國立交通大學

資訊科學與工程研究所

碩 士 論 文

動態相似度協同過濾法的推薦系統

Dynamic Similarity Collaborative Filtering of Recommendation System

研 究 生：林凡鈞

指導教授：李素瑛 教授

中 華 民 國 101 年 6 月

動態相似度協同過濾法的推薦系統

Dynamic Similarity Collaborative Filtering of Recommendation System

研 究 生：林凡鈞　　　Student：Fan-Chung Lin

指導教授：李素瑛　　　Advisor：Suh-Yin Lee

國 立 交 通 大 學

資 訊 科 學 系

碩 士 論 文

A Thesis

Submitted to Department of Computer and Information Science

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer and Information Science

June 2012

Hsinchu, Taiwan, Republic of China

中華民國一百一年六月

動態相似度協同過濾法的推薦系統

研究生：林凡鈞　　　　　　　　　　　　　　　指導老師 ： 李素瑛 教授

國立交通大學資訊科學與工程研究所

摘要

在推薦系統的相關研究中，協同過濾法是其中一種最有效率的方法。然而，極少數的研究考慮到時間對於協同過濾法結果的影響。考慮時間因素的協同過濾法的研究中，多數是利用隨著時間衰減評分值來達到時間的目的。然而，因為評分值表示使用者的興趣程度，衰減評分值可能被誤解成人們對於物品的喜好，會隨著時間改變而有所改變，這與多數的事實不符。人們對於喜歡的東西，通常會維持著相同的看法，很少因為時間而改變。這樣的概念也可能造成推薦系統結果的誤差。

因此，我們提出了一個新的方法於動態協同過濾法上。與其衰減評分值，我們更確信衰減人與人之間的相似度是更為合理的想法。人們之間的關係，會因為時間改變了環境，改變人本身的興趣，而有所改變。大部分的人會與現在同處於相同工作或是念書環境的人們較為類似，而與舊朋友間的相似度，很有可能因為時間而有所改變。我們稱此方法為” 動態相似度協同過濾法” 。此外，我們更提出了進階的應用，利用比較預測值與實際值的結果，能使每個使用者，在每個不同的時間點，都有著適合於個人的相似度衰退值。我們確信，每個人的相似度衰退是不會相同的，甚至同一個人在不同的時間點都會有所不同。因此，這樣的方法，不但解除了我們對於設定相似度衰退參數的設定問題外，更增加了預測的成功率。

在實驗的部分，我們提出了多種驗證的方法，證明我們的方法是更符合人們的行為，並且在執行時間上，有了很大的改進，使得我們的方法更適合實際上的用途。

關鍵字：推薦系統，協同過濾法，動態相似度

Dynamic Similarity Collaborative Filtering of Recommendation System

Student : Fan-Chung Lin Advisor : Prof. Suh-Yin Lee

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

In the researches of Recommendation System, Collaborative Filtering is one of the most effective approaches. With high accuracy in recommendations, however, few researches focus on Dynamic Collaborative Filtering which considers the time influence in Collaborative Filtering. This causes the recommendations inappropriate because the system might make a recommendation which is out of date. On the other hand, most of the existing dynamic Collaborative Filtering works are focused on Dynamic Weight. Dynamic Weight Collaborative Filtering uses decay ratings to achieve dynamic property. In other words, the rating might be multiplied by a decay weight according to the rating time. The older the rating is, the lower the rating becomes. Nevertheless, rating decay can also be interpreted as the changes of users' favor. We believe that people would not actually change their perceptions on the same item because of time.

Hence, we propose a different way in Dynamic Collaborative Filtering called Dynamic Similarity Collaborative Filtering (DSCF). The similarities among users are decayed rather than the ratings. In our opinion, we suppose that time might change the similarities among people. We also propose an enhanced method of DSCF. We feedback the predicted rating via actual value in order to obtain a more appropriate similarity decay rate. The experimental results demonstrate the proposed method has higher accuracy and less computation.

Keyword: Recommendation System, Collaborative Filtering, dynamic similarity

誌謝

感謝我的指導教授李素瑛老師，在碩士班期間對我的教導與照顧。李老師不但在研究方面時常給我寶貴的意見與幫助，在日常生活中，更以身教與言教引領著我，使得我在做人處事上更為成熟與圓滿。李老師對於學生的關心與呵護，讓我終身受益，更永遠銘記於心。並感謝口試委員，沈錳坤教授與彭文志教授，不吝於提供多年的寶貴經驗，充實了本論文的深度與廣度，使得本論文更趨於完善。

此外，資訊系統實驗室的學長姐與學弟妹的照顧與支持，每每讓我感受到家的溫暖與溫馨，尤其是陳以錚學長，不斷地提點與照顧，使得我的研究之路更加地順利，也得到了更多的收穫。另外同期的同學，王偉任、陳姿言與郭立言，感謝你們這一路來的扶持與鼓勵，使得這兩年來過得更為的精采。

最後要感謝我的家人，我的父母，林秀愷先生與楊瑞珍女士，在我身後全力的支持著我，兄弟林凡喬，林凡皓再三地提供精神上的支柱，也感謝共同完成學業的吳佩珊小姐一路來的互相照顧。謹以此文獻給我所敬愛的家人與朋友們。

再次感謝一路走來教誨我，陪伴我，支持我的大家，謝謝。

Table of Contents

List of Figures

Chapter 1 Introduction

Since the internet penetration and internet dependence of people have been increasing, e-commerce becomes a popular subject in academia and business field. On the other hand, people pay more attention on social networks which provides a lot of information such as purchasing behavior on internet or interpersonal relationship. Therefore, social network analysis [33] becomes more important in the area of computer science, especially in data-mining.

In the result of the rise of e-commerce, industries lay more emphasis on the internet to either sell products or acquire advertisement compensation through high traffic volume of online viewers. Thus, webs designers intend to provide more functional services in order to attract more users. Recommendation System [1,26] is an example of the online services. Presently, selling products online becomes more competitive than conventional ways due to lower costs and higher profit margin. However, online shopping could possibly make customers confused since the item sets are usually very huge. This problem could also increase the difficulties while searching products. Recommendation System is a solution which not only helps customer search their targets, but also provides potential interests to customers based on the selected product categories [18]. Moreover, Recommendation System has other functional advantages such as personalization or reduction on workload and overhead.

Generally, there are different types of Recommendation System approaches including Collaborative Filtering [11,12,27], content-based, model-based[35], and hybrid[2,5,6,8,22]. The concept of Collaborative Filtering is that similar people would have similar preferences. For example, if user A and B have high similarity, a certain item which user B likes would also be attractive to user A. Collaborative Filtering is one of the most successful approaches for Recommendation System due to its accuracy and simplicity.

Nevertheless, there is a significant problem on traditional Collaborative Filtering. It has been designed based on static idea which means most of the current approaches of Collaborative Filtering do not consider time factor. They treat data of different time equally which causes a result that the recommendations generated by these approaches might be out of date [29,36].

In order to solve this unreasonable result, few researches focused on improving dynamic Collaborative Filtering [9]. Most of the researches are based on dynamic weight approach. The ratings which represent the favor-level of the user would be multiplied by decay function whose value is relevant to the time of the data. The older the data, the lower the rating becomes. It seems a simple way to achieve the time effect. However, the rating represents the level between like and dislike of the items to the user. The decay of the rating may be interpreted as a user might change his mind from like to dislike for a certain item due to time passing. This does not comport with the fact.

Therefore, we introduce a new concept of dynamic Collaborative Filtering in which the users' similarities decay with time but not the ratings. We divide data into several time-slices and calculate the users' similarities individually to generate the newest similarities by the weighted sum of the older similarities. The idea of our approach is that the people's previous preferences might not change along with time. People might still like the items they used to like, but there might have a lot of difference from their friends used to be similar to the user. The experimental results show that our approach has not only more accuracy but also less computation. Dividing the data by time does not need to calculate the whole data if new data comes. We could only calculate the influence of the incremental part of data.

The contributions of this thesis are as the following:

- We propose a more reasonable dynamic Collaborative Filtering, called Dynamic Similarity Collaborative Filtering (DSCF).

- Dynamic Similarity Collaborative Filtering has improved the accuracy over all existing dynamic Collaborative Filtering approaches. Also, less execution time makes DSCF more appropriate in practice.
- We propose an enhanced algorithm based on DSCF with feedback. This algorithm eliminates the parameter setting required in DSCF to relax the influence of initial parameter.

The rest of the thesis is organized as follows. Chapter 2 illustrates the existing approaches of Recommendation System especially in Dynamic Collaborative Filtering. Chapter 3 points out the drawbacks of the existing Dynamic Collaborative Filtering and introduces the motivation. Chapter 4 describes both DSCF and enhanced DSCF in detail and the experimental results are shown in Chapter 5. Chapter 6 expresses our conclusion and future works.

Chapter 2 Background and Related Work

2.1 Approaches of Recommendation System

Generally speaking, Recommendation System could be categorized to several approaches: Collaborative Filtering, Content-based, Model-based and hybrid.

2.1.1 Collaborative Filtering

The main idea of Collaborative Filtering [24,25] is that, the users, who are similar, would have similar preferences. In other words, if user A and user B are similar, an item which user B likes might be attractive to user A as well.

Based on the assumption, Collaborative Filtering is trying to compute the similarities between active user and other users. Active user represents the target user who would receive recommendations from the system. The Calculation of the similarities in Collaborative Filtering is the way to find out the existing common interests between the active user and other users. In the result of the calculation, people who have more common interests would have higher similarity. As the example in Fig.1, the left side of the figure is the user set and the right side is the item set. The lines between the two sets represent the interested items of the users. We can see the active user, A, is interested in items 1, 2 and 4. To calculate the similarity between user A and user B, we should first identify these two users have common interests on both item 2 and 4. After that, we could use the similarity function to calculate the similarity between these two users by the two common items.

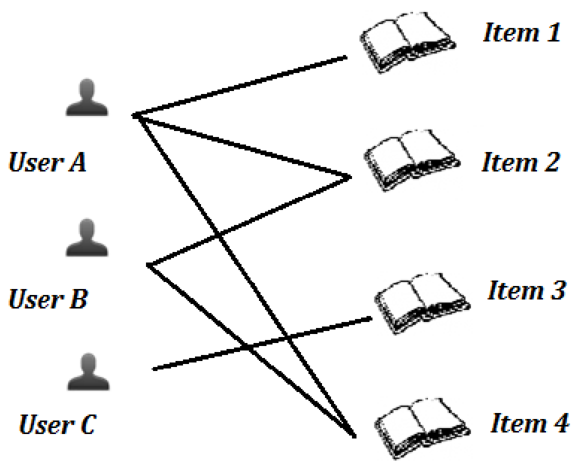

Figure 1: Collaborative Filtering uses the common items to calculate the similarity between users.

The steps of Collaborative Filtering can be summarized as the following:

1. Compute the similarities between the active user and other users.
2. Compute the predicted ratings of the items which have not been rated by the active user.
3. Sort the predicted ratings in decreasing order, and output the top-K items to the Recommendation System as the recommendations to the active user.

In the following, we are going to show some generally used functions of calculating the similarities and the predicted ratings.

The commonly used similarity functions [34] are Pearson Correlation coefficient, Cosine similarity and Conditional Probability. Pearson correlation coefficient is designed as in Eq. (1).

$$sim_{a,u} = \frac{\sum_{i=1}^{m}[(r_{a,i} - \overline{r_a})(r_{u,i} - \overline{r_u})]}{\sqrt{\sum_{i=1}^{m}(r_{a,i} - \overline{r_a})^2 \sum_{i=1}^{m}(r_{u,i} - \overline{r_u})^2}} \quad (1)$$

where $sim_{a,u}$ is the similarity between user *a* and user *u*, *i* is *m*-items which are rated by both user *a* and *u*, $r_{a,i}$ is the rating of user *a* to item *i*, $\overline{r_a}$ is the average-rating of user *a*.

