Temporal Recommendation via Modeling Dynamic Interests with Inverted-U-Curves

Conference Paper · March 2016

DOI: 10.1007/978-3-319-32025-0_20

CITATIONS
4

READS
83

5 authors, including:

Yang Xu
Shandong Normal University
13 PUBLICATIONS **102** CITATIONS
SEE PROFILE

Philip S. Yu
University of Illinois at Chicago
1,539 PUBLICATIONS **89,366** CITATIONS
SEE PROFILE

Temporal Recommendation via Modeling Dynamic Interests with Inverted-U-Curves

Yang Xu[1], Xiaoguang Hong[1(✉)], Zhaohui Peng[1], Guang Yang[1], and Philip S. Yu[2,3]

[1] School of Computer Science and Technology, Shandong University, Jinan, China
zzmylq@gmail.com, {hxg, pzh}@sdu.edu.cn, loggyt@yeah.net

[2] Department of Computer Science, University of Illinois at Chicago, Chicago, USA
psyu@uic.edu

[3] Institute for Data Science, Tsinghua University, Beijing, China

Abstract. How to capture user interest accurately to enhance the user experience is a great practical challenge in recommender systems. Through preliminary investigation, we find that each user has his personalized interest model which may contain multiple kinds of interests, and the strength of each user interest usually has a dynamic evolution process which can be divided into two stages: rising stage and declining stage. The evolution rate of the user interests also differ from each other. Based on this finding, a recommendation framework called SimIUC is proposed, which can identify multiple user interests and adapt the inverted-U-curve to model the dynamic evolution process of user interests. Specifically, SimIUC differs from the traditional user preference based methods which use monotonously decreasing function to model user interest. It can predict the evolutionary trends of interests and make recommendations by inverted-U-interest-based collaborative filtering. We studied a large subset of data from MovieLens and netflix.com respectively. The experimental results show that our method can significantly improve the accuracy in recommendation.

Keywords: Interest modeling · Recommender systems · Inverted U curve

1 Introduction

Personalized recommendation systems that can extract information of interests from a large amount of complex data have been widely used in the case of the rapid expansion of Internet applications. User interest model plays a key role in personalized recommendation systems and it is applied in a variety of applications, e.g., microblog recommendations, product recommendations and music recommendations, etc. The key issue of personalized recommendation is how to accurately obtain the user's interests.

Temporal interaction log of a user's interaction with the system contains the user's interests. Existing studies [6, 12, 13, 15] have generally considered that the more timely the temporal information generated, the higher effectiveness in the rating prediction.

S.B. Navathe et al. (Eds.): DASFAA 2016, Part I, LNCS 9642, pp. 313–329, 2016.
DOI: 10.1007/978-3-319-32025-0_20

Thus, old information has lower effectiveness. Existing user interest models usually use monotonically decreasing function, e.g., linear function, exponential function, to calculate timeliness of information. However, our preliminary investigation shows that user interests have two characteristics. First, the interest pattern is personalized. For example, some of the users keep a clear interest consistently, while some others maintain multiple interests at the same time and also some ones change their interests frequently. Second, the evolution of a user interest is a nonmonotonic process which can be divided into two stages: rising stage and declining stage [1]. The evolution rate of the user interests also differs from each other. Due to these two characteristics, it is hard to accurately model a user's interest using a single monotonously decreasing function. Existing interest models cannot effectively describe interest patterns and interest evolution trends.

In this paper, we propose a new recommendation framework called SimIUC. Under SimIUC, we can effectively identify multiple user interests from his or her temporal interaction log, and adapt the inverted-U-curves to model user interest patterns to reflect the dynamic evolution process of user interests, and then we can predict user's interest evolution trend in the future and make recommendation.

What is Inverted U Curve? The inverted U curve was first advanced by economist Simon Kuznets in the 1950s and 60s, and it was used to describe the relationship between income distribution and economic development [2]. In social psychology, inverted U curve can be used to describe the evolution process of a person's interest. People interact with their interested things, and the larger the number of interactions, the higher the degree of interest [3], but the growth rate has continued to decrease [4]. When the number of interactions reaches a certain threshold, the degree of interest will drop [3]. This is a universal law in the real world, for example, you find a nice dessert in a dessert shop, thus become interested in the dessert shop. You will continue to visit the shop, and a new delicious dessert will attract you a strong interest in the shop. But as you buy more and more dessert in the shop, you may feel that the taste of the dessert in that shop is getting more and more common, while their deficiencies will also be found in this process. Thus a "very good dessert shop" may become a "good dessert shop" after a period of time, and at last, it may just become an ordinary "dessert shop". Inverted-U-curve can properly describe the dynamic evolution process of a user interest like this. To summarize, the major contributions of this paper are as follows:

- In this paper, we consider the dynamic evolution process of user interest which previous studies have rarely mentioned. A novel recommendation framework named SimIUC for modeling the dynamic evolution process of user interest is proposed. SimIUC can be used to predict user interest evolution trend and make accurate recommendation.
- In SimIUC, we design a multiple user interest identification method and an inverted-U-interest model learning algorithm to model user interest pattern and interest evolution trend.
- We systematically compare the proposed SimIUC approach with other algorithms on the dataset of Movielens and Netflix. The results confirm that our new method substantially improves the precision of recommendation.

The rest of this paper is organized as follows. Section 2 presents some notations and the algorithm framework. Section 3 introduces our novel algorithm for interest identification, and Sect. 4 details the construction of the inverted-U-interest model and the recommendation approach. Experiments and discussions are given in Sect. 5. In Sect. 6, we review the related works on temporal dynamics and user interest. Conclusions are drawn in Sect. 7.

