IJICTR International Journal of Information & Communication Technology Research

Volume 4- Number 4- September 2012 (69-77)

Research Note

An Improved Recommender System Based on Forgetting Mechanism for User Interest-Drifting

Rozita Tavakolian
Information Technology Engineering Department
Tarbiat Modares University
Tehran, Iran
rozita_tavakolian@yahoo.com

Mohammad Taghi Hamidi Beheshti
Faculty of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
mbehesht@modares.ac.ir

Nasrollah Moghaddam Charkari
Faculty of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
charkari@modares.ac.ir

Received: September 25, 2011- Accepted: May 30, 2012

Abstract— **Highly effective recommender systems may still face users' interest drifting. One of the main strategies for handling interest-drifting is forgetting mechanism. Current approaches based on forgetting mechanism have some drawbacks: (i) Drifting times are not considered to be detected in user interest over time. (ii) They are not adaptive to the evolving nature of user's interest. Until now, there hasn't been any study to overcome these problems. This paper discusses the above drawbacks and presents a novel recommender system, named *WmIDForg*, using web usage mining, web content mining techniques, and forgetting mechanism to address user interest-drift problem. We try to detect evolving and time-variant patterns of users' interest individually, and then dynamically use this information to predict favorite items of the user better over time. The experimental results on *EachMovie* dataset demonstrate our methodology increases recommendations precision 6.80% and 1.42% in comparison with available approaches with and without interest-drifting respectively.**

Keywords- ***web recommender system; users' interest drifting; forgettin mechachanism; web usage mining; web content mining; time-based hybrid weight.***

I. INTRODUCTION

Web mining is the mining of data related to the World Wide Web. It is categorized into three active research areas according to what part of web data is mined [1]: Web content mining (WCM), which is the process of extracting knowledge from the content of website, for example, contents of documents and their descriptions; Web usage mining (WUM) which studies user access information from log files in order to extract interesting usage patterns; Web structure mining (WSM), which uses links and references within web pages to obtain the underlying topology of the interconnections between web objects. Today, web mining techniques are widely used in web recommender systems [2].

A web recommender system tries to predict the web user interests and preferences, based on data

previously collected from them explicitly or implicitly [3]. Today, most of web recommender systems are based on the combination of web usage mining and web content mining.

In general, one of the major problems is that the user's interest changes over time. The old interest is gradually forgotten, and the new one is created. This phenomenon is called as interest drift. Interest drift is an instance of concept drift, a widely researched problem in machine learning [4]. Users may change the items which have been rated highly in the past. For example, a new mother may be interested in baby toys, although she had no interest in these previously. Or even in a movie recommender system, users may change their preferences in different genres or adopt a new viewpoint on an actor or director. In this regard, a user may currently like romance movies while he might have been interested in action movies some time ago. However, if a recommendation system is not aware of the changes, its prediction will not be appropriate.

One of the main strategies for handling interest drifts is forgetting mechanism [4]. Since the recent observations of the user represent the current interest of the user better than older ones, this mechanism uses a time function to assigns time weights to items so that the items visited recently have more contribution to the recommendation than items visited more previously.

In the recent years, some researchers have focused on addressing user's interest drifting problem. In [5] a hybrid filtering algorithm has been introduced to compute the time weights for different items. Accordingly, a decreasing weight is assigned to the old data. Song et al. developed similarity and difference measures for rule matching to detect changes of customer behavior [6]. In [7] two new data weighting methods: time-based data weight and item similarity-based data weight were proposed to cope with the changes of user interests. Li et al. used a dynamically building decision tree on streaming data, to catch up the interest drifting of users in a traditional collaborative filtering recommendation [8]. Li and Feng developed a computation model to estimate the user interest of the accessed pages through analyzing users' browsing behaviors. They employed time function to give more importance to the recent observations [9]. A personalized service which recommends resources to users at massive education resources net has been proposed in [10]. Min and Han [11] developed a modified collaborative filtering based approach which finds the active user's neighbors according to his/her changing pattern. Their approach didn't pay any attention to interest changes of other users. In [12], a mechanism named *WebProfiler* learns a hierarchical representation of user interests using conceptual clustering, has been proposed. Interests at the top levels of the hierarchy can be seen as long-term interests, while more specific ones can be seen as shorter-term interests. The hierarchy adapts profiles to changes in user interests according to the feedback received from users. Furthermore, a time-based forgetting has been used to give weight to different web paged in the hierarchy.

Available approaches based on interest drifting by the use of forgetting mechanism, have two drawbacks: (1) Drifting times of user interest have been neglected. In order to handle interest drifting problem, they give weights to user observations according to their appearance over time, even if no change occurred.