Cosine similarity is designed as in Eq. (2).

$$sim_{a,u} = \cos(\vec{r}_a, \vec{r}_u) = \frac{\vec{r}_a \cdot \vec{r}_u}{\|\vec{r}_a\| \cdot \|\vec{r}_u\|} \tag{2}$$

where $sim_{a,u}$ is the similarity between user *a* and user *u*, $\vec{r}_a$ is a *nx1* rating matrix of user *a* where *n* represents the numbers of items.

Conditional probability is designed as in Eq. (3).

$$sim_{a,u} = p(I_a | I_u) = \frac{num(I_a \cap I_u)}{num(I_u)} \tag{3}$$

where I_a is the set of ratings of user *a*, *P(x)* is the probability function of *x*, *num(y)* is the number of items in set *y*.

The commonly used predict functions [4] are prediction with user average and prediction without user average. Prediction with user average function is designed as in Eq. (4).

$$p_{a,l} = \overline{r_a} + \frac{\sum_{j=1}^{k} (r_{j,l} - \overline{r_j}) \cdot sim_{a,j}}{\sum_{j=1}^{k} sim_{a,j}} \tag{4}$$

where $p_{a,l}$ is the predicted rating of user *a* on item *l*, $\overline{r_a}$ is average-rating of user *a*, *k* is the number of users who have also rated item *l*, $sim_{a,j}$ is similarity between user *a* and *j*.

Prediction without user average function is designed as in Eq. (5).

$$p_{a,l} = \frac{\sum_{j=1}^{k} r_{j,l} \cdot sim_{a,j}}{\sum_{j=1}^{k} sim_{a,j}} \tag{5}$$

where $p_{a,l}$ is the predicted rating of user *a* on item *l*, *k* is the number of users who have also rated item *l*, $sim_{a,j}$ is similarity between user *a* and *j*.

iExpand model [20] is a novel Collaborative Filtering based Recommender System by user interest expansion via personalized ranking. The goal is to build an item-oriented model-based Collaborative Filtering. This method introduces a three-layer scheme, user–interests–item, which leads to more accurate ranking recommendation results with less computation cost and helps the understanding of the interactions among users, items, and user interests.

In various applications, a record may be rated on several attributes [14.31]. Traditional Collaborative Filtering can only simply returning the recommended items with the highest overall scores but it fails to capture the individual attribute characteristics. In order to enhance the flexibility of CF, Collaborative Filtering with Skyline (CFS) [3], a general framework that combines the advantages of CF with those of the skyline operator is proposed. CFS generates a personalized skyline for each user based on scores of other users with similar behavior. The personalized skyline includes objects that are good on certain aspects, and eliminates those that are not interesting on any attribute combination.

H. Tan et al [28] proposes a Collaborative Filtering recommendation algorithm based on the item classification of the ratings. This approach classifies the items to predict the ratings which are low ratio of rated items to the total of available items. The Collaborative Filtering recommendation method based on item classification prediction can alleviate the sparsity problem of the user-item rating dataset, and can provide better recommendation than traditional Collaborative Filtering.

To solve the problems of Collaborative Filtering, data sparsity and scalability. Q. Li and M. Zhou [17] use a binary tree to store partitioned items. In the process of tree formation, a K-means clustering is used to partition data and generate the neighbor of similar items, and then predication based on a smaller item database is performed.

2.1.2 Content-based Filtering

Content-based filtering is based on the idea that people might have same favor to the similar items. Content-based filtering analyzes the item profiles to find out the similar items. Then it can use these item-relationships to recommend. We take Fig.2 as an example. In the left side of Fig.2 is active user A, the right side is items and the line means a user is interested in the movie. Based on the profile of these movies, we separate movies into different categories. As we show in the Fig. 2, “Harry Potter”, “The Lord of The Rings” and “The Avengers” are fantasy movies. If active user A is interested in “The Lord of The Rings”, we can provide “Harry Potter” and “The Avengers” as recommendations since these two movies are both in the “Fantasy” category.

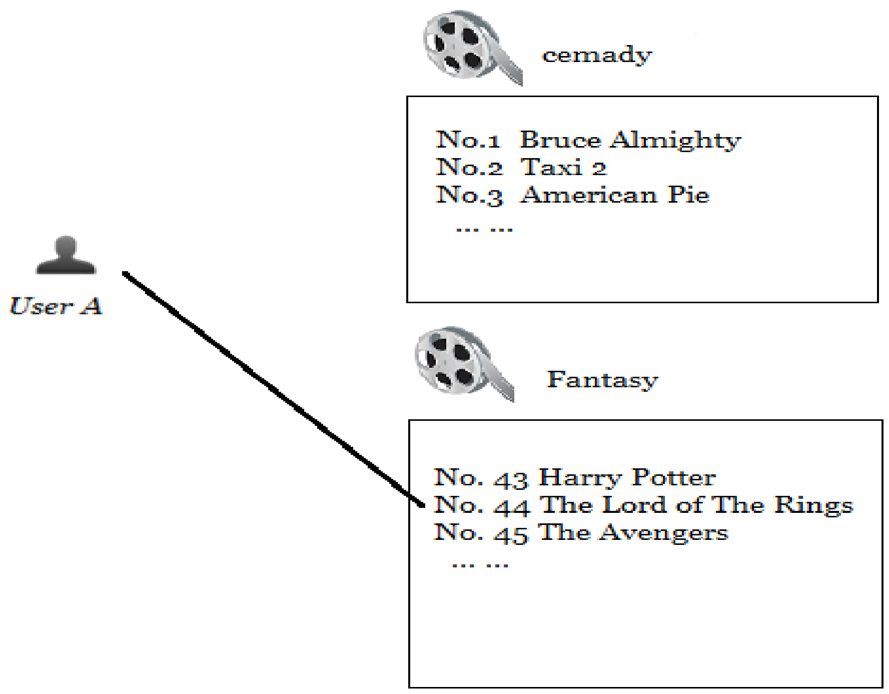

Figure 2: The idea of content-based filtering. People might like similar items.

Content-Based filtering is based on the item feature. It does not need to know about the domain knowledge of the data. However, the accuracy of Content-Based filtering is highly dependent on the profiles of the items. Therefore, when the profiles are not available, it might be a hard work for designing. On the other hand, the qualities of the profiles are also highly influential. Both explicit and implicit description might cause improper recommendations.

Chen et al [7] presents an agent-based personalized recommendation method called Content Recommendation System based on private Dynamic User Profile. The system collects and mines the private data of user at the client side, discovers, stores and updates private Dynamic User Profile at the client side. The system fetches preferred messages from the content server according to Dynamic User Profile. An important usage of this technology is a personalized advertising system in the RSS (Rich Site Summary) reader application.

Peng et al [23] proposes a content-based Recommendation System built on a weighted un-directional graph. The graph describes the content similarity between items bases on the semantic relations of their metadata. Neighbors of a node in the graph construct a ranking list of items to be recommended and there is a ranking list for each item. So it is able to emphasize differences among related items.

2.1.3 Model-based Filtering

Model-based Recommendation System extracts the information from the data to construct a model to represent the data. Therefore, it provides recommendations based on the constructed model rather than computes the whole data each time. It offers the benefits of speed and scalability. Model-based Recommendation System can be structured on many different models. Normally, the model can be probability-model, cluster-model or matrix factorization.

In probabilistic-based filtering, the approach computes the expected value of the items based on the given user profile or item-ratings. It can be represented as below:

$$p_{a,i} = E(r_{a,i}) = \sum_{v=0}^{m} pr(r_{a,i} = v \mid r_{a,k}, k \in I_a) \cdot v \tag{6}$$

where $p_{a,i}$ is the predicted rating of user *a* on item *i*, $r_{a,k}$ is the rating of user *a* on item *i*, *E(x)* is expected value of *x*, *pr(y)* is probability function of *y*, *v* is rating range from 0 to *m*, I_a is rated-item set of user *a*.

The idea of cluster-based filtering is to divide users or items into different clusters and provide recommendation by the clusters. It groups the similar users into same cluster and provides the common interests of the users which are in the same cluster to the active user. On the other hand, it can also group similar items into a cluster and provide the other items of the cluster which active user has interest items in it.

[13] proposes a model algorithm designs for learning predictive models of user preferences. It is based on a generalization of probabilistic latent semantic analysis to continuous-valued response variables. It assumes that the user ratings can be modeled as a mixture of user communities or interest groups. Each community is characterized by a Gaussian distribution on the normalized ratings for each item.

[30] is a graph-based Recommendation System. It considers the co-tagging behavior and adds the similarity relationships into the graph. The algorithm is based on modified random walk with restart. Considering the influence of tags, it has a denser transition matrix and higher accuracy.

Weighted minimum-message ratio (WMR) [21] is a friend Recommendation System which generates a personal friend list by the real message interactions among web members. Communication number is more representative than most of the population variables because they are lack of diversity. Therefore, the Content-based and Collaborative Filtering is not suitable.

Matrix factorization [15] is known as one of the best approaches of Recommendation System in recent years. Matrix factorization models map both users and items into a joint latent factor space R^f of dimensionality *f*, such that user-item interactions are modeled as inner products in that space. Because of that, each item *i* can be represented as a vector $q_i \in R^f$, each user *u* can be represented as a vector $p_u \in R^f$. For a given item *i*, the elements of q_i decides the distance between q_i and other vectors. As the same idea, for a given user *u*, the elements of p_u decides the distance between p_u and other vectors. Therefore, the dot product of a user vector and an item vector, $q_i^T \cdot p_u$, might be seem as the interaction which is represented as ratings in most cases of user *u* on item *i*. The formula is as below:

$$p_{ui} = q_i^T \cdot p_u \tag{7}$$

where p_{ui} is predicted rating of user *u* on item *i*, q_i is vector of item *i*, p_u is vector of user *u*.

Although model-based filtering methods are fast and scalable, however, their accuracies are highly dependent on the relationships between the model and the data. Actually, it is hard to find a nearly perfect model for a dataset. It is difficult to simulate people's behaviors as a single mathematical model. Moreover, people change their favors rapidly nowadays.

2.2 Dynamic Collaborative Filtering

The Collaborative Filtering is known as the best approach of Recommendation System. It only relies on the past user behavior, without being limited by the explicit knowledge such like user profiles or item profiles. Moreover, the simple designs does not cause low accuracy. Collaborative Filtering is one of approaches of Recommendation System with highest accuracy. This is the reason why so many researches find the improvements on the Collaborative Filtering. Most of these works try to merge couples of approaches together to suit a specific dataset, or combine with user analysis or item analysis to improve the qualities of the predictions. However, there are few researches focused on the time effect of Collaborative Filtering.

As we know, Collaborative Filtering is based on the human behaviors. And we also realize that people might change their favor or habits with time. People probably will not still be interested in the same items 10 years after. In the following, we are going to introduce some researches on dynamic Collaborative Filtering.

2.2.1 Time Weight Collaborative Filtering

Time Weight Collaborative Filtering [10] is known as the first work on dynamic Collaborative Filtering. It clearly indicates the serious problem of treating all the ratings equally even if they are in different time. Therefore, it announces the idea that the more recent the data is, the more it contributes to predicting the data. Hence, it enhances the prediction function of Collaborative Filtering as in Eq.(8):

$$o_{i,j} = \frac{\sum_{c=1}^{k} r_{ui} \cdot sim(I_j, I_c) \cdot f(t_{ic})}{\sum_{c=1}^{k} sim(I_j, I_c) \cdot f(t_{ic})} \tag{8}$$

where $o_{i,i}$ is the predicted rating of user *i* on item *j*, *k* is the number of items which are rated by user *i*, $sim(I_j, I_c)$ is similarity between item *j* and item *c*, $f(t_{ic})$ is time function with t_{ic}, t_{ic} is rating time of user *i* on item *c*.

The time function $f(t)$ is a monotonically decreasing function, which reduces uniformly with time *t* and the value of the time function lies in the range (0,1). The paper selects exponential function to ensure the time function decays with the time. However, other functions also meet the functionality, logistic function for instance.

