TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 19/20 pp897–910
DOI: 10.26599/TST.2023.9010050
Volume 29, Number 3, June 2024

A Deep Neural Collaborative Filtering Based Service Recommendation Method with Multi-Source Data for Smart Cloud-Edge Collaboration Applications

Wenmin Lin, Min Zhu, Xinyi Zhou, Ruowei Zhang, Xiaoran Zhao, Shigen Shen*, and Lu Sun

Abstract: Service recommendation provides an effective solution to extract valuable information from the huge and ever-increasing volume of big data generated by the large cardinality of user devices. However, the distributed and rich multi-source big data resources raise challenges to the centralized cloud-based data storage and value mining approaches in terms of economic cost and effective service recommendation methods. In view of these challenges, we propose a deep neural collaborative filtering based service recommendation method with multi-source data (i.e., NCF-MS) in this paper, which adopts the cloud-edge collaboration computing paradigm to build recommendation model. More specifically, the Stacked Denoising Auto Encoder (SDAE) module is adopted to extract user/service features from auxiliary user profiles and service attributes. The Multiple Layer Perceptron (MLP) module is adopted to integrate the auxiliary user/service features to train the recommendation model. Finally, we evaluate the effectiveness of the NCF-MS method on three public datasets. The experimental results show that our proposed method achieves better performance than existing methods.

Key words: deep neural collaborative filtering; multi-source data; cloud-edge collaboration application; stacked denoising auto encoder; multiple layer perceptron

1 Introduction

Nowadays, the explosive development of Internet-of-Things (IoT) has spawned a huge and ever-increasing volume of data from a large cardinality of user devices[1]. The massive and sparse big data resources make it difficult for users to obtain the information that meets their requirements accurately, which causes the problem of "information overload"[2, 3]. As an effective method to address the "information overload" problem, service recommendation approaches[4, 5] could extract user preferences from the big data resources, which could greatly help to reduce the data load and improve user experience. In practice, service recommendation applications have been widely adopted in multiple areas, including news recommendation[6], travel recommendation[7], transportation recommendation[8], IoT device placement recommendation[9], and Web API recommendation[10, 11], to name a few.

The distributed and rich multi-source big data resources raise challenges to the centralized cloud-

• Wenmin Lin, Xinyi Zhou, and Lu Sun are with Alibaba Business School, Hangzhou Normal University, Hangzhou 311121, China. E-mail: wenmin.lin@hznu.edu.cn; zhouxinyigloria@foxmail.com; sunlu@hznu.edu.cn.
• Min Zhu is with Blockchain Laboratory of Agricultural Vegetables, Weifang University of Science and Technology, Shouguang 262700, China. E-mail: zhumin@wfust.edu.cn.
• Ruowei Zhang and Xiaoran Zhao are with School of Computer Science, Qufu Normal University, Rizhao 276827, China. E-mail: ruoweizhang.z@gmail.com; xxiaoranzhao@163.com.
• Shigen Shen is with School of Information Engineering, Huzhou University, Huzhou 313000, China. E-mail: shigens@zjhu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2023-04-04; revised: 2023-05-11; accepted: 2023-05-24

based data storage and value mining approach in terms of economic cost and effective service recommendation methods[12–15]. On one hand, the centralized cloud-based service recommendation methods require the big data to be transferred to the cloud center for further data analysis and recommendation computing, which will bring huge data transmission cost and time cost. Moreover, the cloud server should reserve huge storage space to maintain the big data for analysis. On the other hand, the multi-source data generated by distributed user devices contain various data information, such as user profiles, service attributes, user-service interaction records, making it a challenge to provide accurate recommendation decisions.

Fortunately, the recently emerged cloud-edge collaboration computing paradigm[16] provides a possible solution to enable partial computing and storage resources to be closer to user device side, which could greatly save the cost to build service recommendation models over the huge volume of big data resources. With a typical cloud-edge collaboration architecture, some pre-processing work with low resource consumption could be accomplished on edge servers; while the main work to train service recommendation model with high resource consumption could be executed on cloud servers. Therefore, we could implement the service recommendation method via the popular cloud-edge collaboration computing paradigm.

Moreover, the multi-source data in format of user profiles, service attributes, as well as user-service interaction records produced by distributed user devices poses challenge to produce accurate service recommendation results. Due to the powerful fitting ability, deep learning has been widely adopted in service recommendation, which could extract latent user/service features from various multi-source data to obtain accurate recommendation decisions. The most representative deep learning based service recommendation method is Neural Collaborative Filtering (NCF) proposed by He et al.[17], which adopts the deep neural network to capture the non-linearity relationship between users and services from the implicit user/service interaction records. However, the native NCF method faces data sparsity issue[18, 19], which is caused by the missing data records in the user/service interaction records. How to fuse the multi-source data into the NCF method to improve recommendation accuracy is a challenging problem.

