An expanded Matrix Factorization model for real-time Web service QoS prediction

Jinsheng Hao[1], **Guoping Su**[1], **Xiaofeng Han**[1], **and Wei Nie**[2*]
[1] Information Science and Engineering Department, Xinjiang University
666 Shengli Road, Urumqi, Xinjiang,China
[e-mail: 147807411@qq.com]
[2] School of Electronics and Information Engineering, Shenzhen University
3688 Nanhai Avenue,Nanshan District,Shenzhen,China
[e-mail: wei.nie@szu.edu.cn]
[*]Corresponding author: Wei Nie

Received April 17, 2021; revised July 21, 2021; accepted Septebmer 6, 2021; published November 30, 2021

Abstract

Real-time prediction of Web service of quality (QoS) provides more convenience for web services in cloud environment, but real-time QoS prediction faces severe challenges, especially under the cold-start situation. Existing literatures of real-time QoS predicting ignore that the QoS of a user/service is related to the QoS of other users/services. For example, users/services belonging to the same group of category will have similar QoS values. All of the methods ignore the group relationship because of the complexity of the model. Based on this, we propose a real-time Matrix Factorization based Clustering model (MFC), which uses category information as a new regularization term of the loss function. Specifically, in order to meet the real-time characteristic of the real-time prediction model, and to minimize the complexity of the model, we first map the QoS values of a large number of users/services to a lower-dimensional space by the PCA method, and then use the K-means algorithm calculates user/service category information, and use the average result to obtain a stable final clustering result. Extensive experiments on real-word datasets demonstrate that MFC outperforms other state-of-the-art prediction algorithms.

Keywords: Real-time QoS prediction, matrix factorization, clustering, K-means.

1. Introduction

As the number of services deployed in the cloud, more and more web services are available for users to choose. However, the quality of web services (QoS) with similar functions varies greatly. Therefore, users pay more attention to non-functional attributes (QoS) when they choose web services with corresponding functions [1]. In the ever-changing cloud environment, the QoS value cannot be guaranteed at all times. When the QoS value the working component service changes unexpectedly, it is necessary to find an appropriate candidate service (candidate service with the same function) for the current working component service, and the QoS value is a key indicator for judging whether it is appropriate. Therefore, predicting the QoS value of a candidate service has become an urgent problem to be solved.

QoS prediction models are developing rapidly towards selecting candidate services [1-6]. For example, [1,3] use a model-based Collaborative Filtering (CF) method to predict the QoS values of atomic service. In particular, [1] proposes a real-time prediction model that takes into account the dynamic factors of users and services joining or leaving, but it does not work well under cold-start. [3,6] propose a location-based model that uses the location category information of users and services to alleviate the cold-start problem, but it cannot work in real-time conditions. However, the service call by the user is a continuous operation, and the candidate service is also called through the continuous method. Only the real-time predicting model of the candidate service can be applied in practice. In fact, the most accurate method is the one which invokes web services one by one to test their QoS value in detail [2], but this method is impractical.

In summary, the above methods do not fully consider the real-time and accuracy of the QoS prediction model. However, the web services deployed in the cloud environment require high QoS values, and the QoS prediction methods of candidate services need to meet the following condition:

1.Real-time: The QoS of web service that exist the ever-changing and developmental cloud environment will change in different time periods, because of the impact of dynamic network condition and different server workloads.

2.Accuracy: QoS prediction methods must be accurate in cold-start environment, otherwise unreliable QoS prediction may lead to wrong service when users invoke service or component services call next service.

3.Extensibility: In a dynamic cloud environment, new services with different QoS may become available, and existing candidate services may be stopped by service providers too. With user frequently leaving or joining, the QoS predicting methods needs to have great performance to adapt new users and new services, and can perform prediction steps steadily.

Based on these three points, we propose an extended matrix factorization model (MFC) for real-time web service QoS prediction, as shown in **Fig. 1**. The essential difference from the task studied in [1,3] is that our work focus on predicting the cold-start QoS value of each successive candidate service. Our method is inspired by CF technology [6] and K-means algorithm [7]. Specifically, in terms of model accuracy, in order to alleviate the cold-start problem, we use user/service category group information. Because users/services belonging to the same category group will have similar QoS values, the QoS flow of the web service is first clustered from the user and service levels to obtain user group category and service category group information. In terms of the real-time performance of the model, we first perform dimensionality reduction processing (such as PCA method) on users and services before clustering them to ensure the efficiency of clustering and reduce the complexity of the model. In terms of the scalability of the model, based on the traditional Matrix Factorization model, we first use data conversion technology, online learning technology and adaptive weight technology to expand it into a real-time matrix factorization model, and then incorporate user/service group category information, to improving prediction accuracy.

1.Building group category in real-time QoS matrix data. User group category and service group category are obtained by Principal Component Analysis (PCA)[24] and K-means algorithm. And the information of group category is used to modeling a new regularization term in matrix factorization model.

2.We propose the MFC method which integrates clustering algorithm and matrix factorization model. Moreover, we conducted comprehensive experiments based on real QoS dataset, and the results demonstrate the advantage of MFC over several alternative methods, especially in extremely sparse QoS matrix data.

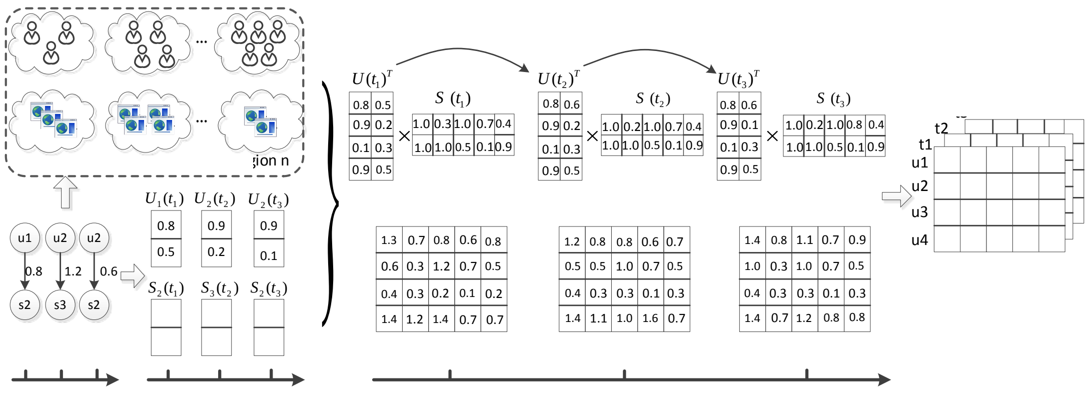


Fig. 1. Overall approach of MFC

2. Related work

In cloud platform, web service have a large number of candidate services, so the prediction of its QoS has a hot research. The most successful method for predicting the QoS of web service is based on the Collaborative Filtering (CF) method [8].

CF can be divided into two different methods: content-based method [9] and model-based method [9, 10, 11] proposed a content-based hybrid method, which combines user-based and item-based technology to predict the reliability of SOS. In [10], the traditional content-based CF algorithm is extended by considering the user's regional information. By predicting the reliability of a atomic web service, [12] using the QoS clustering method predict the reliability of composite services. MF-based method in model-based method has been applied in many studies because of its excellent prediction performance [1,3]. In [8], compared the content-based method and model-based method, obtaining: the model-based is better than the content-based method in predicting accuracy index.

Most of the CF methods on predicting QoS consider only two dimensions: user and service, excluding the time information. Considering the periodicity of the QoS value of web services, the prediction accuracy can be significantly improved. [1] collects a large number of response time throughput of QoS on continuous time slices, which provides a good experimental basis for time-aware QoS prediction. [3] propose an extended model-based CF algorithm for cold-start of QoS prediction. They cluster users and services according to latitude and longitude information of users and services and preference propagation method. They assume that when a user invokes a web service, users in nearby locations often encounter similar QoS, because local users may share the same network infrastructure. Based on the above assumptions, the new user's QoS can be predicted by considering the historical calls of users geographically close to the new user. However, this work is offline without considering the time information.

