ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Dynamic Personalized Recommendation on Sparse Data

N.Shilpa Ambedkar
M.Tech Student
Department of CSE
Sri Venkateshwara Engineering College

Vazed Ahmed
Assistant Professor
Department of CSE
Sri Venkateshwara Engineering College

Abstract*—Recommendation techniques are very important in the areas of E-commerce and other Web-based services. One of the main difficulties is to provide high-quality recommendation on sparse data dynamically. Here in this paper, we proposed a novel dynamic personalized recommendation algorithm, in which data contained in both profile and ratings contents are utilized by exploring latent relations between ratings, a set of dynamic features are designed to describe user preferences in multiple phases, and finally a recommendation is made by adaptively weighting the features. Experimental results on public datasets show that the proposed algorithm is satisfying performance.

Key Terms*—dynamic recommendation, dynamic features, multiple phases of interest.

1 INTRODUCTION

In these days the internet became an indispensable part of our lives, and it provides a platform for enterprises to deliver information regarding products and services conveniently. As the amount of this kind of information is increasing rapidly, one great challenge is making sure that proper information can be delivered quickly to the customers. Personalized recommendation is a desirable way to improve customer satisfaction and retention [1], [2].

We have mainly three approaches to recommendation engines based on different data analysis methods, i.e., rule-based, content-based and collaborative filtering. In these, collaborative filtering (CF) needs only data about past user behavior like ratings, and its two main approaches are the neighborhood methods and latent factor models. The neighborhood methods can be user-oriented or item-oriented. They try to find like-minded users or similar items based on co-ratings, and predict based on ratings of the nearest neighbors. Latent factor models try to learn latent factors from the pattern of ratings using techniques like matrix factorization and use the factors to compute the benfits of items to users. CF has made great success and been proved to perform well in scenarios where user preferences are relatively static.

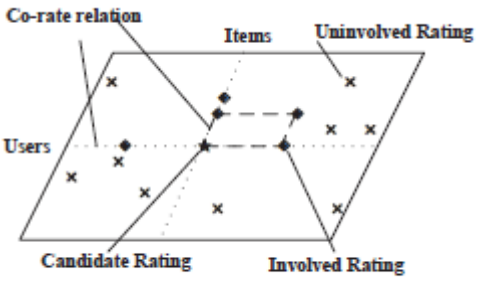
There are two problems that prevent accurate prediction of ratings (in most dynamic scenarios,) – the sparsity and the dynamic nature. Because a user can only rate a very small proportion of all items, the $U \times I$ rating matrix is quite sparse and the amount of information to estimate a candidate rating is less enough. While latent factor models involve most ratings to know the general taste of users, they still have difficulties in catching up with the drifting signal in dynamic recommendation because of sparsity, and it is hard to physically explain the reason of the involving. The dynamic nature decides that users' preferences may drift over time in dynamic recommendation, resulting in different taste to the items in different phases of interest. In our experiences, the interest cycle of differs from user to user, and the pattern how user preferences changes cannot be described by several simple functions. Moreover, CF approaches usually accounter the cold-start problem that is amplified in the dynamic scenario since the rate of new users and new items would be high.

Other researchers attempted to solve the above problems. Hybrid approaches which combine contentbased and collaborative filtering in different

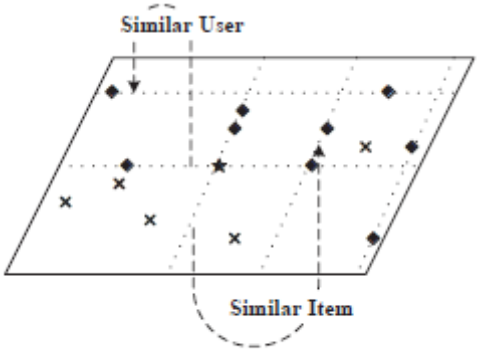
ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

ways were proposed to alleviate the sparsity problem, where more information were mined than just in each of them. Prassas et al, classified items into so many categories by using content information and did choose recent categories for performing Item-Based Collaborative Filtering (IBCF). Kim and Li proposed group similarity by clustering and used it to modify original item-item similarity matrix. The principle of utilization of rating data in these algorithms is shown in Fig. 1.(a). Some approaches emphasize utilization of time information for dealing with the dynamic nature.