That will be right in only situations that user's interest changes every time he/she visits the website or buys something new. In fact, we know that not always user's interest changes, And the trend of changes in user purchase interest is also different. In some periods of time, the user's interest may change quickly, And in other periods, it may last much longer. For example, in a TV show website a user who has tended to watch action movies for one year, suddenly changed his interest to horror movies. And one month later, he got interested in romance movies. In other words, his inertest in an item has been stable for a long time, and it has quickly changed twice in 2 months.

If a user has stable interest during time, some items in his/her profile are still of his/her current interest. In such cases, assigning weights to all observations, which is done by previous approaches, causes these items to lose their impact on prediction. Hence, it results in unsuccessful recommendations. In general, if no change in user interest occurred, no time weight must be used. (2) The other drawback is that they haven't been adaptive to the evolving nature of users' behavior. Navigations of a web user contain gradual evolution of his/her information needs. Therefore, several observations together may represent a unique interest. If an interest have previously changed, its impact must decrease on prediction. Furthermore, if some observations represent one interest of the user, they must be assigned the same weight because they have the same contribution to prediction. As interest drifting is important in recommender systems, considering evolving user interest can increase the precision of recommendations.

In order to overcome the problems above, we develop a novel approach based on forgetting mechanism to (1) trace the behavioral patterns of web users, and then identify evolving and time-variant interest of any user with the lapse of time. (2) and capture the importance of different interests per user and evaluate their impact on recommendation using the information on changes. The results of experiments confirm that our method is able to improve the system's adaptability to user's interest drifting and recommendations precision.

Accordingly, our approach attempts to answer the following questions:

1. *How to detect a change in the user interests?*
2. *How to adapt with the evolving nature of user' interest?*
3. *How the algorithm deals with the above in addressing user interest drifting problem?*
4. *Whether our approach can help improve recommender systems? If the answer is "yes", how much improvement can be achieved?*

To the best of knowledge, no papers have focused on both web usage mining and web content mining to address interest drifting using forgetting mechanism. Our method is based on the combination of web usage mining and web content mining techniques. We name it *WmIDForg* which denotes using Web Mining techniques for user Interests-Drifting based on FORGetting mechanism.

The rest of the paper is organized as follows; Section II describes our proposed approach. The experiments are given in section III. And a short conclusion and some suggestions for future works end the paper in section IV.

II. PROPOSED FRAMEWORK

A. Overall Procedure

Our approach, *WmIDForg*, is based on the mining of users rating in the web and the contents of the retrieved web pages to cope with available problems in users' interest drifting context and improve the ability to adapt to evolving and changing behavior of users. The architecture of *WmIDForg* is shown in Fig. 1.

As it is shown, firstly, clustering is done on web pages. Then, we create a Time-to-Category Rating matrix using the generated clusters and web users' ratings to detect interest drifting of each user individually. Then a new hybrid time-based weight is assigned to interests of each user to reflect their real importance. We apply a weighted association rule mining technique to mine time-variant behaviors. Finally, the generated rules are used to estimate next favorite items for an active user. The entire steps in the proposed methodology appear in a pseudo-code in Fig. 2.

In the following subsections, we will explain each step in detail.

B. Detecting User Interest Drifting

As discussed above, finding drifting times in users' interest is crucial for accurate recommendations. Previous approaches based on interest-drifting don't consider drifting times to be detected in users' interest over time. This module, which includes five steps, purposes to catch the interest changes of users and the times at which the changes occur.

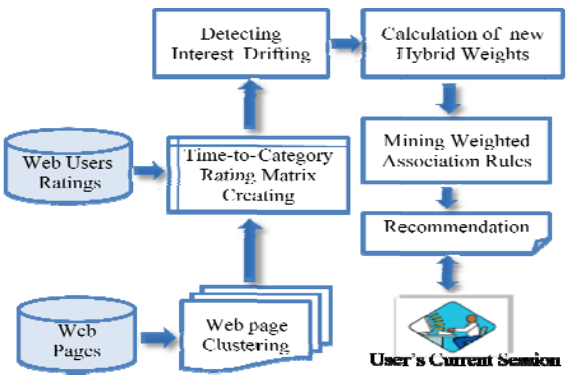

Figure 1. The architecture of *WmIDForg*

Algorithm 1: *WmIDForg*

Input: Web pages, Preprocessed web user logs.
Output: The next favorite item of an active user.
Begin
Step1. Cluster web pages.
For each user u_i :
Step2. Create Time-to-Category Rating Matrix according to descriptions in section B and step 2.
Step3. For each row, and each category k in Time-to-Category Rating Matrix :
3.1. Calculate category rating cr_k using Eq. (1).
Step4,5. Do algorithm2 in Fig. 3 to detect drifting times of interest of u_i.
End For.
Step6. Calculate new weight for all users' ratings using t_r obtained in *Step4,5.* , Eq. (3) and Eq. (5).
Step7. Do Weighted Association Rule Mining.
Step8. Recommend the next favorite item to an active user.
End.