In order to overcome the limitations of previous works, we propose a expanded matrix factorization model for real-time web service QoS prediction, integrating clustering algorithm with matrix factorization model (MFC). The next section will give a detailed description of our method.

3. MFC

In this section, we introduce our method MFC, which include introduction of QoS attributes of web service and formulating the real-time QoS prediction problem.

3.1 QoS attributes

QoS is often used to evaluate the quality of non-functional features of web service. The QoS values are same for different users by service providers measured, such as the price and availability of services. However, due to unpredictable communication links and heterogeneous user environments, the QoS values observed by different users may vary greatly. Therefore, we mainly focus on the QoS values of observation from the perspective of users, such as response time and throughput. Rationally, the QoS values can be specified directly by service providers in Service Level Agreements (SLAs). But in practical applications, there are some factors: (1) Time-varing: the varying network environment, such as the Internet[13], and the different network delays of dynamic service workloads, which generate a lot of QoS attribute values the fluctuate greatly over time. (2) Location-specific: users from different locations usually observe different QoS values on the same service. Similarly, the geographic location distribution of services has a significant impact on the perceived QoS of users. These factors have a great impact on the quality of service, so real-time prediction QoS needs to take these factors into precise prediction.

3.2 Matrix Factorization

Matrix Factorization (MF) is the most widely used method to predict the QoS of a single web service [1,3]. MF [8] solves the Collaborative Filtering (CF) problem by constraining the rank of the QoS matrix, which uses mathematical methods to decompose a matrix into two or more sub-matrices. MF assumes that there are some low-dimensional potential factors to properly partition the matrix. These factors are thought to affect the QoS value implicitly, so the prediction of missing values can be achieved by multiplying the factorization matrix again.

We use MF technology to predict the QoS value of a single web service. The historical QoS matrix of a user invoking a web service is a two-dimensional matrix, in which ij the elements are the QoS values, such as response time and throughput, which are observed when the i user invokes the j web service. The QoS matrix can be decomposed into user matrix U and service matrix S by MF.

Historical QoS data is a sparse matrix R composed of $m \times n$ users and web services. Matrix R can be decomposed into user matrix $U \in R^{m \times d}$ and service matrix $S \in R^{n \times d}$. As shown in (1), d is a potential factor.

$$R \approx U^T \cdot S \tag{1}$$

In order to accurately predict the value of QoS, every element in the U and S must be carefully calculated. In order to complete the calculation, we resolve to minimize the following loss function, as shown in (2):

$$L = \frac{1}{2}\sum_{i,j} I_{ij}\,(R_{ij} - U_i{}^T S_j)^2 + \frac{\lambda}{2}(\sum_i \|U_i\|_2^2 + \sum_j \|S_j\|_2^2) \tag{2}$$

where the first term represents the square error between the observed R_{ij} and predicted values $U_i^T S_j$. Especially, when R_{ij} observed, $I_{ij} = 1$, and 0 otherwise. The second term is the regularization term. It is to solve the occurrence of over-fitting phenomena, and the parameters λ are important to control the regularization term.

Gradient descent [18] is a common algorithm for deriving the solution of U and S by starting with random initialization and iterating until the convergence of the update process. After the U and S solution is obtained, the predicted QoS value is calculated by using the inner product.

3.3 Other recommendations

To simplify the problem description, we formalize the problem of real-time prediction of QoS, as shown in **Fig. 2**.

(a)User, users call services;

(b)Application, different applications can call certain services in different ways according to their functional requirements;

(c)Candidate service, each component service has many candidate services, allowing dynamic replacement of existing services with another candidate service. Service invocation and adaptation actions are supported by the underlying middleware, which can track the call records of each service and record the corresponding QoS values perceived by users.

(d)The QoS record section describes multiple QoS records from different users during historical service invocations. Each record present QoS value of user invoking service in its time slice. Generally, each user only invokes a small group of candidate services at a time, instead of many other services, which leads to the existence of unknown QoS values;

(e)The collected QoS records can be further assembled into a 3D (user-service-time) QoS matrix, where the observed values are represented by numerical values, and the blanks are unknown. In order to make the best service adaptation decision, not only the real-time QoS information of work services but also the real-time QoS information of all candidate services are needed. Therefore, we formulate the real-time QoS prediction of candidate services as a problem, predicting the unknown QoS values of the current time slice under the given historical QoS records. Detailed description is as follows:

Hypothesis, there are n users and m services. A QoS record $R_{ij}(t) \in R^{n \times m}$ present user u_i call service s_j at time slice t. When value is observed, it is defined $I_{ij}(t) = 1, I_{ij}(t) = 0$ when it is unknown. The unknown QoS values are expressed as $\{R_{ij}(t_c) | I_{ij}(t_c) = 0\}$ at time slice t_c , which predicted based on historical observations $\{R_{ij}(t_c) | I_{ij}(t_c) = 1, t \in [t_1, \dots, t_c]\}$.

3.4 Modeling user groups and service groups

In this section, users and services are clustered based on historical invoking records, and user categories and service categories are obtained. New regularization term is modeled according the categories information and added to real-time MF model for accurate QoS prediction.

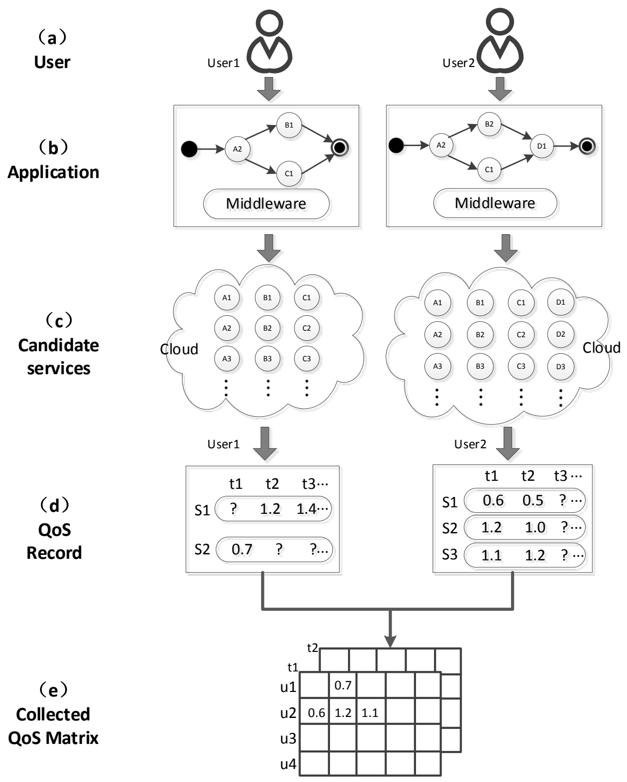


Fig. 2. The problem setting of online QoS prediction

Clustering can effectively and directly aggregate users and services according to certain rules, such as according to geographical location, especially in real QoS dataset [1]. But for real QoS data, it has that the number of services is much larger than the number of users. So we need to consider dimension reduction of service side, and consider the dimension of the QoS matrix, which its number of users is far less than the number of services. When clustering user side, the number of service is very large, service side data is needed to reduce the dimension for improving clustering computational efficiency. When clustering service side, there is no need to reduce dimensions because the number of users is far less than the number of services (if the number of users corresponding to the service is too large, it is also necessary to reduce the dimensionality of user side in clustering service to improve the clustering effect). Here we use the Principal Component Analysis (PCA)[14] method reducing dimension.