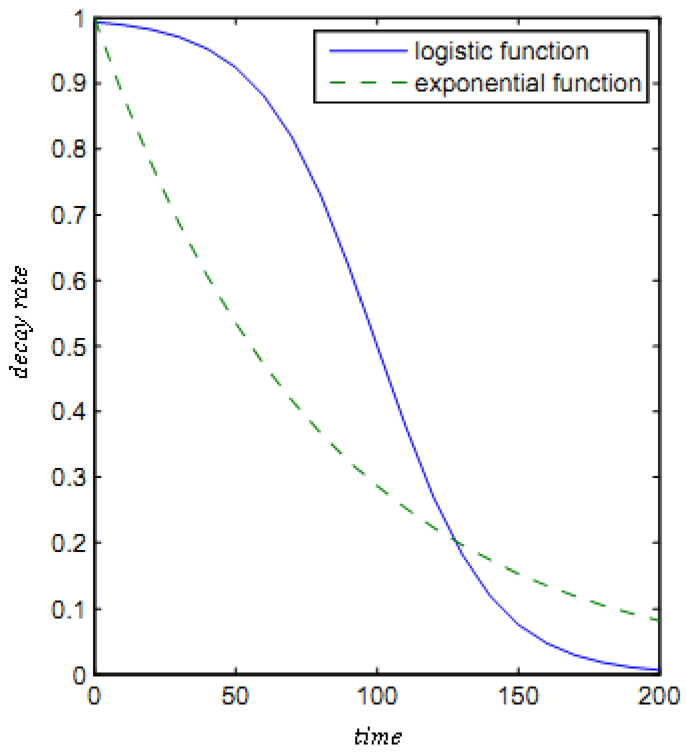

Figure 3: Distribution of two functions of time

Fig. 3 shows the corresponding value of logistic function and exponential function. For the exponential function, we can observe that the gradient of the curve at data point that is close to zero is steeper than the data point which is far away from zero. On the other hand, for the logistic function, the gradient of the curve at the middle data point is steepest. Here, x=0 represents current data. The higher the value of x is, the older in time the data is. In fact, we favor the users' latest purchase interest and focus on the most recent data. Therefore, the exponential function is more suitable for this case. To generate a proper exponential function for the time function, we define a half-life parameter T_0 as Eq.(9), which means, the weight reduces by 1/2 in T_0 days.

$$F(T_0) = \frac{1}{2} \cdot f(0) \tag{9}$$

Then the decay rate λ is defined as Eq.(10)

$$\lambda = \frac{1}{T_0} \tag{10}$$

Finally, the time weighting function is defined as Eq.(11)

$$f(t) = e^{-\lambda \cdot t} \tag{11}$$

From the Eq.(11), we can figure out that the value of the time function is in the range (0,1) and it decreases with the time *t*. The more recent the data, the higher the value of the time function. The exponential function satisfies the requirement well.

Further, we discuss about the half-time parameter T_0. Actually, to design a half-life parameter is to define the decay rate of the weight assigned to each data point. The decay rate of old data is decided by the value of parameter T_0. We know that T_0 is the inverse of decay rate λ. The higher the value T_0, the lower the value λ. The higher the value λ, it is faster for old data to decay and the lower the importance of the historical information compared to more recent data. Fig. 4 shows the different curves of time functions when the value of parameter T_0 is different. In fact, the values should be selected individually in different cases. It is highly relevant to the user's personality. If the user's preference for the type of items is consistent, old data can improve the accuracy of predicting future preference. In this case, the old data should be decayed slowly and a high value T_0 be assigned. Conversely, if the user's preference changes frequently, we should assign a low value of T_0 which means that old rating cannot help predicting the user's future preference. These old data should decay quickly. Therefore, we should select the appropriate parameter T_0 to precisely predict the user future preference according to the user's personalized purchase history.

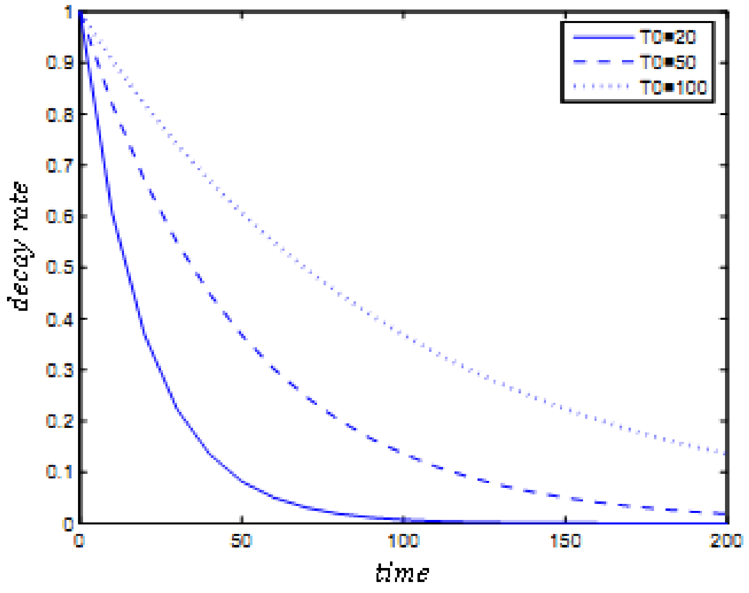

Figure 4: The comparison of different half-time parameter

2.2.2 Personalized Search Recommendation Based on Gradual Forgetting Collaborative Filtering Strategy

The main goal of researches on Recommendation System is to improve the accuracy of the recommendations results. Since the preference of the user may change, reduce the inference of the history information is one way to mimic real life.

In order to reach the idea, Gradual Forgetting strategy [19] is used to lower the weighted value gradually and nonlinearly. The algorithm first assigns every rating a weight which decreases with time by a forgetting function. Then, the system uses the weighted ratings to calculate the similarity between users. The method is called Gradual Forgetting Collaborative Filtering. The forgetting function $h(t)$ is shown as Eq. (12).

$$h(t) = m \times \left(\frac{t - t_{\min}}{t_{\max} - t_{\min}} \right)^2 + 1 - m \tag{12}$$

where $t_{\max} = \max(t_{rate} - t_{start})$, $t_{\min} = \min(t_{rate} - t_{start})$, t_{start} is the start time of the system,

t_{rate} is rating time of a certain item of the active user, *t* is the rating time of the item, *m* is an adjustable parameter which represents the change rate of users' preference and lies in the range (0,1).

With the forgetting function, the Pearson's similarity function is modified in Eq. (13):

$$sim(p_a, p_b) = \frac{\sum_{i=1}^{m}[(p_a(k_i)\times h(t) - \overline{p_a})(p_b(k_i)\times h(t) - \overline{p_b})]}{\sqrt{\sum_{i=1}^{m}(p_a(k_i)\times h(t) - \overline{p_a})^2 \sum_{i=1}^{m}(p_b(k_i)\times h(t) - \overline{p_b})^2}} \tag{13}$$

where $p_a(k_i)$ is the rating of user a on item *i*, $\overline{p_a}$ is average score of user a, h(t) is forgetting function.

Since the field of this paper is about keyword recommendation, the system actually does not have ratings. Therefore, the prior probability of the keyword is selected to replace rating. The formula of prior probability of keyword k_i is shown as Eq.(14)

$$p(k_i) = \alpha \times \frac{N_i}{N} + (1-\alpha)\times\frac{M_i}{M} \tag{14}$$

where $p(k_i)$ is the prior probability of keyword k_i, α is an adjustable parameter in range (0,1) and the definition of N_i and M_i is as follows:

1) In the user's N queries and clicks of keywords, there are N_i times to choose keywords k_i.

2) In the user's M reading web pages, there are M_i web pages which contain keywords k_i.

2.2.3 Dynamic Item-Based Recommendation Algorithm with Time Decay

Recommendation Systems aim to provide personalized advice through mining and discovering the interests and consuming patterns of customers. One of the main motivations of item-based Recommendation System lies on the fact that two items purchased by the same user are likely related to each other. However, this motivation does not mention the fact that the similarity between two purchased items varies according to the time interval between the two purchase actions. For example, it is reasonable to understand that the similarity between two items that are purchased by the same user on the same day may differ from the similarity between two items that are purchased by the same user in the same year.

Therefore, C. Xia et al [32] proposes the dynamic item-based top-N Recommendation System that utilizes time decay to build the models and provide recommendations. The main idea of time decay function is based on the time intervals of two items purchased. For two items, if most of the users who bought both of the items purchased during short time interval, the similarity of the two items might be high. If the purchasing time points of these two items are far from each other for most of users, the similarity between these two items might be low. With the idea mentioned, the new similarity function is generated as in Eq.(15).

$$sim(i,j) = \sum_{k=1}^{n} [R_{ki} \cdot R_{kj} \cdot \theta(|T_{ki} - T_{kj}|)] \tag{15}$$

where R_{ki} denotes a binary value of 1 if user *k* has purchased item *i*, otherwise 0. T_{ki} is the timestamp of user *k* buying item *i*. $|T_{ki} - T_{kj}|$ is the time interval between T_{ki} and T_{kj}. *n* is the number of users who buy both item *i* and *j*. $\theta(x)$ is time decay function and has two primary constraints as follows:

1) $\forall x, \theta(x) \in [0,1]$
2) $\forall x, x'$ if $x \geq x'$ then $\theta(x) \leq \theta(x')$

These two constraints ensure $\theta(x)$ is a decaying function and ensure that the larger time interval is, the stronger the effect of time decay is. Therefore, the time decay function could be presented in three ways.

1) Concave Time Decay Function

$$\theta(x) = \alpha^{x} \tag{16}$$

The parameter α is called concave Time Decay Coefficient and its range is between 0 and 1. Generally speaking, the smaller the α is, the stronger the time decay's effectiveness is. In particular, if α=1, $\theta(x)$ always equals to 1 which means no effectiveness of time decay.

2) Convex time decay function

$$\theta(x) = 1 - \beta^{t-x} \tag{17}$$

t represent the value of the largest time interval between two items in the dataset, x is time interval of two items. The parameter β is called convex time decay coefficient and its range is between 0 and 1. The larger β is, the stringer of the time decay's effectiveness is.

3) Linear time decay function

$$\theta(x) = 1 - \frac{x}{t} \cdot r \tag{18}$$

t represent the value of the largest time interval between two items in the dataset, x is the value of time interval between two items. The parameter r is called linear time decay coefficient and its range is between 0 and 1.The larger the r is, the stringer of the time decay's effectiveness is.

Chapter 3 Motivation

The ideas of Time Weight CF [10] and Gradual Forgetting CF [19] which decay the ratings associated with time called “dynamic weight”. In this method, the rating value would be multiplied by a weight generated from a gradual forgetting function or an exponential function. Since the ranges of the weights are between zero and one, the multiplied ratings would be lower than the primordial ones. The dynamic weight is using this concept to involve the time effect.

Although dynamic weight has considered the time reflections of Recommendation System, it still has a significant problem. In dynamic weight, rating scores would decrease upon time. The change of the ratings could also be interpreted as the change of people’s favors since the ratings represent the level of favor for the users. However, people don’t actually change their favor of the item from “like” to “dislike” even if it was purchased long time ago. For example, many people like the legendary superstar “Michael Jackson” and bought his albums. Since these albums have all released long time ago, most of the rating data are old. In the methods of dynamic weight, these ratings decrease to much lower values which might mislead people that the buyers no longer like those albums, and it is not the fact.

In order to solve this unreasonable situation, we provide a new framework of Collaborative Filtering - Dynamic Similarity Collaborative Filtering (DSCF). The main idea of Collaborative Filtering is to decay the similarities between the active users and their neighbors. Actually, people do not change their favors because of the time, but the relations or similarities among people might probably change because of time. For example, a college student might be more similar to his current classmates than high school classmates in preferences due to the different study environments. Therefore, we adjust the similarities dynamically among users instead of decreasing the ratings for being reasonable.