2 Preliminaries

2.1 Notations

I is a set of all items and U is a set of all users. Our main task in this paper is modeling user interest by analyzing user temporal interaction logs and making recommendations. We first define the data model of user temporal interaction logs. $L_u = \{r_{u1}, r_{u2}, \ldots, r_{ul}\}$ denotes the temporal interaction log of user u which contains l records, and r_{ui} represents u's ith interaction record expressed as a quadruple $(user, item, rating, timestamp)$. $L'_u = \{r'_{u1}, r'_{u2}, \ldots, r'_{uo}\}$ is the processed temporal interaction logs of user u treated by filtering out noise and labeling interest, o records the size of L'_u, and r'_{ui} is an extended record expressed as a five-tuple $(user, item, rating, interest, timestamp)$. We propose algorithms to identify the user's N_u interests and learn N_u inverted-U-interest curves separately. f_{ui} represents user u's ith inverted-U-interest curve, and we denote the inverted-U-interest model of user u as $IUI_u = \{f_{u1}, f_{u2}, \ldots, f_{uN_u}\}$. Table 1 lists the frequently used symbols in this paper.

Table 1. Summary of notations

Symbol	Description
I	the set of items
U	the set of users
$L_u = \{r_{u1}, r_{u2}, \ldots, r_{ul}\}$	temporal interactive log of user u
$L'_u = \{r'_{u1}, r'_{u2}, \ldots, r'_{uo}\}$	temporal interactive log of user u with interest labeled
$IUI_u = \{f_{u1}, f_{u2}, \ldots, f_{uN_u}\}$	the set of inverted-U-interest curves of user u
HIN	the heterogeneous information network
λ	the parameter to govern the influence of feature similarity and interactive similarity
N_u	the number of user interest of user u

2.2 SimIUC Recommendation Framework

We propose a recommendation framework called SimIUC based on inverted-U-interest model. The framework is as shown in Fig. 1. In order to discover users' interests from user temporal interaction logs, SimIUC first needs to calculate the similarity matrix S between each item. In this paper, a new similarity algorithm named FIsim is proposed in SimIUC. In FIsim, we consider the similarity between different items not only based on the item features, but also the similar situation of items in the process of interactions

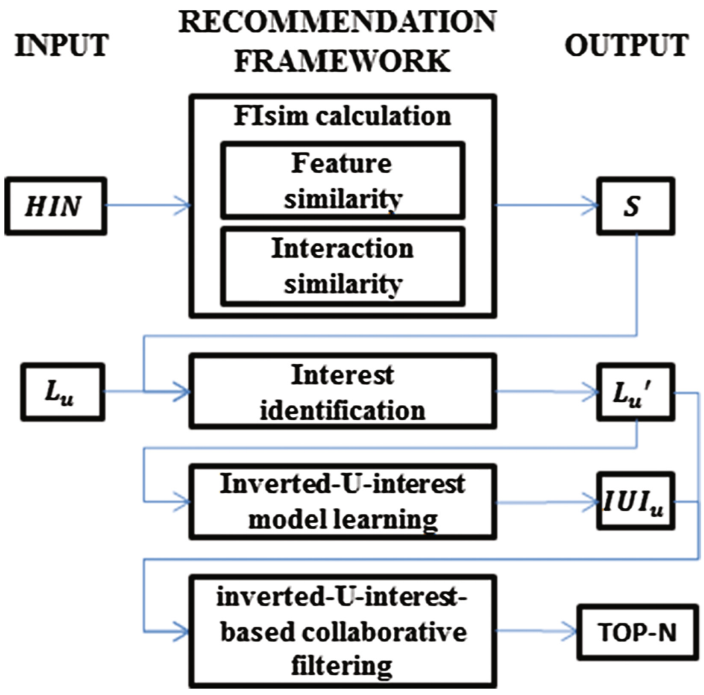

Fig. 1. SimIUC recommendation framework

between users and items. Consequently, the input information in SimIUC is all users' temporal interaction log L and items' feature information FG. Given a target user *u*, SimIUC uses the similarity matrix *S* to cluster and filter items in *u*'s temporal interaction log L_u, getting *u*'s N_u different interests. Then, SimIUC proposes an inverted-U-interest model learning algorithm to learn u's inverted-U-interest model IUI_u. Last, SimIUC makes accurate top-N recommendation through an inverted-U-interest-based collaborate filtering approach.

3 Interest Identification

As discussed in the previous section, in order to accurately discover users' interests from user interaction logs, the first phase of SimIUC is to calculate the similarity between each item. The proposed similarity algorithm FIsim combines the similarities of item feature and user interaction. Then we detail the interest identification algorithm.

3.1 Feature Similarity Calculation

FIsim measures the feature similarity between each item by calculating the similarity between their features in the heterogeneous information network. A heterogeneous information network(HIN) is an undirected weighted graph $G = (V, E, W)$ with an object type mapping function $\varphi : V \rightarrow \mathcal{A}$ and a link type mapping function $\phi : E \rightarrow \mathcal{R}$ and $|\mathcal{A}| > 1$, $|\mathcal{R}| > 1$. Each object $v \in V$ belongs to a particular object type $\varphi(v) \in \mathcal{A}$ and each link $e \in E$ belongs to a particular relationship $\phi(e) \in \mathcal{R}$. $W : E \rightarrow \mathbb{R}^+$ is a weight mapping from an edge $e \in E$ to a real number $\omega \in \mathbb{R}^+$. As an example, a toy IMDB network is given in Fig. 2. It is a typical heterogeneous information network, containing five types of nodes: users (U), movies (M), genres (G), directors (D), and actors (A). G, D and A are called as feature nodes in this paper.

