# A Combination Approach to QoS Prediction of Web Services

Dongjin Yu, Mengmeng Wu, and Yuyu Yin

College of Computer, Hangzhou Dianzi University
Hangzhou China
{yudj,yinyuyu}@hdu.edu.cn, hdu409meng@126.com

**Abstract.** With a growing number of alternative Web services that provide the same functionality but differ in quality properties, the problem of selecting the best performing candidate service is becoming more and more important. However, users can hardly have invoked all services, meaning that the QoS values of some services are missing. In this paper, we propose a combination approach used to predict such missing QoS values. It employs an adjusted user-based algorithm using Pearson Correlation Coefficient to predict the QoS values of ordinary services. For services with constantly poor performance, however, it employs the average QoS values observed by different service users instead. An extensive performance study based on a real public dataset is finally reported to verify its effectiveness.

**Keywords:** Quality of Services, Prediction, Service Selection, Web Services.

## 1 Introduction

With a growing number of alternative Web services that provide the same functionality but differ in quality properties, the problem of selecting the best performing candidate service is becoming more and more important. Due to some inevitable reasons, e.g., location and network environment, the QoS of the same service to different users may be different. For example, the response time for an USA user $u_a$ to invoke web service located in Spain is 5626ms, while that for a Japan user $u_b$ to invoke the same one is 687ms. A user can hardly have invoked all services, meaning that the QoS values of some services that the user has not invoked are missing. Hence, some effective approaches are urgently needed to provide accurate prediction of the QoS values of different Web services for each user without requiring real Web service invocations.

In recent years, researchers have proposed a number of QoS prediction approaches [1-3]. Inspired by the application of Collaborative Filtering (CF) [4] in product recommendation, CF has been extensively employed to predict QoS values using Web service QoS evaluations from different users. It is often classified as memory-based or model-based. For the memory-based ones, all training data are stored in memory. In the prediction phase, similar objects (users or items) are sorted based on their similarities with the active object. Pearson Correlation Coefficient (PCC) is a widely used method to compute the similarities between objects. Based on the data from similar

A. Ghose et al. (Eds.): ICSOC 2012, LNCS 7759, pp. 99–106, 2013.

users or items, a prediction result can be generated. The most analyzed examples of memory-based methods include user-based methods [4-6], item-based methods [7, 8] and fusion methods [9]. For the model-based ones, on the other hand, training data are used to generate a predicting model that is able to predict the missing data. The most analyzed model-based examples include decision tree [4], aspect models [10] and latent semantic models [11, 12].

Many works have been done to predict the missing QoS values. Shao et al. propose a user-based CF algorithm to make similarity mining and predict the QoS of Web services from consumers' experiences [1]. Zheng et al. present a hybrid approach which combines user-based and item-based approach together to predict the QoS of Web services [2]. Chen et al. discover the great influence of a user's location to the accuracy of prediction and propose a region-based hybrid CF algorithm to predict the QoS of services [3]. However, the prediction accuracies of those methods are far from satisfactory.

To improve the prediction accuracy, we propose a combination approach. The basic idea is that we use different methods to predict the QoS values of different services. Considering unstable network environments, we first classify the services into the poor ones with constantly poor performance, and ordinary ones. The final QoS is estimated by different methods based on the category which the target service belongs to. For ordinary target services, it employs an adjusted user-based algorithm using Pearson Correlation Coefficient, or adjusted UPCC, to predict the QoS values. For services with constantly poor performance, however, it employs IMEAN, i.e., uses the average QoS values observed by different service users, instead.

The contributions of this work are as follows. (1) We propose a combination approach to predict missing QoS values. (2) We evaluate the proposed approach experimentally by employing a real-world Web service QoS dataset.

The rest of this paper is organized as follows: Section 2 presents our QoS value prediction approach in detail. Section 3 describes our experiments. Section 4 concludes the paper.

## 2 Prediction Approach

In real world, QoS of service not only relies on the executing environment of services, but also heavily depends on unstable network environment. As a result, some services with constantly poor performance (named as poor services for ease of presentation) may be published on the Internet. Our prediction approach is based on the following assumption: poor services should low the prediction accuracy. The approach has three major steps.