Chapter 4 Dynamic Similarity Collaborative Filtering

4.1 Dynamic Similarity Collaborative Filtering (DSCF)

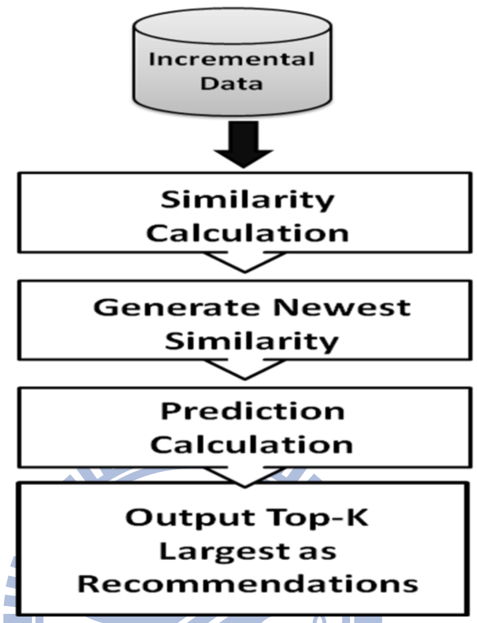
Incremental Data
Similarity Calculation
Generate Newest Similarity
Prediction Calculation
Output Top-K Largest as Recommendations

Figure 5 : The Framework of Dynamic Similarity Collaborative Filtering

In this chapter, we will introduce the similarity calculation and the way to generate the newest similarity in Section 4.2. We will introduce the prediction calculation in Section 4.3. In Section 4.4, we propose an enhance DSCF based on DSCF and we will have some discussion in Section 4.5.

4.2 Similarity calculation of DSCF

In Dynamic Similarity Collaborative Filtering, we initially define a time period *pd*. The time axis could be divided into ordered time slices by the time period *pd* which means the lengths of each time slice are *pd*. The dynamic similarity is defined in Eq.(19):

$$Msim_0^t = \alpha \cdot Msim_0^{t-1} + (1-\alpha) \cdot Msim_{t-1}^t \qquad (19)$$

Here, we use similarity matrix to define Eq. (19). Similarity matrix is a 1*xm* matrix which represents the similarities between the active user and other *m* users and the active user is the user who received recommends by the system. $Msim_0^t$ is the similarity matrix from time 0 to time *t*, α is a system adjustment parameter, called Similarity Decay Value (SDV), ranged between 0 and 1. It represents the decay rate of similarities. The lower α is, the larger decay effect is. The range between *t*-1 and *t* is *pd*. And we choose Pearson's Correlation as our method to calculate the similarity as shown in Eq. (1).

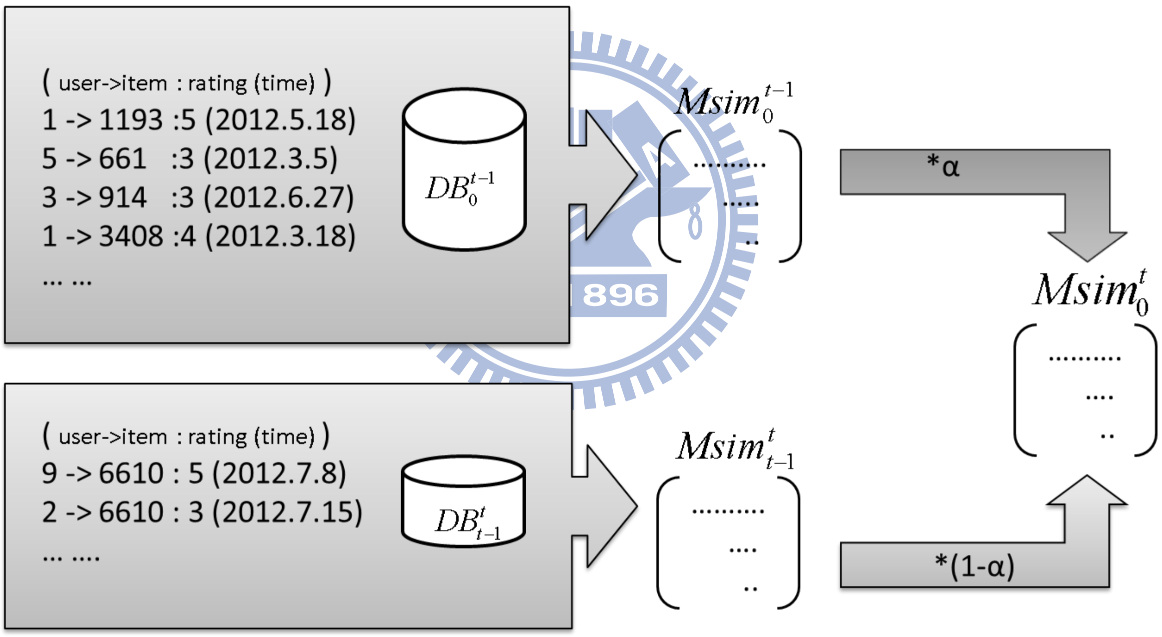

Figure 6: The framework of Dynamic Similarity Collaborative Filtering

The concept of Dynamic Similarity Collaborative Filtering is shown in Fig. 6. When the current time is *t*, the time range is from time 0 (the beginning of the system) to time *t*. The data is divided into previous and current data sections, respectively named as DB_0^{t-1} and DB_{t-1}^t in Fig. 6. The data reflects the relationships among users in the time segments in the two data sections. We individually generate similarities matrix for the two data sections and respectively named as $Msim_0^{t-1}$ and $Msim_{t-1}^{t}$. Afterwards, we combine the two similarity matrixes by weighted sum to generate $Msim_0^{t}$ which represented the similarity matrix from time 0 to time t. In addition, we use the same way to calculate similarity matrix $Msim_{t-1}^{t}$. When the time is *t*-1, the whole data could be divided into the previous one(0 to *t*-2) and the current one(*t*-2 to *t*-1). The two similarity matrixes could be generated as $Msim_0^{t-2}$ and $Msim_{t-2}^{t-1}$. Therefore, $Msim_0^{t-1}$ is the weighted sum of $Msim_0^{t-2}$ and $Msim_{t-2}^{t-1}$. To avoid the confusion, we call the similarity matrix of the current data as "semi-similarity matrix" (the data from last time period to the present time period) and call the similarity matrix of the previous data as "period-similarity matrix" (the data from 0 to the last time period).

4.3 Prediction calculation of DSCF

The second step of Collaborative Filtering uses the similarity values to predict the unknown ratings. In DSCF, we use the latest period-similarity matrix to predict the future. For example, the prediction at time *t* is generated by period-similarity matrix $Msim_0^{t-1}$ in the time *t*-1. As the same way, we use $Msim_0^{t}$ to predict data of next timestamp *t*+1, when the time is *t*.

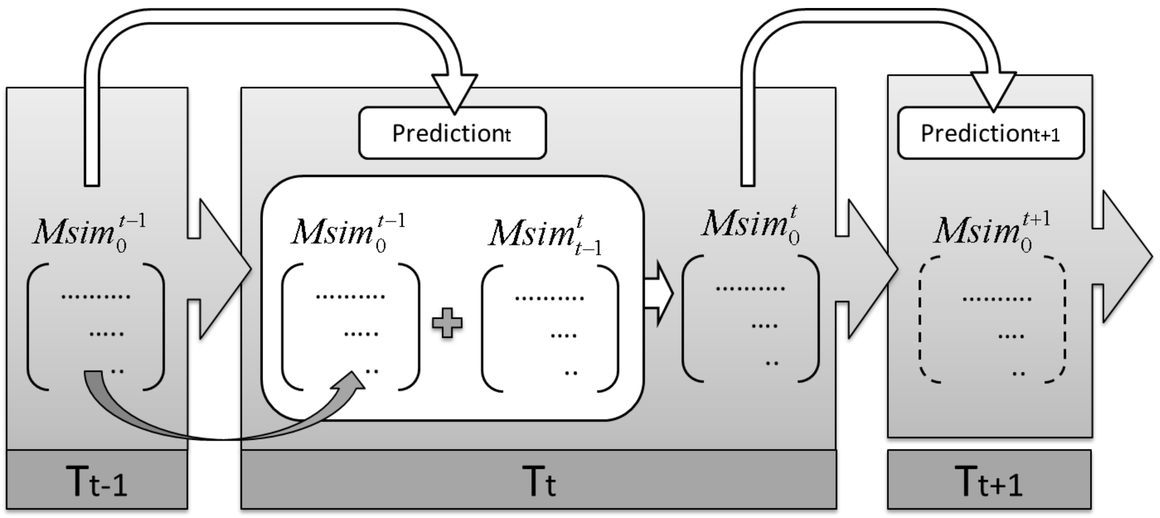

Figure 7: The framework of prediction in DSCF

4.4 Enhanced DSCF

The accuracy of DSCF is much better than the traditional Collaborative Filtering and most of the existing dynamic CFs. However, according to the benefit of time slice division, we have enhanced DSCF by the feedback of Similarity Decay Value (SDV) which is the α in Eq. (19)·

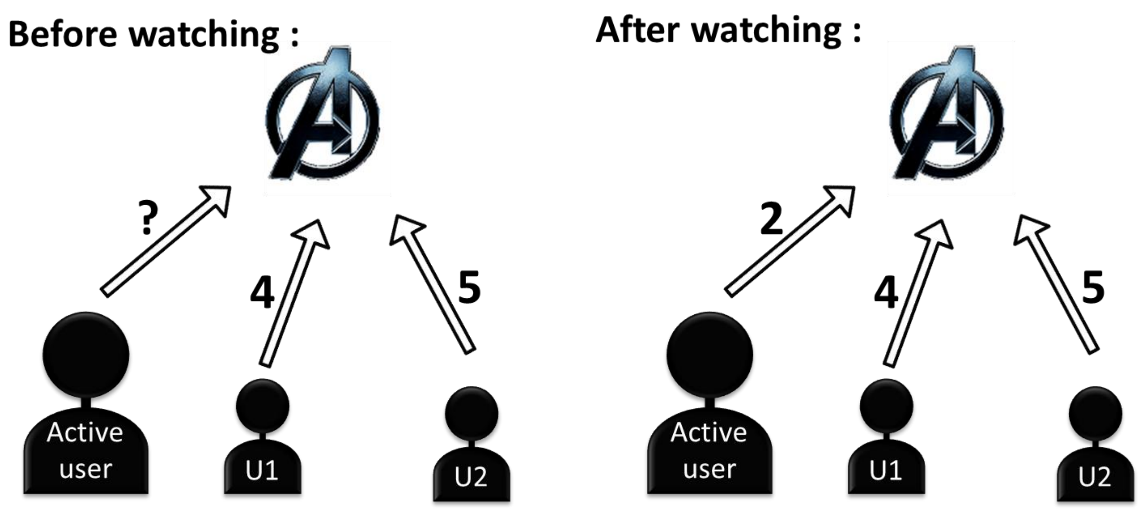

Figure 8: Example of the enhanced DSCF

In DSCF, we fixed SDV to calculate the predicted rating of unknown items. As time goes by, these unknown ratings could be known. As the example in Fig. 8, an active user has not watched "Avengers" yet. With his historical records of other movies' ratings and a fixed SDV 0.2 (set as a system parameter), we might compute the predicted rating of "Avengers" as 3.86 before watching it. Some day in the future, the active user watch "Avengers" and rate it 2. With the actual rating 2, we could then derive a better SDV to be closed to the actual value. Like we show in Eq. (20), we could derive a better SDV 0.68. When SVD is 0.68, the prediction of active user on "Avengers" becomes 1.7 which is closer to the actual value. Thus, we find a more appropriate SVD to the active user than using a fixed system parameter. We call this SDV as ASDV (Actual similarity decay value).