Figure 2. Pseudo code of the *WmIDForg*

Furthermore, in an electronic shop, users have different purchase behaviors, so that their interest changes will not be the same over time. Some users may change their interest quickly during time, while others might have more stable interests over time and their interests change at a low speed. Thus, we intend to detect interest changes per user individually.

We modify the approach which has been presented in [11] to find interest drifts, as follows:

Step 1. Web pages clustering:

We use a hybrid genetic algorithm and simulated annealing [13] with k-means algorithm to categorize web pages based on Euclidean distance. As regards a user has the same tendency towards similar products, despite the [11], we cluster items based on content similarity. Items belonging to a specific category have the same features, and can be used for representing the interest of a user in a specific kind of products. We will show it in the next steps.

Step 2. Creating a Time-to-Category Rating Matrix per user:

In Time-to-Category Rating matrix, each column coincides with one of the categories generated in previous step, while rows denote timeframes which is different time intervals. Number in row i and column I_j show the user rating to item I_j at timeframe i^{th}. The main goal behind Time-to-Category Rating Matrix is to represent the interests of a user in various kinds of items in different timeframes.

Step 3. Category rating Calculation:

The category rating is defined as the average ratings for the items in that category as follows:

$$cr_{u,k} = \sum_{i \in category,k} \frac{1}{RN_k} r_{u,i} \qquad \text{Eq. 1}$$

$cr_{u,k}$ is the derived rating of category k of user u, RN_k is the number of rated items belong to that category. $r_{u,i}$ is the rating of item i of user u. Category ratings are found for each category and each user at different timeframes. Category ratings might be a proper indices to show how much user u is interested in category k^{th}.

Step 4. Correlation Between different interests:

This step calculates the correlation between different interests of each user at different timeframes. The calculation is done by using Auto-Similarity measure (AS) discussed in [11]. Here, an interest change per user is found by Eq. (2). This measure is used for detecting the degree of changes. Auto-Similarity $AS(u)_{T1,T2}$ is defined as the similarity between the category ratings of user u at timeframes $T1$ and $T2$, as follow:

$$AS(u)_{T1,\,T2} = \frac{\sum_k \left(cr(T1)_{u,k} - r(\overline{T1})_u\right)}{\sqrt{\sum_k \left(cr(T1)_{u,k} - r(\overline{T1})_u\right)^2}} \times \frac{\left(cr(T2)_{u,k} - r(\overline{T2})_u\right)}{\sqrt{\sum_k \left(cr(T2)_{u,k} - r(\overline{T2})_u\right)^2}} \quad \text{Eq.2}$$

$cr(T1)_{u,k}$ is user u's category ratings for category k at timeframe $T1$. Moreover, $r(\overline{T1})_u$ is the average rating of the same user u at the same timeframe $T1$.

Step 5. Changes Detection:

If the value of Eq. (2) is less than a predefined threshold δ, we will find out that a change in the user interest has been occurred after time $T1$.

In real world, the timeframe size depends on how fast the user tend to forget his/her interests. If the user preference changes frequently or quickly over time, then a short timeframe will be appropriate. In this situation, more timeframes will be required in the Time-to-Category Rating Matrix to detect all drifts. Conversely, if the user preference is about to be stable over time, then we increase the size of timeframe. Later, we will show that how the timeframe size influences the performance of algorithm.

Algorithm 2 in Fig. 3 describes briefly how to detect time-variant interests of users.

After detecting drifting times of users' interest, we assign them weights as explained in the next subsection.

Algorithm 2: Detecting drifting times of user's interest

Input: Time-to-Category Rating Matrix, Threshold δ.

Output: Drifting times of interest, t_r.

Begin

1. $t_m = 0$.

   / m is the last timeframe in Time-to-Category Rating Matrix.*/*

2. For $j = m - 1$ to 1 :

   2.1. Calculate $AS(u)_{Tj,Tm}$ between timeframes Tj and Tm using Eq. (2).

   2.2. If $|AS(u)_{Tj,\,Tm}| < \delta$ *//a drift occurred.*

   2.2.1 $t_j = Tm - Tj$.

   Else

   2.2.2. $t_j = t_{j+1}$.

End.