Here in this paper, we propose a novel hybrid dynamic recommendation approach. Firstly, for utilizing more information while keeping data consistency, we take item content and user profile to extend the co-rate relation between ratings through each attribute, as shown in Fig. 1.(b). The involved ratings can reflect similar users' preferences and provide useful information. And to enable the algorithm to catch up with the changing of signals quickly and to be updated conveniently, a set of dynamic features are proposed based on time series analysis (TSA) technique, and relevant ratings in each phase of interest are added up by applying TSA to describe users' preferences and items' reputations. Then we propose a personalized recommendation algorithm by adaptively weighting the features according to the amount of utilized rating data. The experimental results show that the proposed algorithm is effective with dynamic data and significantly outperforms previous algorithms.



(a) Common Hybrid filtering[10]



(b) The proposed approach

Fig. 1. Ratings associated in different methods, where star, diamond shape and × represent destination rating, involved rating and uninvolved rating, respectively. In the $U \times I$ plane, ratings along a horizontal line are from the same user and ratings along a vertical line are of the same item. "Similar" here means "identical or close in some attribute of the profiles".

The main aim of this paper can be summarized as follows: (a) More information can be used for recommender systems by investigating the similar relation among related user profile and item content. We utilize the similarity among content in each profile attribute so that more content information is used, especially content in those attributes which are hard to quantify. (b) A novel set of dynamic features is proposed for describing users' preferences, that is more flexible and convenient to model the impacts of preferences in different phases of interest compared with dynamic methods used in previous works, since the features are designed according to periodic characteristics of users' interest and a linear model of the features can catch up with changes in user preferences. (c) An adaptive weighting algorithm is designed to combine the dynamic features for personalized recommendation.

2 THE PROPOSED METHOD

In most cases, the drifting of users' preferences or items' reputations is not too rapid, which makes it possible to describe temporal state of them by using some features. Here we first introduce a way to make use of profiles to extend the co-rating relation, and

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

then we propose a set of dynamic features to reflect users' preferences or items' reputations, and then after we propose an adaptive algorithm for dynamic personalized recommendation.

2.1 Relation mining of rating data

For the sparsity of recommendation data, the main issue of capturing users' dynamic preferences is the lack of useful information, which can be done from three sources – user profiles, item profiles and historical rating records. Classical algorithms highly rely on the co-rate relation, which is rare when the data is sparse. Required ratings are found using the co-rate relation, which is simple, intuitional and physically significant when we go one or two steps along, but it keeps limits to the amount of data used in each prediction.

We try to find a different way to find useful ratings instead of searching neighboring nodes. We notice that when considering the factors which affect a rating $r(u, i)$, we keep more focus on some attributes of u and i in their profiles, instead of the user himself or the item itself. For example, if the movie "Gone with the Wind" is given high ratings by middle-aged people and lower ratings by teenagers with no doubt, we first check on the age attribute in a user's profile when predicting rating, instead of other descriptions of the user or how the user has rated other movies. It may not be necessary to stick only to the co-rate relation, and we introduce the *semi-co-rate* relation between ratings whose corresponding user profiles or item contents have similar or identical content in one or more attributes.

