An Improved Collaborative Filtering Algorithm Based on Multi-Context Awareness

Yuan Bai[1, a], Yu Fang[2, b]

[1]Department of Electronic and Information Engineering, Tongji University, Shanghai, 201804,China

[2]Department of Electronic and Information Engineering, Tongji University, Shanghai, 201804,China

[a]email: 008baiyuan@163.com, [b]email: fangyu@tongji.edu.cn

Keywords: Context-Aware; Collaborative Filtering; Recommender Systems

Abstract. In the research field of recommender systems, people tend to focus on mining relationship between users and items, but less take the additional contextual information into consideration. To solve the problem, we put forward an improved collaborative filtering algorithm based on multiple contexts (MCCF). The algorithm utilizes AHP to calculate the weight vector of context features, then transform contextual information into SimHash sequence based on the weight vector to calculate context similarity. Use the composite similarity which compounds the user preference and the context to structure the target user's nearest neighbor set. Finally, integrate the composite similarity into the traditional collaborative filtering algorithm to predict preferences and recommend items. Experiments show that MCCF is superior to the traditional ones and single context ones in both performance and recommendation accuracy.

1 Introduction

In recent years, recommender systems change from simply based on users or items to based on context-aware recommendation whose purpose is observing users' behavior, understanding users' intent. The technology of multimedia recommendation is growing, but the majority of existing recommendation approaches are based on the multimedia label or users' history information, and less consider the impact of the contextual information on their preferences.

Early 1990s, Mark Weiser[1] proposed the concept of ubiquitous computing, then Schilit[2] introduced context awareness into mobile computing. This technology is one of the core areas of ubiquitous computing, which enables the system automatically discover and utilize location, surroundings and other contextual information to provide users with services and computing resources[3]. With the development of context awareness, application services based on context-aware have aroused widespread concern in many areas. Nowadays, cell phones and other mobile devices built a wealth of sensor function. It has brought great convenience to the acquisition of contextual information, and with the rapid development of mobile sensing and computing technology, which is possible to use real-time analysis of sensor data to explain contextual information. So, how to accurately make full use of contextual information and apply context-aware technologies to recommendation fields has become a new hotspot. It has a long-term significance to solve the typical problem of existing recommender systems and improve the accuracy of recommendation.

Motivated by this, we need some recommendation methods that can perceive users' context to meet personalized needs.

2 Related Work

2.1 Collaborative Filtering Algorithm

The concept of collaborative filtering was first proposed by Goldberg, Nicols, Oki and Terry[4] in 1992. It was applied to Tapestry system, mainly to solve Xerox Company's information overload problem in Palo Alto research center. Collaborative filtering algorithm is widely used in the field of recommender systems. It can recommend for the target user based on his neighbors' preferences information. Its basic idea is to find users' neighbors by similarity computing, and the target user's preferences to the item can be approached by the weighted average of his neighbors' preferences to the same item.

Since recommendation algorithms rely heavily on historical data, the lack of information about new users or new item can't generate a recommendation, which is called cold-start problem. Currently others have proposed some solutions[5] for the cold-start problem, which are mainly divided into two aspects: one is directly combining rating data of traditional collaborative filtering with specific methods to solve. The other one is integrating new users or new items' content attribute information with the traditional collaborative filtering rating data to improve cold-start problems. The introduction of multi-context information belongs to the latter. It can partially alleviate cold-start in the field of conventional recommender systems.

2.2 Context Recommender

Adomavicis and Tuzhilin[6-7] noted earlier, integrating the contextual information into recommender systems will help to improve the accuracy of the recommendation. So they put forward the concept of " Context-aware Recommender Systems, CARS[8]". The traditional "user - item" two-dimensional rating utility Model $R: User \times Item \rightarrow Rating$ can be expanded to multi-dimensional rating utility model $R: User \times Item \times Context \rightarrow Rating$ which include a variety of contextual information. CARS mainly solve the problem that recommends qualified items for different users according to their contextual conditional constraint. In addition, the definition from Dey et al.[9-10] is widely quoted: "Context is any information used to describe the entity state in which the entity can be a person, place or the interaction between the user and the application associated with the object (including user and application itself)".

