# Web Service Recommendation using Optimized Iterative Collaborative Filtering

Wang Binbin, Guo Jie
State Key Laboratory for Novel Software Technology
Nanjing University, 210046
Nanjing, China
{yew.wang, guojie}@mes.nju.edu.cn

Zhou Zuojian, Pan Jingui
State Key Laboratory for Novel Software Technology
Nanjing University, 210046
Nanjing, China
{zzj, panjg}@mes.nju.edu.cn

*Abstract*—**With the explosive growth of web services on the World Wide Web, service recommendation is becoming extremely important to both the service providers and the active users. In this paper, we propose a web service recommendation model which utilizes the prediction of Quality-of-Services (QoS) based on collaborative filtering with optimized iteration. The benefit of employing iteration in collaborative filtering is that the prediction accuracy of QoS values can be raised significantly via recursive refinement. Since such iteration scheme will hinder training performance, a novel optimization strategy is introduced based on the predicting tree. Finally, the optimized model is implemented and deployed to conduct the experiments on a real-world data set, which contains 1.5 million web services invocation information. The experimental results show that our model has achieved better prediction accuracy than other models with similar performance.**

***Keywords-Iteration, Predicting Tree, Collaborative Filtering, Web Service Recommendation, QoS***

## I. INTRODUCTION

In service-oriented architecture, web service technology is increasingly popular because of the support of interoperable machine-to-machine interaction over a network. However, with the diversification of what user needs, as well as the growth of web service number with the same type, it is difficult for users to find web service suited to them through a simple manual method. In addition, when a new web service is introduced, how the service providers looking for the potential users, and pushing new services to users, is becoming a very meaningful problem. If analyzing those problems carefully, we can find that the core problem is to obtain the prediction of web services which the user does not invoke before.

Typically, the nonfunctional characteristics of a web service are described by the Quality-of-Service (QoS), which is formed by a series of features (e.g., *response-time, throughput, failure rate,* etc.)[2]. This QoS value which is generally arranged into a user-item matrix can be obtained by statistically analyzing of the log from the web service providers. Due to the large number of services and the small quantity of services invoked by one user, the user-item matrix mentioned above is usually presented as a sparse one. Besides, the reliability and integrity of the collected data is limited because of the real-world environment, large quantity of users, complex network status and privacy protection[5]. As a result, how to predict the QoS value of unemployed items with the sparse matrix collected before is a challenge to us. To attack this critical challenge, the most useful technology is collaborative filtering, well known approaches in which include the user-based approach, the item-based approach, the cluster-based approach et al[5].

Unfortunately, neither the user-based model nor the item-based model fully utilizes the information of both similar users and similar items. One general property of these models is that the training matrix is utilized only once, but we have found that by iteratively refine the training matrix, the error of the QoS value prediction can be reduced significantly. Stemming from this observation, we propose an iteration-based collaborative filtering model to achieve better accuracy. However, such scheme has a major disadvantage that the overall performance is reduced due to the iteration. Therefore, in the second part of our paper, we present an optimizing scheme to decrease the runtime of training.

In brief, contributions of this paper include:

- To achieve better accuracy, a hybrid iteration-based model is proposed to make personalized QoS value prediction for active user.
- To promote the performance of iteration, an optimized predicting order induced by the PTree is designed to decrease the runtime.

The rest of this paper is organized as follow: Section 2 introduces the related work. Section 3 introduces the new model, and Section 4 describes the optimized strategy. Section 5 shows the experiments, and Section 6 concludes the paper.

## II. RELATED WORK

Traditionally, collaborative filtering can be divided into three major categories[2,8]: the model-based approaches, the memory-based approaches, and the hybrid approaches.

In the model-based approaches, usually a predefined model is trained before the prediction with the training data set, which trends to be expensive, as well as lose useful information for dimensionality reduction techniques. Well known examples of the model-based approach include the SVD model[5,17], the clustering model[11,14,16], the aspect model[4,7], the classification model[3,9,13], et al. However, those work mainly focus on recommending the movies instead of the web services.