Figure 3. Pseudo code of Algorithm 2 for detecting drifting times of user's interest

C. *Hybrid Weight Computation*

In order to capture the real interest of the user to in items, we use forgetting mechanism to assign a hybrid weight to ratings based on combining user ratings and time using Eq. (3) where Wu,j and Ru,j denote new weight of item Ij for user u, and rating user u has assigned to item Ij respectively. $F(t_r)$ is a time function.

$$W_{u,j} = R_{u,j} \times F(t_r). \quad \text{Eq. 3}$$

Several reasons validate the idea of using hybrid weight computing in this paper:

- An item with more rating has more importance to customers.
- Customer preferences are time-sensitive. A later rating on items is more indicative to show the current user interest [14]. In other words, not only high ratings indicate the importance of an item to a user; Recent ratings also should be able to reflect the current interests of a user more accurately. If ratings of items are all time-stamped, the most recently rated items of the user will have a larger impact on the prediction of user preference than items that was rated more previously.

The preferences of users usually change with time. Some preferences change quickly, while others may change at a low speed. As mentioned in section I, previous studies applied a time function to blindly assign a smaller weight to old ratings and larger weight to newer ratings of users [5,7,9,10,14]. They don't detect any drift and multiply the function by all user observations. In the circumstances that a user interest didn't change, it would make the impact of that interest decrease on prediction step, which leads to unsuitable recommendations. In this paper, we apply time function whenever an interest drift occurs.

User history contains gradual evolution of user information needs. Therefore, several observations of the user in the website may indicate a unique interest. Hence, we give equal weight to the observations of an interest.

The following example illustrates our approach in finding individually the interest drift of any user, and then giving weight to each interest:

Suppose that table I is a Time-to-Category Rating Matrix of a user u_i where Tr $(1 \le r \le m)$ is r^{th} timeframe. Tm and $T1$ are respectively the last and the first timeframe at which user u_i has done his/her ratings. Now, the new weights for items are computed.

Firstly, we determine the value of tr as follows:

- Whereas the numbers at time Tm are the last ratings which user has done, they indicate the current interest of the user. Therefore, we set t_r=0 for all ratings at this time.
- In other rows, $t_r \neq 0$. If a drift is detected at timeframe Tr, then t_r will be the time distance between Tr to Tm. For example if Tm is November 1995 and Tr is August 1995, then t_r for all items at timeframe Tr will be equal to 91 days.
- If no interest changes at timeframe Tr, then t_r will be equal with the value of t_r at timeframe $Tr+1$.

Eq. (4) shows the above discussion in brief.

$$t_r = \begin{cases} 0 & if\ r = m \\ t_{r+1}, & if\ r \neq m\ AND\ interest\ \ doesn't\ change. \\ T_m - T_r, & if\ r \neq m\ AND\ interst\ \ changes. \end{cases} \quad \text{Eq. 4}$$

Here, we apply Eq. (5) used in [5] as our time function.

$$F(t) = e^{-\lambda t} \quad \text{Eq. 5}$$

It can satisfy our needs well. As $F(t)$ is a monotonic decreasing function, which reduces uniformly with time t. The value of time weight lies in the range (0,1). In other words, all the data contribute to the recommendation. The more recent the data is, the higher the value of the time function is. A half-life parameter T0 is defined as:

$$F(t_0) = \left(\frac{1}{2}\right) F(0) \quad \text{Eq. 6}$$

TABLE I. TIME-TO-CATEGORY RATING MATRIX

	Category 1			...	Category K		
	I_1	...	I_{n1}		I_{1628-k}	...	I_{1628}
T_m	1	...	5	...	0	...	3
T_{m-1}	2	...	4	...	2	...	0
...	...	...	...	...	...	...	...
T_2	3	...	1	...	4	...	2
T_1	2	...	0		0	...	1

It means that the weight is reduced to 1/2 initial value in $T0$ days. The decay rate λ is defined as:

$$\lambda = \frac{1}{T_0} \quad \text{Eq. 7}$$

Conceptually, the aim of designing a half-life parameter is to define the rate of decay of the weight assigned to each interest. From the Eq. (7), we can see that $T0$ is inversely proportional to λ. The higher the value of λ, the faster the old data decays and the lower the importance of the historical observation compared to more recent data. In other word, if the change between the interests is significant, the larger parameter λ might be selected in Eq. (5) to diminish the impact of older interests. Therefore, the selection of parameter λ is a key issue to the performance of algorithm. We should select different values of parameter $T0$ under different circumstances in experimental step.

After obtaining the real importance value of each item to each user, we extract weighted association rules discussed in the next section.