Based on aforementioned observations, in this paper, we propose a deep neural collaborative filtering based service recommendation method, i.e., NCF-MS, which fuses multiple data sources into the native NCF model to improve the accuracy of recommendation decisions. Moreover, we adopt the popular cloud-edge collaboration computing paradigm to implement the NCF-MS model in a distributed manner. In general, the main contributions of our work are summarized as follows:

- We propose the NCF-MS method, which fuses multi-source data into the native NCF model to overcome the sparsity issue with typical NCF model. The user profiles and service attributes are adopted as auxiliary information to overcome the sparsity issue caused by missing user-service interaction records, so as to improve recommendation accuracy.
- The Stacked Denoising Auto-Encoder (SDAE) component is adopted to extract user and service latent features from the multi-source auxiliary user profiles and service attribute information. The Multiple Layer Perceptron (MLP) component is adopted to train the recommendation model with the extracted latent user/service features.
- The popular cloud-edge collaboration architecture is adopted to implement the proposed NCF-MS model, where the SDAE module is deployed on edge servers to extract latent user/service features, and the MLP module is deployed on cloud server to train the recommendation model.
- A set of simulations on three public datasets are implemented to evaluate the performance of the NCF-MS model.

The reminder of this paper is organized as follows: Section 2 reviews existing works on cloud-edge collaboration architecture based service recommendation models, deep collaborative filtering models, and recommendation model with multi-source data, respectively. Section 3 introduces preliminaries on SDAE, MLP, and NCF model, respectively. Section 4 introduces the system framework for the NCF-MS recommendation model. Section 5 describes our NCF-MS method in detail. Section 6 evaluates the NCF-MF method via a set of experiments on three public datasets. Section 7 concludes this paper and discusses future work.

2 Related Work

Since we explore how to implement a deep collaborative filtering based service recommendation method on multi-source data over the cloud-edge collaboration architecture in this paper, we are going to review existing works on cloud-edge collaboration architecture based service recommendation models, deep collaborative filtering models, and recommendation methods with multi-source data, respectively.

2.1 Cloud-edge collaboration architecture based service recommendation models

Cloud-edge collaboration architecture is a popular system-level computing architecture, where the computing and storage resources of the edge servers are fully taken advantage of to reduce data transfer cost between user devices and the centralized cloud server. Therefore, it is treated as a beneficial complement to the traditional cloud-based centralized systems, which could reduce the load of a cloud platform and improve the execution efficiency of distributed tasks significantly.

However, although the cloud-edge collaboration architecture has been proposed and achieved wide attention from both industry and academia, existing works on adopting this architecture to build distributed service recommendation models are relatively scarce. Most of existing works on cloud-edge collaboration architecture is about how to optimize the offloading process to achieve optimal resource allocation and utilization. For example, Zhou et al.[20] proposed a smart routing method, which uses network coding combined with opportunistic routing to improve energy efficiency in wireless IoT infrastructure, to reduce communication bandwidth and data latency between IoT edge servers and the centralized cloud server. Jia et al.[21] studied a resource optimization method based on CNN model (CroApp) to optimize the resource consumption within the CNN model and cloud-edge collaboration enabled applications.

Moreover, exploring how to build application over the cloud-edge collaboration architecture also achieved wide attention. For example, Xu et al.[22] proposed DisCOV, which is to detect COVID-19 from CXR images with edge-cloud collaboration. Nguyen[23] explored to build recommendation systems over cloud-edge collaboration framework. In his design, the native SVD method is optimized as a distributed method to fit the cloud-edge collaboration framework. However, he only takes the user-service interaction records as input for the SVD recommendation model, which does not consider how to take advantage of the multi-source data such as user profiles and item attributes to improve recommendation accuracy.

In this work, we explore how to build a service recommendation model with multi-source data over the cloud-edge collaboration architecture. In our design, we focus on splitting the working process of the service recommendation model into two major parts: feature extraction process as well as model training process. The feature extraction process is deployed on edge servers; while the model training process is deployed on the cloud server. Therefore, the resources on edge servers could be taken fully advantage of to improve resource utilization of edge servers, as well as reduce data transfer cost between the edge servers and the cloud servers.