In order to make use of contextual information in recommender systems, domestic and foreign researchers conducted a series of studies [11-17]. Kai Lu et al. [13] propose Hidden-parameter model based on user short-term period to improve collaborative filtering algorithm, which belongs to using a single contextual feature (time) to gain improvement. Zhengxing xu et al. [14] studies how to perform travel recommendation by exploiting geotagged signature. Researchers take two types of contextual features (seasons and weather) into account in mining and recommending process. Xinxi Wang et al. [15] present an approach to employ the contextual information of users' real-time activities(working, studying, running, sleeping, walking and shopping) collected by mobile devices, then analyze music content to train a statistical model and use the model to predict which song is suitable for the activity's context. Satoshi Kurihara et al. [16] propose an application recommendation mechanism which uses context features consisting of location and time to recommend an appropriate application for the user in specific context. Ricardo Dias et al. [17] propose a session-based collaborative filtering algorithm by using temporal context to increase the effect of music recommendations. They use temporal context and the diversity of songs to collectively reflect the session, and then create a temporal context vector consisting of four characteristics (Time of day, Weekday, Day of Month, Month) to cluster similar session to generate recommendations. However, the foregoing analysis shows that majority of researchers start from one or two single contextual information and do not take full advantage of context which includes rich content. According to the above problems, the improved point of this paper is to incorporate multiple context features to improve the recommendation accuracy and how to add context model into the collaborative filtering algorithm.

3 The description of MCCF recommendation algorithm

As mentioned above, the traditional collaborative filtering algorithms in section 2.1 ignore the role of users' context. In fact, the time-based and location-based recommendation studied by some researchers now can be seen as one-dimensional context-aware recommendation. However, collaborative filtering algorithm based on a single context does not take into account the overall influence of multiple contexts. Therefore, this section will describe the improved collaborative filtering algorithm based on multiple contexts in detail. First, we will explain how to use AHP to get impact weights of the contextual features. Then we describe how to apply the weight vector W to the context similarity calculation process by SimHash. Finally we integrate context similarity to the traditional collaborative filtering algorithm and give experimental analysis of the complex similarity parameter ɵ.

3.1 AHP to determine the weight of contextual features

In a comprehensive assessment of multiple factors, various factors have different degrees of influence on the final assessment target. So we need to give different weights to factors, such as time, weather, location, status or mood. It can't be determined directly that which of them have the most weight. So we need to sort n kinds of contextual features according to their pros and cons. Here we use the analytic hierarchy process to determine the weight of context features.

Analytic Hierarchy Process (AHP) is formally proposed in the 1970s by T.L Saaty[18](a US operation researcher). It is an important method for qualitative and quantitative analysis of multi-objective decision-making. It can resolve complex problems into a number of layers and a number of factors and make simple comparisons and calculations among the factors. We can get degrees of importance weights for different schemes. How to calculate the sorting weight by judgment matrix is the core work of AHP.

Then we follow the steps of AHP to obtain the weight vector W of context features:

(1) Establish hierarchical model

AHP emphasizes solving problems hierarchically, so we need to analyze the relationship between decision-making objectives and decision factors. The context-aware recommendation is the ultimate goal, so it is the target layer. Factors which affect the target layer are contextual information and user preferences, which are regarded as the criterion layer. The effect weight to the target layer is represented by a vector $W = [\theta, (1-\theta)]$. The value of θ will be described in section 3.3; Sub-criterion layer treats all the contextual features as an assessment for upper layer. In this section we will focus on calculation of characteristic factors' weight contained in contextual information; Scheme layer uses MCCF algorithm described in section 3.3 instead of decision-making method. In summary, in accordance with the evaluation criteria and all the relevant factors in the assessment, we established AHP model shown in Fig. 3.1.