D. Mining Weighted Association Rules

Discovery of association rules has been found successful in many recommender systems. In previous works, all items purchased are treated uniformly. Weighted association rule mining allows different weights to be assigned to different items to reflect their importance to the user. The weights represent as the profitability of different items, or the popularity of different items for the user. Weighted association rule mining pays more attention to items with higher importance [16]. In other words, despite the classic association rule mining which extracts frequent item sets, it is able to find item sets with more importance but less frequency.

In the weighted association rule mining algorithm, input is represented as a sequence of weighted interests obtained in subsection C. The result is expressed as follows:

$$r_a = \{(p_1, w_1), ..., (p_k, w_k) \Rightarrow (p_{k+1}, w_{k+1}), ..., (p_{k+n}, w_{k+n}), S_a, C_a\}$$

where $\{(p_1, w_1), ..., (p_k, w_k)\}$ and $\{(p_{k+1}, w_{k+1}), ..., (p_{k+n}, w_{k+n})\}$ represents the left-hand side and the right-hand side of the a^{th} weighted rule respectively. p_i represents each item, and w_i is the weight of the ith item. S_a and C_a are weighted support and weighted confidence of a^{th} rule respectively.

E. Recommendations with Interest-Drifting

After mining time-variant pattern of users, it is time to recommend to an active user items which meets his/her preferences. Suppose that $S = \langle (t_1, r_1), ..., (t_n, r_n) \rangle$ expresses the current browsing session of an active user u, where ti and r_i indicate ith item and user rating to t_i respectively. Therefore, S represents the current interest of u. in order to better adapt with user's interest drifts, we update the profile of u based on the current session and by using the

method described in subsection B. We obtain $S' = \langle (t_1, w_1^s), ..., (t_n, w_n^s) \rangle$ as the sequence of the weighted interest of *u*. Now, we suggest more useful items to u by matching the sequence *S′* with the mined historical behavior patterns as follows:

As mentioned earlier, each of the weighted association rules are represented as a set of item-weight pairs. We can retrieve a rule which has the most similar left-hand side with *S′*. If the left-hand side of the each rule is represented as a vector $R = \{(p_1, w^r_1), ..., (p_m, w^r_m)\}$, then the similarities might be computed using Eq. (8) [17].

$$\text{Similarity}(R) = \text{Confidence}(R) \times M(S', R)$$

$$M(S,R) = \frac{\sum_k w^r_k \cdot w^{s'}_k}{\sqrt{\sum_k (w^r_k)^2 \times \sum_k (w^{s'}_k)^2}} \qquad \text{Eq. 8}$$

After retrieving a rule with the maximum value of Similarity(R), we recommend to active user items with the highest weight in the right-hand side of the rule.

III. Experimental Results

To gauge how well our proposed recommender system performs, we carried out our experiments on the EachMovie dataset[1] that have been one of the most widely used common datasets in recommender system researches. The *EachMovie* dataset has the rating information on 1,628 movies by 72,916 users within an 18-month-period since 1996. Users have rated various numbers of movies by rating values between 0 and 1. Each vote is accompanied by a time stamp. We have collected the content of each movie from Internet Movie Database (IMDB) website [2] using a web crawler. These contents are web pages include movie title, director, genre, plot summary, cast, award and user comment about each movie. We have selected a subset of 466,245 ratings included of 1,628 movies and 2,880 users such that all of them have rated more than 100 movies. We divided the dataset into a 90% for the training set and 10% for the test set.

In clustering step, web pages have been preprocessed by HTML tag removal, word extraction, stop word removal, stemming, TF-IDF weighting and feature selection [18]. In this regard, a table with 1,628 documents and 4,290 terms has been created. An ART2 neural network has been used for finding the correct number of clusters K and the initial points [19].

Accordingly, K has been found as 96. We used the hybrid genetic algorithm and simulated annealing with Kmeans algorithm [20,21] as our clustering algorithm to minimize the total within cluster variances (TWCV), and run it for several times to get the best result. Table II, presents the minimum, mean and maximum values of TWCV for different 15 runs. The minimum result is finally selected as output.

We reimplemented the algorithm of weighted association rule mining presented in [16]. The values of minimum support and confidence were set to 0.6 and 0.5 respectively.

The leave-one-out method [22] is conducted for our experiments. The quality of recommendation is evaluated by Mean Absolute Error (MAE), which is a widely used metric. It can be defined as the average absolute difference of predictions from the real user ratings. In our experiments, we find MAE on the test set for each user by Eq. (9). Then, the average over the test set of a user is found by Eq. (10) [5]. A smaller value means a better performance.

$$MAE_i = \frac{\sum_{j=1}^{n_i} |ar_{i,j} - r_{i,j}|}{n_i} \qquad \text{Eq.9}$$

$$MAE = \frac{\sum_{i=1}^{m} MAE_i}{m} \qquad \text{Eq. 10}$$

A. Parameter Settings

There are three major parameters in the proposed approach: the timeframes size in section I and subsection A, the threshold δ used to determine the correlation of user interests at different timeframes, and the decay rate λ in Eq. (5). We conducted a series of preliminary experiments to estimate the optimal performance obtained for each parameter when the value of other is constant.