2.2 Deep collaborative filtering models (CFM)

With the continuously development of deep learning (DL) techniques, DL has been widely adopted in various areas and achieves outstanding performances. DL could map multiple and heterogeneous data sources into an identical latent space, and capture accurate data features from the data samples. Therefore, deep learning technique has been widely adopted to extract user/service features in recommendation models.

For example, the auto encoder is a typical feature extraction tool of deep learning techniques. It can encode the data samples and learn latent and low-dimensional data features by the encoding and decoding operations. Li and He[24] proposesd an AutoRec model to capture both user and service latent features to lower the reconstruction losses. Ortega et al.[25] extended AutoRec by adding denosing functions to implement the CFM model, which also fuse auxiliary information to improve recommendation accuracy.

Moreover, there are quite a few works adopting the deep learning techniques to learn the interactions between the user and service features. For example, He et al.[17] proposed to adopt the deep neural network to capture the complex linear and non-linear relationship between user and service latent features. Salakhutdinov et al.[26] firstly adopted the deep learning technique to handle the recommendation prediction problem, and

design a Restricted Boltzman Machine (RBM) based collaborative recommendation model. Vincent et al.[27] extended the model proposed by Salakhutdinov, which capture the similarity and randomness of user's rating scores.

In this work, we take advantage of the powerful fitting ability of deep learning tools to build the multi-source data driven service recommendation model, where the SDAE module is deployed to extract latent user/service features from auxiliary information including user profiles and service attributes; while the MLP module is adopted to fuse multi-source user/service features and train the deep neural collaborative filtering model to predict user/service interaction score.

2.3 Recommendation models with multi-source data

Fusing multi-source auxiliary data into the collaborative recommendation model is one of the most effective solutions to address the data sparsity issue with native collaborative filtering methods. Existing multi-source data driven recommendation models can be divided into two categories: social network based recommendations and text comment based recommendations.

Social network based recommendation methods normally build a social network between users according to their preferences. They believe that users with similar interests tend to accept the services consumed by their friends. Typical social network based recommendation methods include SoRec[28], SocialMF[29], TrustWalker[26], and TrustSVD[30], to name a few. Among existing social network based recommendation methods, one typical solution is to fuse the social information into the neighbourhood based recommendations. For example, TrustWalker considers both the user-service rating scores as well as the comments to recommend similar services. Another solution is to fuse the social information into the model based recommendations. For example, the SocialMF[29] and SoReg[31] consider both the matrix factorization model as all as the trustworthiness to make the recommendation decisions.

Text comment based recommendations, on the other hand, fuse user comments into the collaborative recommendation models to make recommendation decisions. In practice, text comments normally contain user preference and service features, which could be adopted to build user profiles and service profiles. The challenge to fuse text comment into collaborative recommendation models is to extract the user/service features from the non-structural texts. Both typical machine learning methods and deep learning techniques could be adopted to extract user/service features from the text information. Typical text comment based recommendation models include DeepCoNN[32], NARRE[33], and SentiRec[34], for which the common solution is to transfer text into a word vector, then extract user/service feature from text to build user/service profiles.

Based on aforementioned discussion, fusing multi-source data into recommendation model is a common solution to address the data sparsity issue with typical collaborative filtering methods. However, how to consider both the multi-source data and the non-linearity relationship between user and service latent features still need further exploration. In view of this observation, in this work, we propose the NCF-MS method to fuse multi-source data into the deep neural network to capture more accurate user preference and user/service interaction relationships to improve recommendation accuracy.

3 Preliminary

3.1 SDAE

SDAE is an extension of the stacked autoencoder, which is a self-supervising model to extract an unique latent representation for each input data sample. In SDAE, denoising autoencoders are stacked to form a deep network, which is trained to obtain the latent representation for each data sample by minimizing the error of reconstructing the input. As shown in Fig. 1, the SDAE architecture consists of three main layers: the input layer, the latent representation layer, and the

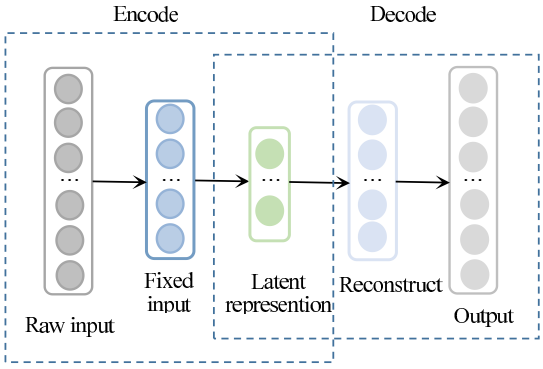


Fig. 1 Stacked denoising auto encoder model.

output layer. The raw input is de-noised to generate the input data sample, which will be encoded via an autoencoder to obtain the latent representation. Furthermore, the latent representation is reconstructed by the decoding layer to reconstruct the data sample. The training process is to learn the parameters of each autoencoder layer, which will help to extract the latent representation of each data sample.