<u>*Prediction phase:*</u>

$$sim_{a,1} = \alpha \cdot sim'_{a,1} + (1-\alpha)\cdot sim''_{a,1} = 0.2*(-0.2) + (1-0.2)*0.8 = 0.6$$

$$sim_{b,1} = \alpha \cdot sim'_{b,1} + (1-\alpha)\cdot sim''_{b,1} = 0.2*(-0.45) + (1-0.2)*0.5 = 0.31$$

$$p = 1.3 + \frac{(5-2.3)*0.6 + (4-1.7)*0.31}{0.6+0.31} = 3.86$$

<u>*Feedback phase:*</u>

$$2 = 1.3 + \frac{(5-2.3)*sim_{a,1} + (4-1.7)*sim_{b,1}}{sim_{a,1} + sim_{b,1}}$$

$$= 1.3 + \frac{(5-2.3)*[\alpha \cdot sim'_{a,1} + (1-\alpha)\cdot sim''_{a,1}] + (4-1.7)*[\alpha \cdot sim'_{b,1} + (1-\alpha)\cdot sim''_{b,1}]}{[\alpha \cdot sim'_{a,1} + (1-\alpha)\cdot sim''_{a,1}] + [\alpha \cdot sim'_{b,1} + (1-\alpha)\cdot sim''_{b,1}]}$$

$$\Rightarrow \alpha = 0.68 \qquad (20)$$

In other words, we first assume the predicted rating as an unknown value, other values (i.e. similarities, SDV, averages, ratings) as known values. Second, we use these known values to solve unknown value. Third, when we know the actual value relative to a predicted rating, we assume SDV as unknown value and other values(i.e. similarities, actual value, averages, ratings) as known values, then solve the unknown values(ASDV). Each actual value would compute an ASDV and an active user might not merely have one ASDV because he has rated more than one item in a single time slice. For an active user with single SDV at a time slice, we take the mean of these ASDVs to replace SDV of the active user at the time slice.

In the following, we are trying to calculate the close form of ASDV to prove that ASDV could be derived when the rest of the values (ratings, averages, similarities) are known.

Assume $A_{a,l} = \overline{r_a} + \frac{\sum_{j=1}^{k}(r_{j,l}-\overline{r_j})sim_{a,j}}{\sum_{j=1}^{k} sim_{a,j}}$, $(r_{j,l}-\overline{r_j}) = D_j$

$$\Rightarrow A_{a,l}-\overline{r_a} = \frac{\sum_{j=1}^{k}(r_{j,l}-\overline{r_j})sim_{a,j}}{\sum_{j=1}^{k} sim_{a,j}} = \frac{\sum_{j=1}^{k} D_j \cdot sim_{a,j}}{\sum_{j=1}^{k} sim_{a,j}} = \frac{\sum_{j=1}^{k} D_j \cdot [\alpha \cdot sim'_{a,j} + (1-\alpha)sim''_{a,j}]}{\sum_{j=1}^{k}[\alpha \cdot sim'_{a,j} + (1-\alpha)sim''_{a,j}]}$$

$$= \frac{\sum_{j=1}^{k} D_j[sim''_{a,j} + \alpha \cdot (sim'_{a,j} - sim''_{a,j})]}{\sum_{j=1}^{k}[sim''_{a,j} + \alpha \cdot (sim'_{a,j} - sim''_{a,j})]}$$

Assume $A_{a,l} - \overline{r_a}$ = X

$$\Rightarrow \mathrm{X} \cdot \sum_{j=1}^{k}[sim''_{a,j} + \alpha \cdot (sim'_{a,j} - sim''_{a,j})] = \sum_{j=1}^{k} D_j[sim''_{a,j} + \alpha \cdot (sim'_{a,j} - sim''_{a,j})]$$

$$\Rightarrow \mathrm{X} \cdot \alpha \cdot \sum_{j=1}^{k}(sim'_{a,j} - sim''_{a,j}) - \alpha \cdot \sum_{j=1}^{k} D_j(sim'_{a,j} - sim''_{a,j}) = \sum_{j=1}^{k} D_j \cdot sim''_{a,j} - \mathrm{X}\sum_{j=1}^{k} sim''_{a,j}$$

$$\Rightarrow \alpha \cdot [\mathrm{X} \cdot \sum_{j=1}^{k}(sim'_{a,j} - sim''_{a,j}) - \sum_{j=1}^{k} D_j(sim'_{a,j} - sim''_{a,j})] = \sum_{j=1}^{k} D_j \cdot sim''_{a,j} - \mathrm{X}\sum_{j=1}^{k} sim''_{a,j}$$

$$\Rightarrow \alpha = \frac{\sum_{j=1}^{k}(D_j \cdot sim''_{a,j} - \mathrm{X} \cdot sim''_{a,j})}{\sum_{j=1}^{k}[\mathrm{X}(sim'_{a,j} - sim''_{a,j}) - D_j(sim'_{a,j} - sim''_{a,j})]} = \frac{\sum_{j=1}^{k}[(D_j - \mathrm{X}) \cdot sim''_{a,j}]}{\sum_{j=1}^{k}[(D_j - \mathrm{X})(sim''_{a,j} - sim'_{a,j})]}$$

(21)

Where $A_{a,i}$ is the actual value of user a on item *i*, $r_{j,l}$ is the rating of user *j* on item *i*, $\overline{r_a}$ is the average rating of user *a*, $sim'_{a,j}$ is period-similarity between user *a* and *j*, and $sim''_{a,j}$ is semi-similarity between user *a* and *j*.

4.5 Discussion of DSCF

In this section, we discuss two issues of DSCF: (1) why is DSCF dynamic. (2) the real-time effect of DSCF

4.5.1 Why is DSCF dynamic

The concept of dynamic in Collaborative Filtering is that newer data should contribute more than older one in similarity computation. To prove DSCF is dynamic, we extend Eq.(19) as follows:

$$
\begin{aligned}
Msim_0^t &= \alpha \cdot Msim_0^{t-1} + (1-\alpha) \cdot Msim_{t-1}^t \\
&= \alpha \cdot [\alpha \cdot Msim_0^{t-2} + (1-\alpha) \cdot Msim_{t-2}^{t-1}] + (1-\alpha) \cdot Msim_{t-1}^t \\
&= \alpha^2 \cdot Msim_0^{t-2} + \alpha \cdot (1-\alpha) \cdot Msim_{t-2}^{t-1} + (1-\alpha) \cdot Msim_{t-1}^t \\
&= \alpha^2 \cdot [\alpha \cdot Msim_0^{t-3} + (1-\alpha) \cdot Msim2_{t-3}^{t-2}] + \alpha \cdot (1-\alpha) \cdot Msim_{t-2}^{t-1} + (1-\alpha) \cdot Msim_{t-1}^t \\
&= \alpha^3 \cdot (1-\alpha) \cdot Msim_0^{t-3} + \alpha^2 \cdot (1-\alpha) \cdot Msim_{t-3}^{t-2} + \alpha \cdot (1-\alpha) \cdot Msim_{t-2}^{t-1} + (1-\alpha) \cdot Msim_{t-1}^t \\
&= \ldots\ldots = \sum_{i=0}^{t-1} \alpha^i \cdot (1-\alpha) \cdot Msim_{t-i-1}^{t-i}
\end{aligned}
$$

(22)

We find that the oldest period-similarity matrix ($Msim_0^1$, similarity between the beginning of the system and first time period) has the highest power of α. Since the adjustment parameter α lies in the range (0,1), higher power would have lower value. In conclusion, the newer similarity matric would have lower decay. Therefore, we proved that our method matches dynamic property.

4.5.2 Real-time effect of DSCF

Real-time property is important for online services. Most internet users do not have patience waiting for web executing for long time. Thus, the algorithm should be as fast as possible and the repeated computations of old data should be as low as possible. In fact, Collaborative Filtering is an efficient algorithm comparing with other works such as some of clustering-based filtering or hybrid-based Recommendation System.

Nevertheless, the repeated computation of old data for Collaborative Filtering might cause significant problems in practice. In all of existing Collaborative Filtering based algorithms, they have only one similarity matrix which represents the relationships among users for whole time. So when the system receives new data, these existing algorithms have to calculate both new and old data. This causes redundant computation. Since the old data would not change, we do not need to calculate the same data repeatedly.

In DSCF, we only have to compute the incremental data to generate the semi-similarity matrix. The old data is reflected in period-similarity matrix and it is not necessary to execute again. Therefore, we could reduce the unnecessary calculation extremely.

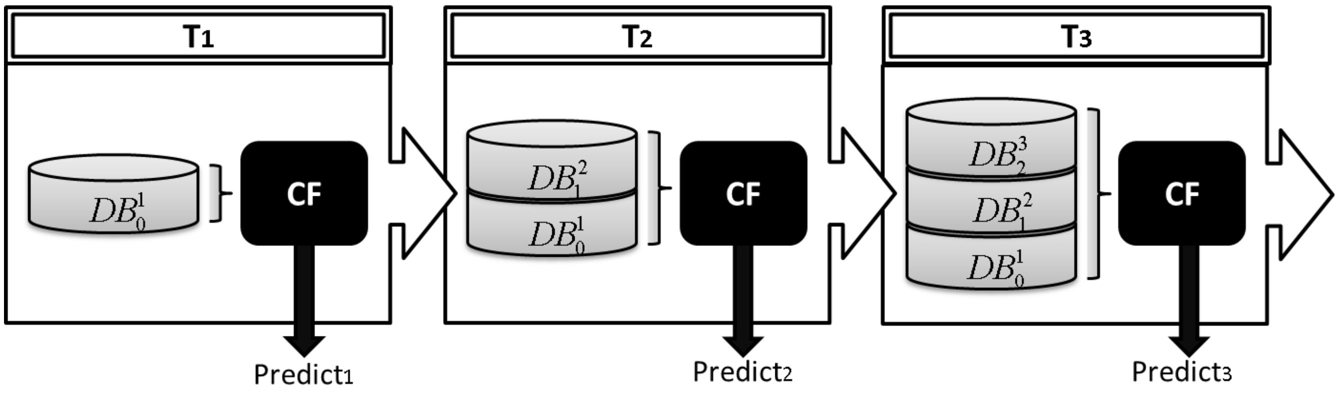

Figure 9: Data execution of other Collaborative Filtering based algorithm

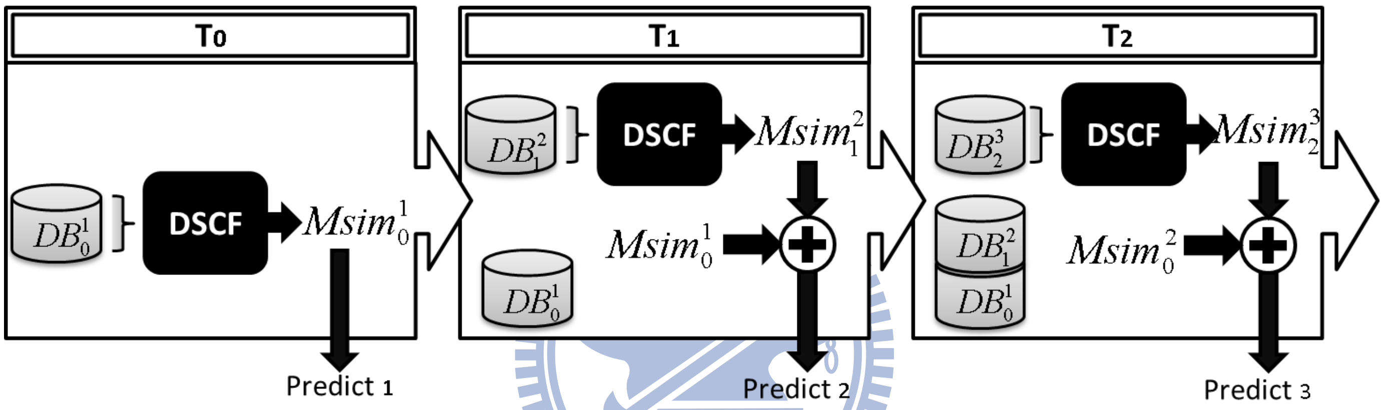

Figure 10: Data computation of DSCF in each time period

Chapter 5 Experimental Result

In this chapter, we present a comparison of accuracy and execution time between DSCF and traditional Collaborative Filtering which do not consider about time and three dynamic Collaborative Filtering we mentioned in Section 2.2: Time Weight CF [10], Gradual Forgetting CF[19] and Decay Function [32] All the programs are compiled in Microsoft Visual Studio 2010 in Windows 7, with Intel core i7-2600k CPU and 16GB main memory. We use two famous datasets in Recommendation System, MovieLens [37] and Netflix [38]. All algorithms are compared in the Mean Absolute Error (MAE), Precision/ Recall and execution time for both datasets. We also show the MAE comparisons in different SDV in DSCF and enhanced DSCF.