1) Find the top *N* poor services using IMEAN;

2) If the prediction service is one of the top *N* poor services, IMEAN is used to predict its QoS performance;

3) If the prediction service is not one of the top *N* poor services, adjusted UPCC is used to predict its QoS performance.

### 2.1 Classification of Services

To find poor services, we first list the services in descending order from their IMEAN values. Then we classify the top *N* services and the others into poor services category and ordinary services category respectively. The formula used to calculate IMEAN values is as follows:

$$P_{u,s} = \frac{\sum_{u_1 \in U_s} r_{u_1,s}}{|U_s|}, \tag{1}$$

where $r_{u_1,s}$ is the vector of QoS values of service s invoked by user $u_1$, $U_s$ is the set of users have invoked s, and $|U_s|$ is number of users have invoked *s*.

Obviously, the number *N* of poor services is the key for our approach. Through some tests, we find the most accurate prediction results using our approach could be acquired when *N* equals to 5. These tests will be shown in Section 3.

### 2.2 Prediction of QoS Values

Because we use different predicting methods for services form different categories, we must judge which category the target service belongs to before the calculation of missing value. If the target service belongs to the poor services category, IMEAN is used. Otherwise, adjusted UPCC is used. Due to IMEAN having been introduced in Section 2.1, we present adjusted UPCC method here.

PCC has been introduced in a number of recommender systems for similarity computation, since it can be easily implemented and achieves high accuracy. The similarity between two users $u_1$ and $u_2$ based on the PCC is computed using the following equation:

$$\phi_{u_1,u_2} = \frac{\sum_{s \in S}(r_{u_1,s} - \bar{r}_{u_1})((r_{u_2,s} - \bar{r}_{u_2})}{\sqrt{\sum_{s \in S}(r_{u_1,s} - \bar{r}_{u_1})^2}\sqrt{\sum_{s \in S}(r_{u_2,s} - \bar{r}_{u_2})^2}}, \tag{2}$$

where $S = S_{u_1} \cap S_{u_2}$ is the subset of Web service items which user $u_1$ and user $u_2$ invoked together, and $\bar{r}_{u_1}$ represents the vector of average QoS values of the user $u_1$.

Although PCC can provide accurate similarity computation, it will overestimate the similarities of service users who are actually not similar but happen to have similar QoS experience on a few co-invoked Web services. To address this problem, we employ an adjusted formula to reduce the influence of a small number of similar co-invoked items. An adjusted PCC formula for the similarity computation between different service users is defined as:

$$\phi'_{u_1,u_2} = \frac{2 \times |S|}{|S_{u_1}| + |S_{u_2}|} \phi_{u_1,u_2}, \tag{3}$$

After calculating the similarities between different users, a set of similar neighbors can be identified. The selection of similar neighbors is an important step for making accurate missing value prediction, since dissimilar neighbors will decrease the prediction accuracy. Traditional Top-*K* algorithms rank the neighbors based on their PCC similarities and select the top *K* most similar neighbors. In practice, some entries in the user-item matrix have limited similar neighbors or even do not have any neighbors. Traditional Top-*K* algorithms ignore this problem and still include dissimilar neighbors to predict the missing value, which will greatly reduce the prediction accuracy. To solve this problem, we propose an enhanced Top-*K* algorithm, where neighbors with PCC similarities smaller or equal to $\xi$ will be excluded. A set of target user *u*' similar neighbors can be found by the following equation:

$$\mathrm{L(u)} = \{\mathrm{u}_a | \mathrm{u}_a \epsilon \mathrm{T(u)}, \phi'_{u_1,u_2} > \xi, \mathrm{u}_a \neq \mathrm{u}\},, \tag{4}$$

where $\mathrm{T(u)}$ is a set of top *K* similar users to the user *u*, and $\xi$ a similarity threshold.

