IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INNOVATION AND APPLICATION OF INTELLIGENT PROCESSING OF DATA, INFORMATION AND KNOWLEDGE AS RESOURCES IN EDGE COMPUTING

Received April 3, 2019, accepted April 23, 2019, date of publication May 6, 2019, date of current version May 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2914737

QoS Prediction for Mobile Edge Service Recommendation With Auto-Encoder

YUYU YIN[1,2], WEIPENG ZHANG[1,2], YUESHEN XU[3,4], HE ZHANG[3], ZHIDA MAI[5], AND LIFENG YU[6]

[1]School of Computer, Hangzhou Dianzi University, Hangzhou 310018, China
[2]Key Laboratory of Complex Systems Modeling and Simulation, Ministry of Education, Hangzhou 310018, China
[3]School of Computer Science and Technology, Xidian University, Xi'an 710126, China
[4]Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou 215006, China
[5]Xanten Guangdong Development Co., Ltd., Foshan 528200, China
[6]Hithink RoyalFlush Information Network Co., Ltd., Hangzhou 310023, China

Corresponding author: Yueshen Xu (ysxu@xidian.edu.cn)

This work was supported in part by The National Key Research and Development Program of China under Grant 2017YFB1400601, in part by the National Natural Science Foundation of China under Grant 61872119 and Grant 61702391, in part by the Natural Science Foundation of Zhejiang Province under Grant LY16F020017, in part by the Shaanxi Province under Grant 2018JQ6050, and in part by the Fundamental Research Funds for Central Universities under Grant JBX171007.

ABSTRACT In the mobile edge computing environment, there are a large number of mobile edge services which are the carriers of various mobile intelligent applications. So how to recommend the most suitable candidate from such a huge number of available services is an urgent task, especially the recommendation task based on quality-of-service (QoS). In traditional service recommendation, collaborative filtering (CF) has been studied in academia and industry. However, due to the mobility of users and services, there exist several defects that limit the application of the CF-based methods, especially in an edge computing environment. The most important problem is the cold-start. In this paper, we propose an ensemble model which combines the model-based CF and neighborhood-based CF. Our approach has two phases, i.e., global features learning and local features learning. In the first phase, to alleviate the cold-start problem, we propose an improved auto-encoder which deals with sparse inputs by pre-computing an estimate of the missing QoS values and can obtain the effective hidden features by capturing the complex structure of the QoS records. In the second phase, to further improve prediction accuracy, a novel computation method is proposed based on Euclidean distance that aims to address the overestimation problem. We introduce two new concepts, common invocation factor and invocation frequency factor, in similarity computation. Then we propose three prediction models, containing two individual models and one hybrid model. The two individual models are proposed to utilize user similar neighbors and service similar neighbors, and the hybrid model is to utilize all neighbors. The experiments conducted in a real-world dataset show that our models can produce superior prediction results and are not sensitive to parameter settings.

INDEX TERMS QoS prediction, service recommendation, auto-encoder, features learning, similarity computation.

I. INTRODUCTION

With the explosive growth of mobile devices, the recent years have witnessed an unprecedented shift of user preferences from traditional desktops and laptops to smartphones and other connected devices. Subsequently, mobile edge services emerge and attract a lot of public attention [1]. Based on the development of mobile services, mobile edge computing (MEC) is developed to significantly improve the quality of experience for uses in mobile environment [2]–[7]. In the paradigm of MEC, as the user requirement is served by a nearby edge node rather than the remote cloud, the end-to-end latency is significantly reduced [8], [9]. As a result, more and more new mobile services have been or are being developed by various enterprises to deliver their business applications in mobile environments. With the number of mobile services growing dramatically, different service providers offer many mobile services with the same or similar

The associate editor coordinating the review of this manuscript and approving it for publication was Ying Li.

functions. It is an urgent task to select suitable mobile services from a large number of service candidates when users travel across the edges of different mobile networks.

There are two types of properties for a service, including functional properties and non-functional properties [10]. The non-functional properties are also known as quality-of-service (QoS). QoS is defined as a set of user experienced properties, such as response time, throughput and reliability [11]. The key criterion in service recommendation and service selection is QoS, which can distinguish the suitable services among different functionally equivalent services. However, it is really hard for a user to invoke all candidate services to acquire all QoS values to make a final decision. Thus, QoS prediction is an indispensable task to finish service recommendation with high quality [11].

Recently, collaborative filtering (CF) methods have been introduced to solve QoS prediction problem and obtained good prediction accuracy [12]. CF methods can be divided into two types, i.e., neighborhood-based CF and model-based CF. The neighborhood-based CF predicts QoS by exploiting historical information from similar users or similar services. A key factor impacting the prediction performance is the similarity computation for similar neighbors selection. But as usually the QoS records are highly sparse, such a high sparsity is easy to lead to poor quality of neighbors identification. The model-based CF usually employs matrix factorization (MF for short) technique [13] and learns latent factors from the QoS records. MF models the user-service invocation relationship as the inner product of user latent vector and service latent vector. It has been verified by many experiments that model-based methods often have better performance than neighborhood-based methods. However, the existing model-based methods can only learn linear relationship between a user and a service, and the inner product cannot catch deep features [14]. In addition, another weakness of CF is the cold-start problem, i.e., how to recommend a service to a user when there are few or no QoS records neither in service side or in user side.

To address these problems, in this paper, we propose an ensemble model that combines CF method and auto-encoder network. The built model combines the neighborhood-based method and the model-based method where side information is also integrated into training process. The proposed approach has two steps, including global features learning and local features learning. In global features learning, we build an improved auto-encoder which learns a non-linear representation of users and services and alleviates the cold-start problem. In local features learning, we introduce two concepts, i.e., common invocation factor and invocation frequency factor, which are used in similarity computation.

In summary, the contributions of this paper are as follows.

- It proposes an improved auto-encoder network, which can deal with sparse QoS records by pre-computing an estimate of missing values, and is verified to be effective in solving cold-start problem.
- It proposes a novel similarity computation method which can alleviate the overestimation problem. It extends Euclidean distance using common invocation factor and invocation frequency factor. The proposed similarity computation method can identify high-quality neighbors.
- It proposes two individual collaborative prediction methods, and one method finishes the prediction problem from user side and the other is from service side. It also proposes a hybrid method that can combine the prediction results of the two individual methods.
- It conducts sufficient experiments on a real-world dataset, and compares the proposed models with many existing models. The experimental results demonstrate the effectiveness of our models.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 and Section 4 presents the framework of our work and describes the proposed models. The experimental results are presented in Section 5. Section 6 concludes the paper and discusses future work.

II. RELATED WORK

The recommender system is applied to handle many problems, and web service recommendation is one of typical recommender systems. Collaborative filtering (CF) is widely used in web service recommendation and service selection [13]. Collaborative filtering algorithm for QoS can be classified into two types [15], including neighborhood-based CF and model-based CF. Most of neighborhood-based methods are developed from the following aspects, including improving the similarity computation, improving the neighbors selection quality or combining different methods.

Neighborhood-based CF algorithms can be classified into user-based CF methods [16], service-based CF methods [17] and hybrid CF methods [18], [19]. Shao *et al.* [20] used the user-based neighborhood collaborative filtering algorithm to obtain similar users, and applied collaborative filtering algorithm to service computing. Wu *et al.* [21] proposed a ratio-based method to compute the similarity and compared the attribute values directly, improving the accuracy of neighbors identification. Zou *et al.* [22] combined neighbors of users and neighbors of services into a unified collaborative filtering algorithm. Li *et al.* [23] studied temporal dynamic characteristics of QoS values in similarity computation. Yin *et al.* [24] discover that mining the potential similarity relations among users or services in CPS is really helpful to improve the prediction accuracy.