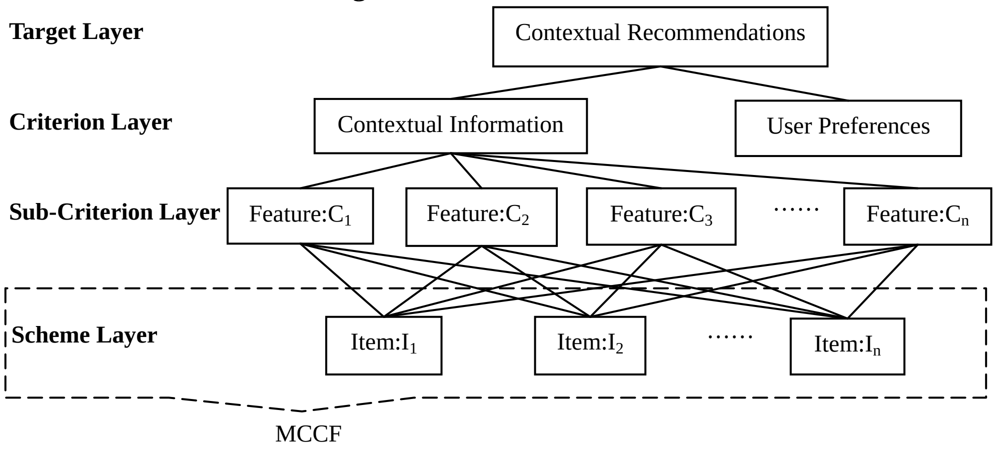


Fig. 3.1 the hierarchical model of Context-aware recommendation

(2) Structure judgment matrix

This step is critical in AHP decision analysis. After establishing a hierarchical model, we apply methods of paired comparisons and 1-9 scale to all factors in sub-criterion layer, which can influence the criterion layer "contextual information" and constitute a judgment matrix. Then we use the matrix to determine all the factors' weight in the same layer, namely weight that $C_1, C_2, C_3, \dots, C_n$ effect contextual information. We mark the weight as a vector:

$$W = [w_1, w_2, w_3, \dots \dots, w_n] \quad (1)$$

We suppose there are n factors involved in the comparison and $A = (a_{ij})_{n\times n}$ is called judgment matrix. Judgment matrix represents the relative importance assessment of all relevant factors for a factor in the upper layer. It must follow consistent matrix method that proposed by Satty and others, namely:

a) Do not put all the factors together to compare, compare them pairwise;

b) Using the relative scale to reduce difficulties when compare the distinct factors, and improve accuracy.

c) The value of a_{ij} in judgment matrix is assigned from 1 to 9 or their reciprocal value according to the following scale. Assignment criteria are as follows:

$a_{ij} = 1$， the factor i and the factor j have the same importance to upper layer;

$a_{ij} = 3$， the factor i is a little more important than the factor j;

$a_{ij} = 5$， the factor i is significantly more important than the factor j;

$a_{ij} = 7$， the factor i is strongly more important than the factor j;

$a_{ij} = 9$， the factor i is vitally more important than the factor j;

$a_{ij} = 2n, n = 1,2,3,4$， the ratio of the importance of factor i and j between $a_{ij} = 2n - 1$ and $a_{ij} = 2n + 1$;

$a_{ij} = 1/n, n = 1,2,\dots,9$， if and only if $a_{ji} = n$。

The characteristic of judgment matrix is $a_{ij} > 0$， $a_{ij} = 1$， $a_{ij} = 1/a_{ji}$。

According to the above method, we can get judgment matrix (Equation 2), which consists of each weight ratio of two factors. Now we assume that the context contains five characteristic factors, which are time C_1, weather C_2, location C_3, mood C_5 and status C_5. Use the method of paired comparison to compare these five contextual features pairwise and gain judgment matrix (Equation 3) as follows:

$$A = \begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \cdots & \frac{w_2}{w_n} \\ \cdots & \cdots & & \cdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & \frac{w_n}{w_n} \end{bmatrix} \quad (2) \qquad A = \begin{array}{c} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \end{array} \begin{bmatrix} 1 & 1/7 & 1/3 & 1/5 & 1/3 \\ 7 & 1 & 4 & 3 & 1/5 \\ 3 & 1/4 & 1 & 1/2 & 1/4 \\ 5 & 1/3 & 2 & 1 & 1/7 \\ 3 & 5 & 4 & 7 & 1 \end{bmatrix} \quad (3)$$

(3) Calculate the weight of context features

The right side of the equation 3 is multiplied by W to obtain the characteristic equation $AW = \lambda W$. Then we should calculate W to satisfy $(A - \lambda E)W = 0$ and $|W| = 1$, which is to calculate eigenvector of the judgment matrix A. Here we use the largest eigenvalue λ = 5.8 of A as the weight vector. In the end, the final result of the equation 3 approximates $W = [0.02, 0.26, 0.08, 0.12, 0.52]$.

3.2 SimHash calculate contextual similarity

SimHash algorithm is proposed by Chariker et al. [19]. The main idea is to reduce the dimension of an arbitrary high-dimensional vector and map it into an f-bit signature (signed by the Hamming distance) to compare two text to determine whether they are similar. Specifically, we change the contextual information into 01 binary string and calculate context similarity of two users by hamming distance, which is the number of different bits between two SimHash binary strings.