The final prediction method employs the data of similar users to predict the missing value of target service *s* to target user *u* as follows:

$$P_{u,s} = \bar{u} + \frac{\sum_{u_1 \in L_u} \phi'_{u,u_1} \times (r_{u_1,s} - \bar{u}_1)}{\sum_{u_1 \in L_u} \phi'_{u,u_1}}, \tag{5}$$

where $L_u$ are a set of user *u's* similar users, and $\bar{u}$ is a vector of average QoS values of different Web services observed by the active user *u*.

# 3 Experiments

## 3.1 Experimental Setup

We have conducted our experiments using a public real-world Web service QoS dataset, which is collected by Zibin Zheng et.al [2]. It contains the records of 1,974,675 Web service invocations executed by 339 distributed service users on 5825 Web services. The record of each invocation contains 2 parameters: Response Time and Throughput. More details about this dataset can be found in [13]. In this paper, we randomly extract 150 users, 100 Web services and use the invocation records between them as the experimental data.

Our experiments are implemented with Matlab 7.0 and MySQL 5.0. They are conducted on a Dell Inspire R13 machine with a 2.27 GHz Intel Core I5 CPU and 2GB RAM, running Windows 7 OS.

## 3.2 Evaluation Metric

In our experiments, *NMAE* is used to evaluate the accuracy of prediction. Mean Absolute Error (*MAE*) is as follows:

$$MAE = \frac{\sum_{U,S} | r_{u,s} - \hat{r}_{u,s} |}{N} \tag{6}$$

where $r_{u,s}$ represents the predicted QoS value of service *s* observed by user *u*, $\hat{r}_{u,s}$ stands for the expected or real QoS value and *N* is the total number of predictions. As we know, services QoS value range may differ tremendously. As an adjustment, NMAE normalizes the differences range of *MAE* by computing:

$$NMAE = \frac{MAE}{\sum_{U,S} \frac{r_{u,s}}{N}} \tag{7}$$

Here, the smaller NMAE means the more accurate QoS prediction.

### 3.3 Performance Comparison

We compare the proposed approach with five typical prediction methods: User Mean(UMEAN), Item Mean(IMEAN), User-based algorithm using PCC (UPCC), Item-based algorithm using PCC (IPCC), and WSRec. UMEAN uses the average QoS of the service user on other Web services to predict the QoS of other Web services, while IMEAN uses the average QoS of the Web service observed by other service users to predict the QoS for the current user. UPCC uses similar users for QoS prediction, while IPCC uses similar services for QoS prediction. WSRec uses a linear combination of the UPCC and IPCC results together. To the best of our knowledge, WSRec is the best one for QoS prediction currently. In the experiment, we set $K$=10 and $\xi = 0.3$. Each experiment is run for 50 times and the average NMAE values are reported.

Table 1 shows the NMAE results of different prediction methods on Response Time and Throughput using 10 and 20 percent densities of the training matrix respectively. For the users in testing matrix, we vary the number of invoked services (given number) as 10 and 20 by randomly removing entries (named as *g*10, *g*20, and *g*30, in Table 1). In addition, the number of training users is varied as 100 and 140.

From Table 1, under all experimental settings, our method obtains smaller NMAE values in all cases, which demonstrates better prediction accuracy. The prediction results when Training users=140 is more accurate than the results when Training users=100. Meanwhile, the increase of the density of a training matrix enhances the prediction accuracy. The reason is that higher density means more training data for the prediction. In addition, the increase of the number of invoked services (*g*10, *g*20, and *g*30) also improves the prediction accuracy. It indicates that the prediction accuracy can be enhanced by providing more QoS values.