With the increase of the number of services and users in the network, the high sparsity problem of service invocation records becomes serious [25]. It is difficult to obtain accurate similar neighbors in neighborhood-based CF methods, while model-based CF methods are still applicable with high sparsity data. Model-based CF methods use machine learning techniques, such as latent factor models [13], clustering-based models [26], [27], and aspect-based models [28]. As a latent factor model, matrix factorization

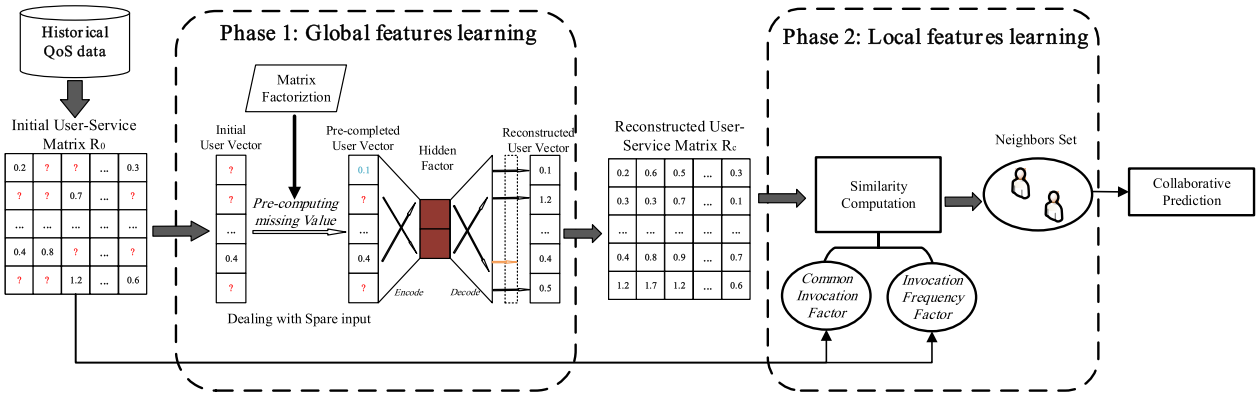


FIGURE 1. The whole framework of the proposed method.

(MF for short) is widely used both in industry and in academia [29]–[32]. Andriy and Salakhutdinov [13] proposed probabilistic matrix factorization (PMF) model that improved the traditional matrix factorization for QoS prediction. Zheng *et al.* [33] combined the neighborhood-based CF and model-based CF to achieve higher prediction accuracy. Yang *et al.* [34] proposed a location-based matrix factorization approach that made full use of context information and neighbors information. Chen *et al.* [35] proposed a fuzzy clustering-based approach to learn latent features of users and services, and designed a latent factor model to learn the features of each cluster.

Recently, deep learning methods have become increasingly popular, but there are not so many methods using deep learning techniques to predict QoS. As one of many deep learning methods, the auto-encoder has strong ability to learn latent features and has a simple network structure. The auto-encoder [36], [37] is an unsupervised neural network, and can encoder itself with its own latent factors. We can use the generalization ability of the auto-encoder to predict QoS. As a model-based learning algorithm, the auto-encoder is not easy to directly learn local features. In contrast, the neighborhood-based algorithm can exploit local features. In this paper, we focus on taking joint advantages of auto-encoder and neighborhood-based collaborative filtering.

III. THE PROPOSED SERVICE RECOMMENDATION FRAMEWORK

Our proposed method consists of two phases, which are shown in Figure 1.

Phase 1 (Global Features Learning): We propose an improved auto-encoder network in user side which can deal with sparse input by precomputing missing value. First, every user vector is extracted from initial user-service invocation matrix R_0, and a part of sparse user vectors are identified. Then the identified sparse vectors will be partially completed by pre-computing the missing value so that we can obtain better global features learning of auto-encoder. At last, all initial user vectors can be reconstructed by auto-encoder network.

Phase 2 (Local Features Learning): A new user-service matrix R_c is constructed by the reconstructed user vectors. Then, we can identify the similar neighbors of users or services using the extended similarity computation method. Finally, we make prediction for missing QoS values by a hybrid model.

IV. THE PROPOSED METHOD: QOS PREDICTION WITH AUTO-ENCODER

A. GLOBAL FEATURES LEARNING

Because there are a large of mobile edge services, a user usually only invokes a small part of services. As a result, a situation often happens with QoS records of more than 95% missing values in service recommendation. However, traditional layer-wise unsupervised pre-training does not work well in high sparsity. One reason is that the current neural networks do not fully exploit the representation power of deep architectures.

Currently, there is no standard way to tackle high sparsity for neural networks. In this paper, we propose an improved autoencoder network which can deal with sparse inputs by pre-computing an estimate of missing values. The architecture is shown in Figure 2. The proposed autoencoder network can capture the nonlinear hidden features from user-service invocation matrix. First, if the inputs are sparse vectors, we fill these vectors until the sparsity becomes lower than a threshold. Then a reconstructed vector $\boldsymbol{y}$ is obtained by the auto-encoder. Finally, the pre-completed vector $\boldsymbol{x}'$ and the reconstructed vector $\boldsymbol{y}$ are used to build reconstruction error training model. The pre-completed vector $\boldsymbol{x}'$ can restore the intrinsic structure of the original input vectors and reduce noise. Also it can make the latent features more stable and more robust, and help the autoencoder learn nonlinear features.

1) DEALING WITH SPARSE INPUTS

We deal with sparse inputs by pre-computing an estimate of missing QoS values in user vectors. The process of dealing with sparse inputs is shown in Figure 3.

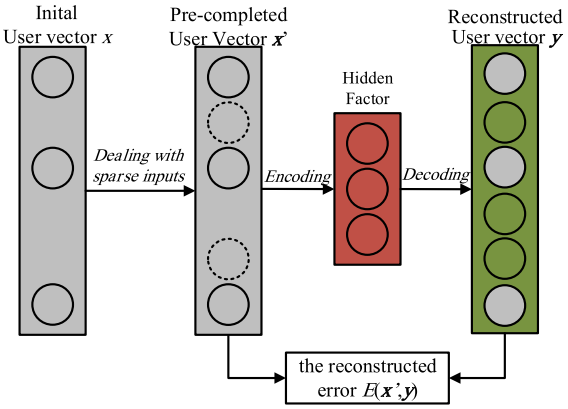


FIGURE 2. The improved auto-encoder with sparse inputs.

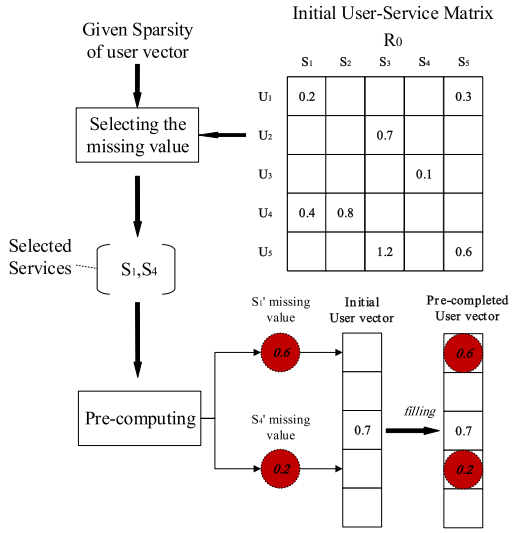


FIGURE 3. The process of dealing with sparse inputs.

First, we will pre-set the sparsity of user vectors x'. If the sparsity of the initial user vector x is smaller than the set sparsity, then we will randomly select some missing QoS values which will be precomputed and fill the initial user vector $\boldsymbol{x}$. Finally, we can obtain the pre-completed user vectors $\boldsymbol{x}'$ with pre-set sparsity.

In the paper, SVD (Singular Value Decomposition) model with bias term (BiasSVD for short) is used to pre-compute the selected missing values. As a model-based CF, BiasSVD performs well in high data sparsity [42]. BiasSVD generates user feature matrix and service feature matrix. The selected missing value computed in BiasSVD is

$$y_{ui} = \sum_{k=1}^{f} PukQik + bu + bi + \mu \tag{1}$$

where f is the number of latent features, P_{uk} is the user feature matrix, and Q_{ik} is the service feature matrix. b_u is the offset value of user u, b_i is the offset value of service i, and μ is the average of all known QoS values.

2) ENCODING

Encoding is the process that converts pre-completed user vector $\boldsymbol{x}'$ to hidden feature vector $\boldsymbol{h}$ using the mapping function f_θ. The nonlinear transformation is followed by an affine transformation.

$$h = f_\theta(x') = s(W_1 \times x' + b_1) \tag{2}$$

where the dimension of the pre-completed user vector $\boldsymbol{x}'$ is d, and the dimension of hidden feature vector $\boldsymbol{h}$ is d'. The parameter set $\theta = \{W_1, b_1\}$, and W_1 is a matrix of size $d' \times d$, and b_1 is the offset vector with dimension being d'. $s(.)$ is an activation function (e.g., sigmoid function or tanh function).

3) DECODING

Decoding is the process of reconverting the hidden feature vector $\boldsymbol{h}$ to d-dimensional reconstructed user vector $\boldsymbol{y}$ using mapping function $g_{\theta'}$. Usually, non-linear transformation is used after affine transformation.

$$y = g_{\theta'}(h) = s(W_2 \times h + b_2) \tag{3}$$

The parameter set $\theta' = \{W_2, b_2\}$.