For example, the input data sample x_i is weighted with parameters (W,b), which will be further mapped with an activation function (e.g., Sigmoid) to produce the latent representation x_l. Moreover, x_l will be mapped via reverse weighting process to produce the output x_o. The training process is to learn the parameters (W,b) iteratively, to minimize the difference between the input x_i and the output x_o. After the training process, the latent representation layer plays the role of encoder to produce the latent feature of the input data sample. The objective function of the SDAE model could be depicted by Formula (1):

$$\min \|x_i - x_o\|^2 + \lambda \Sigma \|W\|^2 \tag{1}$$

where λ is a regularization parameter and $\|\cdot\|$ denotes the Frobenius norm.

In our proposed NCF-MS service recommendation model, we adopt the SDAE module as a feature extraction tool to extract latent user/service features from the auxiliary multi-source data of user profiles and service attributes. Furthermore, the latent user/service features will be adopted as input to train the service recommendation model with an MLP module. The details will be introduced in Section 5.

3.2 MLP

MLP could be treated as a feed forward neural network. Similar to typical neural networks, MLP also has input layer, latent layer as well as output layer. The middle latent layer could be a single layer or multiple layers. As shown in Fig. 2, in MLP, all the neurons are fully connected to the neurons in the next layer. Suppose the initial input of the MLP network is x, the weight of the input layer to the first latent layer is $W^{(1)}$, and the bias is $b^{(1)}$. Therefore, the input of the first latent layer a_1 is

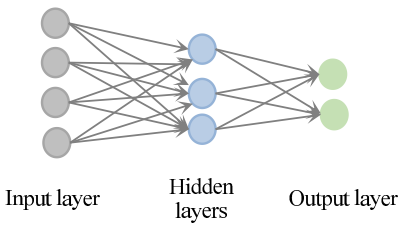


Fig. 2 Multiple layer perceptron model.

$$a^{(1)} = W^{(1)}x + b^{(1)} \tag{2}$$

Each layer takes a non-linear activation function to enhance the non-linearity of each latent feature. Suppose the activation function of the l-th layer is $f^{(l)}$, and the weight of the $(l-1)$-th layer to the l-th layer is $W^{(l)}$, the bias is $b^{(l)}$. Therefore, the output of the l-th latent layer could be noted as $a^{(l)}$:

$$a^{(l)} = f^{(l)}(W^{(l)}a^{(l-1)} + b^{(l)}) \tag{3}$$

The weight for the latent layer to the output layer is $W^{(l+1)}$, and the bias is $b^{(l+1)}$. Therefore, the output of the whole MLP network is y:

$$y = f^{(l+1)}(W^{(l+1)}a^{(l)} + b^{(l+1)}) \tag{4}$$

In the context of collaborative filtering, y is the predicted user-service rating score for user U_u on service I_i: y'_{ui}. And similar to the SDAE model, the objective function for MLP model is to minimize the difference between the predicted user-service rating score y'_{ui} and the actual user-service rating score y_{ui}.

$$\min \|y'_{ui} - y_{ui}\|^2 + \lambda \Sigma \|W\|^2 \tag{5}$$

Similar in Formula (1), λ is a regularization parameter and $\|\cdot\|$ denotes the Frobenius norm.

In this work, we adopt the MLP module to train the recommendation model, where the input of the MLP module is the dot product of latent user/service features, and the output is the predicted user-service rating score. With the training process of MLP, we can obtain the optimal parameters and make recommendation decision for each user.

3.3 NCF

NCF model was firstly proposed by He et al.[17] to overcome the limitation of typical Matrix Factorization model which could only obtain linear relationship between user and service features. In NCF, the MLP model is adopted to learn the arbitrary function between the user/service features and the predicted user-service rating score. The MF model adopts inner product of vectors, which could only learn the linearity function between the user/service latent representation and the rating score. While in NCF, multiple perceptrons are stacked to simulate the complex interactions between user/service latent features and the user-service rating score.

As shown in Fig. 3, the raw user and the service features are mapped into a latent vector via embedding techniques. Furthermore, the user latent vector U_i and the service latent vector I_j are concatenated $U_i \oplus I_j$ to generate the initial input for the NCF module. In NCF module, multiple perceptrons are stacked to learn the parameters for the mapping from the user/service latent vector to the user-service rating score. The objective function of the NCF model is identical to the one adopted in the MLP module, which is to minimize the difference between the predicted user-service rating score and the actual user-service rating score, which is depicted in Formula (5).