Our clustering algorithm is summarized as shown in **Fig. 3** and **Table 1**. First, we reduce the dimension of service side in QoS matrix data to cluster users with fewer features. And we cluster services with no reduce the dimension of user side. K-means algorithm is interpretable and easy to implement, so we use K-means algorithm to cluster users and services [25]. However, it is well known that the clustering results of K-means will be different each time [26]. This problem is related to the distribution of sample points. When the distribution of sample points is more dispersed, the difference will be larger. Especially, the points on the edge of the class will be clustered into different classes in different clustering, which will lead to the non-uniqueness of clustering results. In order to avoid this problem, we repeat the clustering operation, and rounding off the different clustering results. The same method clustering service side. The number of repetitions of clustering users or services needs to be determined according to the number of users and services. The more the number of repetitions, the greater the number of repetitions. The concrete steps of our algorithm are shown in algorithm1. After PCA reduces the dimension of matrix, clustering users and services by k-means algorithm.

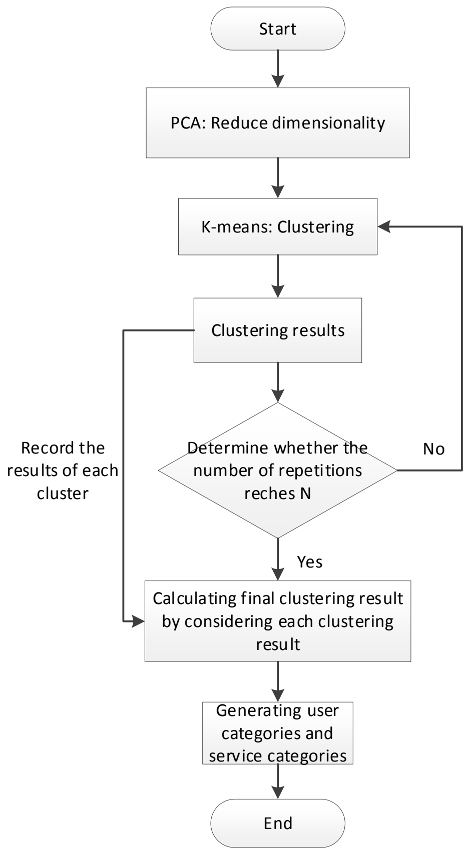

Start
PCA: Reduce dimensionality
K-means: Clustering
Clustering results
Record the results of each cluster
Determine whether the number of repetitions reches N
No
Yes
Calculating final clustering result by considering each clustering result
Generating user categories and service categories
End

Fig. 3. Algorithm for cluster users and services

Table 1. Algorithm 1 Clustering users and services

Algorithm 1 Clustering users and services

Input: QoS matrix
Procedure:

```
1.  Compute the mean value of QoS matrix on all time
    slices – mean QoS matrix
2.  Reduce the dimension of services side in mean QoS
    matrix by PCA, getting new mean QoS matrix.
3.  Clustering users in new mean QoS matrix by K-mean
    algorithm (setting 12 user groups)
    Get ten times clustering results
    For i = 1 : number of row QoS
```

```
        Compute the number of u(i) belong any group in
    clustering every time
         i = i + 1
         Get u(i) belong which user groups
    Get user groups
4.  Cluster services in mean QoS matrix by k-means
    algorithm (setting 100 service groups)
    Get ten times clustering results
    For j = 1 : number of column QoS
        Compute the number of s(j) belong any group in
    clustering every time
         J = j + 1
         Get s(j) belong which service groups
    Get service groups
Output: user groups and service groups
```

After calculating the clustering results of users and services, we get the user groups and service groups. Next, we use clustering results to model new regularization terms, as shown in Equations (3) and (4):

$$min \left\| U_i - \sum_{a \in SG_u(i)} w_a U_a \right\|_2^2 \tag{3}$$

$$min \left\| V_j - \sum_{b \in SG_v(j)} w_b V_b \right\|_2^2 \tag{4}$$

Where U_i, V_j represent factor matrices of target users ans web service, respectively. $SG_u(i)$ and . $SG_u(j)$ represent the group of classes of target user and web service, respectively. U_a is a user a that belong a group with target user i and V_b is a service b that belong a group with target service j . w_a and w_b are weights of user U_a and service V_b, respectively. As shown in (5) and (6):

$$w_a = \frac{sim(i,a)}{\sum_{c \in SG_u(i)} sim(i,c)} \tag{5}$$

$$w_b = \frac{sim(j,b)}{\sum_{d \in SG_v(j)} sim(j,d)} \tag{6}$$

Where $sim(i,a)$ represent similarity between user a that belong a same group with target user i and target user i. Equation (5) indicates that the more similar the user, the greater the impact on the prediction effect, and the same (6) has the same impact on the service.

Specifically, these regularizations imply that the QoS value of the target user or web service should be close to that of the same group of users or web services.

3.5 Clustering-based adaptive matrix factorization

According to the regularization terms obtained in the previous section, a new regularization term is added to the basic MF real-time prediction QoS algorithm in this section, which makes our method better solve the cold-start and improve prediction accuracy.

MF[15] uses mathematical method to decompose a matrix into two or more sub-matrices. The prediction of missing values can be get by multiple sub-matrices. And real-time MF method is that sub-MF model of each time slice is connected by specific parameters. Unlike the basic real-time QoS prediction based on matrix factorization, our method in **Fig. 1** clustered the user-service history QoS values to get user categories and service categories, and then modeled a new regularization terms for predicting QoS. Specifically, **Fig. 1** (a) QoS stream, the collected QoS data are first used as input of clustering algorithm in this method for (b); (b) Clustering user and service, which has given specific clustering operations in the previous section (section 4.2); (c) Online Updating, which sequentially sends the standard data transformation operation to the MF model for real-time updating. This is a continuous model training process with online learning technology; (d) Adaptive Matrix Factorization-Clustering. The Adaptive Matrix Factorization operation is the basic time iteration operation of MF. From (d) can be seen that AMF is the iteration of MF on continuous time slice, in which the model trained in the previous time slice is seamlessly used. To guide the next time slice; (e) Predicted QoS Matrix, real-time QoS prediction using the training model.

4. Data transformation

The real QoS values have different scores from recommendation systems. The scoring ranges in recommendation systems are small, such as 1~5. The range of changes in QoS values is wide, such as the response time ranges from 0~20 seconds, and the throughput ranges from 0~7000kbps. In addition, compared with the score distribution, the distribution deviation of the QoS data is large and has a large variance as shown in **Fig. 4**, which doesn't match the probability hypothesis of low rank matrix factorization. This type of data will reduce the accuracy of the prediction algorithm of MF[15].

In order to solve this problem, we apply Box-Cox to transform QoS data[16]. This method is used to stabilize the data variance and make the data more normal distribution to adapt to the MF model. Box-Cox transformation is rank-preserved, defined by (7):

$$boxcox(x) = \begin{cases} (x^{\partial} - 1)/\partial\,, & if\ \partial \neq 0 \\ \log(x)\,, & if\ \partial = 0 \end{cases} \tag{7}$$

The parameters ∂ control the degree of transformation. It should be noted that due to the monotonous non–decreasing nature of Box-Cox transformation, we have $b_{max} = boxcox(R_{max})$ and $b_{min} = boxcox(R_{min})$. R_{max} and R_{min} represent maximum and minimum QoS values respectively. R_{max} and R_{min} is the maximum and minimum value after data transformation. After the conversion of the QoS data, it is mapped to the range [0,1], such as (8):

$$r_{ij}(t) = (boxcox\left(R_{ij}(t)\right) - b_{min})/(b_{max} - b_{min}) \tag{8}$$

Where $R_{ij}(t)$ is QoS value under user u_i , service S_j and time slice t conditions.