The goal of the auto-encoder is to minimize the reconstructed error between the pre-completed user vector $\boldsymbol{x}'$ and the reconstructed vector $\boldsymbol{y}$, so the purpose of the auto-encoder training is to find the optimal parameters θ and θ' to minimize the reconstructed error in the training set. We use the squared error function $E(x, y) = ||x - y||^2$ to measure the reconstructed error. Auto-encoders often use an error backpropagation algorithm to train the model and update the parameters using gradient descent. The objective function as follow.

$$J_{\theta,\theta'} = \sum_{u=1}^{m} \left\| x'_u - y_u \right\|_2^2 + \frac{\lambda}{2} \left\| \mathrm{W}_1 \right\|_2^2 + \frac{\lambda}{2} \left\| \mathrm{W}_2 \right\|_2^2 \tag{4}$$

where vector $\boldsymbol{x}'_{\mathbf{u}}$ is the pre-completed user vector of user u, vector $\boldsymbol{y}_{\mathbf{u}}$ is the reconstructed user vector of user u, the λ is a parameter controlling regularization. The encoding parameter set θ and the decoding parameter set θ' are updated by gradient descent. The regularization terms of W_1 and W_2 are added to the objective function to avoid the overfitting problem.

B. LOCAL FEATURES LEARNING

In Phase 1, all user vectors as inputs are reconstructed by the improved autoencoder. The elements in reconstructed user vectors are all known, which means all missing values have been predicted. But the prediction accuracy can still need to be further improved. To do this, we aim to take advantage of the ability of local features learning in neighborhood-based CF. The key of neighbor-based CF methods is to accurately identify the similar neighbors using the user-service invocation records. In this section, a novel similarity computation method is proposed to find the similar neighbors by extending the traditional Euclidean distance.

Because the initial user-service invocation matrix R_0 is usually very sparse, it is usually difficult for existing methods to finish similar neighbor identification with high accuracy. Hence, in this paper, we use the reconstructed user-service invocation matrix R_c for neighbors identification. R_c is built

by the reconstructed user vectors so that it is a full matrix. However, the use of R_c is likely to lead to a situation that some users or services preferences in R_0 cannot be learned due to the reconstruction. Also, the accuracy of neighbors identification may be harmed. In order to address this problem, we propose two new concepts, which are common invocation factor and invocation frequency factor. The two factors are obtained according to the initial user-service invocation matrix R_0.

1) COMMON INVOCATION FACTOR

The common invocation factor of users (CIFU for short) is based on the number of all services commonly invoked by any two users. The common invocation factor of services (CIFS for short) is based on the number of all users invoking any two services. The intuition behind the two new concepts comes from the observation that two users are likely to share more similar invoking preference if they have commonly invoked more services before.

In order to compute CIFU, we first divide all services into three types: *com*, *any*, *non*.

The type *com* indicates that the services invoked by any two users, so the subset $L_{com,u,v}$ can be generated as follow.

$$L_{com,u,v} = M_u \cap M_v \tag{5}$$

where u and v represent two different users, and M_u and M_v represent the services set that are invoked by users u and v.

The type *any* indicates that the services are invoked by only one user, so the subset $L_{any,u,v}$ can be generated as follows.

$$L_{any,u,v} = (M_u \cap \overline{M_v}) \cup (\overline{M_u} \cap M_v) \tag{6}$$

The type *non* indicates that the services have never been invoked by any user, so the subset $L_{non,u,v}$ can be generated as follows.

$$L_{non,u,v} = \overline{M_u} \cap \overline{M_v} \tag{7}$$

The common invocation factor of users $CIFU_{\mathrm{tag,u,v}}$ can be computed as follows.

$$CIFU_{tag,u,v} = \sqrt{\frac{\|S\|}{\|L_{tag,u,v}\|}} \tag{8}$$

where *tag* is the type of services and S represents the set of all services.

Similarly, we can divide all users into three subsets $L_{com,i,j}$, $L_{any,i,j}$, $L_{non,i,j}$, where i and j represent two different services. The common invocation factor of services $CIFS_{\mathrm{tag,i,j}}$ can be computed as follows.

$$CIFS_{tag,i,j} = \sqrt{\frac{\|U\|}{\|L_{tag,i,j}\|}} \tag{9}$$

where U represents the set of all users.

2) INVOCATION FREQUENCY FACTOR

Inspired by the concept of *inverse document frequency* (IDF for short), we can have such an observation that two users are likely to be more similar, if they have invoked some services which are rarely invoked by other users. In order to describe such invocation preference, we put forward two concepts, i.e., the users invocation frequency factor and the services invocation frequency factor.

The users invocation frequency factor F_u is computed as

$$F_u = e^{-(x_u - m)} \tag{10}$$

where x_u indicates the number of services invoked by user u, and m represents the average number of services invoked by all users.

Similarly, the services invocation frequency factor F_i is computed as

$$F_i = e^{-(x_i - n)} \tag{11}$$

where x_i indicates the number of users that have invoked service i, and n represents the average number of users invoking services.

3) EXTENDED EUCLIDEAN DISTANCE

In this section, we integrate common invocation factor and invocation frequency factor into Euclidean distance to produce a new distance computation method.

The new distance is computed as follows.

$$d_{u,v} = \frac{\sum_{tag}^{\{com,any,non\}} \sum_{i}^{LN_{tag,u,v}} \left(q_{u,i} - q_{v,i}\right)^2 \cdot CIFU_{tag,u,v} \cdot F_i}{\left|N_{u,v}\right|} \tag{12}$$

where $CIFU_{tag,u,v}$ represents the common invocation factor of users and F_i represents the services invocation frequency factor. $q_{u,i}$ and $q_{v,i}$ represent the QoS value in the reconstructed matrix R_c. $LN_{tag,u,v} = L_{tag,u,v} \cap N_{u,v}$, and $L_{tag,u,v}$ represents the subset built by the initial user-service invocation matrix R_0. $N_{u,v}$ represents the services invoked commonly by u and v in the reconstructed matrix R_c.

Similarly, the distance between two services is computed as follows.

$$d_{i,j} = \frac{\sum_{tag}^{\{com,any,non\}} \sum_{u}^{LN_{tag,i,j}} \left(q_{u,i} - q_{u,j}\right)^2 \cdot CIFS_{tag,i,j} \cdot F_u}{\left|N_{i,j}\right|} \tag{13}$$

where $CIFS_{tag,i,j}$ represents the common invocation factor of services and F_u represents the users invocation frequency factor. $q_{u,i}$ and $q_{v,i}$ represent the QoS value in the reconstructed matrix R_c. $LN_{tag,i,j} = L_{tag,i,j} \cap N_{i,j}$, and $L_{tag,i,j}$ represents the subset bulit by the initial user-service invocation matrix R_0. $N_{i,j}$ represents the users that have invoked service i and service j in the reconstructed matrix R_c.

4) SIMILARITY CONPUTATION

In this section, a novel similarity computation method is proposed based on the new distance computation method.

The similarity between two users can be computed as follows.

$$S_{u,v} = T^r(d_{u,v}|\beta) = \left(\frac{1}{d_{u,v}+1}\right)^{\beta} \quad \beta > 1 \tag{14}$$

where $S_{u,v}$ represents the similarity between user u and user v, and T^r $(d_{u,v}|\beta)$ is the function which converts distance $d_{u,v}$ to similarity $S_{u,v}$.

Similar to the similarity between two users, the similarity between services is computed as follows.

$$S_{i,j} = T^r(d_{i,j}|\beta) = \left(\frac{1}{d_{i,j}+1}\right)^{\beta} \beta > 1 \tag{15}$$

where $S_{i,j}$ represents the similarity between service i and service j. $T^r(d_{i,j}|\beta)$ is the function which converts distance $d_{i,j}$ to similarity $S_{i,j}$.

Compared to the existing similarity computation method based on Euclidean distance, two important differences can be found in our proposed method. One is that we extend the traditional Euclidean distance. The other is that we improve the converting function by introducing the exponent β. Because β is always larger than 1, our method can take more advantage of distance than existing methods, and has better ability of similarity discrimination.

C. QoS PREDICTION

After we compute the similarity between users and similarity between services, we then select the most similar K users and services for QoS prediction. In detail, in user side, the prediction value is as follows.

$$\hat{q}^U_{u,i} = \frac{\sum\limits_{v \in NS_u \cap N_i} S_{u,v} \cdot q_{v,i}}{\sum\limits_{v \in NS_u \cap N_i} S_{u,v}} \tag{16}$$

where NS_u represents the neighbors set of user u, and $S_{u,v}$ represents the similarity between user u and user v. $q_{u,i}$ represents the QoS value in the initial user-service invocation matrix R_0, and N_i represents the users that have invoked service i. The proposed method is named as SAE-U (Sparse-dealing Auto-Encoder for Users).