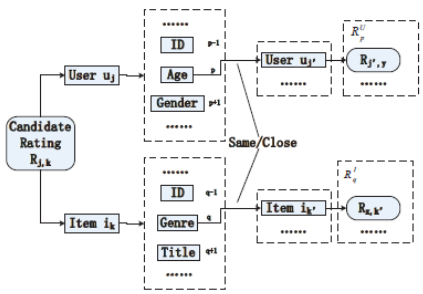


Fig. 2. Finding neighboring ratings in the new relation

Let $\boldsymbol{U=\{u_j\}^m_{j=1}}$ be the entire user set with $|U| = m$, $\boldsymbol{I= \{i_k\}^n_{k=1}}$ be the entire item set with $|I| = n$, R be a $m \times n$ matrix such that its element Rj,k refers to the rating user $\boldsymbol{u_j}$ gave to item $\boldsymbol{i_k}$, and T be the corresponding time matrix such that $\boldsymbol{T_{j,k}}$ denotes the timestamp of $\boldsymbol{R_{j,k}}$. We note the set whose ratings is *semi-co-rate* related with the candidate rating via the p-th attribute in user profile as R^U_p, and similarly we define R^I_q, as shown in Fig. 2. If we note the set whose rating is co-rate related with the candidate rating via user as R^U_0 and similarly we define R^I_0, we have $R^U_0 = (\cap_p R^u_p)$ and $R^I_0 = (\cap_q R^i_q)$. Clearly the *semi-co-rate* is much looser than the co-rate relation, and now that we have found much more related ratings via the relation instead of co-rate, we take only one step neighboring nodes, $(U_p R^u_p)$ $U(U_q R^i_q)$ in this newly defined graph to keep consistency of utilized data.

To avoid overwhelming computation in finding R^U_p (p =1, 2, ...) and R^I_q (q = 1, 2, ...) for all user ratings and to solve the difficulty in quantification of some contents, classification and clustering are done as the content of each attribute in profiles of users and items, and R is then separated into rating subsets $R^{U,c}{}_p$ (c = 1, 2, ...) or $R^{I,c}{}_q$ (c = 1, 2, ...) accordingly in actual implementation and c is the class number. For example, "Age" is the p-th attribute in user description, then R is divided into six subsets for this attribute. With the help of clustering techniques like K-Means on content of "Age", we divide user age into different ranges on the basis of relevant users' ages of all ratings in R. Then for user rating $R_{j,k}$, we can conveniently find its neighboring ratings in our algorithm by reaching relevant subsets, which is done by matching each attribute content of u_j and i_k to the nearest subsets. We limit the size of each separated subset in online calculation, and the when new ratings are added in, then earliest ratings will be removed from the rating subset. Through these techniques, we introduced a general relation between ratings and an extended way of information mining in personalized recommendation.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

2.2 Dynamic feature extraction

Three kinds of methods were proposed in concept drift to deal with the drifting problem as *instanceselection*, *time-window* (usually time decay function) and*ensemble learning*. Koren [4] also proposed an algorithm to isolate transient noise in data using temporal dynamics to help recommendation. These methods are useful to make progressin precision of dynamic recommendation, but some of the weaknesses of these methods are : decay functions cannot precisely describethe evolution of user preferences and only isolating transientnoise cannot catch up with the change in data.

Here we propose a set of dynamic features to describe users' multi-phase preferences in consideration of computation, flexibility and accuracy. It is impossible to learn weights of all ratings for every user, but it is possible to learn the general weights of ratings in the user's different phases of interest if the phases include ranges of time that are long enough. For convenience of notation, we relabel all subsets $R^{U,c}{}_p$ and $R^{I,c}{}_q$ acquired through the extended information mining as R_s (s = 1, 2, ...).

To enable the features to describe users' preferences, we divide each rating subset R_s into several secondary subsets $R^d{}_s$ (d = 1, 2, ...) with the help of the time distances between each rating in R_s and the candidate rating $R_{j,k}$, where each secondary subset is manually assigned with a range of time-distance, and then we use some basic algorithms loke *time series analysis* (TSA) calculate the features on each secondary subsets.