In particular, when $\partial = 1$, Box-Cox data conversion was a common linear normalization.The distribution of QoS data after data conversion steps in shown in **Fig. 5**, which has a good effect for MF.

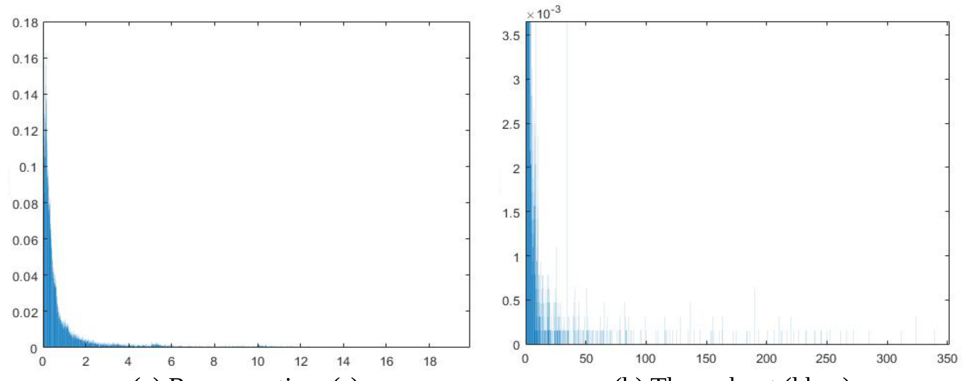

(a) Response time (s) (b) Throughput (kbps)

Fig. 4. Data distribution

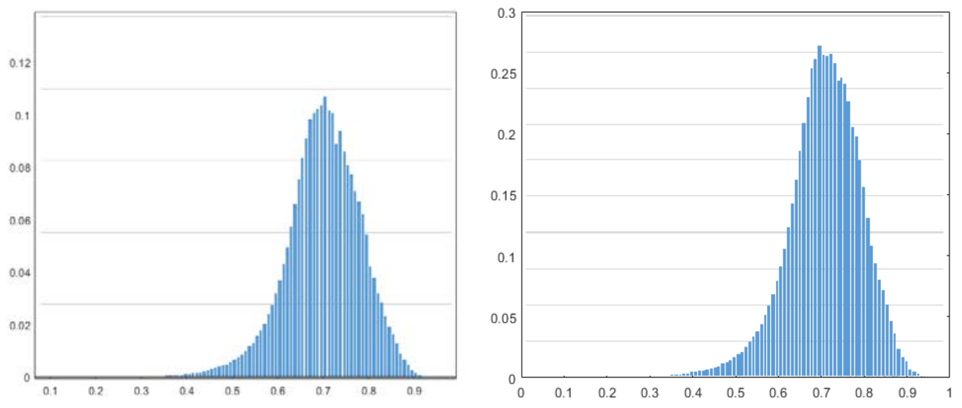

(a) Response time of transformed (s) (b) Throughput of transformed (kbps)

Fig. 5. Transformed data distribution Data distribution

5. Modeling

The purpose of this section is to establish a complete real-time QoS prediction model based on MF and clustering. Real-time QoS modeling can be represented by the interaction between users and services, representing the QoS between users u_i and services s_j in time slice t, as shown in (9):

$$\hat{R}_{ij}(t) = U_i(t)^T S_j(t) \tag{9}$$

In order to adapt to normalized QoS data $r_{ij}(t)$, we use Logistic function $g(x) = 1/(1+e^{-x})$ to map $\hat{R}_{ij}(t)$ in the range [0,1]. Then by applying Equation (9) and equations (3) ,(4) of C part of III, we can define the loss function as shown in (10):

$$L(t) = \frac{1}{2}\sum_{i,j} I_{ij}(t)(r_{ij}(t) - g_{ij}(t))^2 + \frac{\lambda}{2}\left(\sum_i \|U_i(t)\|_2^2 + \sum_j \left\|S_j(t)\right\|_2^2\right) + \frac{\gamma}{2}\left(\sum_i \left\|U_i - \sum_{a\in SG_u(i)} w_a U_a\right\|_2^2 + \sum_j \left\|V_j - \sum_{b\in SG_v(j)} w_b V_b\right\|_2^2\right) \tag{10}$$

Where $g_{ij}(t)$ is $g(\hat{R}_{ij}(t))$. When $r_{ij}(t)$ was observed, $I_{ij}(t) = 1$. Other, $I_{ij}(t) = 0$.

However the traditional MF model minimizes the sum of squared errors and uses absolute error measures for prediction accuracy evaluation. In fact, the absolute error measure is not suitable for the evaluation of QoS prediction because of the wide range of QoS values. For example, for a given QoS value $q_{s1} = 1, q_{s2} = 100$, the threshold is set to: $q_{s1} < 5, q_{s2} > 90$. There are now two predictions: (a) $q_{s1} = 8, q_{s2} = 99$; (b) $q_{s1} = 0.9, q_{s2} = 92$; It can be seen that (a) there is a minimum MAE (mean square error), but it does not meet the threshold, while (b) meets the threshold, but the MAE is not as small as (a). Therefore, we need to minimize the relative error of the QoS prediction and derive the corresponding loss function as (11)

$$L(t) = \frac{1}{2}\sum_{i,j} I_{ij}(t)(\frac{r_{ij}(t) - g_{ij}(t)}{r_{ij}(t)})^2 + \frac{\lambda}{2}\left(\sum_i \|U_i(t)\|_2^2 + \sum_j \left\|S_j(t)\right\|_2^2\right) \\ + \frac{\gamma}{2}\left(\sum_i \left\|U_i - \sum_{a\in SG_u(i)} w_a U_a\right\|_2^2 + \sum_j \left\|V_j - \sum_{b\in SG_v(j)} w_b V_b\right\|_2^2\right) \tag{11}$$

By summing up all historical time slices $L(t)$, the minimum global loss function is $L = \sum_{t=1}^{t_c} L(t)$.

6. Online study

The gradient descent [17] method can be used to solve the above minimization problem. However, it is often used for offline work, so it can't easily adapt to time varying QoS values. Online learning algorithms are needed to maintain continuous and incremental updates using sequentially observed QoS data. To this end, this section uses a widely used online learning algorithm: stochastic gradient descent (SGD) [17] to train our MFC model. For each QoS record $(t, u_i, s_j, r_{ij}(t))$ observed by a user u_i invoking a service s_j in a time slice t, there is a point-by-point loss function, as shown in (12):

$$\Delta L = \frac{1}{2}(\frac{r_{ij} - g_{ij}}{r_{ij}})^2 + \frac{\lambda}{2}\left(\|U_i\|_2^2 + \left\|S_j\right\|_2^2\right) \\ + \frac{\gamma}{2}\left(\left\|U_i - \sum_{a\in SG_u(i)} w_a U_a\right\|_2^2 + \left\|V_j - \sum_{b\in SG_u(j)} w_b V_b\right\|_2^2\right) \tag{12}$$

There are also $L = \sum_{t=1}^{t_c}\sum_{i=1}^{n}\sum_{j=1}^{m} I_{ij}(t)\Delta L$, sum all observed QoS records. Our MFC model incrementally updates each observed QoS record $(t, u_i, s_j, r_{ij}(t))$ using rules, as shown in (13) and (14):

$$U_i \leftarrow U_i - \eta[(g_{ij} - r_{ij})g'_{ij}S_j/r_{ij}^2 + \lambda U_i + \gamma(U_i \\ - \sum_{a\in SG_u(i)} w_a U_a)] \tag{13}$$

$$S_j \leftarrow S_j - \eta[(g_{ij} - r_{ij})g'_{ij}U_i/r_{ij}^2 + \lambda S_j + \gamma(V_j \\ - \sum_{b\in SG_v(j)} w_b V_b)] \tag{14}$$

Where g'_{ij} is $g'(U_i^T S_j)$, and derivative of $g(x)$ is $g'(x) = e^x/(e^x + 1)^2$, η is to control the learning rate of each interaction step.