4 System Framework for NCF-MS Recommendation Model

As mentioned in Section 1, we take the popular cloud-edge collaboration architecture to implement the NCF-MS recommendation model in this work. In this section, we will introduce the system framework of the NCF-MS recommendation model and describe how the cloud-edge collaboration architecture works to support the NCF-MS recommendation model.

4.1 System framework

Figure 4 depicts the system framework of the cloud-edge collaboration architecture based NCF-MS recommendation model. As shown in Fig. 4, there are three major parts in the system framework for the NCF-MS recommendation model: user devices, an edge server group and a centralized cloud server, respectively. User devices are responsible for collecting multi-source data, including user profiles, service attributes as well as the user-service interaction records, which constitutes the training dataset for

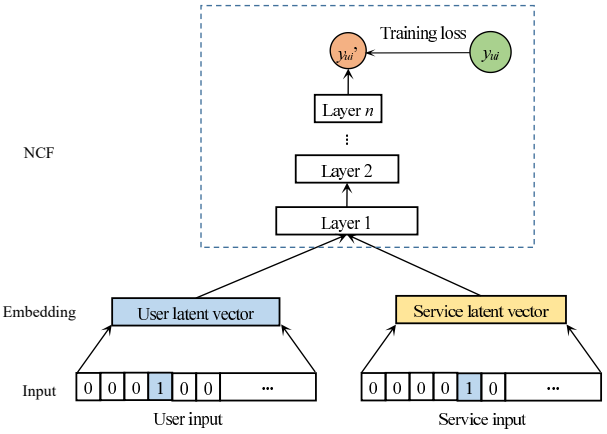


Fig. 3 NCF model.

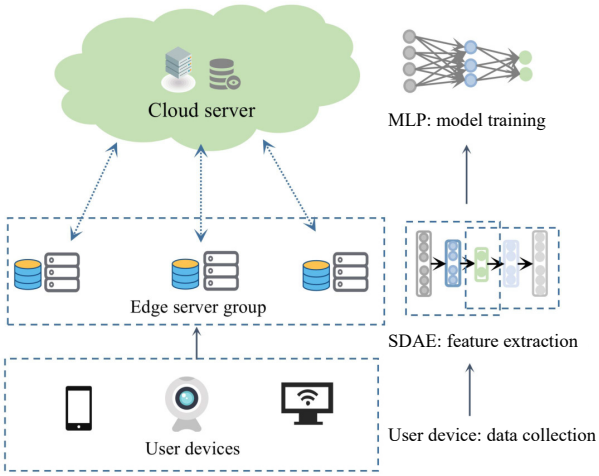


Fig. 4 System framework for NCF-MS model.

further recommendation computing. The edge servers are deployed close to the distributed user devices, which are to handle the user/service latent feature extraction. While the centralized cloud server is taking the user/service latent feature from the edge server as input to train the recommendation model to make recommendation decisions.

In our design, the NCF-MS recommendation model consists of two major components: an SDAE module and an MLP module. The SDAE module is deployed on edge servers that are close to user devices. And the SDAE module is responsible for extracting latent user/service features from the user profiles and service attributes collected by user devices. On the other hand, the MLP module is deployed on the cloud server, which concatenate user/service features received from edge servers, and train the recommendation model via a deep neural network to predict user-service rating score. By dividing the model training process into fine-grained feature extraction process and model training process, we could take full advantage of the resources on edge servers and cloud servers, therefore to improve resource utilization rate and save economic cost in terms of data transfer and time consumption.

4.2 Working process of NCF-MS recommendation model

With the support of the cloud-edge collaboration computing architecture, the working process of the NCF-MS recommendation model mainly consists of three phases:

- **Phase 1.** Feature Extraction: In this phase, the multi-source data including user profiles and service

attributes are transferred to the edge server group, which will adopt the SDAE module to extract user/service latent features.

- **Phase 2.** Model Training: With the latent user/service features extracted by the SDAE component, cloud server will concatenate the latent user/service features extracted from user profiles and service attributes, and take the MLP deep neural network to train the recommendation model.
- **Phase 3.** Recommendation Prediction: With the trained recommendation model, we could calculate the predicted rating score for each user/service pair and recommend uncalled services for each target user.

The details of feature extraction, model training, and recommendation prediction will be introduced in Section 5.

