage when predicting in the future. This is because both methods are designed to tackle the temporal dynamics in recommender systems. Other than better performance in terms of prediction accuracy, our CE method is more stable timeSVD++ as the RMSE of the *CE* method would not raise in the long future.

Thirdly, all the methods have regular drop of performance in nearly every 7 days and these drops happen on the weekends. We find that on the weekends, the users usually browse different items, comparing to those they view on the weekdays. This suggests that we may need to consider a different model for user behaviors on the weekends, which is beyond the scope of this paper, and is a potential future extension.

4.4 Size of Time Intervals in Training

We have analyzed the confidence intervals of CE's parameter estimations, and conclude that the performance of CE can be improved monotonously when increasing the number of time intervals in training. In this section, we show the experimental results of varying the size T of time intervals in CE's training. Experimental results are shown in Figure 2.

In the experiments, we gradually increase the size of time intervals for training the CE model. "CE_5Train", "CE_15Train", "CE_30Train", and "CE_50Train" represent the training with the data from 5, 15, 30, and 50 consecutive days before testing respectively. For testing, we use data from August 22, 2013 onwards. On one hand, we empirically justify Proposition 2, which states that the reliability of the CE model increases monotonously as the size of the training time intervals increases. As can be seen in Figure 2, when the size of the training time intervals increases from 5 to 50, the RMSE of CE decreases for all future predictions. Because the decrease of RMSE reflects that the model better captures the changes of the user's interests, we conclude that the reliability of the model increases when the size of the training

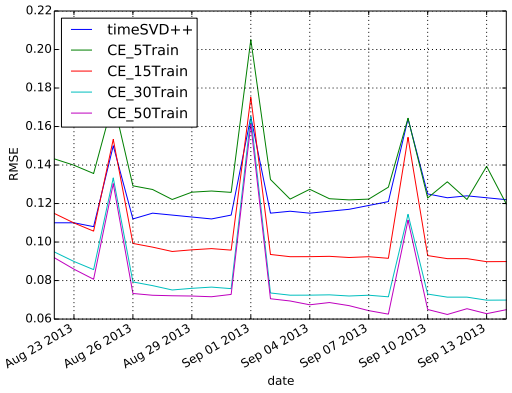

Figure 2: Different sizes T of the Time Intervals in Training. As T increases, the performance always becomes better and eventually approaches a limit.

time intervals increases. On the other hand, when the size T of the time intervals for training is larger than a certain value, i.e. 50, increasing T does not lead to significant decrease of RMSE. To balance the computational cost and the prediction accuracy, we propose to use 30 time intervals for training the CE model on our dataset.

5 Related Works

Our work is mostly related to collaborative filtering with temporal dynamics, concept drift and Markov decision process (MDP).

Collaborative Filtering is an approach of making automatic prediction (filtering) about the interests of a user by collecting interests from many users (collaborating). Some of the best results are obtained based on matrix factorization techniques [Marlin, 2003; Singh and Gordon, 2008; Koren *et al.*, 2009; Lu *et al.*, 2015]. However, most of collaborative filtering methods assumed that a user's interests towards a fixed item set do not change, which does not hold in most current online shopping scenarios.

Being aware of the change of users' interests, several recent works have attempted to integrate temporal dynamics into collaborative filtering methods. A major approach is by introducing a time decay function to weight the item set according to the data age [Koren, 2009; Liu *et al.*, 2010; Nakatsuji *et al.*, 2012]. In practice, the changes in users' interests are too complex to be modeled by a time decay function. The time decay function may only be able to approximate some short-term changes in users' interests, but fail to model the long-term ones. Besides, the time decay approaches are usually not able to explore when would users change their interests. Currently the temporal dynamics researches cover many specific scenario [Pálovics *et al.*, 2014; Liu and Aberer, 2014; Liu, 2015], and we are the first to explore the evolution of user's interest by mining a complete dataset from a real world recommender system.

Some works try to capture the changes of users' interests by taking sequential patterns into account. A major approach is by utilizing the state-based models, such as the Markov chain model [Rendle *et al.*, 2010; Zhao *et al.*, 2012; Wang and Zhang, 2013; McAuley and Leskovec, 2013]. However because the item set is extreme large in the online shopping scenario, it would usually not be possible to compute the transition functions over the item set effectively nor efficiently.

Concept drift is a phenomenon when the observed data is generated from a distribution that changes over time [Tsymbal, 2004]. The strategies can be summarized into three groups. The first is to discount the data that are not relevant for prediction [Cunningham *et al.*, 2003]. The second is to build an ensemble of models, each of which is fitted to a different subset of the data [Street and Kim, 2001]. The third is mining data instances ordered by time, which is known as data stream mining [Fan, 2004; Gao *et al.*, 2007]. Though those research problems are related to deal with the changing of data distributions, the settings of our Collaborative Filtering for recommender systems is quite different: while most existing techniques under concept drift tracked a single concept, the user of a recommender system often shows multiple interests, which usually influence each other and only associated with limited data instances [Koren, 2009].

Markov decision process An MDP-based recommender system views the problem of generating recommendations as a sequential optimization problem [Shani *et al.*, 2002]. Because of the nature of Markov decision process, this approach needs a strong initial model built with plenty of data, which is not practical in the sparse dataset of real-world applications.