Among the memory-based approaches[10,14,15], the user-based[6] and item-based[12] are the most frequently used models suffering from data sparsity and scalability. Similarity computation methods[2,8], used in those models are pearson correlation coefficient (PCC), adjusted cosine similarity (ACS), vector space similarity (VSS), et al. For example, the user-based collaborative filtering models using PCC (UPCC)[20], which makes the QoS value prediction according to the similar users' value, decreases the prediction accuracy when the data set is sparse. The same problems also exist in the item-based models (e.g., IPCC).

Recently, a kind of hybrid models, which is a joint consisting of the value predicted by single models is proposed[12,21]. *WSRec*[21] uses the confidence weight and an extra parameter to combine the prediction of UPCC and IPCC. Since limited prediction accuracy can those models achieve, method of iteration can be added to make better prediction. To the best of our knowledge, few works focus on the convergency of iteration and the optimized predicting order. Models proposed in [1,19], which just repeat basic cluster-based or neighbor-based model simply ignore the optimization and costs more time to carry on the iteration. In this paper, we propose an optimized iterative collaborative filtering model to extend the idea of hybrid and iteration, which could raise both the accuracy and performance significantly.

## III. Iteration-based Collaborative Filtering

In this section, we introduce an iteration-based hybrid collaborative filtering model (ICF), and describe its key components in detail.

### A. *Overview*

Definition of the iteration in our model is described as follows.

$$F(t)=\frac{\sum_{n=1}^{N}\left|q_{u,i}^{t}-q_{u,i}^{t-1}\right|}{N} \quad \text{(Eq. 1)}$$

The objective function we designed is the mean absolute error which is between missing value predicted during two successive iterations.

### B. *Key Components*

#### 1) *Similarity Computation*

This step is to compute the similarity between different users or items, so that we can get the similar neighbors and predict the missing value in the training matrix. A variety of similarity calculation methods existed include ACS, PCC, VSS, among which PCC outperforms others due to its easy implementation and high accuracy[14,20]. As a result, we employ the PCC to compute the similarity in our model.

In UPCC, the missing QoS value $q_{u,i}$ is predicted according to the items invoked by the similar users of user *u*. PCC is employed to calculate the similarity between user *u* and other user $u_s$ based on the items they commonly have. Therefore, the similarity calculation can be written as:

$$Sim(u,u_s)=\frac{\sum_{i\in I}\left(q_{u_s,i}-\overline{u_s}\right)\left(q_{u,i}-\overline{u}\right)}{\sqrt{\sum_{i\in I}\left(q_{u_s,i}-\overline{u_s}\right)^2}\sqrt{\sum_{i\in I}\left(q_{u,i}-\overline{u}\right)^2}} \quad \text{( Eq. 2)}$$

where $q_{u,i}$ is the vector of QoS value of item *i* adopted by user *u*, $\overline{u}$ is the average QoS value of items that user u adopted. Besides, $I=I_u\cap I_{u_s}$ is the set of items that both user *u* and $u_s$ have invoked. If $I=\varnothing$, which means that there is no element in *I*, the value of *Sim(u,u<sub>s</sub>)* will regard as *null*, not *0*, because we do not have any information to carry out the similarity computation. Based on the previous definition, the similarity of two users, i.e. $Sim(u,u_s)$, is in the interval of [-1,1], where larger PCC value trends to indicate more similarity between two users. Similar method can also be used in IPCC.

#### 2) *Similarity Selection*

In our model we have made some modifications to the original Top-K method to increase its robustness. For instance, if PCC value is negative, there is little similarity even no similarity in those two users or items. As a result, when entries in the user-item matrix have limited similar neighbors, the Top-K method still selects the dissimilar neighbors to predict the missing value, which will seriously reduce the accuracy of the prediction. To solve this problem, an extra parameter $\delta(0<\delta<1)$ is employed to exclude the dissimilar neighbors whose PCC value is non-positive. Similarity selection used in UPCC and IPCC is described by the following two equations.

$$S(u)=\left\{u_s\left|Sim(u,u_s)_{(k)}>\delta,1\le k\le K\right.\right\} \quad \text{( Eq. 3)}$$

*S(u)* used in UPCC contains at most K elements selected by Top-K whose PCC value must be bigger than $\delta$. Similar method can also be used in IPCC.