5.1 Evaluation methods

To evaluate the accuracy of each methods, we propose three measurements: Mean Absolute Error(MAE), Precision and Recall which are defined as follows.

$$MAE = \frac{\sum_{i=1}^{N} |x_i - y_i|}{N} \tag{23}$$

where N is the number of active user's ratings. x_i identifies the predicted rating for the i-th item of active user. y_i is the true rating for the i-th item of active user.

$$precision = \frac{\#hits}{\#recommendations} \tag{24}$$

where $\#recommendations$ is the number of recommendations. $\#hits$ is the number of items which are both relevant and recommended.

$$recall = \frac{\#hits}{\#relevent} \tag{25}$$

where $\#relevent$ is the number of relevant.

Besides, we calculate execution time to show the real-time property of our proposed method.

In order to evaluate the prediction qualities, we have to assume some ratings in the dataset as unknown ratings, predict the unknown ratings and then compute the errors between predicted ratings and actual ratings. Therefore, we divided the whole dataset as 80% training set and 20% testing set which means 20% of the ratings is treated as unknown rating. We use the training set to calculate the similarity matrix of users and predict the rating of items in the testing set.

5.2 Dataset MovieLens

MovieLens is provided and maintained by University of Minnesota. It collected over 100M ratings of 3,900 movies made by 6,040 users. We use this dataset to evaluate in terms of MAE, precision/recall and execution time in different algorithms.

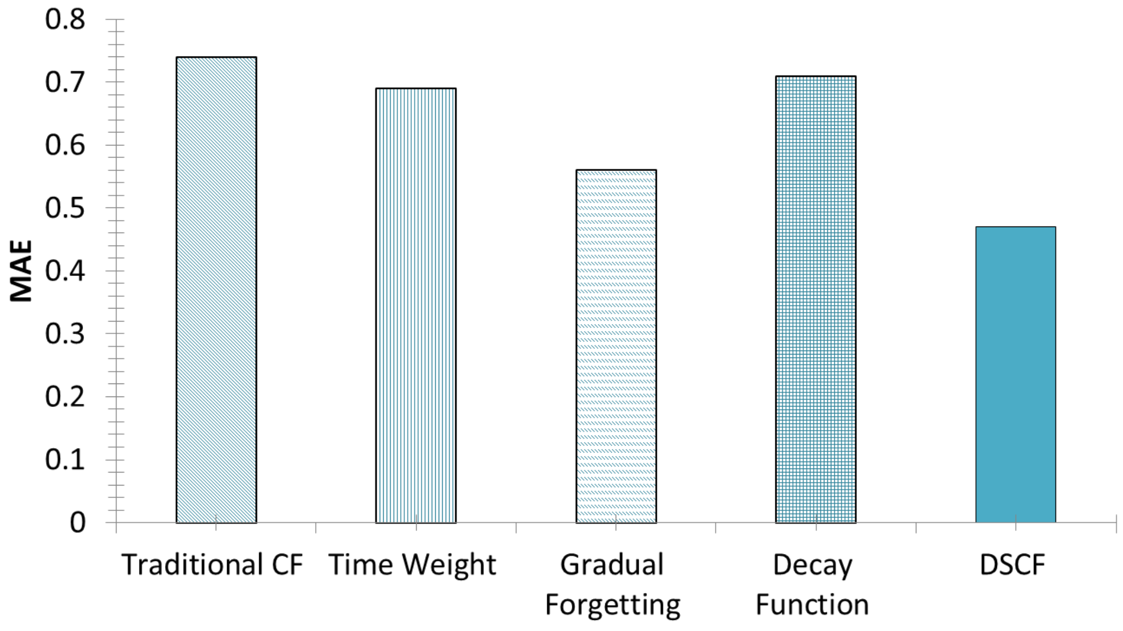

Figure 11: MAE comparison for dataset "MovieLens"

Fig.11 shows the MAEs of traditional CF, Time Weight CF, Gradual Forgetting, Decay Function and DSCF for dataset MovieLens. The result shows our method, DSCF, has the best quality than other works. The other dynamic CFs also have good results in MAE. Traditional CF which does not consider about time has the worst result. This means it is significant to consider the time influence in Collaborative Filtering.

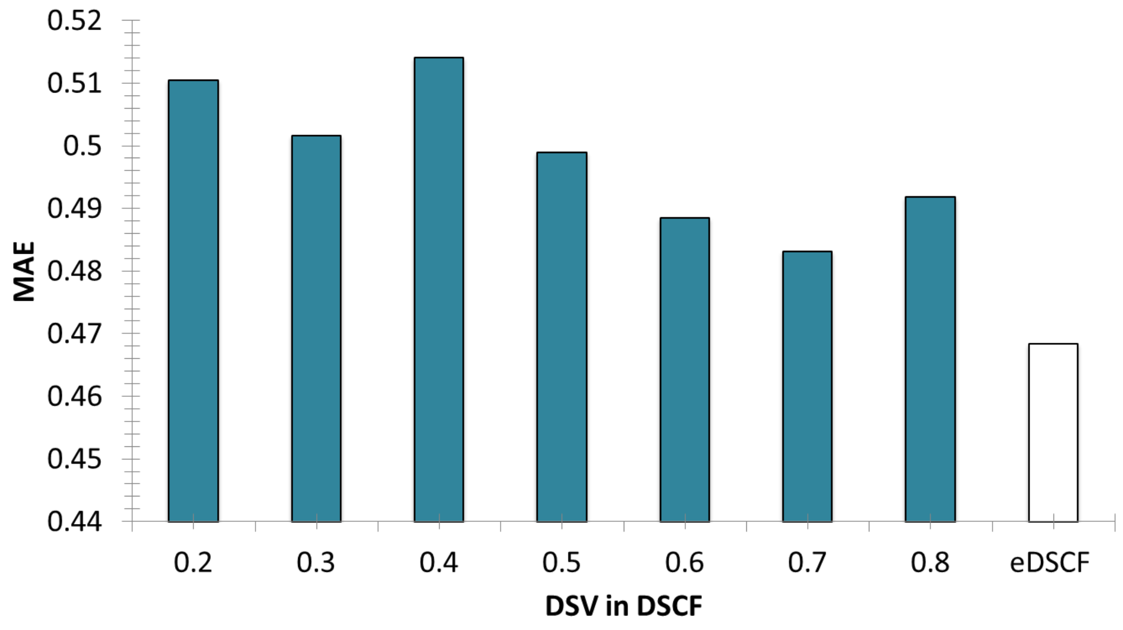

Figure 12: MAE of different Decay Similarity Value and enhanced DSCF for "MovieLens"

Fig. 12 shows the MAEs in different Similarity Decay Value (SDV) of DSCF and the MAE of enhanced DSCF for Movielens. We find that in DSCF, the decision of SDV has no regular rule. This means SDV is highly dependent on different data. Heuristic method might be a way to find SDV. However, enhanced DSCV can automatically generate a proper SDV. Also, with different SDV for different users, the accuracy also have improved.

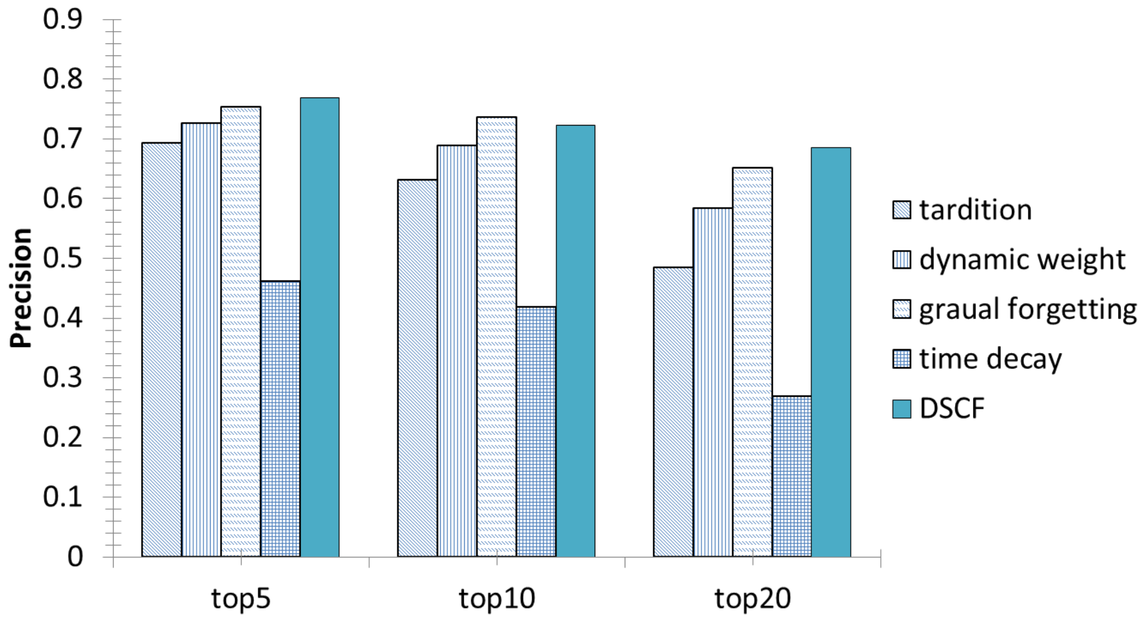

Figure 13 : Comparison of precision for dataset “MovieLens”

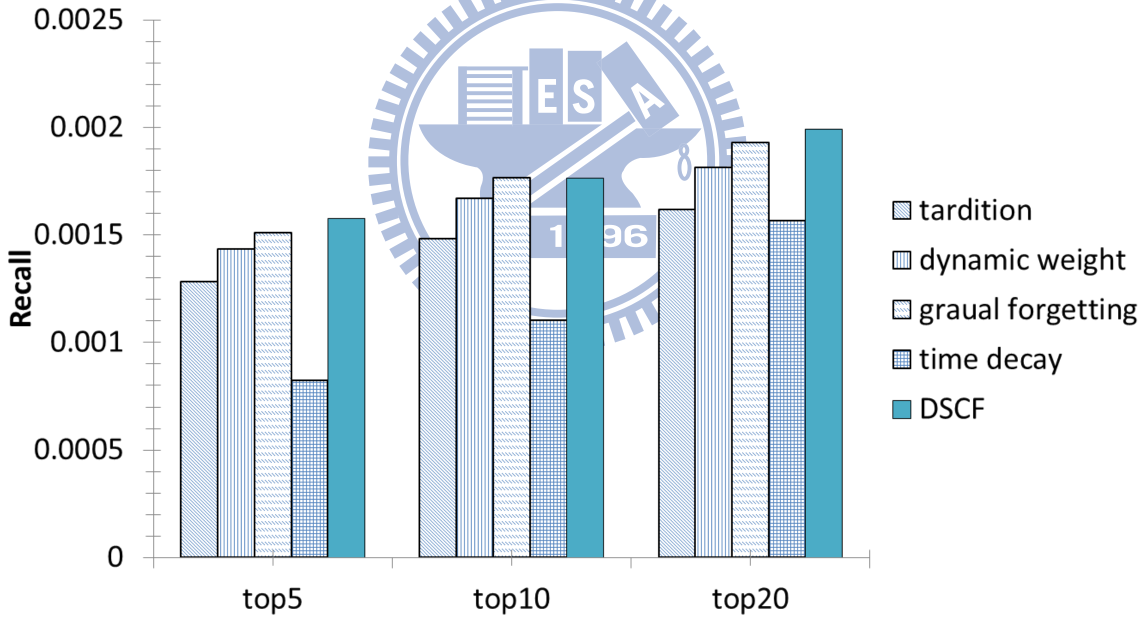

Figure 14: Comparison of recall for dataset “Movielens”

In order to guarantee that the accuracies of our proposed method are confident, we use precision and recall evaluation measurements to further prove the accuracies. Fig. 13 shows the precision of traditional CF, Time Weight CF, Gradual Forgetting, Decay Function and DSCF for dataset MovieLens, It has similar result as Fig.11, DSCF has best accuracy than other works. Collaborative Filtering without considering time has the worst accuracy.