By the thought of SimHash algorithm [20], we can convert every user's contextual information into a binary signature and store them in advance, then calculate context similarity directly through their Hamming distance. Afterwards, the method both saves storage space of dataset and speeds up computing speed of context similarity.

（1） Segmentation: separate all the contextual text we have collected into different context features.

（2） Hash: transform each feature into a hash value by hash algorithm, in which contextual information can be converted into a digital sequence. It can greatly improve computing performance when we calculate the figures' similarity, and reduce the dimensionality of multi-dimensional context.

（3） Weighting: make use of hash generated in the previous step (2) to form weighting numeric string in accordance with different context feature's weight. The weight vector $W = [w_1, w_2, w_3, \dots\dots, w_n]$ has been given in section 3.1, Here we need to multiply each of the weights $w_i \in W$ by 10, because the decimal value should be converted to a value between 0 to 10 and then be weighed. Greater weight indicates this feature is more important than others to the user's preferences.

(4) Merge. Calculate the sequence value of all the contextual features and eventually get a cumulative string.

(5) Dimensionality reduction: convert every bit of the sequence following this: a bit string more than 0 is signed as 1, less than 0 is signed as 0.

(6) Convert the digital sequence string to 01 string, and form the final SimHash signature.

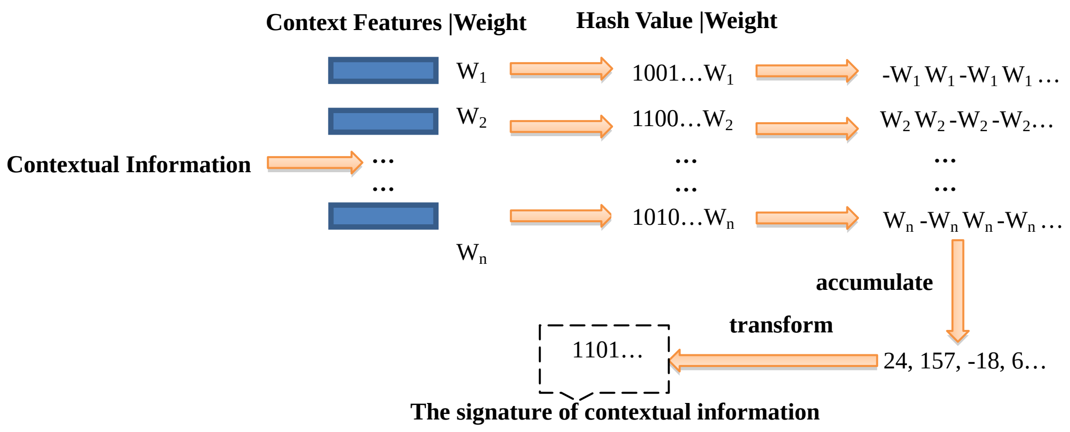


Fig. 3.2 SimHash calculate the sequence of contextual information [21]

Firstly generate an f-bit digital sequence to for each context feature. Generative rules must meet that the corresponding sequence is uniformly random distribution and its binary sequence must be unique to the same context feature. For example, the f-bit hash value of time feature C_1 is expressed as 1001 ..., then corresponding f-bit sequence to this feature is $1,-1,-1,1,\ldots$. Namely one bit of hash value is 1, its corresponding sequence string bit is set to 1, otherwise -1. Then to weight string sequence, the sequence of time feature C_1 is weighted as $w_1,-w_1,-w_1,w_1,\ldots$, then accumulate the weighted sequence corresponding to the various features of contextual information. The summation of accumulated sequence value represents contextual information. Finally, in order to obtain an f-bit signature, we need to compress it further. If one bit in the summation sequence is greater than 0, then the corresponding bit in final signature is 1, if less than 0, the bit is 0. This compression operation is equivalent to leaving the information of the serial string's quadrant, and an f-bit signature can represent up to 2^f quadrants. Eventually the resulting binary string is the context's signature.

By the above conversion, the contextual information is converted to SimHash sequence saves the storage space. Here we sign the context similarity of user u and user v as $\delta(c_u,c_v)$, then calculate $\delta(c_u,c_v)$. The number of different values between two binary strings corresponding to contexts SimHash is called the Hamming distance. We can calculate the similarity of the two contexts by Hamming distance, so the similarity can be calculated by $\delta(c_u,c_v)=1-\text{Hamming Distance}/f$.