5 NCF-MS: Deep Neural Collaborative Filtering Based Service Recommendation Model with Multi-Source Data

In this section, we will introduce the details of our proposed NCF-MS recommendation model.

The overall design of the NCF-MS recommendation model is depicted by Fig. 5. As shown in Fig. 5, the NCF-MS model consists of two major parts: the feature extraction part and the model training part. The feature extraction part is responsible for extracting user and service features from the user-service interaction records, the user profiles as well as the service attribute information. Also, the extracted user/service features are fused with the concatenate operation, which will be transferred to the model training part to simulate the interaction between users and service to predict the user-service rating score.

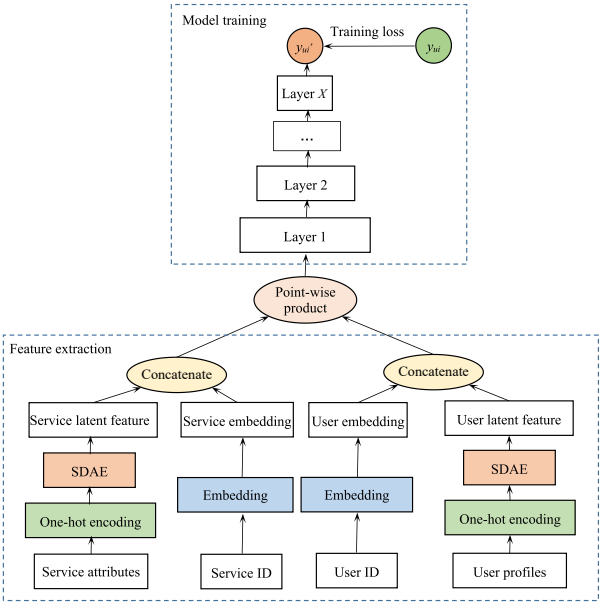


Fig. 5 Deep collaborative filtering model with multisource data.

More specifically, in NCF-MS, both the user feature vector and service feature vector consist of two parts. Take user feature vector for example, for a user u, its feature vector U_u consists of the user embedding P_u generated by the embedding component as well as the latent feature V_u extracted from the SDAE component. Similarly, for a service i, its feature vector I_i consists of the service embedding P_i and the latent feature V_i extracted from the SDAE component. Moreover, the user feature vector could be represented as $U_u = P_u \oplus V_u$; and the service feature vector could be represented $I_i = P_i \oplus V_i$. Furthermore, with the extracted user feature vector and the service feature vector, the model training component adopts an MLP module to train the parameters of the NCF-MS model.

The MLP model could be depicted by Eq. (6):

$$y'_{ui} = f(U_u, I_i | U_u, I_i, \Theta) \tag{6}$$

In Eq. (6), y'_{ui} is the predicted user-service rating score generated by the MLP module. f is the function to depict the interaction between the user feature vector and the service feature vector. Θ is the parameters for the function f. In MLP, f could be represented by Eq. (7):

$$f(U_u, I_i | U_u, I_i, \Theta) = \rho_{\text{out}}(\rho_x(\cdots(\rho_1(U_u, I_i) + b_1)\cdots) + b_x) \tag{7}$$

In Eq. (7), ρ represents the activation function of the MLP network for corresponding layer; while b is the bias parameter as explained in Section 3.2.

The details of the feature extraction and model training processes of the NCF-MS model is described in the following section.

5.1 Feature extraction

As described in aforementioned section, the feature extraction module is responsible for extracting user/service feature vector from the user-service interaction records, the user profiles, as well as the service attribute information.

Step 1. Feature extraction from user-service interaction record.

Definition 1 User-Service interaction record. In our solution, the user-service interaction record dataset

could be depicted as a matrix $Y = M \times N$, where M is the number of users and N is the number of services. Each record y_{ui} in Y refers to whether user u has interacted with service i or not.

$$y_{ui} = \begin{cases} 1, & \text{if user } u \text{ invoked service } i; \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

For each user u and service i in Y, we will firstly encode u and i with a unique and numeric vector with fixed size via the embedding technique. For u, its embedding could be denoted as P_u; while for service i, its embedding could be denoted as P_i.

The recommendation model with such implicit user-service interaction record could be formalized as to find a prediction function to predict the missing values in the user-service interaction record matrix, which is equal to obtain the optimal parameters in Eq. (7) to make accurate prediction for missing user-service interaction scores. In the native NCF recommendation model, it firstly takes embedding technique to generate the user embedding and service embedding, which will be further trained via the MLP module to generate the predicted user-service rating score. However, in practice, the user-service interaction record matrix is normally sparse. The pure user embedding and service embedding could only provide limited user/service features, which could limited the performance of the prediction accuracy. Therefore, we propose to fuse multi-source auxiliary information into the recommendation model to depict more accurate user features and service features to improve the recommendation accuracy.