The earlier ratings in the theory of time series analysis, should impact the predictive features less, and thus they should have lower weights. So if we perform TSA algorithm on a secondary subset of R (i.e. $R^d{}_s$) to get a feature $fea_{s,d}$, there is a formulation as:

$$fea_{s,d} = \sum_{l=1}^{o} \frac{w_l}{w} R^d_{s,l}, \qquad (1)$$

where $\#R^d{}_s = o$, $R^d{}_{s,l}$ (l = 1, 2, ..., o) are the rating values which are from the subset $Rd\ s$ and listed in reversed time order. And positive weight parameters w_l, (l = 1, 2, ..., o) and normalization factor w should satisfy

$$\begin{cases} w = \sum_{l=1}^{o} w_l, \\ w_{l_1} \geq w_{l_2} \ if \ l_1 < l_2. \end{cases} \qquad (2)$$

Since the subsets are updated frequently, index smoothing, which is a classic TSA algorithms, is chosen as the basic TSA algorithm:

$$\begin{cases} R^d_s = \{R_{j',k'} | R_{j',k'} \in R_s \ and \ T_{j,k} - T_{j',k'} \geq T_d\}, \\ fea_{s,d} = \sum_{l=1}^{o} \mu(1-\mu)^{l-1} R^d_{s,l}, \end{cases} \qquad (3)$$

Here $R^d{}_s$ (d = 1, 2, ...) are the secondary subsets, T_d (d = 1, 2, ...) are a sequence of time differences manually set, $R^d{}_{s,l}$ (l = 1, 2, ..., o) are the rating values listed in reversed order in the subset, μ is the forgetting factor for index smoothing. We have tested different values for mu in the experiments and set μ = 0.95 empirically.

All $f\ ea_{s,d}$ (d = 1, 2, ...) and the sizes of $R^d{}_s$ (d = 1, 2, ...) are recorded as dynamic features. With the dynamic features, we have to optimize their weights to get the best estimation of the user rating, and in this way we transformed the training of a recommendation model into weight learning across different secondary rating subsets. Now that the features are related to phases of interest and latent relations between ratings, we would see how the preferences differ with each other in impacting the candidate rating by analyzing optimal weights of the features. We can also see in Eq(3) that the feature extraction need not heavy computation. Finding all R^d needs only comparison in time s one by one, and the computation of $f\ ea_{s,d}$ is very efficient, The proposed algorithm is termed as *Multiple Phase Division* (MPD). In this way we have proposed a flexible way of feature extraction.

2.3 Adaptive weighting algorithm

As features like $f\ ea_{s,d}$ (s = 1, 2, ..., d = 1, 2, ...) gained by applying *Multiple Phase Division* are all

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

normalized rating values, in other words, as content of user and item profiles have been quantified in the feature extraction, it is convenient for us to organize them for accurate rating estimation by adaptive weighting. Sizes of the relevant subsets are also recorded in MPD and can reflect data density.

$R_{j,k}$ is used to note the estimated rating ˆ that user u_j could give to item i_k at time point $T_{j,k}$, and the adaptive linear model can be formulated as: formula (4)

where sizes of relevant subsets are used as prior information in weighting the features to improve recommendation accuracy, $f\,ea_{s,d}$ (s = 1, 2, ..., d = 1, 2, ...) are the features calculated in Eq.(3), $R^d{}_s$ (s = 1, 2, ..., d = 1, 2, ...) denote their relevant secondary rating subsets, b_{uj} and b_{ik} are binary functions denoting the relating state of candidate rating and relevant subset and $\alpha_{s,d}$ and β are weighting parameters which should balance the weights of features and data density, or, balance the affection of data consistency and quantity of information. In detail, buj (s) = 1 if $R_{j,k}$ is *semi-co-rate* related with all ratings in secondary subset R_S through attribute of the user uj denoted by s, else buj (s) = 0, bik (s) = 1 if $R_{j,k}$ is *semi-co-rate* related with all ratings in secondary subset R_S through attribute of the item i_k also denoted by s, else bik (s) = 0.

$$\hat{R}_{j,k} = \sum_s \sum_d (\alpha_{s,d} + \beta(\#R_s^d)) b_{u_j}(s) b_{i_k}(s) fea_{s,d}, \quad (4)$$