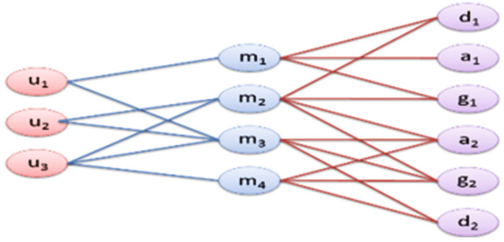

Fig. 2. An example of HIN

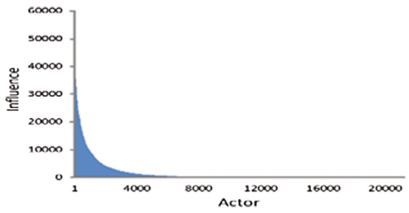

Fig. 3. Long-tailed distributions of actor influence

In G, the weight of the edges between items and features represents the influence from feature to item and depends mainly on two factors. One is the global influence of features, which is an important consideration in people's content-consumption behaviors because users tend to choose popular objects to interact. The more profound influence the item features has, the more possibility the user chooses the item. The other factor is association strength among features and item, which depends on the rank between the same kinds of features of the item. The higher rank the feature has, the greater association with the item it has. For example, the strength of association of the first actor is much higher than the fifth actor in a movie.

In G, the feature node v_f connects with the items which own feature f, and the set of these items denoted as I_f. The global influence of v_f is defined as:

$$p\left(v_f\right) = \left|\cup_{v_i \in I_f} \cup_{v_u \in U_i} \{v_u\}\right| \tag{1}$$

where U_i is the set of user nodes v_u which is connected with item node v_i. The definition means, given a feature node v_f, its influence is the number of users who have interacted with items which own feature f. In the example shown in Fig. 2, the influence of director d_1 is 3 and actor a_1 is 1. The weight of the edges between the item and the feature is measured as:

$$w\left(v_f, v_i\right) = e^{-\frac{r\left(v_f, v_i\right) \cdot p_{0.2}}{p\left(v_f\right)}} \tag{2}$$

where $r(v_f, v_i)$ is the rank of feature node v_f in the same type of features of item i, and $p_{0.2}$ is a parameter for adjusting the range of $w(v_f, v_i)$. Figure 3 is the distribution of the actor influence in Movielens dataset. The influence of item features presents a long-tailed distribution. We denote n_i as the total number of feature nodes belonging to the feature type i in *HIN*. According to the Pareto Principle [10], we take the (0.2•n_i)th maximum influence value as the value of $p_{0.2}$.

When calculating feature similarities between items in *HIN*, we introduce the concept of meta-path mentioned in [11]. $\mathbb{P}$ denotes the set of specified meta-paths. $P = (A_1 A_2 \ldots A_n)$ is a meta-path, and $p = (v_1 v_2 \ldots v_n)$ is a path instance of the meta-path. It has been shown in [11] that long meta-paths are not quite useful in calculating similarity. So when we calculate the similarity between two items, we only need to use those meta-paths whose length is no more than 5 between the two items. For one path instance, we calculate the product of every edge's weight of it as the similarity value on this path instance. Then, the feature similarity between two items i_p, i_q is the sum of the similarity values on all path instances of all meta-paths between nodes v_{i_p}, v_{i_q}. The feature similarity between two items is defined as:

$$FeatureSim\left(v_{i_p}, v_{i_q}\right) = \sum\nolimits_{P \in \mathbb{P}} \sum\nolimits_{p \in P} \prod\nolimits_{k=1}^{n-1} w\left(v_k, v_{k+1}\right). \tag{3}$$

3.2 Interaction Similarity Calculation

FIsim considers that interactions between the user and the system are driven by interests, and the user's interests remain constantly for a period of time. Consequently, items connected with the same user have a certain similarity called interaction similarity.

The algorithm we proposed measures the interaction similarity from two dimensions: time and rating. FIsim considers that the smaller the interaction time interval or rating difference, the more similar between the two items, because the user's interest and evaluation standard remain constantly for a period of time.

The interaction similarity between the items i_p, i_q is measured as:

$$\mathrm{sim}\left(v_{i_p}, v_{i_q}\right) = \sum\nolimits_{u \in (U_p \wedge U_q)} e^{-\left[(1-\theta)\left(\frac{t_{up}-t_{uq}}{t_u}\right)^2 + \theta\left(\frac{r_{up}-r_{uq}}{\bar{r}_u}\right)^2\right]} \tag{4}$$

$$\mathrm{InteractSim}\left(v_{i_p}, v_{i_q}\right) = \frac{\sum\nolimits_{v_a, v_b \in I} I(sim(v_a, v_b) \neq 0)}{\sum\nolimits_{v_a, v_b \in I} I(sim(v_a, v_b) \neq 0) sim(v_a, v_b)} sim(v_{i_p}, v_{i_q}) \tag{5}$$

where $I(\cdot)$ is an indicator function, U_p is a set of user nodes which connected with item p, t_{up} is the interaction time of user u with item p, t_u is the time of latest interaction, r_{up} is u's rating of p and $\bar{r}_u$ is the average rating of u. The value range of interaction similarity is limited by formula 5. λ is used to govern the influence of feature similarity and interaction similarity. The FIsim is given by the following equation:

$$\mathrm{FIsim}\left(v_{i_p}, v_{i_q}\right) = (1 - \lambda) \cdot FeatureSim\left(v_{i_p}, v_{i_q}\right) + \lambda \cdot InteractSim(v_{i_p}, v_{i_q}). \tag{6}$$