As shown in (a) and (c) of **Fig. 1**, whenever a new QoS record is observed, the user u_i will change the feature vector U_i slightly, while the service s_j will change the feature vector S_j slightly. Online learning eliminates the need for retraining the entire model, enabling our MFC model to quickly adapt to new QoS observations and allowing easy integration of new users and services.

However, the above online learning algorithm may not perform well under the high stirring rate of users (i.e., continuous joining or leaving). The convergence of the algorithm is controlled by learning rate η, but fixed η help to quickly move them to the correct position. However, for user u_1 call existing service s_2 , the feature vectors S_2 may have converged. When users u_1 adjust the feature vectors of services s_2 , they have large prediction errors with non-convergent feature vectors, which may increase the prediction errors rather than reduce the prediction errors. Therefore, online predicting model also requires to resist the loss of users and services.

In order to achieve this goal, we propose to use adaptive weights to control the step size when updating the model. Simply, accurate users should not move too much according to inaccurate services, inaccurate users need to move more than accurate services, and vice verse. We express the average error e_{u_i} of users u_i as well as the average error e_{s_j} of services s_j as. Therefore, we set two weights w_{u_i} and w_{s_j} for users u_i and services s_j , such as (15) and (16):

$$w_{u_i} = e_{u_i}/(e_{u_i} + e_{s_j}) \tag{15}$$

$$w_{s_j} = e_{s_j}/(e_{u_i} + e_{s_j}) \tag{16}$$

Where $w_{u_i} + w_{s_j} = 1$. To update the e_{u_i} and e_{s_j}, we apply the exponential moving average[18], which is a weighted average, giving more weight (β controlled) to the latest data, as shown in (17) and (18):

$$e_{u_i} = \beta w_{u_i} e_{ij} + (1 - \beta w_{u_i}) e_{u_i} \tag{17}$$

$$e_{s_j} = \beta w_{s_j} e_{ij} + (1 - \beta w_{s_j}) e_{s_j} \tag{18}$$

Where e_{ij} is relative error of QoS record, between g_{ij} and r_{ij} .as shown in (19) :

$$e_{ij} = |r_{ij} - g_{ij}|/r_{ij} \tag{19}$$

After getting the updated weights w_{u_i} and w_{s_j} for each interaction, we finally redefine (13) and (14), such as (20) and (21):

$$U_i \leftarrow U_i - \eta w_{u_i}[(g_{ij} - r_{ij}) g'_{ij} S_j / r_{ij}^2 + \lambda U_i + \gamma (U_i - \sum_{a \in SG_u(i)} w_a U_a)] \tag{20}$$

$$S_j \leftarrow S_j - \eta w_{s_j}[(g_{ij} - r_{ij}) g'_{ij} U_i / r_{ij}^2 + \lambda S_j + \gamma (V_j - \sum_{b \in SG_v(j)} w_b V_b)] \tag{21}$$

Where U_i , S_j are data in current time slice t_c.

7. Experiment

In this section, we conducted a set of experiments on a real QoS dataset of web services to evaluate our MFC from various aspects, including prediction accuracy, efficiency and parameter analysis. We use Matlab to implement the algorithm. The hardware environment is Intel Core i5 3.30GHz processor, and the memory size is 12GB.

7.1 QoS attributes

In our experiment, we focused on two QoS attributes, response time (RT) and throughput (TP), both of which are important in representing the nonfunctional quality of the service. Response time represents the time between the user initiating the request and receiving the response, while throughput represents the data transfer rate (such as kbps) of the user invoking the service.

We used a public real-word dataset: WS-Dream[1]. The dataset has been widely used in the research community since its publication. It contains approximately 40.9 million QoS records, recording response time and throughput values for service invocations for 142 users and 4,500 Web services on 64 continuous time slices spaced at 15 minutes intervals.

Table 2 provides some basic statistics for WS-Dream dataset. Both QoS attributes have a wide value range: the response time range is 0~20s (average 1.33s), and the throughput range is 0~7,000kbps (average 11.35kbps). We have drawn the original data distribution diagram of response time and throughput, as shown in **Fig. 4**. The results show that the data distribution is seriously deviated. **Fig. 5** depicts the normal data distribution obtained through the data transformation in DATA TRANSFORMATION of D part of III.

7.2 Evaluating indicator

As for the evaluation index of QoS prediction, we use two indexes to measure the accuracy of QoS prediction**.**

7.2.1 MRE (Median Relative Error)

MRE adopts the median of all paired relative errors, and its calculation formula is as follows (22):

$$MRE = median_{I_{ij(t)=0}}\{\hat{R}_{ij}(t) - R_{ij}(t)/R_{ij}(t)\} \tag{22}$$

Table 2. Units for Magnetic Properties

Property	Value
QoS Records	40,896,000
Users	142
Services	4,500
Timeslices	64
Time Interval	15min
Rt Range	0~20s
Rt Average	1.33s
Tp Range	0~7,000kbps
Tp Average	11.35kbps

Where $R_{ij}(t)$ is real QoS value, $\hat{R}_{ij}(t)$ is the corresponding predicted QoS value.

7.2.2 NPRE (Ninety-Percent Relative Error)

NPRE is 90% of all paired relative errors.

7.3 Accuracy comparison

To evaluate our proposed method and performance, we compared our real-time QoS prediction MFC method with four other methods that are considered to have QoS prediction potential[1, 10, 19-21]. Although these methods were not initially used for real-time service tuning, they were included in the comparison to compare algorithm accuracy.

AMF: Adaptive Matrix Factorization online predictive QoS algorithm[1], by using the real QoS data collected by my own team, the traditional matrix decomposition model is extended and the characteristics of real QoS data are fully considered through data transformation, online learning and adaptive weighting.

UIPCC: The user-based CF method (UPCC) uses similarity between users to predict QoS values, the item-based CF method (IPCC) uses similarity between services to predict QoS values. [19]combine the two and make full use of similar information between users and services to predict QoS.

PMF: this is a widely used predictive model based on MF. [20]PMF has been applied to offline QoS prediction. For our real-time QoS prediction problem, we use PMF under each time slice, and finally take the mean of all the predicted results under the time slice.

WSPred: [10]this method processes the time perception QoS prediction problem of Web services through the fusion of time information. It uses the tensor factorization model[22], which is based on the extension of low-rank MF and represents the 3D (user, service, time) QoS matrix.

NTF: Non-negative Tensor Factorization (NTF) is a method proposed by[21], which further extends WSPred and applies non-negative constraints to the tensor factorization model of time-sensing QoS prediction.

At the same time, in order to test performance of our method on sparse data, the optimal parameters obtained through our experiment are as follows: λ= 0.0003, γ=0.001/0.005/0.01 (see the parameter analysis in section 4.4 for details), a comparative experiment of sparse data was carried out.

The results of our comparative experiment are shown in **Table 3** and **Table 4**, respectively the results of the two predictors MRE and NPRE. We can see from the table that our method MFC has more accurate prediction effect than other algorithms (bold mark). The "Improve" column shows the average improvement under different densities of QoS matrix. For the response time (RT) attribute, MFC achieves an improvement of 3.3%~43.9% on MRE and 34.8%~658.9% on NPRE. Similarly, for the throughput (TP) attribute, MFC achieves an improvement of 4.2%~25% on MRE and 29.2%~1298% on NPRE. In particular, the AMF model has the optimal effect in all comparison algorithms. It is an extension of the traditional MF model and generates good accuracy by considering the characteristics of QoS attributes. However, the AMF model does not take into account the category of users and services that generate QoS values. Since QoS data is the attribute value generated by the user invoking the service, so QoS data has the characteristics of users and services. Therefore, our model takes category information of users and service into consideration in the AMF model to produce better prediction effect.