It is difficult to solve all parameters in Eq.(4) at once, hence we use sequential optimization. Let

$$\delta_{s,d} = \alpha_{s,d} + \beta(\#R^d), s \quad (5)$$

in Eq.(4) and we first solve for the combined weights $\delta s,d$ (s = 1, 2, ..., d = 1, 2, ...) by minimizing the differences between prediction results of the recommendation algorithm and the real rating values in the training set, where RLS algorithm [17] could be used for optimization, i.e.,

$$E = \sum_{R_{j,k} \in R_{Train}} (\hat{R}_{j,k} - R_{j,k}))^2, \quad (6)$$

Here *RT rain* is the training set or known rating set. But we notice that a user's preferences or an item's reputations are commonly affected by only a few principle factors, indicating that using more features might also bring noise into the recom- mendation. So we changed the destination of the optimization and limited the quantity of the features by regularization, and the training problem can be formulated as:

$$\begin{aligned} &\min_{\delta} : \sum_{R_{j,k} \in R_{Train}} (\hat{R}_{j,k} - R_{j,k}))^2 + \lambda ||\delta||_1, \\ &\text{with} : 0 \le \delta_{s,d} \le 1 \ and \ \sum_s \sum_d \delta_{s,d} = 1. \end{aligned} \quad (7)$$

This is a normal LASSO optimization problem which can be solved via ADMM . Provided the δs are solved, we turn to the second step of the sequential optimization: to solve αs and β. To deal with the uncertainty in solving αs and β from Eq.(5), we introduce the generalization error like in SVM [19]. Here the generalization the generalization error is

$$\max(\textstyle\sum_s \sum_d \alpha_{s,d}^2, \ \sum_s \sum_d \beta^2 (\#R_s^d)^2),$$

And we minimize it to gain satisfying performance as:

$$\begin{aligned} &\min_{\alpha,\beta} : \max(\sum_s \sum_d \alpha_{s,d}^2, \ \sum_s \sum_d \beta^2 (\#R_s^d)^2), \\ &\text{with} : \forall s, d, \ \alpha_{s,d} + \beta(\#R_s^d) = \delta_{s,d}, \ \alpha_{s,d} \ge 0, \ \beta \ge 0. \end{aligned} \quad (8)$$

This optimization problem has explicit solution as:

$$\begin{cases} \beta = \dfrac{\sum_s \sum_d \delta_{s,d}}{2 \sum_s \sum_d (\#R_s^d)}, \\ \alpha_{s,d} = \delta_{s,d} - \beta(\#R_s^d) \ for \ all \ s, d. \end{cases} \quad (9)$$

Now we have a practical way of solving all the parameters. Firstly we solve δs from Eq.(7) using Lasso algorithm, then use Eq.(8) and Eq.(9) to compute αs and β.

3.EXPERIMENTS

3.1 Datasets

Here we collect MovieLens 100k data1 and Netflix Competition data2 from online movie recommender

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

service, and are two datasets in studying personalized recommendation. These two datasets contain abundant rating records which last in a reasonable time, and they are different in composition and dynamic nature of data. We use them for our case study. Time distances between the target rating and historical ratings are defined as time of interest, and we manually assigned 6 time intervals to classify times of interest into multiple phases, i.e., within 1 day, 1 to 7 days, 1 to 4 weeks, 1 to 3 months, 3 to 12 months and more than a year.

3.2 Evaluation

Accuracy indicator which is used frequently for predictive algorithms, Root-mean-square error (RMSE), is also used to evaluate the proposed algorithm.The training and testing data are randomly chosen for the experiments is not suitable for the evaluation of dynamic recommendation. With respect to general causality, it is a critical fact in dynamic recommendation that we can use only historical data but not future data for current prediction in real applications. Unfortunately, the fact is often ignored in previous studies. In traditional RMSE evaluations (even for the Netflix competition), training and testing data are randomly sampled and the train and test split is not based on time. This would produce current prediction based on future data. Even if it is guaranteed that testing instances of each user/item come later than its training instances, the aforementioned issue still exists in algorithms like *IBCF* and *latent factor models* due to the utilization of other users' future ratings.