3.3 Interest Identification

Using the FIsim algorithm described in last section, we can get the similarities between any two items and construct similarity matrix S. The identification of user multiple interests follows a cluster filtering approach, which clusters the items in L_u and uses the principal component extraction method to filter out the noise. We treat each item in L_u as a singleton candidate cluster. For each pair of items (p, q) in L_u, we will put (p, q) in a descending sequence SEQ if the similarity of (p, q) is no smaller than a predefined threshold β. Next, we remove (p, q) from SEQ in sequence, and the two clusters which contain item p and q respectively will be merged if the similarity between them is no less than a predefined threshold γ until SEQ is empty. $sim(p, q)$ is calculated as:

$$sim(p, q) = \frac{1}{|c(i)||c(j)|} \sum\nolimits_{i_p \in c(i)} \sum\nolimits_{i_q \in c(j)} S(p, q) \tag{7}$$

where $c(i)$ is the cluster which contains item i, and $S(p.q)$ is similarity between items p, q. We apply the principal component extraction method to the clustering result. The N_u maximum clusters which make $\sum_{i=1}^{N_u} size(c(i)) / \sum_{i=1}^{n} size(c(i)) \geq \eta$, where η is the threshold of the first N_u principal components' cumulative contribution rate, n is the number of clusters in the clustering result and $size(c(i))$ is the number of items in cluster $c(i)$. They are selected as N_u kinds of the user interest and other clusters are considered as noise to be removed. In this paper, we set η to 0.8. Finally, we update the item records in L_u with interest tag. The pseudocode of interest identification is shown in Algorithm 1 which is inspired by Kruskal algorithm [21]. Through multiple user interest identifications, each of the remaining items in the user temporal interaction log is assigned into a corresponding interest.

Algorithm 1. Interest identification

```
Input  : $S$: the similarity matrix of all items, $L_u$: the temporal interactive log of the user $u$,
         $\beta$: the similarity threshold of items, $\gamma$: the similarity threshold of clusters
Output: $L'_u$: the temporal interactive log with interest labeled of the user u
 1:  for each item $i_p$, $i_q$ in $L_u$ do
 2:     if $FIsim(i_p, i_q) \geq \beta$ then
 3:        add $(p, q)$ in Seq;
 4:  Queue Q ← Seq.sort("order by $FIsim(i_p, i_q)$ descending");
 5:  each item $i_k$ in $L_u$ as a singleton cluster $c(i_k)$;
 6:  while $Q$ is not empty do
 7:     $(i_p, i_q)$ = Q.top();
 8:     Calculate the similarity $sim(p, q)$ between $c(i_p)$ and $c(i_q)$ according to formula 7;
 9:     if $sim(p, q) \geq \gamma$ then
10:        merge $c(i_p)$ and $c(i_q)$ as a new cluster;
11:  apply the principal component extraction method to the clustering result;
12:  select $N_u$ maximum clusters which satisfied $\sum_{i=1}^{N_u} size(c(i)) / \sum_{i=1}^{n} size(c(i)) \geq \eta$ as user's $N_u$
     interests;
13:  $L'_u$ ← update $L_u$ with interest tag;
14:  return $L'_u$;
```

4 Learning Inverted-U-Interest Model and Making Recommendation

After interest identification, SimIUC presents a learning algorithm to learn the inverted-U-interest model and then introduces an inverted-U-interest-based collaborate filtering approach to make top-N recommendation.

4.1 Inverted-U-Interest Model

Interest Pattern. Influenced by behavior habits, every user has a personal interest pattern. There are four kinds of user interest patterns through analyzing user log, which are *specific interest pattern*, *multiple interest pattern*, *interest shifting pattern* and *random noise pattern*. We cannot make accurate predictions about future behaviors of the user with random noise pattern because when the user interacts with the system, it shows a strong randomness. So we do not consider this type of users in this paper. For the other three interest patterns, SimIUC characterizes them by building inverted-U-interest model.

Learning Algorithm of Inverted-U-Interest Model. Time is an important aspect in fitting interest curve. Because of the difference between user behavior and external environment constraints, the frequency of interaction of different users varies considerably even with the same interest. For example, two users A and B both like watching comedy movie. Due to work reasons, user A watches movies only on weekends, but user B watches movies every day. If we measure the interestingness using the same time dimension, user A's interestingness to comedy movie would be far lower than B's.

But this is not the case. Consequently, the modeling results will be greatly intervened with the interaction frequency if we use the same time dimension for all users in modeling. As illustrated in [3], the evolution of user interest relates to interaction times. Our interest modeling approach uses interaction times to build the time dimension, which eliminates the impact of interaction frequencies. The learning samples are created from L'_u, and user's inverted-U-interest model is learned by the learning algorithm SimIUC proposed. The pseudocode of learning algorithm of inverted-U-interest model is shown in Algorithm 2.