In service side, the prediction value is computed as

$$\hat{q}^S_{u,i} = \frac{\sum\limits_{j \in NS_i \cap N_u} S_{i,j} \cdot q_{u,j}}{\sum\limits_{j \in NS_i \cap N_u} S_{i,j}} \tag{17}$$

where NS_i represents the neighbors set of service i, $S_{i,j}$ represents the similarity between service i and service j, and N_u represents the services invoked by user u. The proposed method is named as SAE-S (Sparse-dealing Auto-Encoder for Services).

To improve prediction accuracy, we propose a hybrid method that combines SAE-U and SAE-S, where the prediction value is computed as

$$\hat{q}^H_{u,i} = \alpha \times \hat{q}^S_{u,i} + (1\text{-}\alpha) \times \hat{q}^U_{u,i} \tag{18}$$

where parameter α is to control the weight of SAE-U and SAE-S. The hybrid method is named as SAE-H (Sparse-dealing Auto-Encoder for Hybrid).

V. EXPERIMENTS AND EVALUATION

A. DATASET

We conducted experiments in a public dataset WSDream for QoS prediction [38]. WSDream dataset contains 339 users, 5825 services, and a total of 1974,675 invocation records. There are two types of QoS properties, i.e., response time and throughput. We used response time as the evaluation property in this paper. With data statistics analysis, we find that all values of response time are between 0 and 20 seconds, and 91% of response time values are less than 2 seconds.

B. EVALUATION METRICS

To measure prediction accuracy, we use two metrics, including Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE). MAE is defined as the mean errors of the predicted values.

$$MAE = \frac{\sum\limits_{(u,i) \in T} \left|\hat{q}_{u,i} - q_{u,i}\right|}{|T|} \tag{19}$$

NMAE is given by

$$NMAE = \frac{\sum\limits_{(u,i) \in T} \left|\hat{q}_{u,i} - q_{u,i}\right|}{\sum\limits_{(u,i) \in T} q_{u,i}} \tag{20}$$

where T represents the test set, $q_{u,i}$ represents the real QoS value, and $\hat{q}_{u,i}$ represents the predicted QoS value. A smaller MAE and NMSE indicate a higher prediction accuracy.

C. EXPERIMENT SETTING

In a real-world environment, the user-service invocation matrix formed by service invocation records is of high sparsity, and users usually only invoke a small number of services. In order to truly reflect the real service invocation scenario, we randomly select a part of QoS records in WSDream dataset as the training set, and the remaining QoS records is as the test set. In this paper, there are five different sparsity levels to be used, which are 2.5%, 5%, 10%, 15% and 20%. We repeat ten times for each test case, and report the average result.

D. PERFORMANCE COMPARISON

In order to evaluate the performance of our method, we implemented several well-known QoS prediction methods, which are as follows.

1) UPCC (User-based PCC) [16]. The neighborhood-based CF that utilizes similarities between users, which

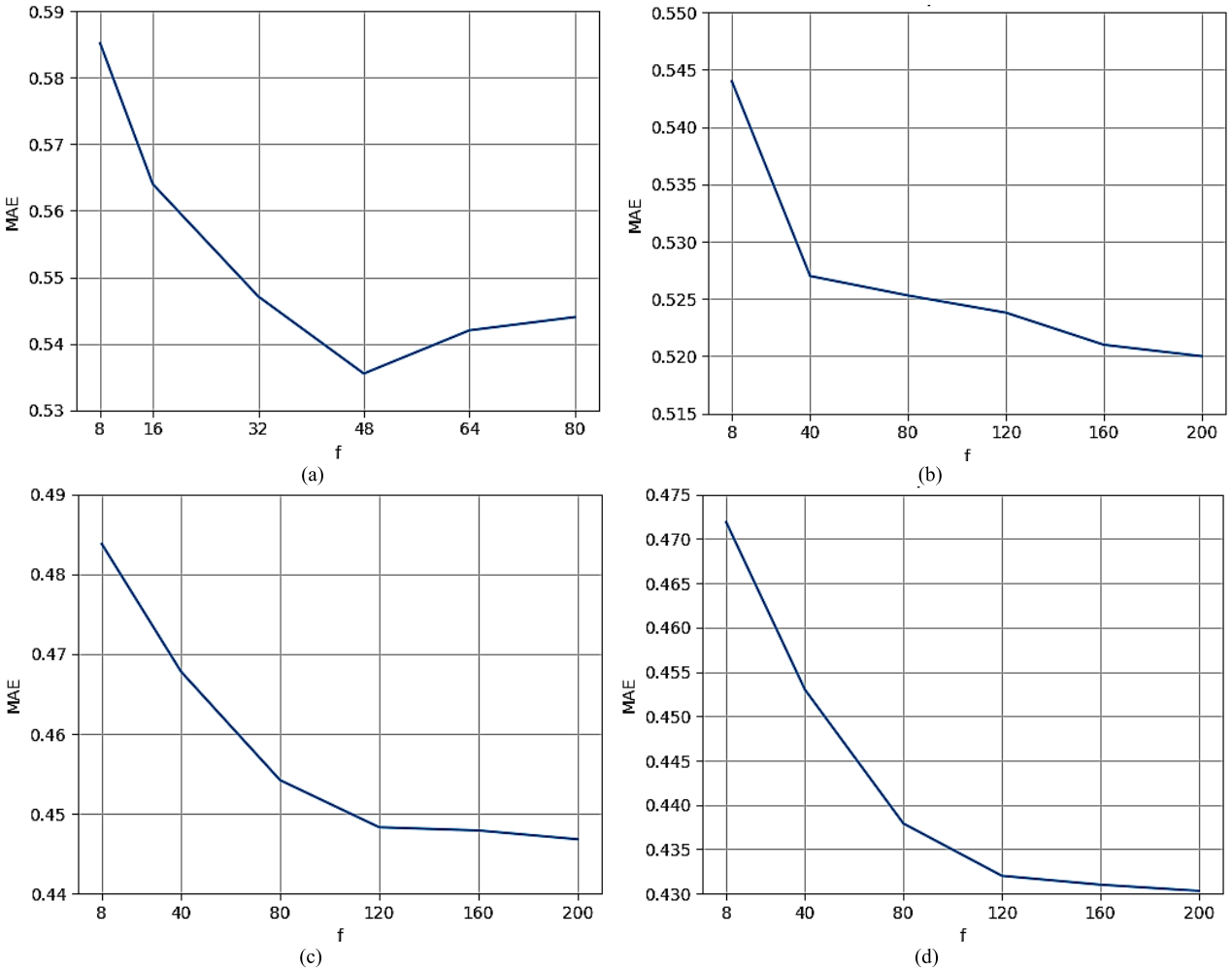


FIGURE 4. Impact of *f*. (a) Training set density = 2.5%. (b) Training set density = 5%. (c) Training set density = 15%. (d) Training set density = 20%.

are computed by Pearson correlation coefficient (PCC for short).

2) IPCC (Item-based PCC) [17]. The neighborhood-based CF that utilizes similarities between services, which are computed by Pearson correlation coefficient.
3) WSRec [38]. A hybrid approach that linearly combines the results of UPCC and IPCC
4) LFM (Latent Factor Model) [30]. LFM decomposes the user-service invocation matrix using dimensionality reduction to learn latent features and produces prediction results based on latent features.
5) CAP (Credibility-Aware Prediction) [39]. CAP is a credibility-aware QoS prediction method, which employs two-phase K-means clustering algorithm.
6) H-RBM (Hybrid Restricted Boltzman Machine) [40]. A two-stage QoS prediction algorithm based on restricted Boltzmann machine.
7) AutoEncoder [41]. A QoS prediction algorithm based on AutoEncoder.
8) SAE-U: The proposed method based on users' similarity computation, and is explained in Section IV (Eq. 16).
9) SAE-S: The proposed method based on services' similarity computation, and is also explained in Section IV (Eq. 17)
10) SAE-H: The proposed method based on hybrid similarity computation, and is explained in Section IV (Eq. 18).

E. IMPACT OF PARAMETERS

1) IMPACT OF *F*

The parameter f is the number of hidden units in model SAE-U. When the number of hidden units is small, it is difficult to learn latent features from known QoS records. But if the number of hidden units is too large, the model will become more complicated, and it is likely to learn features that are not of importance, which will lead to overfitting problem. Therefore, it is necessary to exploit the selection of the number of hidden units. Figure 4 shows the effect of different hidden units on prediction accuracy at different training set densities. A lower training set density indicates less training data.