3.3 Multi-context collaborative filtering algorithm (MCCF)

The context of user is dynamic and transient. With context switching, user's interests and satisfaction will change dynamically. Therefore, accurate understanding of the user's contextual information to apply this information to recommender systems is a key step when we design a recommendation algorithm. On the other hand, according to the traditional collaborative filtering ideas, it is unable to find neighbor set because new users lack historical information, so the algorithm could not give appropriate recommendations, which is referred to user cold-start problem. MCCF in this paper can rely on contextual information of the new user to measure the potential similarities between other users, and then provide users with recommendation items. It successfully solves new user cold-start problem to some extent.

The MCCF recommending models include three key steps: First, compute the similarity of user preferences and context respectively, then compound them according to a certain percentage; Secondly, screen a neighbor set according to the composite similarity. This step still uses the method of traditional collaborative filtering; Finally, a recommendation stage infers user's favorite items based on the user's nearest neighbors rating information.

In step one, calculation method of similarity still uses Pearson correlation coefficient and makes improvements by introducing the context dimension. Pearson correlation coefficient is known as a product-moment correlation coefficient, which reflects the amount of two variables linear statistical relevance. It is proposed by British statistician Pearson in the 20th century. Suppose there are two users: u and v, their Pearson correlation coefficient can be calculated by Equation 4.

$$Sim(u,v)=\frac{\sum_{j\in I_{u,v}}(r_{u,j}-\overline{r}_u)(r_{v,j}-\overline{r}_v)}{\sqrt{\sum_{j\in I_{u,v}}(r_{u,j}-\overline{r}_u)^2\times\sum_{j\in I_{u,v}}(r_{v,j}-\overline{r}_v)^2}} \tag{4}$$

We add contextual information dimension C to the original Pearson correlation coefficient (Equation 4). The rating score $r_{u,j}$ of user u to item j is changed to the rating score $r_{u,C_u,j}$, which means user u rates item j in the context C_u and the user v is same. Then add context similarity $\delta(c_u,c_v)$ calculated from SimHash in chapter 3.2. Final formula 5 is an improved Pearson correlation coefficient. The similarity becomes the recombination of user similarity and context similarity, θ determines combination ratio. We use θ to control the importance of user similarity and context similarity.

$$Sim(u,v,c_u,c_v)=\theta\frac{\sum_{j\in I_{c(u,v)}}(r_{u,c_u,j}-\overline{r}_{u,c_u})(r_{v,c_v,j}-\overline{r}_{v,c_v})}{\sqrt{\sum_{j\in I_{c(u,v)}}(r_{u,c_u,j}-\overline{r}_{u,c_u})^2\times\sum_{j\in I_{c(u,v)}}(r_{v,c_v,j}-\overline{r}_{v,c_v})^2}}+(1-\theta)\delta(c_u,c_v) \tag{5}$$

About the value of ɵ we find, when ɵ tends to 1, the less the proportion of context similarity, the smaller the corresponding impact on collaborative filtering algorithm, meanwhile MAE (Mean Absolute Error) will stabilize. When ɵ = 1, the coefficient of context similarity becomes 0 and the algorithm degenerate to traditional collaborative filtering algorithm. The experimental results can be seen in Fig. 3.3. When the value of ɵ is around 0.4 to 0.5, MAE in all three algorithms tends to be minimal. At the same time, MCCF forecast quality over the entire range is always better than a single context collaborative filtering algorithm. This algorithm ɵ MCCF argument is set to 0.5.

We still use the idea of traditional collaborative filtering algorithms to select neighbors: Finding the nearest neighbors of the target user u to build a user set $V=\{v_1,v_2,\ldots,v_k\}$ in the user space. It should satisfy $u\notin V$, the similarity $Sim(u,v_1)$ of user v_1 and user u is the highest and the similarity $Sim(u,v_1)$ of user v_2 and user u follows behind, and so on.