To provide a better simulation of practical recommendation systems' working in evaluation, we split training and testing data based on time in the same way and evaluate the accuracies of dynamic recommendation algorithms as follows:

1) Sort the entire dataset in normal time order, use a certain training ratio to determine its splitting.
2) Use the earlier part as the training set to adjust all parameters in the recommendation algorithm.
3) Run algorithm on testing set, generate estimated rating for each user-item pair in testing set.
4) Compare estimated ratings and real ratings in the testing set, and calculate RMSE.
5) Use different ratios and repeat last four steps.

3.3 Experiment setup

Comparison of proposed algorithm with some representative and widely-used dynamic recommendation algorithms is done here. In the comparisons, all the competing algorithms were in online updating forms and their parameters were set to their empirically best. Here is the brief introduction of them.

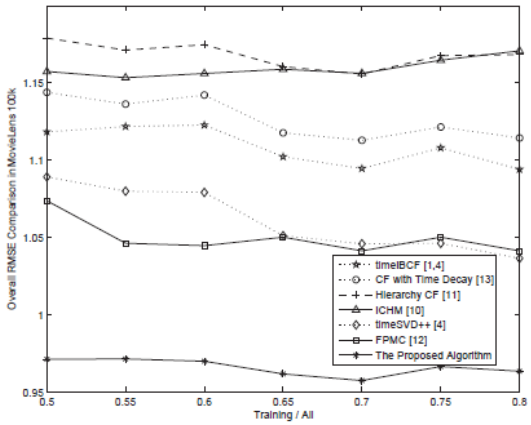
TimeSVD++ [4] is extended from SVD++ by accounting for temporal dynamics. TimeIBCF is extended from IBCF by accounting for temporal dynamics. Factorized Personalized Markov Chain (FPMC) [12] combines *matrix factorization* and *markov chain* together to handle both the sparsity and the dynamic nature of dynamic personalized recommendation. IBCF with time decay weights the similarities of CF by time decay functions to deal with the dynamic problems. Hierarchy CF is a hybrid of content-based and collaborative filtering methods using category information in items' content. ICHM introduces group similarity by clustering to modify similarities in IBCF, and K-Means is adopted for the clustering of item content.

3.4 Comparison results

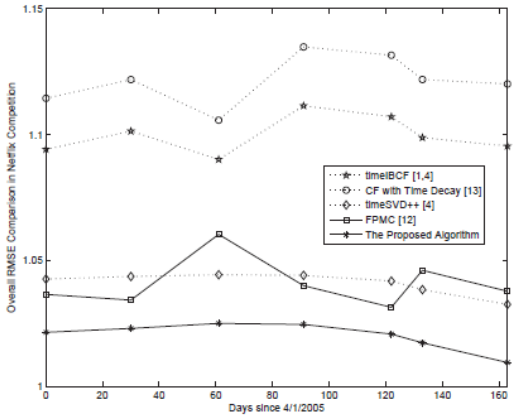
The performances of the algorithms on MovieLens 100k and Netflix Competition are reported is shown in fig 3. The proposed algorithm significantly outperforms the other algorithms in accuracy on MovieLens 100k and has comparable performance with timeSVD++ on Netflix Competition. On MovieLens 100k, the RMSE of the proposed algorithm is consistently the lowest, and the average RMSE of the proposed method is about 10%-15% lower than other algorithms. On Netflix Competition, the accuracy of the proposed algorithm is comparable with timeSVD++, due to the lack of user profiles and lack of information in item profiles. The

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

experimental results also show that the proposed algorithm and timeIBCF are robust with time evolving, indicating that the proportion of ratings by new users or of new items does not change a lot.



(a)



(b)

Fig. 3. Accuracy comparison (a)MovieLens 100k and (b)Netflix Competition.

Comparing Fig. 3(a) and Fig. 3(b), we can infer that (i) the accuracies of algorithms would be enhanced when data, especially recent data, gets dense, and
(ii) the utilization of profile content in the proposed algorithm is effective and helps improve the quality of recommendation. Compared to timeSVD++ and timeIBCF, in which only rating information is utilized, hybrid approaches make use of more information and may achieve better recommendation accuracies if the information mined is sufficient and the dynamic nature of data is well handled. The experimental results also show that users' preferences could be well described and learned by the MPD- based features.