Algorithm 2. Learning inverted-U-interest model

```
Input  : $L'_u$: the temporal interactive log with interest labeled of the user u
Output: $IUI_u$: the set of inverted-U-interest models of user u
1:   for each interest $k$ of user $u$ do
2:       Set $x$ of sample points in interest $k$'s learning sample are $[1,2,\ldots,|L_u'|]$;
         Set y of sample points to the number of interactions between the user and items belong-
3:   ing
         to $u$'s interest $k$ when user $u$ has interacted with system $x$ times.
4:       //use the improved sigmoid function and the least square method to fit sample points//
5:       $f_{uk} \leftarrow \arg\min_{a,b,c} \sum_i (Y_i - \hat{Y}_i)^2 = \arg\min_{a,b,c} \sum_i [Y_i - (\frac{a}{1+be^{-cX_i}})]^2$;
6:       Inverted-U-interest curve $f_{uk}'$ is the derivative of $f_{uk}$;
7:       $IUI_u.add(f_{uk}')$;
8:   return $IUI_u$;
```

Through identifying the multiple user interests, N_u different interests of target user u have been identified and each item in L'_u is assigned into the corresponding interest. In L'_u, the items belonging to u's interest k have been selected to construct the learning sample of u's interest k. A learning sample of interest k is a group of ordered pairs (x, y), in which x represents the number of interactions between user u and the system, and y is defined as the number of interactions between user u and items belonging to u's interest k. For example, there are two different interests identified for user u, and at the time of $x = 20$, which means that user u has interacted with the system for 20 times, the number of interactions between user u and items belonging to his first interest is 12 and that of the second interest is 8, then the sample point $(20, 12)$ is contained in u's first interest learning sample, and the sample point $(20, 8)$ is contained in the second one. The discrete ranges of both x and y are $[1, |L'_u|]$. Each interest learning sample is dealt with in lines 1 to 4 of Algorithm 2. In lines 5 to 6, we improve the sigmoid function as the fitting function and use the least square method to fit sample points. The prediction is given by the following equation:

$$\hat{Y} = \frac{a}{1 + be^{-cX}} \tag{8}$$

This equation involves three parameters. The least square method defines the estimate of these parameters as the values which minimize the sum of the squares between the measurements and the model. This amounts to minimizing the expression:

$$\varepsilon = \sum\nolimits_i (Y_i - \hat{Y}_i)^2 = \sum\nolimits_i [Y_i - (\frac{a}{1+be^{-cX_i}})]^2 \tag{9}$$

Thus, the best parameter setting (including a, b and c) of fitting curve should be

$$\arg\min_{a,b,c} \sum\nolimits_i (Y_i - \hat{Y}_i)^2 = \arg\min_{a,b,c} \sum\nolimits_i [Y_i - (\frac{a}{1+be^{-cX_i}})]^2 \tag{10}$$

N_u is the number of user interest of user u. We can get N_u fitting curves $\{f_1,f_2,\ldots,f_{Nu}\}$ by fitting the sample points of each interest. The inverted-U-interest curve f_k' of interest k is the derivative of f_k:

$$f_k'(x) = \left(\frac{a_k}{1+b_k e^{-c_k x}}\right)' = a_k c_k g(x)[1-g(x)], g(x) = \frac{1}{1+b_k \cdot e^{-c_k x}} \tag{11}$$

where a_k, b_k, c_k are the three parameters to be learned. The inverted-U-interest model of user u is the set of inverted-U-interest curves, i.e. $IUI_u = \{f_1',f_2',\ldots,f_{Nu}'\}$.

Case Study. The results of learning the inverted-U-interest model for the 51th user in Movielens dataset is shown in Fig. 4. The user interests are identified by interest identification algorithm. The size of L_{51}' become 36, after filtering out 4 noise records from user's temporal interaction log L_{51}, and two kinds of user interests are identified. The fitting results of two interest learning sample of 51th user is shown in Fig. 4(a), and the inverted-U-interest model has been learned is shown in Fig. 4(b).

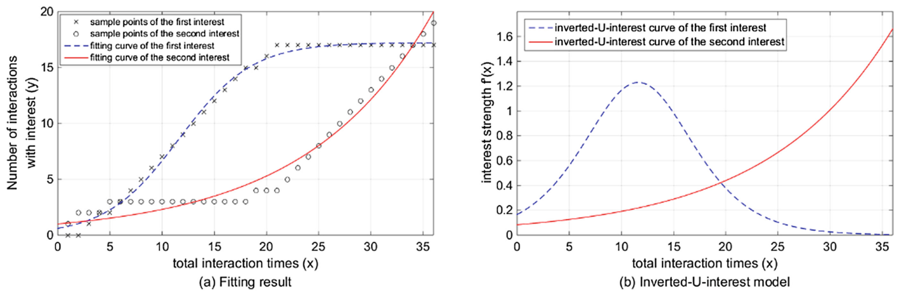

Fig. 4. The results of learning IUI model for the 51th user in Movielens data set

In Fig. 4(a), '✕' and '○' represent sample points of the user's first interest and the second interest respectively. Each point represents that when the user has interacted with system x times, the number of interactions between him and the items belonging to his corresponding interest is y. The dotted line and the full line respectively describe the fitting result of the user's first interest and the second interest. Figure 4(a) has shown that during the first half of interactions, the user mainly interacts with the items belonging to his first interest, but in the second half, the user shifts his attention to items belonging to his second interest.