6 Conclusion

We proposed a Collaborative Evolution (CE) model to embed an evolutionary view of users' profiles to MF. In the CE model, two processes, i.e. the collaborative learning of the users' profiles in each time interval and the evolutionary learning of the user's profiles along the consecutive time intervals, are iteratively guided by each other. As an output of the CE model, we aim to predict users' profiles in a certain day of the future by the vector autoregressive model obtained from the evolutionary learning. This evolutionary learning of the users' profiles is a breakthrough of the current collaborative filtering techniques. The CE method is particularly useful in a scenario where the users demonstrate frequently changing behaviours and long-term predictions in the future are needed, such as the online shopping scenario. We perform experiments on the real-world dataset from the online shopping website *www.51buy.com*, which contains more than 180 days of records for $1,000,000$ users and $270,000$ items. The dataset will be released and a sample has been published online. With the evolutionary learning of users' profiles, we enhanced the current online shopping recommender systems by providing predictions for the long *future*.

7 Acknowledgement

We thank the support of China National 973 project 2014CB340304 and Hong Kong CERG projects 16211214 and 16209715. Sinno Jialin Pan is supported by the NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020.

References

[Adomavicius and Tuzhilin, 2005] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state of the art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749, 2005.

[Bennett and Lanning, 2007] James Bennett and Stan Lanning. The netflix prize. In *KDD cup and workshop*, 2007.

[Cunningham *et al.*, 2003] Pádraig Cunningham, Niamh Nowlan, Sarah Jane Delany, and Mads Haahr. A case-based approach to spam filtering that can track concept drift. In *ICCBR*, 2003.

[Dueck and Frey, 2004] Delbert Dueck and Brendan Frey. Probabilistic sparse matrix factorization. *University of Toronto technical report PSI-2004-23*, 2004.

[Fan, 2004] Wei Fan. Systematic data selection to mine concept-drifting data streams. In *KDD*, pages 128–137. ACM, 2004.

[Gao *et al.*, 2007] Jing Gao, Wei Fan, Jiawei Han, and S Yu Philip. A general framework for mining concept-drifting data streams with skewed distributions. In *SDM*. SIAM, 2007.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[Koren, 2009] Yehuda Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456. ACM, 2009.

[Lathia *et al.*, 2010] Neal Lathia, Stephen Hailes, Licia Capra, and Xavier Amatriain. Temporal diversity in recommender systems. In *SIGIR*, pages 210–217. ACM, 2010.

[Liu and Aberer, 2014] Xin Liu and Karl Aberer. Towards a dynamic top-n recommendation framework. In *RecSys*, pages 217–224. ACM, 2014.

[Liu *et al.*, 2010] Nathan N Liu, Min Zhao, Evan Xiang, and Qiang Yang. Online evolutionary collaborative filtering. In *RecSys*, pages 95–102. ACM, 2010.

[Liu, 2015] Xin Liu. Modeling users dynamic preference for personalized recommendation. In *IJCAI*, 2015.

[Lu *et al.*, 2015] Zhongqi Lu, Zhicheng Dou, Jianxun Lian, Xing Xie, and Qiang Yang. Content-based collaborative filtering for news topic recommendation. In *AAAI*, pages 217–223, 2015.

[Lütkepohl, 2005] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Cambridge Univ Press, 2005.

[Marlin, 2003] Benjamin M Marlin. Modeling user rating profiles for collaborative filtering. In *NIPS*, 2003.

[McAuley and Leskovec, 2013] Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *WWW*, pages 897–908. International World Wide Web Conferences Steering Committee, 2013.

[Middleton *et al.*, 2004] Stuart E Middleton, Nigel R Shadbolt, and David C De Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88, 2004.

[Mnih and Salakhutdinov, 2007] Andriy Mnih and Ruslan Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2007.

[Nakatsuji *et al.*, 2012] Makoto Nakatsuji, Yasuhiro Fujiwara, Toshio Uchiyama, and Hiroyuki Toda. Collaborative filtering by analyzing dynamic user interests modeled by taxonomy. In *ISWC*, pages 361–377. Springer, 2012.

[Neumaier and Schneider, 2001] Arnold Neumaier and Tapio Schneider. Estimation of parameters and eigenmodes of multivariate autoregressive models. *ACM Trans. Math. Softw.*, 27(1):27–57, 2001.

[Pálovics *et al.*, 2014] Róbert Pálovics, András A Benczúr, Levente Kocsis, Tamás Kiss, and Erzsébet Frigó. Exploiting temporal influence in online recommendation. In *RecSys*, pages 273–280. ACM, 2014.

[Rendle *et al.*, 2010] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820. ACM, 2010.

[Salakhutdinov and Mnih, 2008] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887. ACM, 2008.

[Shani *et al.*, 2002] Guy Shani, Ronen I Brafman, and David Heckerman. An mdp-based recommender system. In *UAI*, pages 453–460. Morgan Kaufmann Publishers Inc., 2002.

[Singh and Gordon, 2008] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *KDD*, pages 650–658. ACM, 2008.

[Street and Kim, 2001] W Nick Street and YongSeog Kim. A streaming ensemble algorithm (SEA) for large-scale classification. In *KDD*, pages 377–382. ACM, 2001.

[Tsymbal, 2004] Alexey Tsymbal. The problem of concept drift: definitions and related work. Technical report, Computer Science Department, Trinity College Dublin, 2004.

[Wang and Zhang, 2013] Jian Wang and Yi Zhang. Opportunity model for e-commerce recommendation: right product; right time. In *SIGIR*, pages 303–312. ACM, 2013.

[Zhao *et al.*, 2012] Gang Zhao, Mong Li Lee, Wynne Hsu, and Wei Chen. Increasing temporal diversity with purchase intervals. In *SIGIR*, pages 165–174. ACM, 2012.