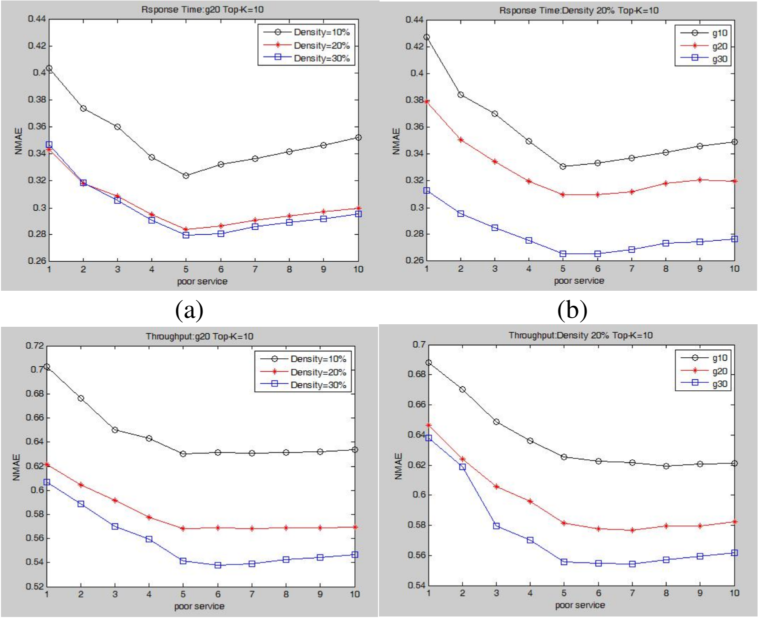
### 3.4 Evaluation of the Top *N*

In our paper, parameter *N* denotes the number of the poorest services. To study the impact of the parameter *N* on the efficiency of our prediction approach, we vary the value of *N* from 0 to 10 with a step value of 1. Figure 1 (a) and (c) show the accuracy of prediction, with given number of 20 (g20), with 10, 20, and 30 percent density of training matrix of Response Time and Throughput, respectively. Figure 1 (b) and (d) show the accuracy of 20 percent density with given number 10, 20, and 30 of Response Time and Throughput, respectively.

**Table 1.** NMAE Comparison with Different Prediction Approaches

| Density | Methods | T=100 | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | | Response Time | | | Throughput | | |
| | | g10 | g20 | g30 | g10 | g20 | g30 |
| 10% | UMEAN | 0.9809 | 0.972 | 0.9448 | 1.1335 | 1.083 | 1.0812 |
| | IMEAN | 0.5368 | 0.5124 | 0.5154 | 0.6182 | 0.614 | 0.6258 |
| | UPCC | 0.5443 | 0.4587 | 0.4272 | 0.8277 | 0.7168 | 0.7018 |
| | IPCC | 0.437 | 0.4127 | 0.3981 | 0.7466 | 0.6906 | 0.6886 |
| | WSRec | 0.4362 | 0.4107 | 0.3958 | 0.7463 | 0.6896 | 0.6871 |
| | Our Method | 0.3972 | 0.358 | 0.3407 | 0.6913 | 0.6456 | 0.6322 |
| 20% | UMEAN | 0.9861 | 0.9771 | 0.9827 | 1.1152 | 1.0676 | 1.055 |
| | IMEAN | 0.5173 | 0.5047 | 0.4911 | 0.5916 | 0.6204 | 0.6031 |
| | UPCC | 0.5012 | 0.4358 | 0.4043 | 0.7521 | 0.6717 | 0.626 |
| | IPCC | 0.3714 | 0.3585 | 0.3409 | 0.6562 | 0.6347 | 0.6154 |
| | WSRec | 0.3702 | 0.3568 | 0.3394 | 0.6547 | 0.6317 | 0.6115 |
| | Our Method | 0.3596 | 0.3279 | 0.306 | 0.6378 | 0.6055 | 0.5769 |
| Density | Methods | T=140 | | | | | |
| | | Response Time | | | Throughput | | |
| | | g10 | g20 | g30 | g10 | g20 | g30 |
| 10% | UMEAN | 1.0597 | 0.94 | 0.9059 | 1.1128 | 1.0793 | 1.0159 |
| | IMEAN | 0.4792 | 0.493 | 0.4668 | 0.6151 | 0.6106 | 0.6007 |
| | UPCC | 0.5409 | 0.4347 | 0.3718 | 0.7981 | 0.735 | 0.6608 |
| | IPCC | 0.397 | 0.3744 | 0.3555 | 0.7075 | 0.6985 | 0.6445 |
| | WSRec | 0.395 | 0.3727 | 0.3534 | 0.7061 | 0.6963 | 0.6425 |
| | Our Method | 0.373 | 0.3296 | 0.2875 | 0.6722 | 0.6321 | 0.59 |
| 20% | UMEAN | 1.0208 | 0.9285 | 0.8551 | 1.189 | 1.0811 | 1.026 |
| | IMEAN | 0.4737 | 0.4571 | 0.4759 | 0.5914 | 0.568 | 0.5601 |
| | UPCC | 0.4886 | 0.3867 | 0.3476 | 0.8243 | 0.6503 | 0.5849 |
| | IPCC | 0.3386 | 0.3291 | 0.3059 | 0.7062 | 0.604 | 0.5725 |
| | WSRec | 0.3376 | 0.3272 | 0.3039 | 0.7052 | 0.6006 | 0.5689 |
| | Our Method | 0.3303 | 0.3097 | 0.2877 | 0.6605 | 0.5688 | 0.553 |