Traditional collaborative filtering algorithms use preference information of a neighbor to an item to determine the target user's preference to the same item. The neighbor set is a user set which has similar preference to the target user. MCCF proposed in this paper not only considers the user's own interest preference but also takes the similarity of two users' context into account. Namely when choosing a neighbor set, we should replace the original similarity with the complex similarity of user preference and contextual information. Accordingly, the satisfaction condition that above user set $V=\{v_1,v_2,\ldots,v_k\}$ should satisfy becomes: $u\notin V$, and the similarity $Sim(u,v_1,c_u,c_{v_1})$ of v_1 and u is highest , the similarity $Sim(u,v_2,c_u,c_{v_2})$ of v_2 and u follows, and so on.

In the third stage, the similarity of user preference $Sim(u,i)$ in the corresponding original score prediction formula should be replaced with a composite similarity $Sim(u,i,c_u,c_i)$. Formula 6 is the result, which $p_{u,c_u,k}$ indicates a predicted score of the target user u for the item k in the context c_u , $\overline{r_u}$ is the average of all rating scores given by user u, N is the most similar neighbor set of the target user u among all users.

$$p_{u,c_u,k}=\overline{r}_u+\frac{\sum_{i=1}^{N}Sim(u,i,c_u,c_i)\times(r_{i,c_i,k}-\overline{r}_{i,c_i})}{\sum_{i=1}^{N}Sim(u,i,c_u,c_i)} \tag{6}$$

This average weighted strategy comprehensively considers the user's rating conditions for all items. When the number of items evaluated by the user is numerous, this method will have a good recommendation effect, but when there are fewer number of ratings, some individual ratings for items will have a greater influence on the average score $\overline{r_u}$, In this situation, the average rating score can’t correctly reflect the user’s rating score for the majority items, so we'll set a threshold during the pre-processing of data to filter invalid users whose score times less than this threshold.

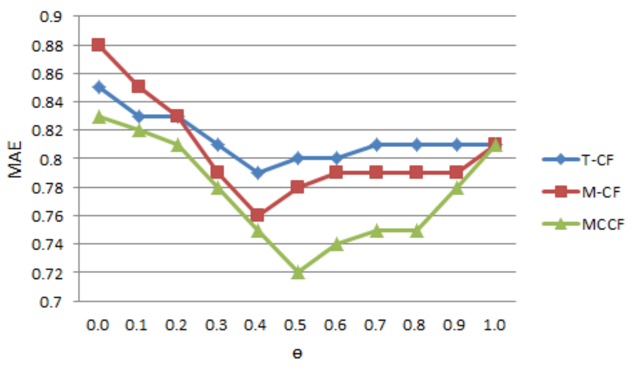
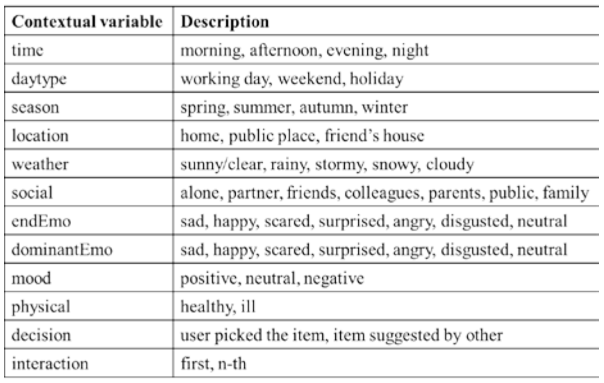


Fig. 3.3 When the neighboring set K=10, distribution of ɵ corresponding to the value of MAE



Contextual variable	Description
time	morning, afternoon, evening, night
daytype	working day, weekend, holiday
season	spring, summer, autumn, winter
location	home, public place, friend’s house
weather	sunny/clear, rainy, stormy, snowy, cloudy
social	alone, partner, friends, colleagues, parents, public, family
endEmo	sad, happy, scared, surprised, angry, disgusted, neutral
dominantEmo	sad, happy, scared, surprised, angry, disgusted, neutral
mood	positive, neutral, negative
physical	healthy, ill
decision	user picked the item, item suggested by other
interaction	first, n-th

Table 4.1 The description of LDOS-CoMoDa Dataset’s situational context features

4 Experiments and Analysis

4.1 DataSet

In order to verify the effectiveness and feasibility of the proposed method in this paper and that it has optimum accuracy compared with the traditional collaborative filtering recommendation algorithm, we use two datasets to conduct the experiment.

(1) LDOS-CoMoDa public dataset

The dataset is collected by researchers in Ljubljana University, they collected 1611 evaluation records from a total of 89 users on 946 films. These users who participated this survey have an average age of 27 years old and come from 16 different cities in six countries. LDOS-CoMoDa is a rich contextual information rating dataset, in addition to basic information of movies and users, it also contains 12 kinds of contextual features, as shown in Table 4.1.