- Experiment (1): the impact of timeframe size

In the first experiment, we change the timeframe size parameter from 1 to 4 months. The impact of using different values is demonstrated in Fig. 4. This parameter value depends on how fast users tend to change the interest over time. As discussed before, when the speed of changes is high, then the size of time frame should be selected smaller to be able to find all drifts. Conversely, if the changes happen less quickly over time, the timeframe size can be selected larger. It is found that timeframe size dramatically influences the performance of the algorithm. It is necessary to assign different values to this parameter to achieve the best performance.

- Experiment (2): the impact of δ

We change the value of δ to evaluate its influence on the performance of the algorithm. Results are shown in Fig. 5. As it is seen, MAE values increase with δ. Recall that in Section II, if the similarity of two interests is less than δ, it means a drift has been occurred. Obviously, the lower the value of δ is, the less drifts detected. Conversely, if δ increases, more interest drifts are detected. It is possible that two

[1] www.research.compaq.com/SRC/eachmovie
(It' s no longer available in that website.)
[2] www.imdb.com

interests are similar but if the value of δ is selected too large, then they may consider distinct. Therefore, it seems reasonable that the higher value of δ results the higher value of MAE.

- Experiment (3): the impact of λ

To experimentally determine the impact of the decay rate λ on the quality of the prediction, we selectively vary the parameter to 20, 50 and 100. Therefore, the value of λ will be 0.05, 0.02, and 0.01 respectively. Fig. 6 shows the MAEs of the proposed algorithm in different cases.

TABLE II. THE TOTAL WITHIN CLUSTER VARIANCES FOR THE CLUSTERING RESULTS

	$TWCV_{min}$	$TWCV_{average}$	$TWCV_{max}$
Result	5562.324229	5609.873818	5659.292831

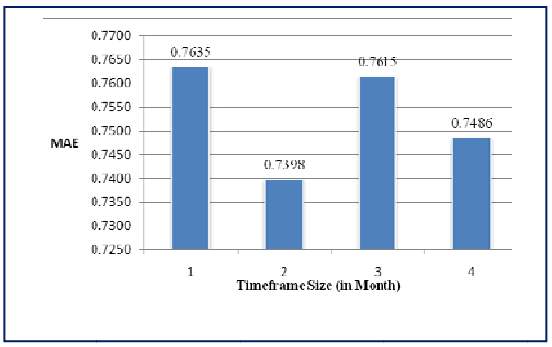

Figure 4. The impact of timeframe size

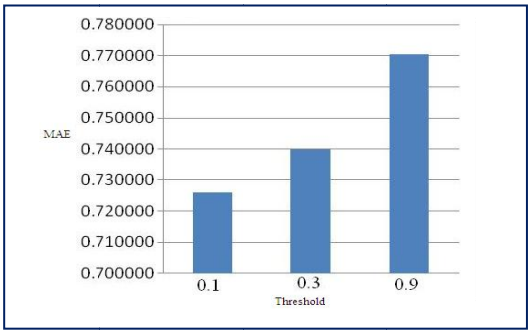

Figure 5. The impact of threshold

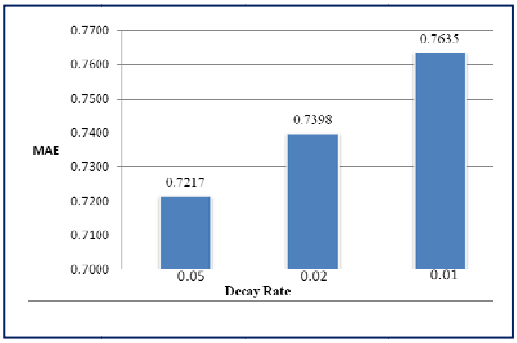

Figure 6. The impact of decay rate

B. Overall Performance

Some experiments have been conducted to demonstrate the effectiveness and the efficiency of our proposed approach. We compare our results with the approach of [5] and the approach that doesn't consider interest drifting. Ma et al. [5] used collaborative filtering with interest drifting where drifting time is not considered to be detected in user interests over time, and then the time function is multiplied by each rating of user even if no change occurred. We re-implement this method and compare it with our proposed method. For the second algorithm, we use a classic weighted association rule mining algorithm without interest drifting where the weight of items is only user ratings. Here, we called it *C-WARM*.