Observing from Figure1, we can draw the conclusion that the value of *N* impacts the predicted results significantly. In addition, when *N*=5, the most accurate results are provided for both Response Time and Throughput.



(a) (b)

(c) (d)

**Fig. 1.** Impact of the top *N*

# 4 Conclusion

In this paper, we present a combination approach to predict the missing QoS values. Different from the previous methods, considering unstable network environment, we classify the services into poor services category and ordinary service category, and use different predicting methods based on the category which the target service belongs to. The experiments based on a public dataset prove that our prediction approach outperforms the existing methods.

In the future, we will explore the service selection framework. Furthermore, we will also collect more QoS data of Web services to improve the scale of our experiments.

**Acknowledgments.** This research was partially supported by the National Natural Science Foundation of China with grant number of 61100043 and the Natural Science Foundation of Zhejiang with grand number of LY12F02003. The authors would also like to thank anonymous reviewers who gave valuable suggestion to improve the quality of the paper.

# References

1. Shao, L., Zhang, J., Wei, Y., Zhao, J., Xie, B., Mei, H.: Personalized QoS prediction for web services via collaborative filtering. In: IEEE International Conference on Web Services, Salt Lake City, USA, pp. 439–446 (2007)

2. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Qos-aware web service recommendation by collaborative filtering. IEEE Transactions on Service Computing 4(2), 140–152 (2011)
3. Chen, X., Zheng, Z., Liu, X., Huang, Z., Sun, H.: Personalized QoS-aware web service recommendation and visualization. IEEE Transactions on Services Computing PP(99) (2011)
4. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: 14th Conference on Uncertainty in Artificial Intelligence, Madison, USA, pp. 43–52 (1998)
5. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: ACM SIGIR Conference, Berkeley, USA, pp. 230–237 (1999)
6. Jin, R., Chai, J.Y., Si, L.: An automatic weighting scheme for collaborative filtering. In: ACM SIGIR Conference, Sheffield, UK, pp. 337–344 (2004)
7. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Transaction on Information System 22(1), 143–177 (2004)
8. Sarwar, B., Karypic, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: International World Wide Web Conference, Hong Kong, China, pp. 285–295 (2001)
9. Wang, J., Vries, A.P., Reinders, M.J.: Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In: ACM SIGIR Conference, Seattle, USA, pp. 501–508 (2006)
10. Si, L., Jin, R.: Flexible mixture model for collaborative filtering. In: International Conference on Machine Learning, Washington, USA (2003)
11. Hofmann, T.: Collaborative filtering via Gaussian probabilistic latent semantic analysis. In: ACM SIGIR Conference, Toronto, Canada, pp. 259–266 (2003)
12. Hofmann, T.: Latent semantic models for collaborative filtering. ACM Transaction on Information System 22(1), 89–115 (2004)
13. Zheng, Z., Zhang, Y., Lyu, M.R.: Distributed QoS evaluation for real-world web services. In: International Conference on Web Services, Miami, USA, pp. 83–90 (2010)