#### 3) *Basic Prediction*

Once the PCC value and the similar neighbors are available, a basic prediction is performed according to:

$$PU\left(q_{u,i}\right)=\overline{u}+R(u)\cdot\frac{\sum_{u_s\in s(u)}\frac{Sim(u,u_s)\left(q_{u_s,i}-\overline{u_s}\right)}{R(u_s)}}{\sum_{u_s\in s(u)}Sim(u,u_s)}$$

$$R(u)=\max(u)-\min(u)$$

$$\text{(Eq. 4)}$$

where $PU\left(q_{u,i}\right)$ used in UPCC is a vector of predicted QoS value of the entry $q_{u,i}$. $\overline{u}$ and $\overline{u}_s$ is the average QoS value of items that user *u* and $u_s$ adopted. In addition, *R(u)* and $R(u_s)$ which are the difference between maximum and minimum are employed to normalize the predicted value.

In order to avoid the cases that the predicted value is too large or too small, we have employed clamp operation for PU and PI, respectively, i.e.

$$PU^{'}\left(q_{u,i}\right)=\begin{cases}\min(u) & ifPU\left(q_{u,i}\right)<\min(u)\\ PU\left(q_{u,i}\right) & if\ \min(u)\le PU\left(q_{u,i}\right)\le\max(u)\\ \max(u) & ifPU\left(q_{u,i}\right)>\max(u)\end{cases} \quad \text{(Eq. 5)}$$

which could remove the wrong predicted value produced in extreme conditions. Similar method can also be used in IPCC.

#### 4) *Compositive Prediction*

The combination of user-based and item-based collaborative filtering model is proposed to fully utilize the information of similar users and similar items[12,21]. Since the prediction accuracy of UPCC and IPCC is different, two confidence weights are employed using the method below.

$$con(u)=\sum_{u_s\in s(u)}\frac{\left(Sim(u,u_s)\right)^2}{\sum_{u_s\in s(u)}Sim(u_s,u)} \quad \text{(Eq. 6)}$$

After calculating the confidence weight, we use an additional parameter $\lambda$ to indicate how much the final prediction relies on the UPCC and IPCC[21]. Consequently, the prediction equation for hybrid model is expressed as:

$$P\left(q_{u,i}\right)=w(u)\times PU^{'}\left(q_{u,i}\right)+w(i)\times PI^{'}\left(q_{u,i}\right) \quad \text{(Eq. 7)}$$

where Eq.8 and Eq.9 represent model weights of UPCC and IPCC, respectively.

$$w(u)=\frac{con(u)\times\lambda}{con(u)\times\lambda+con(i)\times(1-\lambda)}$$

(Eq. 8)

$$w(i)=\frac{con(i)\times(1-\lambda)}{con(i)\times\lambda+con(i)\times(1-\lambda)}$$

(Eq. 9)

## IV. PTREE-BASED OPTIMIZATION

In this section, we present an optimized model (OICF) based on a specially designed data structure to promote the performance of iteration.

### A. PTree

As we can see, it is unnecessary to predict all the missing entries in the training matrix during the whole iteration. For example, the missing entries whose similar neighbors needn't predicting can just be predicted only in the first iteration. Though the predicted value may get better in the next iteration, the prediction accuracy gains quite less than the missing entries whose similar neighbors also need predicting. In addition, for the entries whose similar neighbors need predicting, first we can predict the similar neighbors, then make prediction for missing value using the predicted similar neighbors.