In our solution, we take the user profiles and the service attributes as auxiliary information to build the multi-source data driven collaborative recommendation model. And the feature extraction module will extract user/service features from the auxiliary information and fuse them with the raw user/service embeddings to generate the user/service feature vector as input for subsequent model training module.

Step 2. Feature extraction from user profiles.

Definition 2 User profile. For each user u in the user profiles of NCF-MS method, he/she can be formalized as a quad-tuple $u =< UserID, Gender, Age, Carrer >$.

For each attribute in user profiles, we firstly encode it with the one-hot encoding method[35]. For example, there are M users involved in the user-service interaction matrix Y, the one-hot encoding of the $UserID$ attribute is a $0-1$ numeric vector with length of M. Similarly, the one-hot encoding of the *Gender* and *Carrer* attributes is a $0-1$ binary numeric vector with length of 2 and x (i.e., x is the number of *Career* attributes), respectively. For *Age* attribute, we divide the age into m ranges and encode the *Age* attribute to a $0-1$ binary numeric vector with length of m.

With one-hot encoding method, we could map each user into a $0-1$ numeric vector with fixed size $M+2+x+m$. Furthermore, the SDAE module is adopted to extract the key features from the user profile. For user u, the latent user feature extracted by the SDAE module could be represented as V_u.

Step 3. Feature extraction from service attributes.

Definition 3 Service attribute. Similarly, for each service I_i in service attributes, it can be formalized as a multi-tuple I_i = <*ServiceID*, *ServiceType*, *ServiceFunction*>.

The latent service feature extraction process is similar to process of feature extraction from user profiles. Therefore, we omit the details here. Similarly, for each service i, the latent service feature extracted by SDAE module could be represented as V_i.

5.2 Model training

With the feature extraction process, we could obtain P_u and V_u, which refers to the user features from embedding technique as well as the latent user feature extracted from the auxiliary user profiles, respectively. Similarly, P_i and V_i are the embedded service feature and the latent service feature, respectively. In our NCF-MS method, we fuse them via concatenate operation.

$$\begin{cases} U_u = concatenate(P_u, V_u), \\ I_i = concatenate(P_i, V_i) \end{cases} \tag{9}$$

U_u and I_i are the user feature vector for user u and the service feature vector for service i, respectively. U_u and I_i are the input vectors for the NCF-MS model, which could be multiplied via inner product and trained by the MLP module to predict the user-service interaction score y'_{ui}. In our proposal, we adopt the MLP module to simulate the interaction between the user and service feature vector. Therefore, with the non-linearity fitting ability of the MLP neural network, we could calculate the accurate interaction relation between users and services.

As shown in Fig. 5, the input of the MLP module is the element-wise product of the user latent feature vector and the service latent feature vector, which will

be trained with the x-th layer neural networks, to calculate the predicted user-service interaction score. Furthermore, we could formulate the score prediction problem as model training problem to learn the optimized parameters for the MLP model. Since the user-service interaction score is a binary value, for which $y_{ui}=0$ represents no interaction between U_u and I_i; while $y_{ui}=1$ means U_u has interacted with I_i. Therefore, the objective of the MLP model in our recommendation context is to maximize the likelihood value for all positive and negative user-service interaction scores.

$$\rho(y,\ y-|U,I,\theta)=\Pi_{(u,i)\in y}y'_{ui}\Pi_{u,i\in y-}(1-y'_{ui}) \tag{10}$$

Take the negative log value for Eq. (10), we can obtain the loss function in Eq. (11), which can be further transferred to the expression depicted by Eq. (12).

$$L=-\Sigma_{(u,i)\in y}\log y'_{ui}-\Sigma_{(u,i)\in y-}\log(1-y'_{ui}) \tag{11}$$

$$L=\Sigma_{(u,i)\in yUy-}\{y_{ui}\log y'_{ui}+(1-y_{ui})\log(1-y'_{ui})\} \tag{12}$$

The loss function of Eq. (12) is actually a binary categorization problem. We could take the Stochastic Gradient Decreasing (SGD) method to calculate the optimal parameters. Algorithm 1 depicts the training process of the NCF-MS model.

5.3 Recommendation prediction

With the trained NCF-MS model, the predicted interaction score for each user-service pair could be calculated correspondingly. Furthermore, we could recommend services with higher predicted interaction score for each user.