Each interest of users has its corresponding inverted-U-interest curve constructing user's inverted-U-interest model. Figure 4(b) demonstrates the inverted-U-interest model of the 51th user where x-axis represents the number of interactions between the 51th user and the system, and y-axis represents the interest strength. Figure 4(b) has shown that the interest of the 51th user has an obvious shifting in the process of interacting with the system. The old interest (dotted line in Fig. 4(b)) declines while the new interest (full line in Fig. 4(b)) rises fast. So we infer that the interest pattern of this user belongs to the "*interest shifting pattern*", in which SimIUC will focus more on the new interest in recommendation.

4.2 Inverted-U-Interest-Based Collaborative Filtering Approach

When making recommendation for x times interaction of the target user u, our basic idea is to consider u as the source to be injected with user preferences. The propagation process of user preferences is shown in Fig. 5. Preferences injected into the user u will be propagated to items i in L_u' through the corresponding interest, and tend to propagate to K nearest neighbor items i' which construct candidate item set C. For each i in C, the estimated preference $p_{u,i,x}$ of user u on item i is measured as:

$$P_{u,i,x} = \sum\nolimits_{j \in L_u'} \sum\nolimits_{f'k \in IUI_u} e^{f'k(x)} \cdot I(j \in interest_k) \cdot S(i,j). \tag{12}$$

where $I(\cdot)$ is an indicator function, f_k' is the inverted-U-interest curve of $interest_k$, and $S(i,j)$ is the similarity between item i and j. We sort $P_{u,i,x}$ for all no-interacted neighbor items, and then return top-N no-interacted neighbor items to user u.

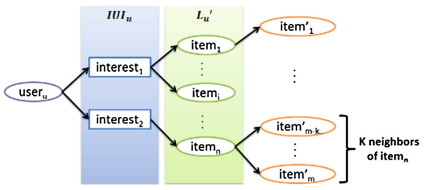

Fig. 5. The propagation process of user preferences

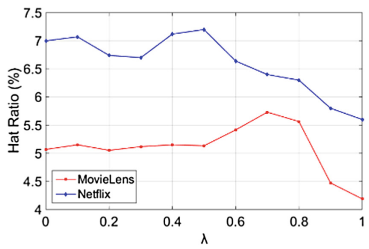

Fig. 6. The impact of λ on hit ratio in both datasets

5 Experiments

We have conducted a set of experiments to examine the performance of our recommendation method compared with the baselines. We begin by introducing the experimental settings, and then analyze the evaluation results.

5.1 Experimental Settings

Data Description. There are two real datasets in our experiments: Movielens [22] and Netflix [23]. Movielens and Netflix are the most widely used common datasets in recommendation research projects. The Movielens dataset contains 6,040 users who have issued 999,209 explicit ratings on a 5-point likert scale, referring to 3,883 movies.

The Netflix dataset was made available in 2006 as a part of the Netflix Prize, in which each user rated at least 20 movies, and each movie was rated by 20–250 users. It has been widely used as a large-scale data set for evaluating recommenders. As our goal is to make accurate top-*N* recommendation by analyzing item features and user temporal interaction logs, we extend the movie features of the Netflix dataset through using MovieLens dataset. 6000 users and related movies in Netflix dataset are randomly selected as experimental data. The global statistics of these two datasets used in our experiments are shown in Table 2.

Table 2. Characteristics of datasets

Name of dataset	MovieLens	Netflix
Number of users	6,040	6,000
Number of items	3,883	4,158
Avg. # of rated items/user	257.3	359.7
Number of ratings	999,209	2,158,471

Table 3. Disassembled algorithms

Disassembled algorithm	FIsim	IUI model
SimIUC	○	○
CosIUC	╳	○
SimCF	○	╳
CosCF	╳	╳

Evaluation Metric. We adopt the All-But-One evaluation method and use Hit Ratio [5] as the metric for the top-*N* recommendation. Our datasets were splitted into two subsets, the training set and the test set. For every user, the latest item he rates is selected as test data and the remaining items are used as training data.

When making recommendation, we use SimIUC to generate a recommendation list of N items named $R(u,t)$ for each user u at time t. If the test item of the user u appears in $R(u,t)$, we call it a hit. The Hit Ratio is calculated in the following way:

$$\text{Hit Ratio} = \frac{\sum_u I(T_u \in R(u,t))}{|U|} \tag{13}$$

where $I(\cdot)$ is an indicator function, $R(u,t)$ is a set of top-N items recommended to user u at time t, T_u is the test item that the user u has actually interacted with at time t.

Compared Methods. We examine the performance of the proposed SimIUC approach by comparing it with three other top-*N* recommendation algorithms, including ItemKNN [9], TItemKNN [6], and IFCM-IFC [12].

Item-based collaborative filtering (ItemKNN) is famous recommendation algorithm which could predict unknown item ratings for a given user by referencing item rating information from other similar items. TItemKNN is a time weighted item-based collaborative filtering method by reducing the influence of old data when predicting users' further behaviors. TItemKNN was designed for rating prediction task but it can be easily extended to top-N recommendation for temporal data. IFCM-IFC method adapts the memory forgetting curve to model the user interest for temporal recommendations.

5.2 Evaluations

Impact of Parameter λ. We first focus on analyzing the parameter λ, which governs the influence of feature similarity and interaction similarity. In the first experiment, we vary the parameter λ from 0, 0.1 to 1. The number of the nearest neighbor *K* is set to 100, and *N* is set to 20. The results of using different constant λ on both datasets are demonstrated in Fig. 6.

The results have shown that λ is important in determining the Hit Ratio, and ignoring either interaction similarity ($\lambda = 0$) or feature similarity ($\lambda = 1$) cannot generate good results. Optimal results can be gotten by combining feature similarity and interaction similarity together. The optimal value of λ is about 0.7 in MovieLens and is about 0.5 in Netflix. In the following experiments, λ is set to 0.7 in Movielens and 0.5 in Netflix.