Based on this analysis, we design a data structure named PTree to find the predicting order of the missing entries. Firstly, the missing matrix which records the index of the missing entries is needed. Secondly, an adjacency matrix which stores the missing entries and their similar neighbors can be acquired in the step of similarity selection. Finally, with the missing matrix(*mis_mat*) and the adjacency matrix(*adj_mat*), the predicting tree which is stored in the depth matrix(*dep_mat*) can be constructed using the Algorithm 1.

```
Algorithm 1 Create PTree
Procedure PTree(depth, K)
  For $mis\_mat[i] \in mis\_mat$ do
      If $mis\_mat[i] \notin dep\_mat$ then //Create root
          dep_mat[k][1] = I //index of the missing item
          dep_mat[k][2] = depth //depth of each node
          k = k + 1
      End if
      depth = depth - 1
      For j = 1 to K do //construct child nodes
          If $adj\_mat[i][j] \notin dep\_mat$ then
          dep_mat[k][1] = adj_mat[i][j]
          dep_mat[k][2] = depth
           k = k + 1
          End if
      End for
      If depth > 0 then //constructing for child node
       For j = 1 to K do
         id = adj_mat[i][j]
         If $mis\_mat[id] \in mis\_mat$ then
          PTree(depth, K)
         End if
       End for
      End if
  End for
End Procedure
```

Generally, the parameter *depth* equals to *T* which is the maximum number of iteration. The computational complexity of constructing the PTree is O(*NlogN*) where *N* the total number of missing entries.

### B. Optimization Strategy

According to the description in Algorithm 1, each node in the PTree contains two fields: index of the missing entry and depth of the node. The predicting order can be acquired by the level traversal of PTree according to the sorted depth. Based on the predicting order induced by PTree, we design an optimization strategy to promote the performance of the brute force iterative model.

- In the $t^{th}$ iteration, we just predict the missing entries whose depths are not less than *t*

For example, *T=4*, *depth=4*, in the first iteration, all the missing entries are predicted. In the second iteration, only the entries whose *depth* is larger than 2 are predicted again. With this strategy, we can reduce the runtime of the offline part rapidly.

## V. EXPERIMENT

### A. Experimental Setup

In our experiments, a series of programs implemented with *Matlab* are running on a PC with Intel Xeon X7460 CPU and 64G RAM. Additionally, the data set we used in this paper is from *WS-DREAM*[18,22].19 Two features of the QoS value collected in the data set are response-time and throughput, both of which contain a user-item matrix(339*5825). To evaluate the prediction accuracy and performance of different models, 10 user-item matrices(150*200) without missing value are selected from the original data set to serve as the real world QoS data.

Tab. 1 Parameter Description

| Parameter | Descriptions |
|---|---|
| *Train_user* | the number of the training users |
| *Given_number* | the number of QoS of an active user |
| *Density* | density of the training matrix |
| *K* | the number of similar neighbors |
| $\delta$ | the threshold of PCC value |
| *T* | the maximum number of iteration |
| $\lambda$ | the model weight of hybrid model |

150 service users are divided into training users and active users according to *train_user*. We randomly remove the entries in the training matrix according to *density* to generate sparse matrix with various sparsity. *given_number*

is provided to select the entries of active user as the missing value. Tab.1 describes all the parameters in the experiments.

### B. *Accuracy*

In this paper, mean absolute error(MAE) is employed to measure the prediction accuracy of our model in comparison with other collaborative filtering models. The definition of MAE and RMSE is shown by the following equations.

$$MAE(t)=\frac{\sum_{n=1}^{N}\left|q_{u,i}^{*}-q_{u,i}^{t}\right|}{N} \quad \text{(Eq. 10)}$$

In these equations, $q_{u,i}^{*}$ denotes the real QoS value of web service $i$ invoked by user $u$, while $q_{u,i}^{t}$ denotes the QoS value predicted by different models. $N$ is the number of missing entries.

To find the convergency of iteration, we evaluate MAE on throughput with increasing number of iterations (up to 40), showing in Fig.2.

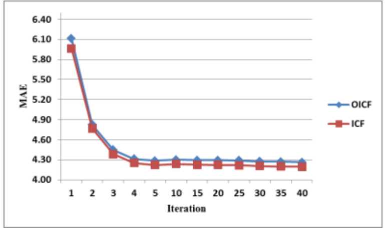

Fig. 1 Convergency of Iteration

Two curves in Fig.1 illustrate that MAE of ICF and OCIF diminishes rapidly with the increasing of iteration number. Note that after roughly 4 iterations, both curves tend to be stable, which means the MAE convergency. Therefore, four iterations seems sufficient for most cases, which is used in the succeeding experiments.