(2) MovieLens 100k dataset

MovieLens 100k dataset comes from the official website of MovieLens, and it collected by GroupLens research team in Minnesota University lasting for seven months. The dataset provides 100 000 score records from 943 user on 1682 films. Although it does not contain situational contextual information of users when they enjoy the film, it carries rich information including user contextual information (age, gender, occupation) and film contextual information (title, year, release date, genres), we can combine these two kinds of context to replace situational contextual information.

4.2 Evaluation Index

Recommendation accuracy is the basic indicator to evaluate a recommendation algorithm, the experiment in this paper use Mean Absolute Error (MAE) to evaluate the accuracy of the proposed algorithm.

$$MAE = \frac{\sum_{i=1}^{N} | p_i - q_i |}{N} \qquad (7)$$

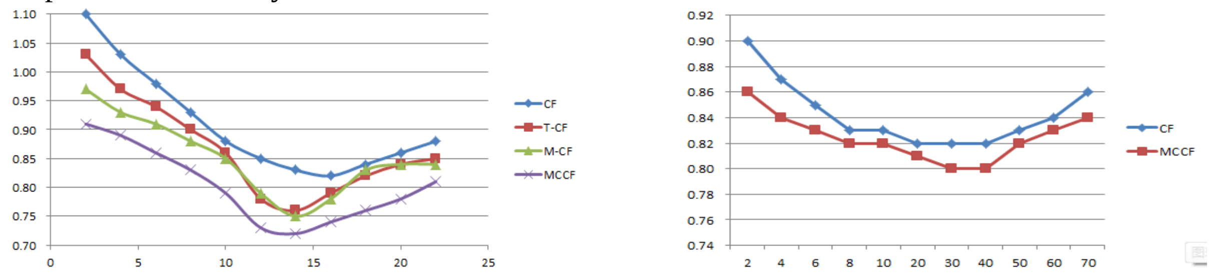
4.3 Experimental results and analysis

In this paper, experiments are respectively based on LDOS-CoMoDa dataset and MovieLens 100k dataset. Compare MCCF with traditional collaborative filtering algorithm (CF), single context collaborative filtering algorithm to conduct a comprehensive assessment. The results are shown in Fig. 4.1, 4.2 (x-axis represents the size of neighbors, y-axis represents MAE), the proposed algorithm MCCF effectively improves the recommendation accuracy, its performance is superior to all other methods, thereinto single context collaborative filtering algorithm takes second place and the traditional CF is worst. In the whole interval, we can see that the effect of recommendation based on a variety of contexts is better than a single context, MCCF has been at best range.

(1) The contrastive experiment based on public LDOS-CoMoDa dataset

The experiment respectively uses traditional collaborative filtering algorithm, single context collaborative filtering algorithm based on single context (including time-based T-CF and mood-based M-CF) and the proposed collaborative filtering algorithm MCCF to forecast preferences . Fig. 4.1 shows the comparative results of four kinds of algorithms.

Fig. 4.1 shows, MCCF has lower MAE than traditional CF and single context CF in the circumstance of different neighbor set and can achieve best results. The trend is to be reduced to the optimum value and then gradually increase. In contrast, when K <10, curve is steeper, with the size of neighbor set increasing MAE promotes significantly, when K ranges between 10 and 15, MAE drops to the lowest point and later tends to stabilized in a certain extent. On the whole, MCCF has a higher recommendation quality than the other methods, meanwhile the size of neighbor set also has some impact on the recommendation accuracy, thus confirms the fact that contextual information can improve the accuracy of recommendation.



(a) On LDOS-CoMoDa (b) On MovieLens 100k

Fig. 4.1 The contrastive experiment based on two kinds of datasets

(2) The contrastive experiment based on MovieLens 100k dataset

The dataset previously have been classified in accordance with 80% training set and 20% testing set, every user has only one score to a film, so we can use user contextual information and film contextual information together to replace situational contextual information required herein. As can be seen from Fig. 4.2, curve has the roughly same trend with Fig. 4.1, when selecting a different size of neighbor set, MCCF has a lower MAE than the traditional collaborative filtering algorithm. Meanwhile, when the size of neighbor set ranges from 30 to 40, the algorithm has highest accuracy.