Comparison to TItemKNN and IFCM-IFC with Different α. The decay rate α in TItemKNN and IFCM-IFC are used to control the attenuation rate of the influence of old data. In this section, we compare the Hit Ratio of SimIUC with TItemKNN and IFCM-IFC under different α on both datasets. When tuning α, *K* is set to 100, and *N* is set to 20. The results of how Hit Ratios of TItemKNN and IFCM-IFC change against α are shown in Fig. 7. Because there is no parameter α existing in SimIUC, its Hit Ratio is drawn as a straight line. The results have shown that SimIUC outperforms other algorithms when $\alpha \in [0.1,1]$ in both datasets. For TItemKNN, the optimal α is about 0.7 in Movielens, and is about 0.3 in Netflix. For IFCM-IFC, the optimal α is about 0.6 in Movielens, and in Netflix, it is about 0.2.

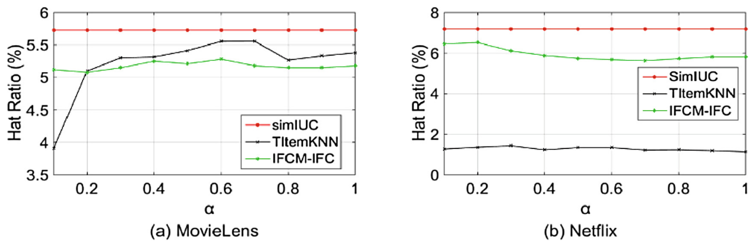

Fig. 7. The impact of α on hit ratio

Overall Accuracy Performance. In this section, we have evaluated the overall accuracy performance of SimIUC and the other three top-*N* recommendation algorithms, ItemKNN, IFCM-IFC, and TItemKNN. In the experiment, the number of the nearest neighbors *K* is set to 100, and other parameters are set to the optimal values obtained from previous experiments.

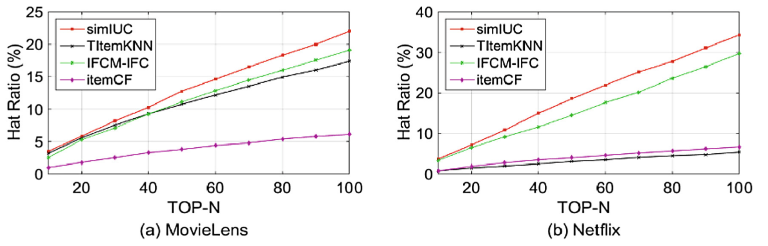

Fig. 8. Comparisons on hit ratio of recommendation

The comparison results have been shown in Fig. 8, where SimIUC exceeds all the baselines under different size of recommendation list. By using IFCM-IFC as the benchmark, SimIUC outperforms IFCM-IFC 10.43% to 29.31% on Netflix dataset, and improves it up to 10.42% to 38.34% on MovieLens dataset. The experiment proves that the inverted-U-interest model plays an important role in improving recommendation accuracy, and SimIUC can get better accuracy of item recommendation.

Disassembly Analysis of SimIUC. The performance of each part of the SimIUC recommendation framework is analyzed in this section. In the experiment, FIsim has been replaced with cosine similarity, and the inverted-U-interest model has been removed by setting $e^{f'_k(x)}$ in formula (12) to 1. The three disassembled algorithms are shown in Table 3, in which the label "◯" means the corresponding part of SimIUC has been preserved, and the label "╳" means the corresponding part of SimIUC has been removed or replaced.

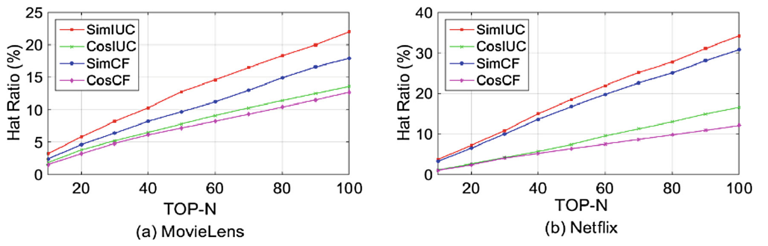

Fig. 9. Disassembly analysis

Figure 9 demonstrates the hit ratios of the disassembled algorithms under different size of recommendation list. It has shown that SimIUC and CosIUC have achieved significant improvement in hit ratio over SimCF and CosCF by introducing inverted-U-interest model. The hit ratio of SimIUC compared with CosIUC has a significant improvement by replacing the similarity method with FIsim, and SimCF also has a notable improvement compared with CosCF on both datasets. The results lead us to conclude that both combining the feature similarity with the interaction similarity and introduction of inverted-U-interest model are necessary and effective.

6 Related Work

Temporal Dynamics in Recommendation. Temporal information is widely used in the framework of information spreading [18, 20], matrix-factorization [14], item-based CF [6] or graph [19]. Modeling temporal dynamics is indispensable when designing recommender systems.

Koren [14] models the temporal dynamics via a time-aware factorization model to predict movie ratings for Netflix. Quan [7] propose the Geographical-Temporal influences Aware Graph (GTAG) to exploit both geographical and temporal influences in time-aware POI recommendation. Chen [8] designs a Gibbs sampling algorithm to learn the receptiveness among users over time. Xiang [19] proposes a Session-based Temporal Graph (STG) to model long-term and short-term preferences and strengthen the impact of current interests via Injected Preference Fusion (IPF) method. Lathia et al. [16] and Wang [17] improve temporal diversity of recommendation across time. Compared to these work handling the temporal dynamics in different ways, SimIUC focus on modeling the dynamic evolution process of the user interests which is composed of two stages: rising stage and declining stage, and making accurate top-N recommendation based on the user interest model.