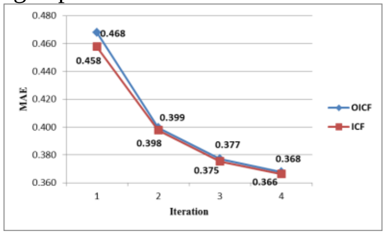

Fig. 2 MAE on response-time

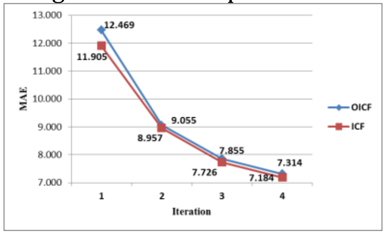

Fig. 3 MAE on throughput

In Fig.2 and Fig.3, we compare predicting errors of ICF and OCIF on *response-time* and *throughput*, with 4 iterations. As we expected, the MAE reduce rapidly with the increasing of iteration, while the difference between ICF and OICF is small. This means the MAE of OICF is acceptable for both response-time and throughput. Evidently, using iteration greatly promots the accuracy of QoS value prediction.

### C. *Performance*

As is illustrated in Fig.4 and Fig.5, though the runtime of both models are higher than the non-iteration methods, the optimization strategy can improve the performance to some extent. Even for 4 times, the running time of OICF is still not very high, which is acceptable in most recommender systems.

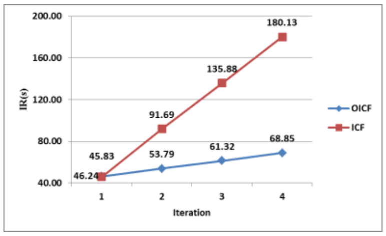

Fig. 4 train_user=100

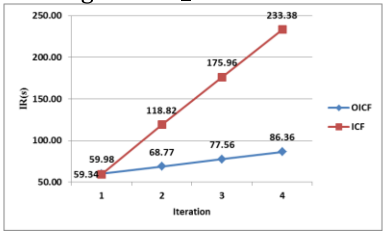

Fig. 5 train_user=120

## VI. Conclusion

In this paper, we propose an iteration-based collaborative filtering method used in the web service recommender systems. One key point of our methodology is the choice of calculating the similarity functions more than once in order to raise its accuracy via recursive refinement. Besides, an optimization strategy is designed to promote the performance of iteration. Experimental results show that this iterative process and its corresponding optimization strategy can achieve better prediction accuracy with relatively high performance.

Further research based on the work in this paper includes the implementation of the web services recommender system. Additionally, the optimization strategy according to the predicting order could be adopted in other collaborative filtering models to reduce the cost of runtime. Furthermore, the model we proposed could be used to make other recommendation, such as the medical services, the e-commerce, et al.

## Acknowledgment

This work is supported by Natural Science Foundation of Jiangsu Province of China (BK2010373) and Postgraduate Cultivation and Innovation Foundation of Jiangsu Province (CXZZ11_0045).

## References

[1] G. Adomavicius and A. Tuzhilin, “Toward the Next Gen- eration of Recommender Systems: A Survey of the State- of-the-Art and

Possible Extension", IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734-749, 2005.

[2] X. Su and T. M. Khoshgoftaar, "A Survey of Colla- borative Filtering Techniques", Advances in Artificial Intelligence, vol. 2009, no. 4, pp. 1-19, 2009.

[3] S. Zhang, W. Wang, J. Ford and F. Makedon, "Using Singular Value Decomposition Approximation for Colla- borative Filtering", the 7th IEEE international Confer- ence on E-Commerce Technology (CEC'05), pp. 257-264, 2005.

[4] H. Polat and W. Du, "SVD-based Collaborative Filtering with Privacy", Proceedings of the 2005 ACM symposium on Applied (SAC'05), pp. 791-795, 2005.