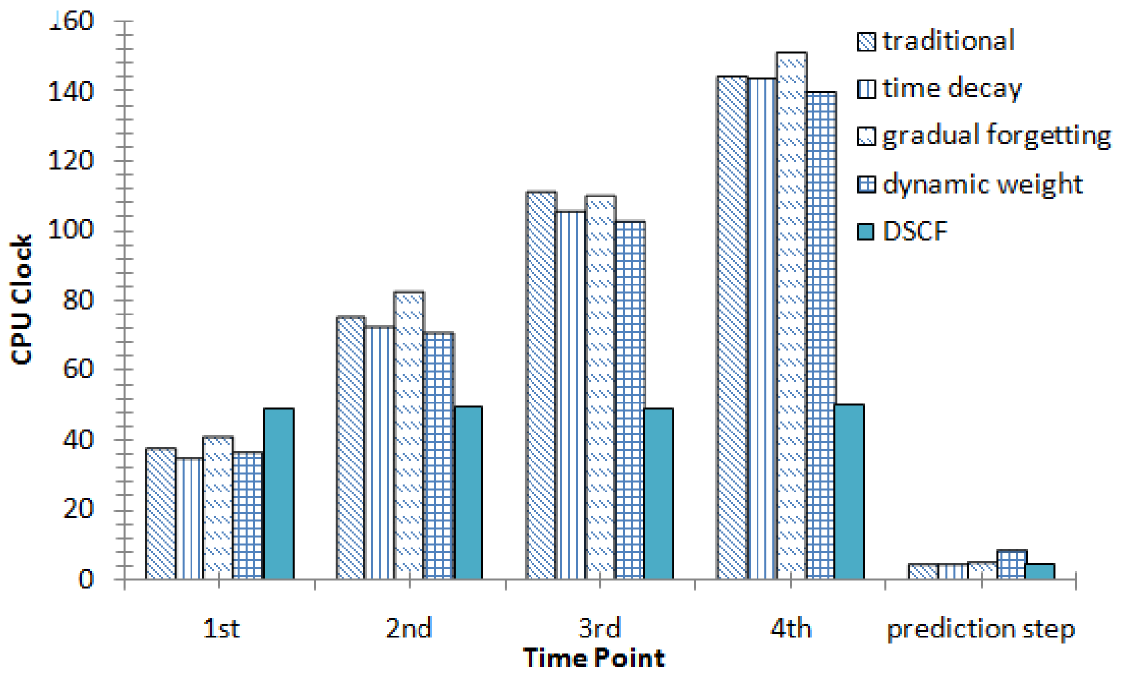

Figure 15: Comparison of execution time for dataset “MovieLens”

To evaluate the execution time, we simulate a system using whole year data and output recommendations at the end of a season. So we divide the whole data into four parts by the time which every part have the same time slice length. We calculate similarities once of each part and record the execution time. And compute the prediction in the end of the whole data. Fig. 15 shows the execution time of each method. Our method has better performance in execution time since we have only execute the incremental data at each timepoint. Other works have to execute all of data from the beginning to the end. Therefore, our method has better design in practice.

5.3 Dataset Netflix

The second dataset we used is Netflix. It is constructed to support the Netflix Prize. The data has over 100M ratings of 17,770 movies made by 480,189 users. We also compare the algorithms in terms of MAE, precision/recall and execution time for this dataset.

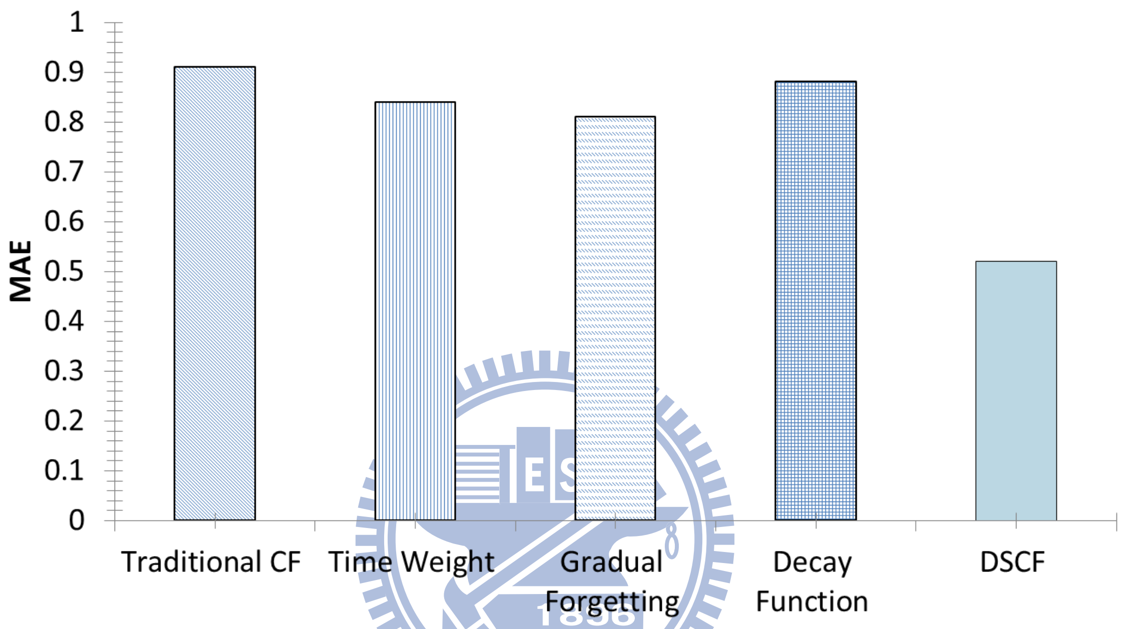

Figure 16 : Comparison of MAE for dataset "NetFlix"

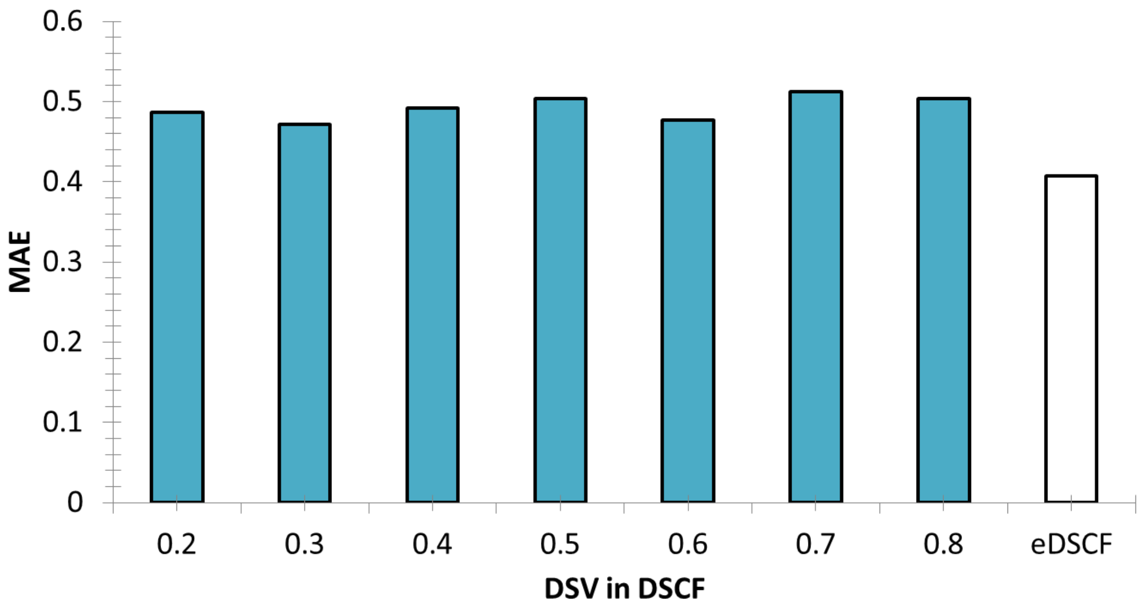

Figure 17 : Comparison of MAE in different DSV and enhanced DSCF for dataset

“Netflix”

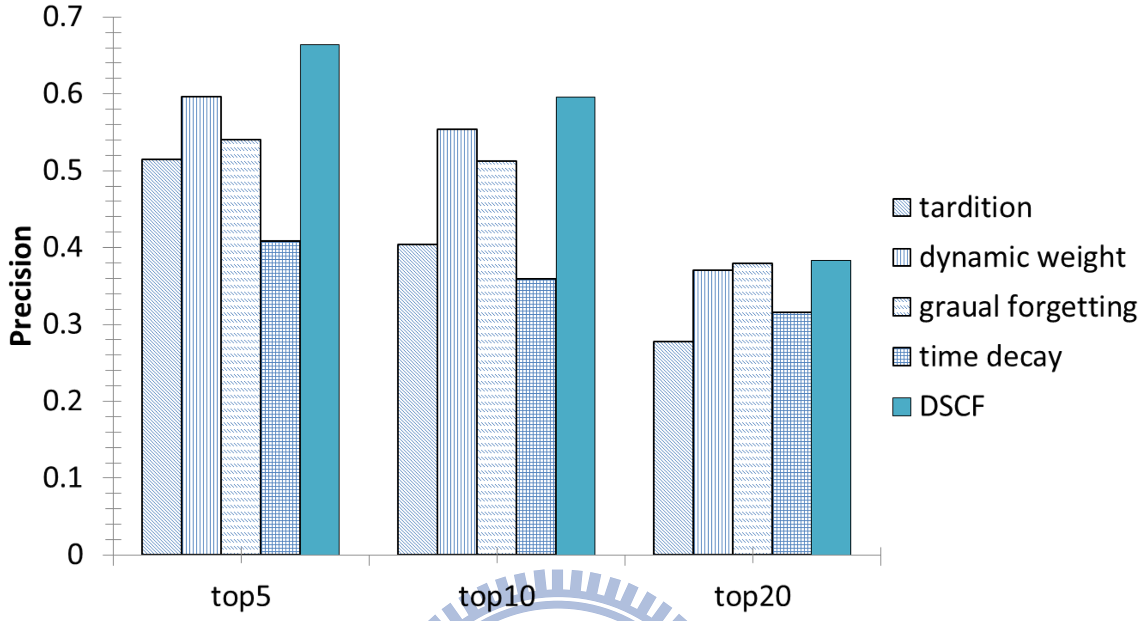

Figure 18 : Comparison of Precision for dataset “Netflix”

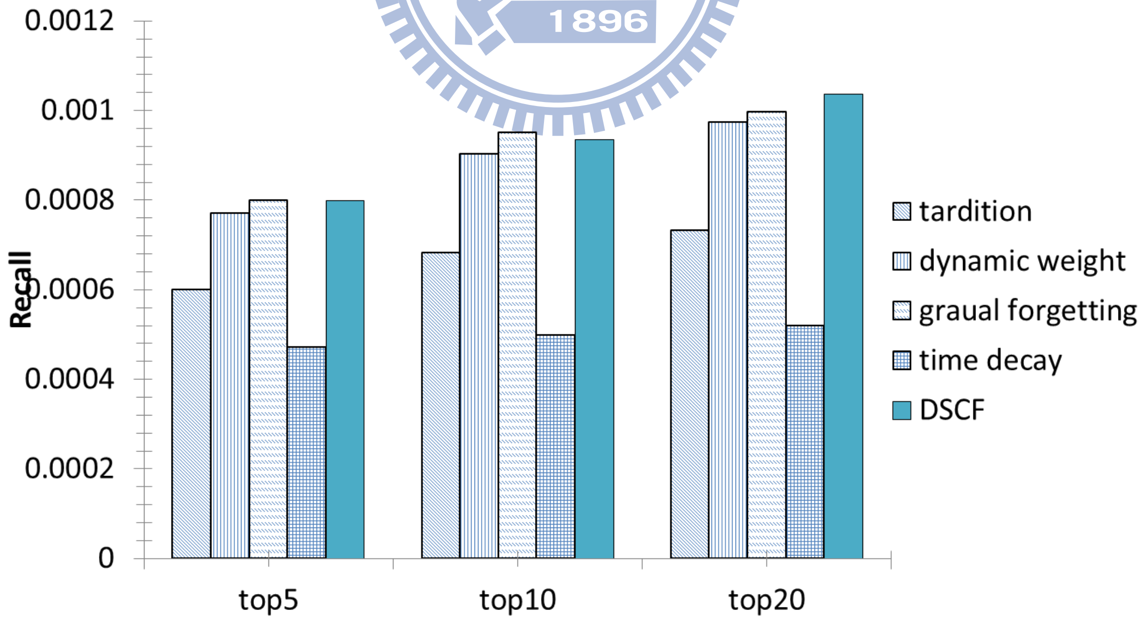

Figure 19 : Comparison of recall for dataset “Netflix”