It can be seen from Figure 4, in the case of training set density being 2.5%, the MAE value decreases first

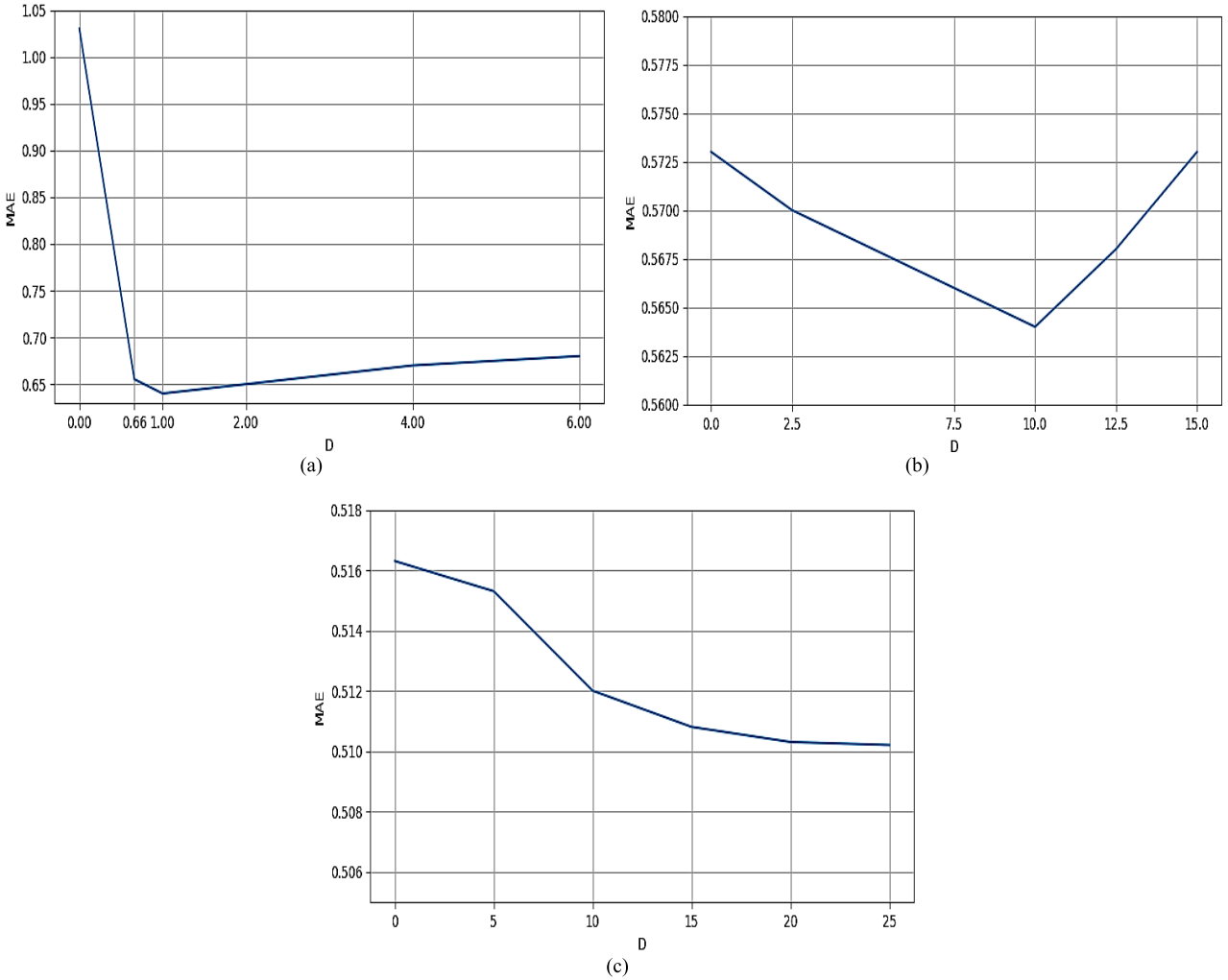


FIGURE 5. Impact of *D*. (a) Training set density = 1%. (b) Training set density = 2.5%. (c) Training set density = 5%.

TABLE 1. Accuracy comparison (a smaller value means a higher accuracy).

Model	Training Set Density (d) —— response time dataset									
	d = 2.5%		d = 5%		d = 10%		d = 15%		d = 20%	
	MAE	NMAE	MAE	NMAE	MAE	NMAE	MAE	NMAE	MAE	NMAE
UPCC	0.856	1.046	0.769	0.948	0.720	0.921	0.652	0.819	0.592	0.748
IPCC	0.836	1.021	0.732	0.903	0.712	0.911	0.678	0.852	0.650	0.821
WSRec	0.704	0.860	0.679	0.837	0.621	0.794	0.603	0.758	0.602	0.760
LFM	0.698	0.853	0.578	0.712	0.564	0.722	0.543	0.683	0.535	0.676
CAP	0.650	0.794	0.545	0.673	0.504	0.618	0.486	0.596	0.463	0.568
H-RBM	0.633	0.773	0.523	0.635	0.489	0.625	0.468	0.588	0.457	0.578
AutoEncoder	0.581	0.641	0.522	0.562	0.477	0.520	0.451	0.495	0.435	0.472
SAE-U	**0.552**	**0.618**	**0.518**	**0.552**	**0.453**	**0.494**	**0.421**	**0.461**	**0.408**	**0.439**
SAE-S	**0.528**	**0.580**	**0.482**	**0.510**	**0.397**	**0.436**	**0.359**	**0.389**	**0.351**	**0.375**
SAE-H	**0.511**	**0.576**	**0.471**	**0.507**	**0.384**	**0.424**	**0.354**	**0.378**	**0.344**	**0.368**

and then increases with the increase of the number of hidden units. At the cases of training set density being 5%, 15%, and 20% sparsity, the MAE value decreases gradually until convergence, and the turning point is all around 120. In this paper, f is set to 120 as default value.

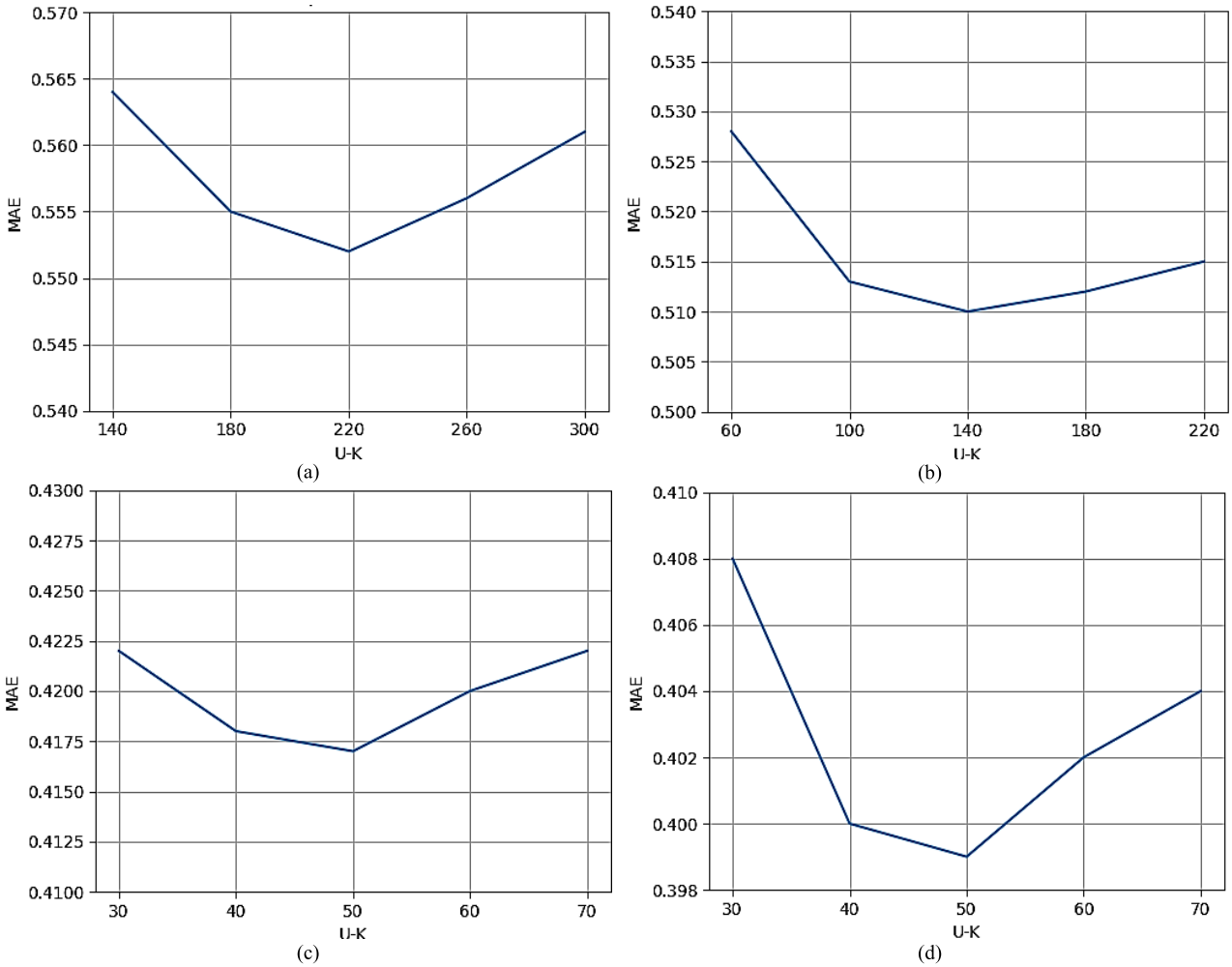


FIGURE 6. Impact of K in user side. (a) Training set density = 2.5%. (b) Training set density = 5%. (c) Training set density = 15%. (d) Training set density = 20%.