[5] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu and Z. Chen, "Scalable Collaborative Filtering Using Cluster-based Smoothing", Proceedings of the 28th ann- ual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'05), pp. 114-121, 2005.

[6] Q. Yu and M. Rage, "On Service Community Learning: A Co-cluster Approach", IEEE International Conference on Web Services, pp. 283-290, 2010.

[7] F. Wagner, F. Ishikawa and S. Honiden, "QoS-aware Au- tomatic Service Composition by Applying Functional Cl- ustering", IEEE International Conference on Web Serv- ices, pp. 89-96, 2011.

[8] T. Hofmann, "Latent semantic models for collaborative filtering", ACM Transactions on Information Systems, vol. 2022, no. 1, pp. 89-115, 2004.

[9] L. Si and R. Jin, "Flexible Mixture Model for Colla- borative Filtering", Proceedings of the Twentieth Inter- national Conference on Machine Learning (ICML'03), pp. 259-266, 2003.

[10] G. Wu, J. Wei, X. Qiao and L. Li, "A Bayesian network based Qos assessment model for web services", IEEE International Conference On Services Computing, pp. 498-505, 2007.

[11] X. Chen, X. Liu, Z. Huang and H. Sun, "RegionKNN: A Scalable Hybrid Collaborative Filtering Algorithm for Personalized Web Service Recommendation," IEEE Inte- rnational Conference on Web Services, pp. 9-16, 2010.

[12] X. Su, T. M. Khoshgoftaar and R. Greiner, "Imputation- Boosted Collaborative Filtering Using Ma- chine Learing Classifiers", Proceedings of the 2008 ACM symposium on Applied computing(SAC'08), pp. 949-950, 2008.

[13] X. Su, T. M. Khoshgoftaar and R. Greiner, "Imputed Neighborhood Based Collaborative Filtering", IEEE/WI- C/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'08), pp. 633-639, 2008.

[14] K. Yu, A. Schwaighofer, V. Tresp, X. Xu and H.-P. Kriegel, "Probabilistic Memory-Based Collaborative Filt- ering", IEEE Transactions on Knowledge and Data Engi- neering, vol. 16, no. 1, pp. 56-69, 2004.

[15] L. Shao, J. Zhang, Y. Wei, J. Zhao, Bingxie and H. Mei, "Personalized QoS Prediction for Web Services via Collaborative Filtering", IEEE International Conference on Web Services, pp. 439-446, 2007.

[16] J. Wang, A. P. Vries, and M. J.T.Reinders, "Unifying User-based and Item-based Collaborative Filtering App- roaches by Similarity Fusion", Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'06), pp. 501-508, 2006.

[17] Z. Zheng and M. R. Lyu, "Collaborative Reliability Prediction of Service-Oriented Systems", IEEE Inter- national Conference on Software Engineering, pp. 35-44, 2010.

[18] Z. Zheng, H. Ma, M. R. Lyu and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filter- ing", IEEE Transactions on Services Computing, vol. 4, no. 2, pp. 140-152, 2011.

[19] A. Abdelwahab, H. Sekiya, I. Matsuba, Y. Horiuchi and S. Kuroiwa, "Collaborative Filtering Based on an Iterative Prediction Method to Alleviate the Sparsity Pro- blem," Proceedings of the 11th International Conference on Information Integration and Web-based Applications and Services (iiWAS'09), pp. 375-379, 2009.

[20] Z. Zhang, P. Cuff and S. Kulkarni, "Iterative collabora- tive filtering for recommender systems with sparse data", IEEE International Workshop on Machine Learning for Signal Processing, pp. 1-6, 2012.

[21] Y. Zhang, Z. Zheng and M. R. Lyu, "Exploring Latent Features for Memory-Based QoS Prediction in Cloud Computing," Proceedings of the 30th IEEE Symposium on Reliable Distributed Systems(SRDS'11), pp. 4-7, 2011.

[22] Z. Zheng, Y. Zhang and M. R. Lyu, "Distributed QoS Evaluation for Real-World Web Services", Proceedings of the 8th International Conference on Web Services (ICWS'10), pp. 83-90, 2010.