As the efficiency of common IBCF is high in all recommendation algorithms [1], [3], we listed the computational time cost of the proposed algorithm and IBCF on the dataset of MovieLens 100k in Table 1 to illustrate its efficiency. We can see that the proposed approach has a comparable computational cost with IBCF. The size of testing set and the difference of time consumed between two algorithms are inversely proportional.

TABLE 1
Computational cost of the proposed algorithm and IBCF [1] on MovieLens 100k

#Test Set	Common IBCF [1]	Proposed Approach
50000	1.8 s	1.8 s
40000	1.5 s	1.5 s
30000	1.2 s	1.3 s
20000	0.8 s	1.0 s

We applied the proposed algorithm on the data in each single phase defined before, and the RMSEs are calculated separately according to the definition of users' multiple phases of interest. In Fig. 4, we presented the RMSEs of the proposed recom mendation algorithms using the data in different phases of interest at different training ratios. It is clear that the proposed algorithm is quite robust in the phases.

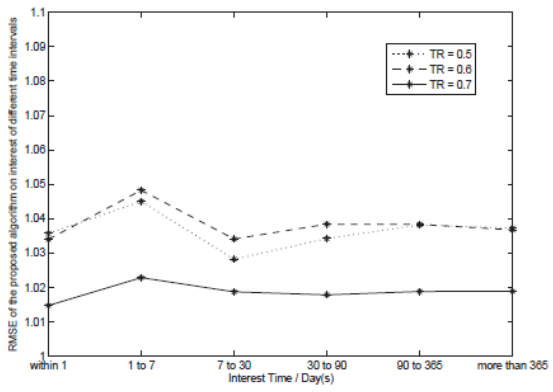


Fig. 4. Accuracy of the proposed algorithm on different phases of interest. TR means training ratio.

Comparing Fig. 4 with Fig. 3(a), we can see that the pro- posed algorithm has better performances than other algorithms even when it uses only the data in some single

phases. We can see that the accuracies become higher when we use all the data, which illustrated that mining and making use of more related data can provide more useful information.

4.CONCLUSION

In this paper, we did propose a dynamic personalized recommendation algorithm for sparse data, in which more rating data is utilized in one prediction by involving more neighboring ratings through each attribute in user and item profiles. A set of dynamic features are designed to describe the preference information based on TSA technique, and finally a recommendation is made by adaptively weighting the features with the help of information in multiple phases of interest. Experimental results indicate that the proposed algorithm is effective, and its computational cost is also reasonable one.

5. REFERENCE

[1] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: WWW, 2001, pp. 285–295.

[2] P. Brusilovsky, A. Kobsa, W. Nejdl (Eds.), The Adaptive Web, Methods and Strategies of Web Personalization, Lecture Notes in Computer.

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

[4] Y. Koren, Collaborative filtering with temporal dynamics, Communications of the ACM 53 (4) (2010) 89–97.

[5] L. Candillier, F. Meyer, M. Boull´e, Comparing state-of-the-art collaborative filtering systems, in: P. Perner (Ed.), MLDM, Vol. 4571 of Lecture Notes in Computer Science, Springer, 2007, pp. 548–562.

[6] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, H. Kriegel, Probabilistic memory-based collaborative filtering, IEEE Transactions on Knowledge and Data Engineering 16 (1) (2004) 56–69.

[7] F. Fouss, A. Pirotte, J. Renders, M. Saerens, Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation, IEEE TKDE 19 (3) (2007) 355–369.

[8] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.

[9] S. Boutemedjet, D. Ziou, Long-term relevance feedback and feature selection for adaptive content based image suggestion, Pattern Recognition 43 (12) (2010) 3925–3937.

[10] B. M. Kim, Q. Li, C. S. Park, S. G. Kim, J. Y. Kim, A new approach for combining content-based and collaborative filters, J. Intell. Inf. Syst. 27 (1) (2006) 79–91.