2) IMPACT OF D

The parameter D is set sparsity in dealing with sparse inputs (Section IV). Figure 5 shows the impact of set sparsity D on prediction of SAE-U model. Figure 5(a) presents the experimental result at 1% training set density, and the MAE value decreases quickly along with D rising. After the value of D continuously increases, the MAE value slowly increases, and the optimal value of D at 1% training set density is around 1. Figure 5(b) presents the experimental result at 2.5% training set density, and the optimal value of D at 2.5% training set density is around 10. Figure 5(c) presents the experimental result at 5% training set density, the optimal value of D at 5% training set density is around 20.

3) IMPACT OF K IN USER SIDE

The parameter K in user side is the number of similar users that are selected in user's collaborative filtering. Due to data sparsity problem, the actual number of similar users that should be selected is usually smaller than K. Figure 6 presents the impact of different values of K on prediction accuracy in different sparsity levels. The present experimental results are MAE results.

It can be seen from Figure 6, with the increase of K, the MAE result decreases first. The reason is that the number of similar users available for prediction is gradually increasing, and the prediction results of similar neighbors are more reliable. After K reaches a certain value, the MAE results start to rise because the number of similar users continues to increase, and a part of neighbors with low similarity join the prediction of QoS value. These less similar users harm the similarity of the overall similar neighbors. The best number of users at training set densities of 2.5%, 5%, 10%, 15%, and 20% is around 100 to 200. The optimal number of users becomes smaller as the training set density becomes higher. This is because there are fewer neighbors that can be used for prediction under low training set densities, such as 2.5% and 5%. As the density increases, the number of available neighbors increases, and the value of optimal K becomes smaller accordingly.

4) IMPACT OF K IN SERVICE SIDE

The parameter K in service side is the number of similar services that are selected for service-based collaborative filtering. Due to high data sparsity, the actual number of

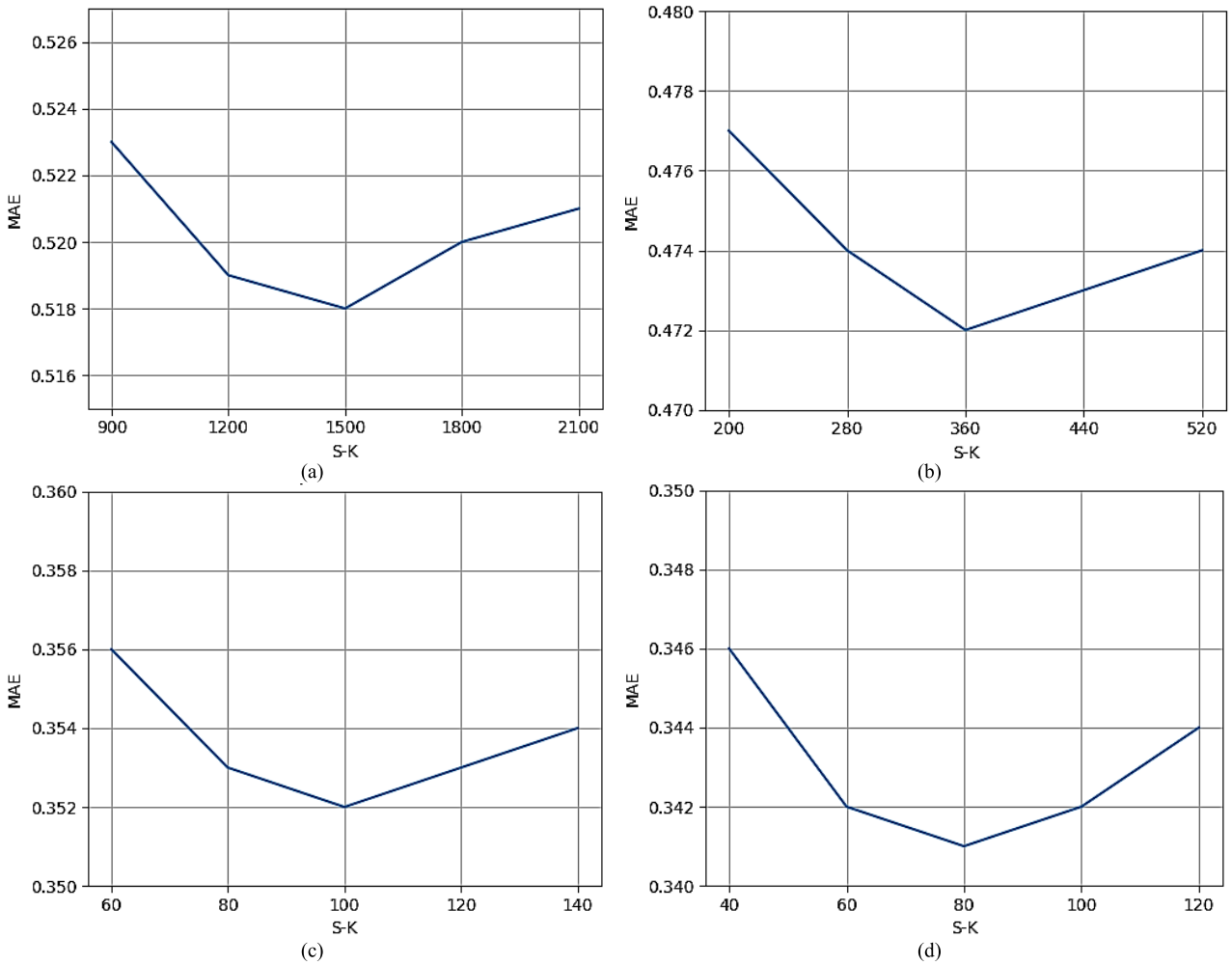


FIGURE 7. Impact of K in service side. (a) Training set density = 2.5%. (b) Training set density = 5%. (c) Training set density = 15%. (d) Training set density = 20%.

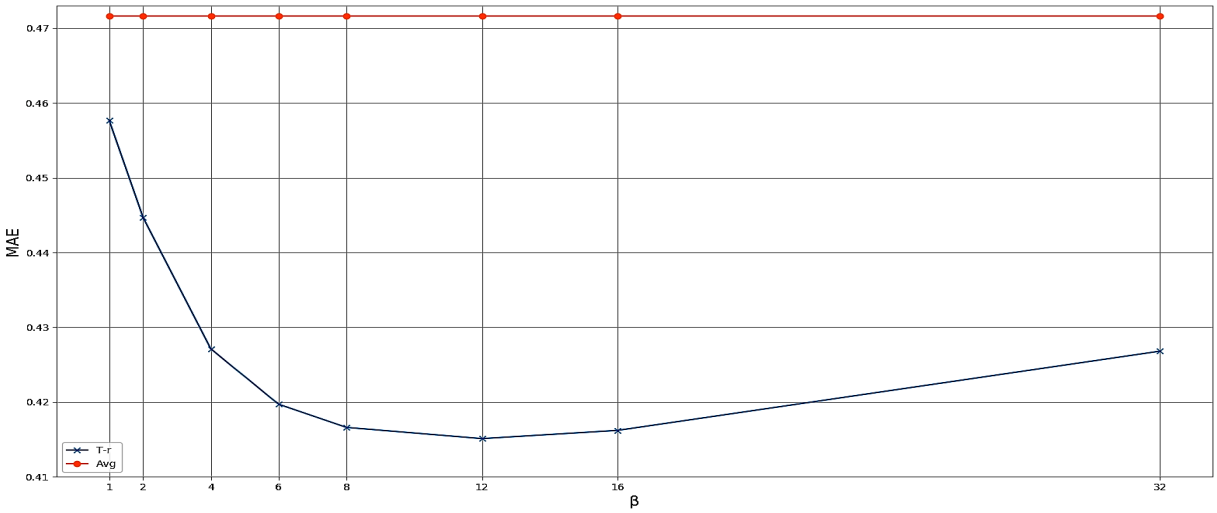


FIGURE 8. Impact of β.

similar services that should be selected is usually smaller than K. Figure 7 shows the impact of different values of K on prediction accuracy in different sparsity levels. The present experimental results are MAE values.