User Interest in Recommendation. In the past few years, how to capture user interest accurately has become the key issue and challenge in personalized recommendation. User interest changes over time. There are many studies using monotonously decreasing function to model user interest. Koychev [15] uses linear functions to simulate the decay of user interest. Ding et al. [6] and Andreas [13] both propose a time weighted item-based collaborative filtering method (TItemKNN) by reducing the influence of old data when predicting user's further behaviors. Chen [12] adapts the memory forgetting curve to model the human interest-forgetting curve for music recommendation. However, the method has two disadvantages. One is that the latest data is not always important while old data is not worthless all the time. The other is that users have a personalized interest model, and there are differences in the amount of interests, interest strength, interest forgetting rate and the trend of user interest evolution. It is not enough reasonable to use uniform interest model to all users.

The paper focuses on discovering user interest patterns and the trend of interest evolution by mining user interaction logs. To the best of our knowledge, this is the first work that use inverted U curves to model user interests in recommendation.

7 Conclusion and Future Work

A user's interests have a dynamic evolution process, including two stages, growth and decay. Modeling and leveraging this dynamic process for temporal recommendation poses great challenges. In this paper, we proposed a novel recommendation framework named SimIUC, which can identify multiple user interests and model the dynamic evolution process of user interests with the inverted-U-curves. Based on that, SimIUC can predict user interest evolution trend in the future and make more accurate recommendation. The experimental results have shown a significant improvement in the accuracy of our top-N recommendation method compared with the baselines. In our future works, we will try to predict the potential transfer directions of a user's interests, and acquire new interests a user just formed.

Acknowledgements. This work is supported by NSF of China (No. 61303005), 973 Program (No. 2015CB352500), NSF of Shandong, China (No. ZR2013FQ009), the Science and Technology Development Plan of Shandong, China (No. 2014GGX101047, No. 2014GGX101019). This work is also supported by US NSF grants III-1526499, and CNS-1115234.

References

1. Silvia, P.J.: Exploring the Psychology of Interest. Oxford University Press, New York (2006)
2. Stewart, B., Mark, M.: The U-curve adjustment hypothesis revisited: a review and theoretical framework. J. Int. Bus. Stud. **22**, 225–247 (1991)
3. Zajonc, R.B.: Attitudinal effects of mere exposure. J. Pers. Soc. Psychol. **19**(2), 77–78 (1968)
4. Stigler, G.J.: The adoption of the marginal utility theory. Hist. Polit. Econ. **4**(2), 571–586 (1972)
5. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: CIKM 2001, pp. 247–254 (2001)
6. Ding, Y., Li, X.: Time weight collaborative filtering. In: CIKM 2005, pp. 485–492 (2005)
7. Quan, Y., Gao, C., Aixin, S.: Graph-based point-of-interest recommendation with geographical and temporal influences. In: CIKM 2014, pp. 659–668 (2014)
8. Chen, W., Hsu, W., Lee, M.L.: Modeling user's receptiveness over time for recommendation. In: SIGIR 2013, pp. 373–382 (2013)
9. Deshpande, M., Karypis, G.: Item-based top- N recommendation algorithms. ACM TOIS **22**(1), 143–177 (2004)
10. Newman, M.: Power laws, pareto distributions and Zipf's law. Contemp. Phys. **46**(5), 323–351 (2005)
11. Sun, Y., Han, J., Yan, X., Wu, T.: Pathsim: meta path-based top-k similarity search in heterogeneous information networks. In: VLDB 2011 (2011)
12. Chen, J., Wang, C., Wang, J.: Modeling the interest-forgetting curve for music recommendation. In: MM 2014, ACM (2014)
13. Toscher, A., Jahrer, M., Bell, R.M.: The bigchaos solution to the Netflix Grand prize (2008)
14. Koren, Y.: Collaborative filtering with temporal dynamics. In: KDD 2009, pp. 447–456 (2009)

15. Koychev, I., Schwab, I.: Adaptation to drifting user's interests. In: ECML 2000 Workshop: Machine Learning in New Information Age (2000)
16. Lathia, N., Hailes, S., Capra, L., Amatriain, X.: Temporal diversity in recommender systems. In: SIGIR 2010, pp. 210–217 (2010)
17. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: KDD 2003, pp. 226–235 (2003)
18. Senzhang, W., Xia, H., Philip, S.Y., Zhoujun, L.: MMRate: inferring multi-aspect diffusion networks with multi-pattern cascades. In: KDD 2014, pp. 1246–1255 (2014)
19. Liang, X., Quan, Y., Zhao, S., Chen, L., Zhang, X.: Temporal recommendation on graphs via long-and short-term preference fusion. In: KDD 2010, pp. 723–732 (2010)
20. Senzhang, W., Sihong, X., Xiaoming, Z., Zhoujun, L., Philip, S.Y., Xinyu, S.: Future influence ranking of scientific literature. In: SDM 2014, pp. 749–757 (2014)
21. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Am. Math. Soc. **7**, 48–50 (1956)
22. MovieLens: http://grouplens.org/datasets/movielens
23. Netflix: http://www.netflixprize.com