Table 3. QoS Prediction Accuracy (MRE)

QoS Methods		MRE(%)									
		D=0.5	D=1	D=2	D=3	D=4	D=5	D=10	D=15	D=20	**Improve**
RT	UIPCC	0.89	0.88	0.88	0.87	0.85	0.75	0.66	0.64	0.62	**43.9%**
	PMF	0.73	0.72	0.71	0.70	0.65	0.61	0.61	0.58	0.57	**31.1%**
	WSPred	0.69	0.67	0.66	0.64	0.62	0.59	0.54	0.52	0.49	**25.8%**
	NTF	0.64	0.63	0.62	0.60	0.58	0.56	0.51	0.51	0.48	**22.7%**
	AMF	0.49	0.45	0.42	0.39	0.37	0.36	0.32	0.30	0.29	**3.3%**
	MFC	**0.45**	**0.41**	**0.37**	**0.35**	**0.33**	**0.32**	**0.30**	**0.29**	**0.28**	—
TP	UIPCC	0.80	0.73	0.66	0.69	0.67	0.62	0.60	0.52	0.45	**25%**
	PMF	0.77	0.66	0.62	0.60	0.57	0.54	0.52	0.51	0.44	**19.30%**
	WSPred	0.68	0.57	0.53	0.50	0.49	0.44	0.43	0.42	0.36	**10.20%**
	NTF	0.64	0.54	0.52	0.50	0.48	0.46	0.43	0.41	0.37	**9.30%**
	AMF	0.58	0.51	0.47	0.45	0.42	0.41	0.38	0.36	0.32	**4.20%**
	MFC	**0.54**	**0.45**	**0.40**	**0.38**	**0.37**	**0.37**	**0.34**	**0.34**	**0.31**	—

Table 4. QoS Prediction Accuracy (NPRE)

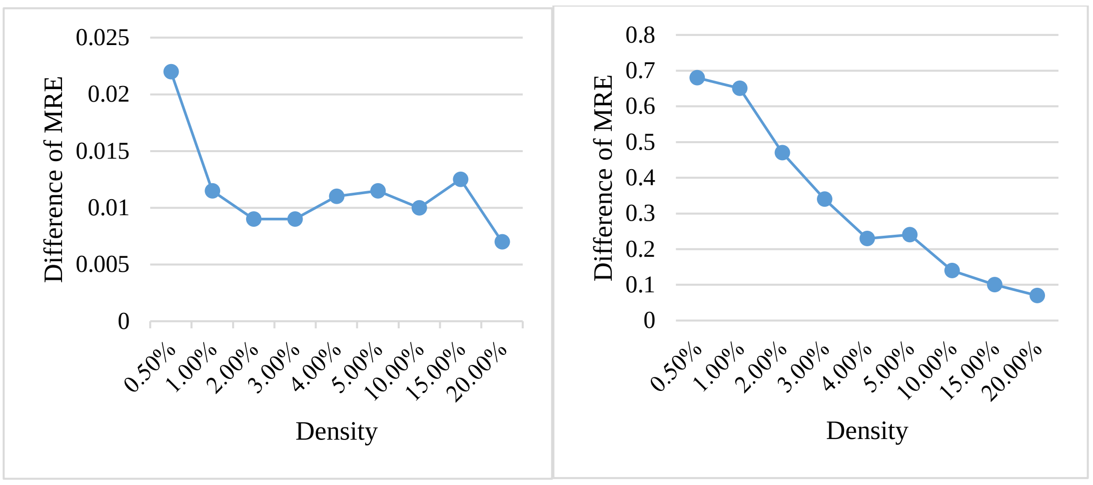
QoS Methods		NPRE(%)									
		D=0.5	D=1	D=2	D=3	D=4	D=5	D=10	D=15	D=20	**Improve**
RT	UIPCC	12.30	9.99	9.21	8.11	7.33	6.48	5.49	4.86	4.62	**658.9%**
	PMF	4.31	3.43	3.08	2.71	2.49	2.56	2.79	2.96	3.03	**203.1%**
	WSPred	5.68	4.45	3.91	3.70	3.44	3.35	3.13	2.83	2.68	**267.5%**
	NTF	4.99	4.18	3.80	3.61	3.42	3.23	3.11	3.05	2.98	**258.6%**
	AMF	1.99	1.79	1.62	1.45	1.28	1.21	1.02	0.95	0.91	**34.8%**
	MFC	**1.33**	**1.11**	**1.09**	**1.01**	**0.97**	**0.95**	**0.88**	**0.89**	**0.87**	—
TP	UIPCC	16.01	15.01	14.87	14.71	14.55	14.34	14.12	14.98	13.28	**1298%**
	PMF	3.51	3.66	3.45	3.20	2.91	2.85	3.13	3.34	3.42	**160.4%**
	WSPred	3.31	3.68	3.41	3.51	3.23	3.32	3.45	3.51	3.55	**177.0%**

NTF	3.12	3.00	2.96	3.00	3.10	3.36	3.44	3.43	3.45	**153.7%**
AMF	3.40	2.71	2.27	1.93	1.77	1.66	1.46	1.27	1.19	**29.2%**
MFC	**2.71**	**2.11**	**1.89**	**1.71**	**1.62**	**1.46**	**1.31**	**1.13**	**1.11**	—

7.4 Effect and efficiency of MFC

Here we need to analyze the effectiveness and efficiency of our method MFC. As shown in **Fig. 6**, we compare our method with AMF method in predicting the difference in accuracy. There are two differences: on MRE and on NPRE. We take the mean values of RT and TP of MFC at different densities of QoS matrix, and the mean values of RT and TP of AMF method at different densities of QoS matrix. Then we can get the predicting gap between MFC and AMF method at different densities of QoS matrix. We can see from **Fig. 6** that with the increase of density, the difference between MFC and AMF method is decreasing. This shows that our method has better prediction effect when the density of QoS matrix is very sparse. This is because our method incorporates user category and service category. When predicting unknown QoS values, similar users and similar services will be taken into account, and the prediction accuracy is improved. With the density increases, the gap between our method and AMF method decreases gradually on MRE and NPRE. This is because that, when the density of QoS matrix is not too, MF has more reliable data (compared with sparse data) when calculating the inner product of user term matrix and service term matrix with stochastic gradient descent algorithm. And MF prediction ability can be better reflected. The performance of predicting the QoS value based on user category and service category is weakened, which leads to the larger matrix density and the less obvious our method is. Therefore, it can be said that our method not only improves the prediction accuracy in all density, but also solves the cold start problem to a certain extent.

However, MFC also has drawbacks. In addition to the above-mentioned performance is not obvious when the density of QoS matrix is large, the algorithm complexity of MFC is too high, which consists of two parts: Calculating K-means results and Solving matrix factorization. The specific complexity of each part is shown in **Table 5**.



(a)Change rate of MRE (b) Change rate of NPRE

Fig. 6. Change rate of difference between our method and AMF.