In Figure 7, with the increase of K, the MAE decreases first, and after K reaches a certain value, the MAE value starts to rise. The optimal number of services at training set densities of 2.5%, 5%, 10%, 15%, and 20% is also different.

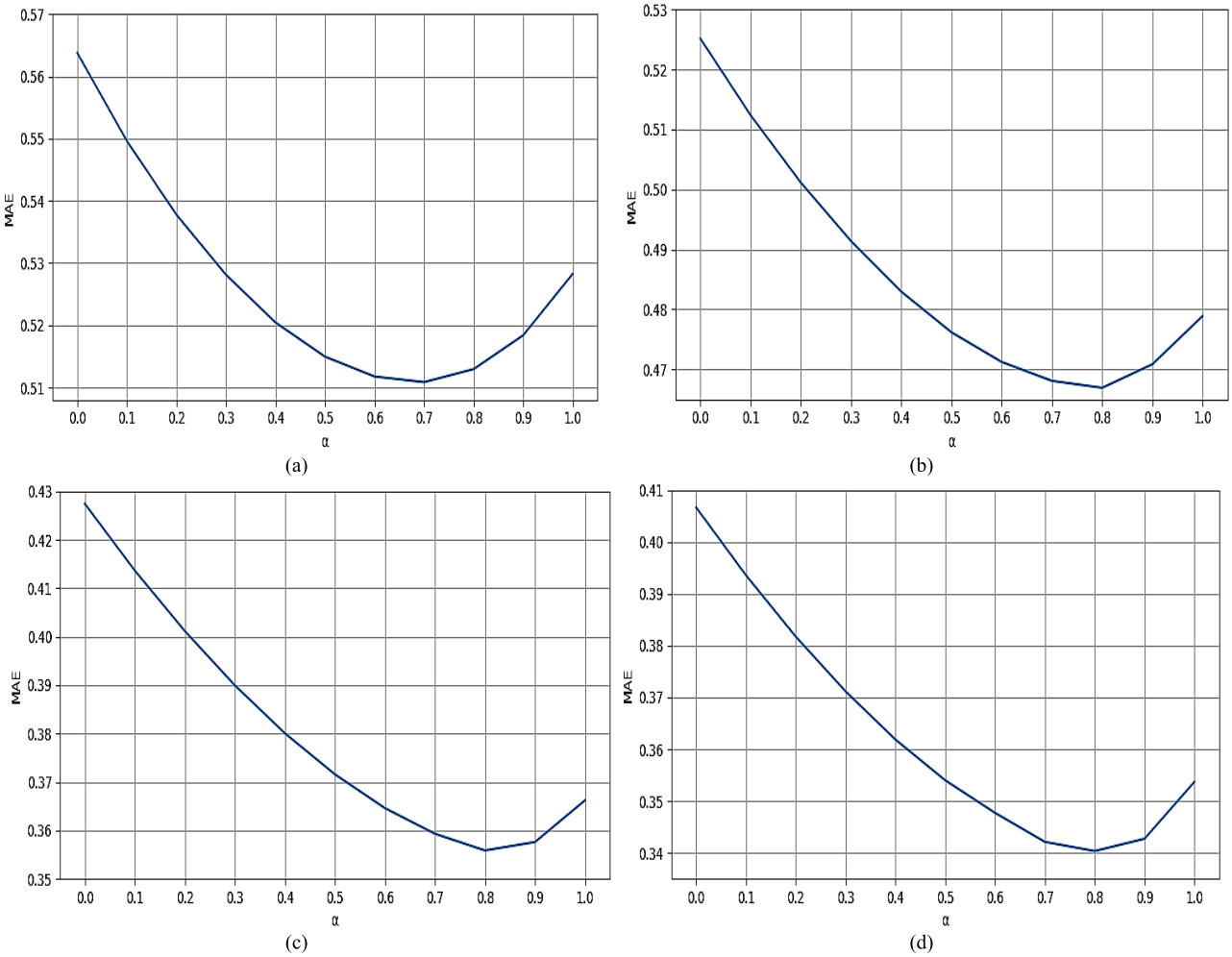


FIGURE 9. Impact of α in SAE-H. (a) Training set density = 2.5%. (b) Training set density = 5%. (c) Training set density = 15%. (d) Training set density = 20%.

The optimal number of services becomes smaller as the training set density increases.

5) IMPACT OF β

The parameter β is used to control the smoothness of the mapping function in similarity mapping. In this section, we evaluate the impact of parameter β on prediction accuracy of SAE-U method. The training set density is 20%.

In Figure 8, the *T-r* curve represents the $T^{\mathrm{r}}(d|\beta)$ mapping function, and *Avg* curve represents the prediction method that utilizes the mean of all neighbors. The MAE values of the *T-r* curve decrease first and then increase, while the curve of Avg remains unchanged.

As β increases, the MAE results of $T^{\mathrm{r}}(d|\beta)$ decrease. It is because QoS prediction requires more neighbors with lower distances. After β increases to a certain value, the MAE results of $T^{\mathrm{r}}(d|\beta)$ begin to increase. It is because QoS prediction overuses the historical records of neighbors with lower distances.

6) IMPACT OF α

The parameter α is used to control the weight of SAE-U and SAE-S in hybrid model. The parameter α is in range of 0 to 1. The experimental results are shown in Figure 9. From the experimental results in Figure 9, it can be seen that the most suitable value of parameter α falls in range of 0.7 to 0.8.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a method combining auto-encoder and collaborative filtering method. The proposed method uses the auto-encoder improved by the substitution strategy to obtain nonlinear latent features, and further recovers a part of unknown QoS values to alleviate the high sparsity problem. Then, we improve the traditional method of similarity computation obtaining more accurate neighbors and improve the contribution of real neighbors. Finally, we propose two models to produce the final QoS prediction results from user side and service side respectively. The results of extensive experiments on a real-world dataset verify the effectiveness of our method.

In future work, on the one hand, we plan to investigate other neural network models as basic methods, such as convolutional neural network (CNN). On the other hand, we plan to study the function and mechanism of context information in QoS prediction.

REFERENCES

[1] S. Deng, Z. Xiang, J. Yin, J. Taheri, and A. Y. Zomaya, "Composition-driven IoT service provisioning in distributed edges," *IEEE Access*, vol. 6, pp. 54258–54269, 2018.

[2] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[3] S. Deng, H. Wu, W. Tan, Z. Xiang, and Z. Wu, "Mobile service selection for composition: An energy consumption perspective," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 3, pp. 1478–1490, Jul. 2017.

[4] G. Q. Xu, Y. Zhang, A. K. Sangaiah, X. H. Li, A. Castiglione, and X. Zheng, "CSP-E2: An abuse-free contract signing protocol with low-storage TTP for energy-efficient electronic transaction ecosystems," *Inf. Sci.*, vol. 476, pp. 505–515, Feb. 2019.

[5] H. Gao, W. Huang, X. Yang, Y. Duan, and Y. Yin, "Toward service selection for workflow reconfiguration: An interface-based computing solution," *Future Gener. Comput. Syst.*, vol. 87, pp. 298–311, Oct. 2018.

[6] X. Zeng, G. Xu, X. Zheng, Y. Xiang, and W. Zhou, "E-AUA: An efficient anonymous user authentication protocol for mobile IoT," *IEEE Internet Things J.*, to be published.

[7] S. Deng, L. Huang, J. Taheri, J. Yin, M. Zhou, and A. Y. Zomaya, "Mobility-aware service composition in mobile communities," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 555–568, Mar. 2017.

[8] T. Yu, Y. Zhang, and K.-Y. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," *ACM Trans. Web*, vol. 1, no. 1, 2007, Art. no. 6.

[9] H. Gao, Y. Duan, H. Miao, and Y. Yin, "An approach to data consistency checking for the dynamic replacement of service process," *IEEE Access*, vol. 5, pp. 11700–11711, 2017.

[10] H. Gao, H. Miao, L. Liu, J. Kai, and K. Zhao, "Automated quantitative verification for service-based system design: A visualization transform tool perspective," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 28, no. 10, pp. 1369–1397, 2018.

[11] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for Web services composition," *IEEE Trans. Softw. Eng.*, vol. 30, no. 5, pp. 311–327, May 2004.

[12] Y. Liu, A. H. Ngu, and L. Z. Zeng, "QoS computation and policing in dynamic Web service selection," in *Proc. 13th Int. World Wide Web Conf. Alternate Track Papers Posters*, New York, NY, USA, 2004, pp. 66–73.

[13] M. Andriy and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, Toronto, ON, Canada, 2008, pp. 1257–1264.