The results are evaluated based on MAE metric and showed in Fig. 7. It is found that our proposed approach performs 6.80% and 1.42% better than [5] and *C-WARM* respectively.

Our approach, *WmIDForg*, outperforms for following reasons:

- As discussed in section II and subsection C, not only ratings indicate the importance of an item to a user. Customer preference changes with time, and therefore earlier observations of users have less impact to predict their current information needs. *C-WARM* considers only ratings of users on different movies. Hence, the precision of our approach which is able to catch the real importance value of each item per user, would be better than *C-WARM* which isn't adaptive to drifts.

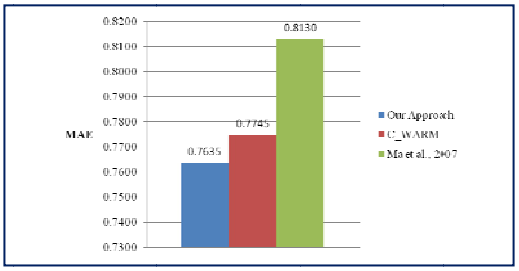

Figure 7. MAE values of different algorithms on *EachMovie*

- Ma et al. and other similar approaches neglect drifting times of user behaviors, and multiply the time function in all observations of the user to deal with their interest drifting. It diminishes the impacts of all items. In some situations where the user has a stable interest or his/her interest drifting occurs gradually over time, it makes the impact of items which are still of his/her current interest decrease, which finally results in poor recommendations. In other words, their method is useful in only situations where the interest of all users changes after each time user visits the website. That is not always true. We used time function only whenever an interest drift occurs. Our method is able to track and identify interest drifts of any user and decrease the score of any interest when a drift takes place.

- On the other hand, web user observations contain gradual evolution of his/her information needs and therefore several observations together represent a unique interest. These observations must have the same contribution in the predictions. Despite previous studies, we found evolving interests of users' and then assigned equal weight to all the items related to one interest.

IV. Conclusion and Future Work

In this paper, we developed a novel approach, named *WmIDForg*, based on web usage mining and web content mining techniques which individually detect evolving and time-variant patterns of user's interest. Moreover, it computes real importance degree to each interest of the user. The proposed method consists of four phases of (1) detecting the sequence of different interests per user over time, (2) computing the importance value of different items for the user using a hybrid weight, (3) mining time-variant pattern and (4) finally recommendation.

We conducted some experiments to evaluate the prediction quality on the EachMovie dataset and compared them with the approach of [5] and the classic approach that doesn't take interest drifting into account. The results show the improvement of our proposed method in the ability to adapt to user's interests drifting and making recommendations.

As future work, we intend to evaluate WmIDForg on other dataset. To our knowledge, no papers has focused on both web usage mining and web content mining to address interest drifting using forgetting mechanism. We would also like to extend our method with other web mining techniques such as web structure mining. According to the discuss in section III and subsection A, timeframe size has considerable impact on the performance of proposed method. It can be useful to obtain its appropriate value. We plan to use a sliding-window method [23], and apply heuristics methods on it to adjust properly the timeframe size for each user.

In a shopping market, users' interest may change periodically. As an example, a user may have different buying habit in the weekends compared to the other days. In such situations, it seems no interest change occurs. Considering such periodic changes in detecting different user interests over time may increase the precision of recommendation system. How to handle the periodic changes can be the further research.

References

[1] M.H. Dunham, Data Mining: Introductory and Advanced Topics, Prentice Hall, 2003.

[2] M. Eirinaki and M. Vazirgiannis, "Web mining for web personalization", Journal ACM Transactions on Internet Technology (TOIT), vol. 3, 2003, pp. 1-27.

[3] H. Liu and V. Kešelj, "Combined mining of Web server logs and web contents for classifying user navigation patterns and predicting users' future requests", Data & Knowledge Engineering, vol. 61, 2007, pp. 304-330.

[4] I. Koychev, "Tracking Changing User Interests through Prior-Learning of Context", In de Bra, P., Brusilovsky, P., Conejo, R. (eds.): Adaptive Hypermedia and Adaptive Web Based Systems. Lecture Notes in Computer Science, Springer-Verlag, vol. 2347, 2002, pp. 223-232.

[5] S. Ma, X. Li, Y. Ding and M. E. Orlowska, "A Recommender System with Interest-Drifting", In Web Information Systems Engineering – WISE 2007, Lecture Notes in Computer Science, vol. 4831, pp. 633-642, 2007.

[6] H. S. Song, J. K. Kim and S. H. Kim, "Mining the change of customer behavior in an internet shopping mall", Expert Systems with Applications, vol. 21, 2001, pp. 157-168.