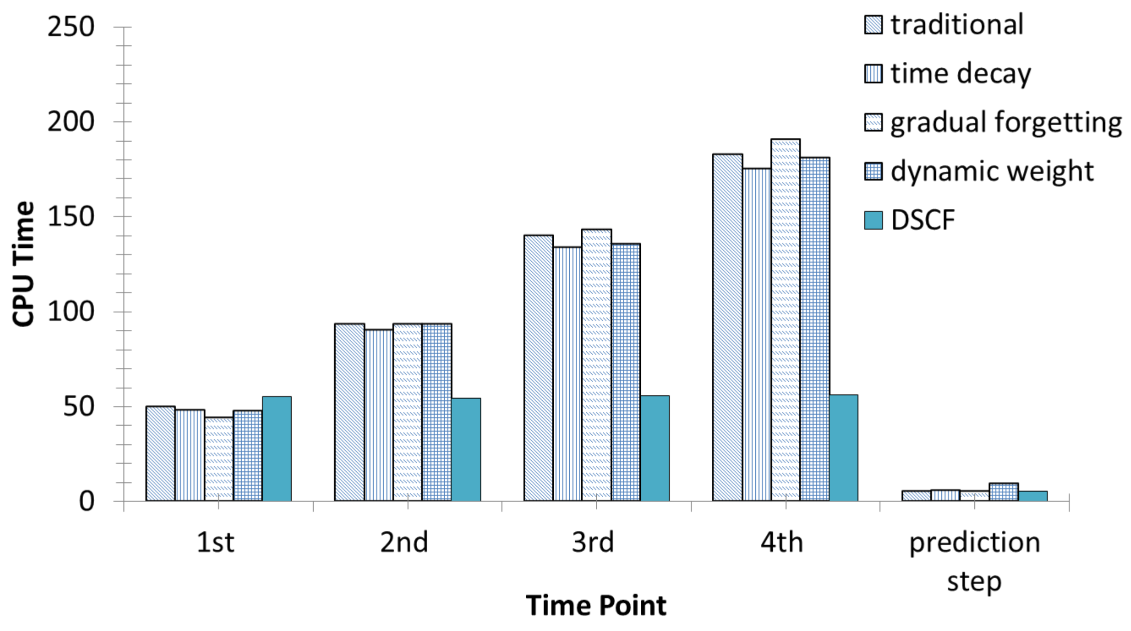

Figure 20: Comparison of Execution time for dataset “Netflix”

According to Fig. 16 to Fig. 20, we almost has the same conclusion as in dataset MovieLens. In accuracy evaluations, DSCF has the highest accuracy than the existing works .DSCF with verification is higher than every value of SDV. And we always has the best performance in execution time.

Chapter 6 Conclusions and Future Work

In this thesis, we propose a new framework on Dynamic Collaborative Filtering based on the decay of similarities among users in order to solve the unreasonable pheromone of other Dynamic Collaborative Filtering works. The experimental results show that our proposed method has higher accuracies on both MAE and precision/recall measurements. In the other words, our predictions present well on both of the items which users potentially have interests or not. This ensures our predictions satisfy some users who have special interests. Furthermore, the results show that our assumption, people do not actually change their favors on an item but the similarities among people change because of time, is correct. In our architecture we divide the data into several segments. This manner not only supports incremental similarity calculation, but reduces the repeated computation of old data. Thus, our work fits real-time property better than other works and also be more appropriate in practice.

For the future work, we have some suggestions:

(1) We want to design formulas for dynamic CF. In our work, the similarities and prediction calculation formulas are the same as traditional CF. The formulas are designed as static, in other words, they have no time factor. We achieve dynamic property by multiplying a decay coefficient on similarities. We believe that a well-designed dynamic formula will not only raise the accuracies but also improve the execution performance.

(2) Practically, a Recommendation System should be hybrid-based. Because the complexity of human-behavior and various items in the system, it has to cooperate with many analyzed algorithms to reinforce the accuracy. Nonetheless, too many calculations may cause lower performance. Accordingly, we look forward to develop a well-designed hybrid-based Recommendation System.

Bibliography

[1] G. Adomavicius and A. Tuzhilin, “Toward the Next Generation of Recommender System: A Survey of the State-of-the-Art and Possible Extensions,” IEEE Transactions Knowledge and Data Eng., vol.18, no.6, pp. 734-749, 2005.

[2] M. Balabanvic and Y. Shoham, “Fab: Content-based, Collaborative Recommendation,” Comm. ACM, vol.40, no. 3, pp. 66-72, 1997.

[3] I. Bartolini, Z. Zhang, and D. Papadias, “Collaborative Filtering with Personalized Skylines,” Knowledge and Data Engineering, IEEE Transactions, vol. 23, Issue 2, pp. 190 – 203, 2011.

[4] J. Breese, D. Heckerman and C. Kadie, “Empirical Analysis of Predictive Algorithm for Collaborative Filtering,” Proc. 14th Conf. Uncertainty in Artificial Intelligence, 1998.

[5] R. Burke, ”Hybrid web recommender systems,” The Adaptive Web LNCS, vol. 4321, pp. 377–408, 2007.

[6] R. Burke, “Hybrid Recommender Systems: Survey and Experiments,” User modeling and user-adapted interaction, vol. 12, Number 4, pp. 331-370, 2002.

[7] T. Chen, W. Han, H.Wang, Y. Zhou, B. Xu and B. Zang, “Content recommendation system based on private dynamic user profile,” Machine Learning and Cybernetics, vol. 4, pp. 2112 – 2118, 2007.

[8] M. Claypool, A. Gokhale, and T. Miranda, “Combining content-based and collaborative filters in an online newspaper, ” Proc. of the ACM SIGIR-99 workshop on recommender systems-implementation and evaluation, 1999.

[9] Y. Ding, X. Li and M. Orlowska, “Recency-based collaborative filtering,” Proceedings of the 17th Australasian Database Conference, vol. 49, pp. 99-107, 2006.

[10] Y. Ding and X. Li, “Time weight Collaborative Filtering,” Proceedings of the 14th ACM international conference on Information and knowledge management, pp. 485 – 492, 2005.

[11] D. Goldberg, D. Nichols, B. Oki and D. Terry, “Using Collaborative Filtering to weave an information tapestry,” Communications of the ACM - Special issue on information filtering, vol. 35, Issue 12, pp. 61 – 70, 1992.

[12] J. Herlocker, J. Konstan and J. Riedl, “An empirical analysis of design choices in neighborhood-based collaborative filtering systems,” Information Retrieval, pp. 287–310, 2002.

[13] T. Hofmann, “Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis,” Proceedings of the 26th annual international ACM SIGIR conference of Research and Development in Information retrieval, pp. 259-266, 2003.

[14] I. Konstas , V. Stathopoulos and J. Jose, “On social networks and collaborative recommendation,” The 32nd international ACM SIGIR conference on Research and development in information retrieval, pp. 195-202,2009.

[15] Y. Koren, R. Bell and C. Volinsky, “Factorization Techniques for Recommender Systems,” Computer, vol. 42, Issue 8, pp. 30 – 37, 2009.

[16] M. L., C. N. and J.D.J. Perez-Alcazar, “A comparison of several predictive algorithms for collaborative filtering on multi-valued ratings,” ACM symposium on Applied computing, pp. 1033-1039, 2004.

[17] Q. Li and M. Zhou, “Research and design of an efficient collaborative filtering predication algorithm,” Parallel and Distributed Computing, applications and technologies, pp. 171-174, 2003.

[18] G. Linden, B. Smith and J. York, “Amazon.com recommendations: item-to-item Collaborative Filtering,” Internet Computing IEEE, vol. 7, Issue 1, pp.76-80, 2003.

[19] C. Liu and J. Cheng, “Personalized Search Recommendation Based on Gradual

Forgetting Collaborative Filtering Strategy," Intelligent Information Technology Application, pp. 475-478, 2009.

[20] Q. Liu, E. Chen, H. Xiong, C. Ding and J. Chen, "Enhancing Collaborative Filtering by User Interest Expansion via Personalized Ranking," Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions, vol. 42, Issue 1, pp. 218 – 233, 2012.

[21] S. Lo and C. Lin, "WMR-A graph-based algorithm for friend recommendation," Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence, pp. 121–128, 2006.

[22] M. Pazzani, "A Framework for Collaborative, Content-Based and Demographic Filtering," ARTIFICIAL INTELLIGENCE REVIEW, vol. 13, Numbers 5-6, pp. 393-408, 1999.

[23] T. Peng, W. Wang, X. Gong, Y. Tian, X. Yang and J. Ma, "A Graph Indexing Approach for Content-Based Recommendation System," Multimedia and Information Technology (MMIT), vol. 1, pp. 93 – 97, 2010.

[24] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm and J. Ried, "Grouplens: an open architecture for collaborative filtering of netnews," ACM conference on Computer supported cooperative work, pp. 175-186, 1194.

[25] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-Based collaborative Filtering recommendation systems," International World Wide Web Conference, pp. 285-295, 2001.

[26] J. Schafer, J. Konstan and J. Riedl, "E-Commerce Recommendation Applications," Data Mining and Knowledge Discovery, vol. 5, Numbers 1-2, pp. 115–153, 2001.

[27] L. Si and R. Jin, "Flexible Mixture Model for Collaborative Filtering," Twentieth international conference on machine learning, Washington DC, 2003.

[28] H. Tan and H. Ye, "A Collaborative Filtering Recommendation Algorithm Based

On Item Classification,” Computing in the Global Information Technology (ICCGI), Fifth International Multi-Conference, pp. 66 – 70, 2010.

[29] T. Tang, P. Winoto and K. Chan, “On the Temporal Analysis for Improved Hybrid Recommendations,” Web Intelligence, pp. 214-220, 2003.

[30] Z. Wang, Y. Tan and M. Zhang,”Graph-based recommendation on social networks,” Proceedings of International Asia–Pacific Web Conference, pp. 116–122. , 2010.

[31] R. Wong, J. Pei,, A. Fu and K. Wang, “Mining Favorable Facets,” Proc. 13th ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining, 2007.

[32] C. Xia, X. Jiang, S. Liu, Z. Luo and Z. Yu, “Dynamic Item-Based Recommendation Algorithm with Time Decay,” Natural Computation (ICNC),Sixth International Conference, pp. 242-247, 2010.

[33] S. Yu, ”The dynamic competitive recommendation algorithm in social network services,” Information Sciences, vol. 187, pp. 1–14, 2012.

[34] C. Zeng, C. Xing and L. Zhou, “Similarity measure and instance selection for collaborative filtering,” Proceedings of the 12th international conference on World Wide Web, pp. 652 – 658, 2003.

[35] C. Ziegler, G. Lausen and L. THieme, “Taxonomy-driven Computation of Product Recommendations,” Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 406 – 415, 2004.

[36] Y. Zhao, C. Zhang and S. Zhang, “A Recent-Biased Dimension Reduction Technique for Time Series Data,“ Advances in knowledge discovery and data mining, Lecture Notes in Computer Science, vol. 3518, pp. 107-176, 2005.

[37] “MovieLens,” http://www.grouplens.org/node/73/

[38] “Netflix,” http://www.netflixprize.com/