[14] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, Perth, WA, Australia, 2017, pp. 173–182.

[15] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 14th Conf. Uncertainty Artif. Intell.*, San Francisco, CA, USA, 1998, pp. 43–52.

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," *ACM Conf. Comput. Supported Cooperat. Work*, Chapel Hill, NC, USA, 1994, pp. 175–186.

[17] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. Int. World Wide Web Conf.*, Hong Kong, 2001, pp. 285–295.

[18] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware Web service recommendation by collaborative filtering," *IEEE Trans. Services Comput.*, vol. 4, no. 2, pp. 140–152, Apr./Jun. 2011.

[19] Y. Yin, W. Xu, Y. Xu, H. Li, and L. Yu, "Collaborative QoS prediction for mobile service with data filtering and slopeone model," *Mobile Inf. Syst.*, vol. 2017, pp. 1–14, Jun. 2017.

[20] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and M. Hong, "Personalized QoS prediction for Web services via collaborative filtering," in *Proc. 14th IEEE Int. Conf. Web Services*, Salt Lake City, UT, USA, Jul. 2007, pp. 439–446.

[21] X. Wu, B. Cheng, and J. Chen, "Collaborative filtering service recommendation based on a novel similarity computation method," *IEEE Trans. Services Comput.*, vol. 10, no. 3, pp. 352–365, May 2017.

[22] G. Zou, M. Jiang, S. Niu, H. Wu, S. Pang, and Y. Gan, "QoS-aware Web service recommendation with reinforced collaborative filtering," in *Proc. Int. Conf. Service-Oriented Comput.*, Hangzhou, China, 2018, pp. 430–445.

[23] J. Li, J. Wang, Q. Sun, and A. Zhou, "Temporal influences-aware collaborative filtering for QoS-based service recommendation," in *Proc. IEEE Int. Conf. Services Comput.*, Honolulu, HI, USA, Jun. 2017, pp. 471–474.

[24] Y. Yin, F. Yu, Y. Xu, L. Yu, and J. Mu, "Network location-aware service recommendation with random walk in cyber-physical systems," *Sensors*, vol. 17, no. 9, p. 2059, 2017.

[25] M. Grčar, D. Mladenič, B. Fortuna, and M. Grobelnik, "Data sparsity issues in the collaborative filtering framework," in *Proc. Int. Workshop Knowl. Discovery Web*, Berlin, Germany, 2005, pp. 58–76.

[26] G.-R. Xue *et al.*, "Scalable collaborative filtering using cluster-based smoothing," in *Proc. 28th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Salvador, Brazil, 2005, pp. 114–121.

[27] Y. Yin, Y. Xu, W. Xu, M. Gao, L. Yu, and Y. Pei, "Collaborative service selection via ensemble learning in mixed mobile network environments," *Entropy*, vol. 19, no. 7, p. 358, 2017.

[28] P. Singla and M. Richardson, "Yes, there is a correlation:—From social networks to personal behavior on the Web," in *Proc. Int. Conf. World Wide Web*, Beijing, China, 2008, pp. 655–664.

[29] S. Zhang, W. Wang, J. Ford, F. Makedon, and J. Pearlman, "Using singular value decomposition approximation for collaborative filtering," in *Proc. IEEE Int. Conf. E-Commerce Technol.*, Munich, Germany, Jul. 2005, pp. 257–264.

[30] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[31] Y. Yin, L. Chen, Y. Xu, and J. Wan, "Location-aware service recommendation with enhanced probabilistic matrix factorization," *IEEE Access*, vol. 6, pp. 62815–62825, 2018.

[32] Y. Yin, S. Aihua, G. Min, X. Yueshen, and W. Shuoping, "QoS prediction for Web service recommendation with network location-aware neighbor selection," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 26, no. 4, pp. 611–632, 2016.

[33] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative Web service QoS prediction via neighborhood integrated matrix factorization," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 289–299, Jul. 2012.

[34] Y. Yang, Z. Zheng, X. Niu, M. Tang, Y. Lu, and X. Liao, "A location-based factorization machine model for Web service QoS prediction," *IEEE Trans. Services Comput.*, to be published.

[35] S. Chen, Y. Peng, H. Mi, C. Wang, and Z. Huang, "A cluster feature based approach for QoS prediction in Web service recommendation," in *Proc. IEEE Symp. Service-Oriented System Eng.*, Bamberg, Germany, Mar. 2018, pp. 246–251.

[36] G. Sampson, D. Rumelhart, J. McClelland, and T. P. R. Group, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*. Cambridge, MA, USA: MIT Press, 1987, p. 871.

[37] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

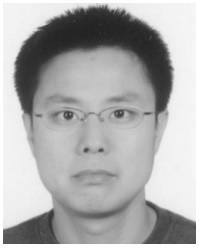
[38] Z. Zheng, H. Ma, M. R. Lyu, and L. King, "WSRec: A collaborative filtering based Web service recommender system," in *Proc. IEEE Int. Conf. Web Services*, Los Angeles, CA, USA, Jul. 2009 pp. 437–444.

[39] C. Wu, W. Qiu, Z. Zheng, X. Wang, and X. Yang, "QoS prediction of Web services based on two-phase K-means clustering," in *Proc. IEEE Int. Conf. Web Services*, New York, NY, USA, Jun./Jul. 2015, pp. 161–168.

[40] L. Chen, Y. Yin, Y. Xu, L. Chen, and J. Wan, "Two-phase Web service QoS prediction with restricted Boltzmann machine," in *Proc. Int. Conf. Service-Oriented Comput.*, Hangzhou, China, 2018, pp. 592–600.

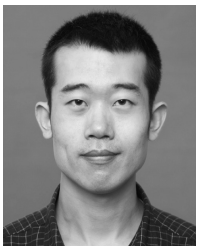
[41] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, Florence, Italy, 2015, pp. 111–112.

[42] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proc. KDD Cup Workshop*, San Jos, CA, USA, 2007, pp. 5–8.



YUYU YIN received the Ph.D. degree in computer science from Zhejiang University, in 2010. He is currently an Associate Professor with the College of Computer, Hangzhou Dianzi University, and is engaged as a Supervisor of the Master's Students with the School of Computer Engineering and Science, Shanghai University, Shanghai, China. During the past ten years, he has published more than 40 papers in journals and refereed conferences, such as *Sensors*, *Entropy*, IJSEKE, *Mobile Information Systems*, ICWS, and SEKE. He organized more than 10 international conferences and workshops, such as FMSC2011–2017 and DISA2012/2017–2018. His research interests include service computing, cloud computing, and business process management. He is a member of the China Computer Federation (CCF) and CCF Service Computing Technical Committee Member. He was a Guest Editor of the *Journal of Information Science and Engineering* and the *International Journal of Software Engineering and Knowledge Engineering*, and as a Reviewers of the IEEE TRANSACTION ON INDUSTRY INFORMATICS, the *Journal of Database Management*, and *Future Generation Computer Systems*.

WEIPENG ZHANG received the bachelor's degree in computer science from Hangzhou Dianzi University. She is currently pursuing the master's degree in computer science from Hangzhou Dianzi University and also from the Key Laboratory of Complex Systems Modeling and Simulation, Ministry of Education, Zhejiang, China. Her research interests include recommendation systems and machine learning.



YUESHEN XU received the Ph.D. degree from Zhejiang University. He was a Co-Trainer of the Ph.D. student with the University of Illinois at Chicago. He is currently a Lecturer with the School of Computer Science and Technology, Xidian University. He has published more than 20 papers in international journals or conferences, such as ESWA, EAAI, WISE, WAIM, and *Sensors*. His research interests include recommender systems, text mining, and service computing. He is a member of ACM and CCF. He is the Reviewer of several journals and conferences, including IEEE TSC, KBS, the IEEE ACCESS, JNCA, *Neurocomputing*, and ICDCS.

HE ZHANG received the bachelor's degree in software engineering from Xidian University, where she is currently pursuing the master's degree with the School of Computer Science and Technology. Her research interests include recommender systems, service computing, and text mining.

ZHIDA MAI received the bachelor's degree in automation from Xidian University. He is currently a Senior Technical Manager and an Applied Researcher with Xanten Guangdong Development Co., Ltd. He has managed several large software projects that develop information systems based on service-oriented architecture. He has also participated in several practical research programs, which aim to study the application of recommender systems in real-world e-commercial platforms.

LIFENG YU received the master's degree from the Department of Computer Science, Zhejiang University, Hangzhou, China, in 2006. He is currently a Senior Engineer with Hithink RoyalFlush Information Network Co., Ltd. He has published more than ten papers in related international conferences and journals. His current research interests include cloud computing, computer networks, and data processing.