[7] CX Xing, FR Gao., SN. Zhan and LZ. Zhou, "Collaborative Filtering Recommendation Algorithms Incorporated with User Interest Change", Journal of Computer Research and Development, vol.44, no.2, 2007, pp. 296-301.

[8] X. Li, J. M. Barajas and Y. Ding, "Collaborative filtering on streaming data with interest-drifting", Intelligent Data Analysis, vol.11, no.1, 2007, pp.75-87.

[9] Y. Li and B. Q. Feng, "Page Interest Estimation Model Considering User Interest Drift", in Proceedings of 2009 4th International Conference on Computer Science & Education, pp. 1893-1896, July 2009.

[10] Y. Jing, X. Li and S. Zhoung, "Research on Personalized Services for Users of Education Resources Net", in 2009 Ninth International Conference on Hybrid Intelligent Systems, Computer Society, vol. 3, pp 195-197, 2009.

[11] S. H. Min and I. Han, "Detection of the customer time-variant pattern for improving recommender systems", Expert Systems with Applications, vol. 28, 2005, pp. 189–199.

[12] D. Godoy and A. Amandi, "Interest Drifts in User Profiling: A Relevance-Based Approach and Analysis of Scenarios", The Computer Journal Advance, vol. 52, no. 7, 2009, pp. 771-788.

[13] A. E. Eiben and J. E. Smith, Introduction to Evolutionary Computing, 1st ed., Springer, Berlin, 2003.

[14] Y. Ding and X. Li, "Time weight collaborative filtering", in the CIKM 2005. Proceedings of the 14th ACM international conference on Information and knowledge management, ACM Press, New York, NY, USA, pp. 485–492, 2005.

[15] Y. Hong and Z. Li, "A Collaborative Filtering Method Based on the Forgetting Curve", Journal of Nanjing University(Natural Science), vol. 1, 2010, pp. 183-187.

[16] N. Zhou, J. X. Wu, S. L. Zhang, H. Q. Chen and X. R. Zhang, "Mining Weighted Association Rules with Lucene Index", In: Proc. of WiCom'07. Shanghai, China, pp. 3697-3700, September 2007.

[17] YS Kim, "Streaming association rule (SAR) mining with a weighted order-dependent representation of Web navigation patterns", Expert Systems with Applications, vol. 36, 2009, pp. 7933–7946.

[18] C. D. Manning, P. raghavan and H. Schutze, An Introduction to Information Retrieval, Cambridge University Press, 2008.

[19] M. D. Shieh, W. Yan, Ch.H. Chen, "Soliciting customer requirements for product redesign based on picture sorts and ART2 neural network", Expert Systems with Applications, vol. 34, 2008, pp. 194–204.

[20] S.F Hwang, R.S He, "Improving real-parameter genetic algorithm with simulated annealing for engineering problems ", Advances in Engineering Software, vol. 37, 2006, pp. 406-418

[21] D. Gong, F. Pan; X. Sun, "Research on a novel adaptive genetic algorithm" Industrial Electronics, 2002. ISIE 2002. Proceedings of the 2002 IEEE International Symposium on , vol. 1, pp. 357 – 360, November 2002

[22] L.B. Marinho and L.S. Thieme, "Collaborative Tag Recommendations", Data Analysis, Machine Learning and Applications, 2008, pp. 533-540.

[23] G. Giannakopoulos and T. Palpanas, "the Effect of History on Modeling Systems' Performance: The Problem of the Demanding Lord", Data Mining (ICDM), 2010 IEEE International Conference on, Sydney, pp. 809-814, December 2010.

Rozita Tavakolian received her B.Sc. degree in computer engineering (software) from Shiraz University in 2006 and the M.Sc. degree in Information Technology engineering from Tarbiat Modares University of Tehran in 2010. Her research area includes Evolutionary Algorithms, Web Mining, and Web Recommender Systems.

Mohammad Taghi Hamidi Beheshti received his M.Sc. and Ph.D. degrees in electrical engineering from Wichita State University, Wichita, KS. in 1987 and 1992 respectively. He is an associate professor in department of ECE, Tarbiat Modares University, Tehran, Iran. His research interests are robust optimal control of singularly perturbed systems, and quality of service of communication systems.

Nasrollah Moghaddam Charkari received his M.Sc. and Ph.D. degrees in computer engineering and information systems engineering from Yamanashi University, Japan in 1992 and 1995 respectively. He is an associate professor in faculty of ECE, Tarbiat Modares university, Tehran, Iran. His research topics include parallel processing, AI, robotics, computer vision, and image processing.