Table 5. The computational complexity of MFC

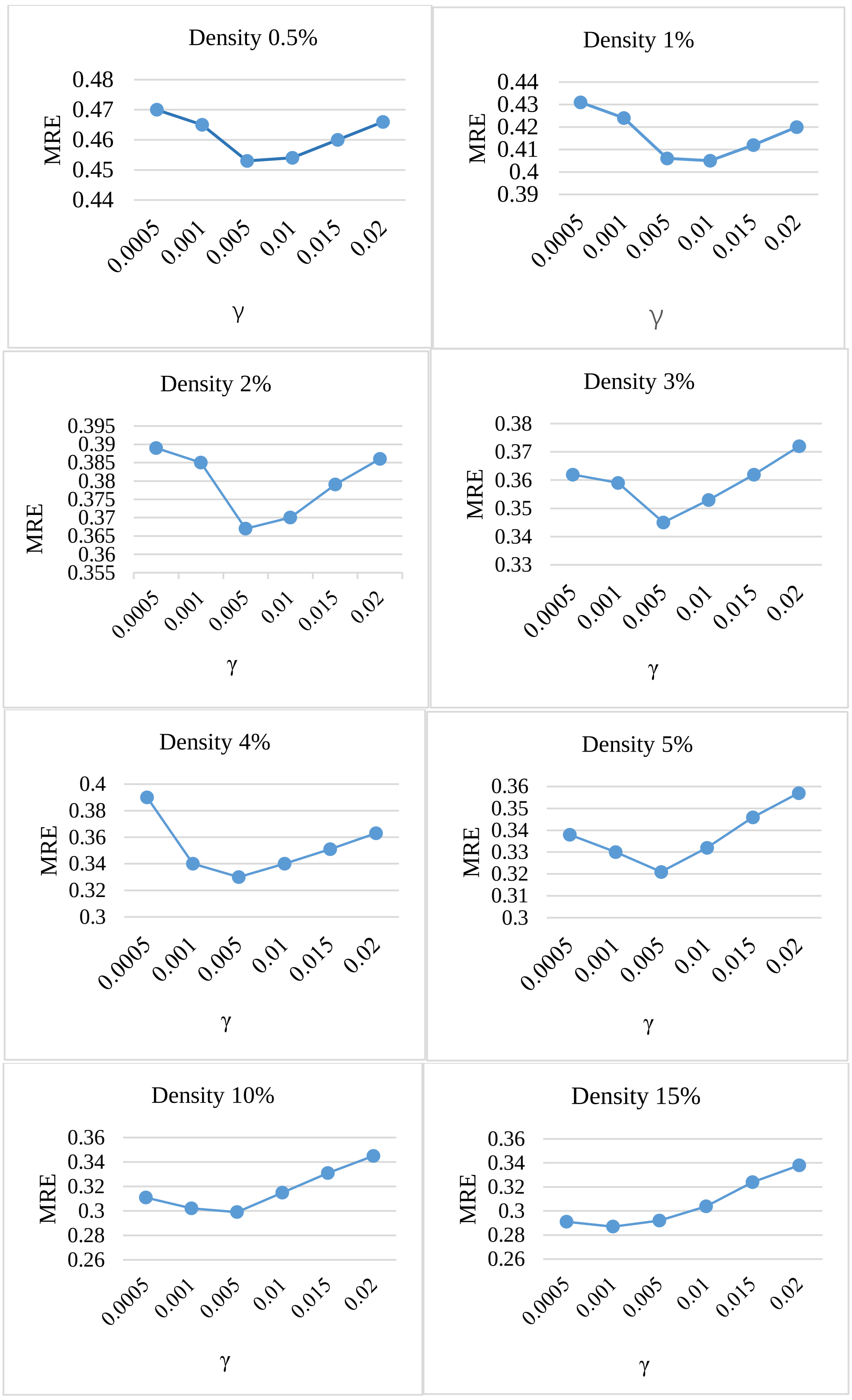
The major procedures	**Computational complexity**
Calculating K-means results	$O(m^2 + n^2)$
Solving matrix factorization	$O(i\rho d)$

Where m is the number of users, n is the number of service. i is the interaction times of gradient descent algorithm in MF model, ρ is the average size of data set used, and d is the number of potential factors in MF model.

From **Table 5**, we can see that the complexity of MFC is $O((m^2 + n^2) + i\rho d)$, and the complexity of AMF algorithm is $O(i\rho d)$, from this formula alone, the complexity of our method is far greater than that of AMF algorithm. However, it should be noted that **Fig. 6** shows that MFC has better performance when the density of the QoS matrix is very small. In other words, the number of users and the number of services corresponding to the density of the QoS matrix is very small, so there is little difference between $O((m^2 + n^2) + i\rho d)$ and $O(i\rho d)$. However, with the increase of the density of the QoS matrix, MFC not only has a low prediction accuracy, but also has a high algorithm complexity. And the algorithm complexity increases with the increase of density of QoS matrix.

7.5 Selection of Parameters

In order to evaluate our method, that is, to incorporate user categories and service categories into the basic real-time MF model, we set different values of parameters γ to find the values that make the algorithm the most accurate. There are two main parameters in the algorithm: the regularization parameter λ of avoiding over-fitting and the regularization parameter γ of clustering category. If the parameter γ is 0, MFC will become the basic AMF model. If the value of the parameter γ is large, the user category and service category will have a great impact on the predicted effect. On the basis of previous experience [1], we set the parameter λ as 0.0003, because the prediction accuracy is the highest at 0.0003. Therefore, we analyze the influence of parameters on predicting accuracy. As shown in **Fig. 7**, we take six values of parameters γ: 0.0005, 0.001, 0.005, 0.01, 0.015, 0.02. It can be observed that MRE decreases when γ is 0 to 0.001 or 0.005 or 0.01, which means that the use of category regularization term can improve the predicted accuracy. However, when the parameters γ reach a certain value (0.001 or 0.005 or 0.01), MRE begins to increase, which is due to the occurrence of over-fitting phenomenon, which makes the predicted accuracy no longer improve. So concluded that adding basic real-time MF model to the category regularization term is helpful to improve the predicted accuracy, but not arbitrary γ values are helpful to improve the predicted accuracy. Fixed values are recommended: 0.001 or 0.005 or 0.01.


Density 0.5%
MRE
0.48
0.47
0.46
0.45
0.44
0.0005
0.001
0.005
0.01
0.015
0.02
γ
Density 1%
MRE
0.44
0.43
0.42
0.41
0.4
0.39
γ
Density 2%
MRE
0.395
0.39
0.385
0.38
0.375
0.37
0.365
0.36
0.355
γ
Density 3%
MRE
0.38
0.37
0.36
0.35
0.34
0.33
γ
Density 4%
MRE
0.4
0.38
0.36
0.34
0.32
0.3
γ
Density 5%
MRE
0.36
0.35
0.34
0.33
0.32
0.31
0.3
γ
Density 10%
MRE
0.36
0.34
0.32
0.3
0.28
0.26
γ
Density 15%
MRE
0.36
0.34
0.32
0.3
0.28
0.26
γ

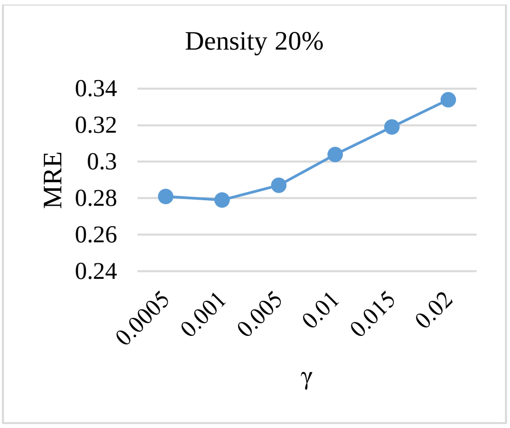


Fig. 7. Importance of parameter γ .

8. Discussion

In this paper, we propose an extended matrix factorization model (MFC) for real-time web service QoS prediction. Based on our analysis and conclusions of predicting candidate service QoS model, we use user/service category group information to extend the traditional matrix factorization, improving the prediction accuracy and at the same time makes our model more suitable for real-time QoS prediction under cold-start conditions. However, our model has some limitations. Therefore, we discuss the complexity and improvements of our model, so as to provide some guidance for future work.