5 Conclusion

This paper proposes an improved collaborative filtering recommendation algorithm based on multiple context to solve the problem that recommender systems ignore the effect of environmental information, and gives an solution to new users' cold-start problem. Experimental results show that MCCF can really improve the recommendation accuracy and provide better recommendation results. Finally, this paper relates only a limited context features, in fact, more and more comprehensive contextual information and how to tune their impact weights, can achieve better recommendation quality and also becomes the focus of future researches.

References

[1]Mark Weiser. The computer for the 21st century[J]. Scientific American,1991,265(3):66-75.

[2]Bill N. Schilit, Norman Adams and Roy Want . Context-aware Computing Applications[J].IEEE Workshop on Mobile Computing Systems and Applications, Washington, D. C.,USA:IEEE Computer Society, 1994.

[3]Licai Wang, Xiangwu Meng, Yujie Zhang. Context-Aware recommender systems[J].Journal of Software,2012, 23(1):1-20.

[4]David Goldberg, David Nichols, Brian M. Oki and Douglas Terry. Using collaborative filtering to Weave an information Tapesty[J].Communications of the ACM,1992,35(12):61-70.

[5]Dongting Sun, Tao He, Fuhai Zhang. Research on cold start Problem of Recommender Systems[J]. Computer and Modernization, 2012(5):59-63.

[6]Adomavicius G, Tuzhilin A. Context-Aware recommender systems[M]. In: Recommender Systems Handbook. Berlin: Springer-Verlag,2011. 217-253.

[7]Adomavicius G, Sankaranarayanan R, Sen S, Tuzhilin A. Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach[C]. ACM Trans. Inf. Syst., 2005,23(1):103–145.

[8]Adomavicius G, Ricci F. RecSys'09 Workshop3: Workshop on context-aware recommender systems (CARS 2009)[J]. In: Proc. of the RecSys 2009. New York: ACM Press, 2009. 423-424. http://dl.acm.org/citation.cfm?id=1639806.

[9]Anind K. Dey. Understanding and Using Context[J]. Personal and Ubiquitous Computing Journal, 2001,5(1):4-7.

[10] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a Better Understanding of Context and Context-Awareness[C]. Handheld and Ubiquitous Computing, 1999(1707):304-307.

[11]Katsuhiko Kaji, Keiji Hirata, Katashi Nagao. A Music Recommendation System Based on Annotations about Listeners' Preferences and Situations[J]. IEEE Computer Society, 2005.

[12]Donghai Guan, Qing Li, Sungyoung Lee, Youngkoo Lee. A Context-Aware Music Recommendation Agent in Smart Office[C]. Fuzzy Systems and Knowledge Discovery Lecture Notes in Computer Science, 2006(4223):1201-1204.

[13]Kai Lu, Kuanyuan Zhang, Bin Wang. CICF: A Context Information Based Collaborative Filtering Algorithm[J]. Journal of Chinese Information Processing, 2014(28):122-128.

[14] Zhenxing Xu, Ling Chen, Gencai Chen. Topic based Context-aware travel Recommendation method exploiting geotagged photos[J]. Neurocomputing,2015(155):99-107.

[15] Xinxi Wang, David RosenBlum, Ye Wang.Context-Aware Mobile Music Recommendation for Daily Activities[J]. ACM Multimedia 2012,99-108.

[16] Satoshi Kurihara, Koichi Moriyama and Masayuki Numao. Context-aware application prediction and recommendation in mobile devices[J]. 2013 IEEE/WIC/ACM International Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT).

[17] Ricardo Dias, Manuel J. Fonseca .Improving Music Recommendation in Session-Based Collaborative Filtering by Using Temporal Context[J]. Tools with Artificial Intelligence(ICTAI), 2013 IEEE 25th International Conference,2013:783-788.

[18] TL Saaty. The Analytic Hierarchy Process[J]. Proceedings of the Second International Seminar on Operational Research in the Basque Provinces, 1980, 4(29):189-234.

[19] M. Charikar. Similarity estimation techniques from rounding algorithms[J]. In Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pages 380-388, 2002.

[20] Caitlin Sadowski and Greg Levin. Simhash: Hash-based similarity detection[J]. Technical report, Technical report, Google, 2007.

[21]Leoncom, The Duplicated Web Pages Removal of SimHash and Google[EB/OL]. http://leoncom.org/?tag=simhash, 2011-04-12/2015-04-23.