Complexity issues. Through the discussion in the previous section, we know that our method has a high complexity, which is caused by the operation of extracting group of categories of users and services. But the complexity of our K-means algorithm is less than the complexity of propagating clustering categories through preference [3]. Obviously, the characteristics of users and services have good expressive power for prediction under collaborative filtering, but its disadvantage is that it will increase the complexity of the model. Therefore, how to improve the prediction effect without spending a lot of calculation cost is an urgent problem to be solved.

Potential improvements in QoS prediction. Our method currently uses the K-means clustering algorithm to extract the characteristics of users and services. However, the K-means algorithm is unstable, especially for large-scale QoS matrices, the distribution of sample points is very scattered, and the K-means clustering effect will be greatly reduced. This problem is worthy of further exploration. A more stable clustering algorithm can be used to extract the characteristics of users and services. In addition, in this paper, the exponential moving average is used to adjust the learning step parameters. However, time information will be more useful, such as the application of time series technology to change point detection.

9. Conclusion and future work

We proposed a expanded matrix factorization model for real-time web service QoS prediction. Our method firstly preprocesses data by PCA, then uses K-means algorithm to cluster users and web services, and the new regularization item of model using category group information of users and services, expanding the traditional matrix factorization model. The validity of our method is verified by experiments on real dataset. Finally, the limitations and potential of our method are discussed. Our future work aims to propose a low complexity prediction model that suitable the cold-start QoS prediction problem.

References

[1] Zhu, J, et al., "Online QoS prediction for runtime service adaptation via adaptive matrix factorization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 10, pp. 2911-2924, Oct. 2017. Article (CrossRef Link)

[2] Metzger, A., et al., "Towards pro-active adaptation with confidence: augmenting service monitoring with online testing," in *Proc. of 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems*, pp. 20-28, 2010. Article (CrossRef Link)

[3] Ryu, D., K. Lee, and J. Baik, "Location-based Web Service QoS Prediction via Preference Propagation to address Cold Start Problem," *IEEE Transactions on Services Computing*, vol. 14, no. 3, pp. 736-746, Apr. 2018. Article (CrossRef Link)

[4] Cheng T, Wen J, Xiong Q, et al., "Personalized Web Service Recommendation Based on QoS Prediction and Hierarchical Tensor Decomposition," *IEEE Access*, vol. 7, pp. 62221-62230, Apr. 2019. Article (CrossRef Link)

[5] Li S, Wen J, Luo F, et al., "Time-Aware QoS Prediction for Cloud Service Recommendation Based on Matrix Factorization," *IEEE Access*, vol. 6, pp.77716-77724, Nov. 2018. Article (CrossRef Link)

[6] D. Wu, X. Luo, M. Shang, Y. He, G. Wang and X. Wu, "A Data-Characteristic-Aware Latent Factor Model for Web Services QoS Prediction," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1-1, 2020. Article (CrossRef Link)

[7] Herlocker, J.L., et al., "Evaluating collaborative filtering recommender systems,"*ACM Transactions on Information Systems (TOIS)*, vol. 22, no 1, pp. 5-53, Jan. 2004. Article (CrossRef Link)

[8] Hartigan, J.A. and M.A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society*. *Series C (Applied Statistics)*, vol. 28, no 1, pp. 100-108, 1979.

[9] Zheng, Z. and M.R. Lyu, "Personalized reliability prediction of web services," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 22, no 2, pp. 12, pp. 1-25, Mar. 2013. Article (CrossRef Link)

[10] Zheng, Z. and M.R. Lyu, "Collaborative reliability prediction of service-oriented systems," in *Proc. of 2010 ACM/IEEE 32nd International Conference on Software Engineering*, vol. 1, pp. 35-44, May, 2010. Article (CrossRef Link)

[11] Zhang, Y., Z. Zheng, and M.R. Lyu, "WSPred: A time-aware personalized QoS prediction framework for Web services," in *Proc. of 2011 IEEE 22nd International Symposium on Software Reliability Engineering*, 2011. Article (CrossRef Link)

[12] Zheng, Z., et al., "Collaborative web service qos prediction via neighborhood integrated matrix factorization," *IEEE Transactions on Services Computing*, vol. 6, no 3, pp. 289-299, Jan. 2012. Article (CrossRef Link)

[13] Yang, Y., et al., "Generalized aggregate Quality of Service computation for composite services," *Journal of Systems and Software*, vol. 85, no 8, pp. 1818-1830, Aug. 2012. Article (CrossRef Link)

[14] Strom, D. and J.F. van der Zwet, "Truth and lies about latency in the cloud," Netherlands: Interxion, 2012. [Online]. Available:https://www.interxion.com/whitepapers/truth-and-lies-of-latency-in-the-cloud

[15] Mohammed, A.A., et al., "Human face recognition based on multidimensional PCA and extreme learning machine," *Pattern Recognition*, vol. 44, no 10, pp. 2588-2597, Oct. 2011. Article (CrossRef Link)

[16] Mnih, A. and R.R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. of the 20th International Conference on Neural Information Processing Systems*, pp. 1257-1264, 2007. Article (CrossRef Link)

[17] Sakia, R., “The Box-Cox transformation technique: a review,” *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 41, no 2, pp. 169-178, June 1992. Article (CrossRef Link)
[18] Shapiro, A. and Y. Wardi, “Convergence analysis of gradient descent stochastic algorithms,” *Journal of optimization theory and applications*, vol. 91, no 2, pp. 439-454, Nov. 1996. Article (CrossRef Link)
[19] de Souza, M.J.S., et al., “Examination of the profitability of technical analysis based on moving average strategies in BRICS,” *Financial Innovation*, vol. 4, no 1,pp. 3, Feb. 2018. Article (CrossRef Link)
[20] Zheng, Z., et al., “WSRec: A Collaborative Filtering Based Web Service Recommender System,” in *Proc. of IEEE International Conference on Web Services*, July 6-10, 2009. Article (CrossRef Link)
[21] Zheng, Z., et al., “Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization,” *IEEE Transactions on Services Computing*, vol. 6, no 3, pp. 289-299, July. 2013. Article (CrossRef Link)
[22] Zhang, W., et al., “Temporal QoS-aware Web Service recommendation via Non-negative Tensor Factorization,” in *Proc. of International Conference on World Wide Web*, pp. 585–596, 2014. Article (CrossRef Link)
[23] Kolda, T.G. and B.W. Bader, “Tensor Decompositions and Applications,” *Siam Review*, vol. 51, no 3, pp. 455-500, Aug. 2009. Article (CrossRef Link)
[24] A. Kim, C. Wang and S. Seo, "PCA-CIA Ensemble-based Feature Extraction for Bio-Key Generation," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 7, pp. 2919-2937,July. 2020. Article (CrossRef Link)
[25] A. Raza, M. F. Khan, M. Maqsood, B. Haider and F. Aadil, "Adaptive k-means clustering for Flying Ad-hoc Networks," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2670-2685, June 2020. Article (CrossRef Link)
[26] J. Kim, M. Ryu and S. Cha, "Approximate k values using Repulsive Force without Domain Knowledge in k-means," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 3, pp. 976-990, Mar. 2020. Article (CrossRef Link)

Jinsheng Hao received his Master's degree from Wuhan University of Techology. Now, he is a Ph. D candidate, with Information college Department, Xinjiang University, Urumqi, China. His current interests are network security and cloud computing etc.

Guoping Su received the Ph.D. from the Beijing University of Aeronautics and Astronautics. He is a doctoral supervisor of Xinjinag University. His research interests include network security and Information science.

Xiaofeng Han received her Master’s degree from Xinjiang University. She is currently a PhD student in School of Computer Science, Beijing University of Posts and Telecommunications, Beijing. Her research interests include user behavior analytics, recommender systems and intelligent information processing.

Wei Nie received the Ph.D. from the University of Electronic Science and Technology. He is a Lecturer of Shenzhen University. His research interests include network security and Software Defined Network.