6 Experimental Evaluation

In this section, we simulate a set of experiments on three public datasets to verify our proposed NCF-MS method. The experimental setting of our simulation is depicted in Table 1.

The three datasets adopted in our experiment are: the Santander Bank dataset[36], the Amazon AToys[37] dataset as well as the Amazon AMusic[37] dataset, respectively. The Santander Bank dataset contains the consumption records of 709 411 users on 24 bank products; while the Amazon AToys and Amazon AMusic are two public datasets provided by Amazon that records the user's rating score and reviews on Amazon products. Limited by the memory of the device, as well as to improve the recommendation accuracy, we filter the primitive datasets to select the records as experimental dataset. In our rule, the number of users is less than 5000. Also, each user involved in the dataset should interacted with at least 5 services; while each service should be invoked by at least 20 users. With this processing rule, the datasets are summarized in Table 2.

Algorithm 1 Training process of NCF-MS model

EdgeServerExecute($UserProfile$, $ServiceAttributes$)
  $UserVector \leftarrow$ NULL
  $ServiceVector \leftarrow$ NULL
  for each user u in $UserProfile$ **do**
    $P_u \leftarrow onehot\_Encoding(u)$
    $P_u \leftarrow$ SDAE(P_u)
    $V_u \leftarrow$ Embedding(u)
    $U_u \leftarrow P_u \oplus V_u$
    add U_u into $UserVector$
  for each service i in $ServiceAttributes$ **do**
    $P_i \leftarrow onehot\_Encoding(s)$
    $P_i \leftarrow$ SDAE(P_i)
    $V_i \leftarrow$ Embedding(i)
    $I_i \leftarrow P_i \oplus V_i$
    add I_i into $ServiceVector$
  return $UserVector$, $ServiceVector$

`CloudServerExecute`($UserVector$, $ServiceVector$):
    initialize the parameters w and b of the MLP module
  for each epoch $i=1,2,...,E$ **do**
    $y' \leftarrow$ MLP($UserVector \odot ServiceVector$)
    maximize Eq. (12) to update w and b
  return w, b

Table 1 Experimental setting.

CPU	Intel (R) Core (TM) i7-10870H CPU @ 2.20 GHz
Memory	8 GB
Development tool	Python (Tensorflow)
System environment	Window 1064 bits

6.1 Baselines

In our simulation, we compare our proposed method with three existing baseline methods:

ItemKNN: ItemKNN is a typical item-based collaborative filtering method. It calculates the similarity between services based on the user-service interaction records, and recommend to users with similar services according to their history consumption records. We take ItemKNN as comparison method to

Table 2 Simulation on three public datasets.

Dataset	Attribute			
	Number of users	Service	Rating	Rating sparsity
Santander	5000	15	3882	0.9676
Amazon AToys	3137	33 953	84 642	0.9992
Amazon AMusic	1733	12 489	46 087	0.9978

check how NCF-MS method works by comparing with typical CF method.

NCF: NCF is the first typical collaborative recommendation model which adopts the neural network to calculate the interaction between users and service to make the recommendation decision. We take NCF method as the baseline method to check the effectiveness of integrating multi-source data into the native NCF method.

BPR: BPR adopts the implicit feedback to make the recommendation decision. Moreover, it adopts matrix factorization and the pair-wise loss function to conduct model training. We take the BPR method as baseline to check how deep learning methods work than typical matrix factorization methods to predict user-service rating score.

We implement the four recommendation methods based on the Tensorflow library. To learn the optimal parameters of the methods, we initialize the model parameters following random Gaussian distribution, and we adopt the Adam model in the model training process. The learning rate is set to 0.001. The number of MLP modules is set to 3 according to the performance of the simulation results.

6.2 Evaluation metrics

We adopt two evaluation metrics to evaluate our proposed method.

HR refers to hit rate, i.e., the recall rate of the top-K recommendation results, for which the calculation equation is depicted by Eq. (13). In Eq. (13), N refers to the whole number of users, $hits(i)$ indicates whether the interacted services for the i-th user is in the recommendation list or not.

$$\mathrm{HR} = \frac{1}{N}\sum_{i=1}^{N} hits(i) \tag{13}$$

NDCG refers to the normalized cumulative loss gain, which evaluates the sequence of the services in the recommendation list. In Eq. (14), p_i is the location of the services interacted by the i-th user in the recommendation list. And p_i equals to infinity if the services are not involved in the recommendation list.

$$\mathrm{NDCG} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\log_2(p_i+1)} \tag{14}$$

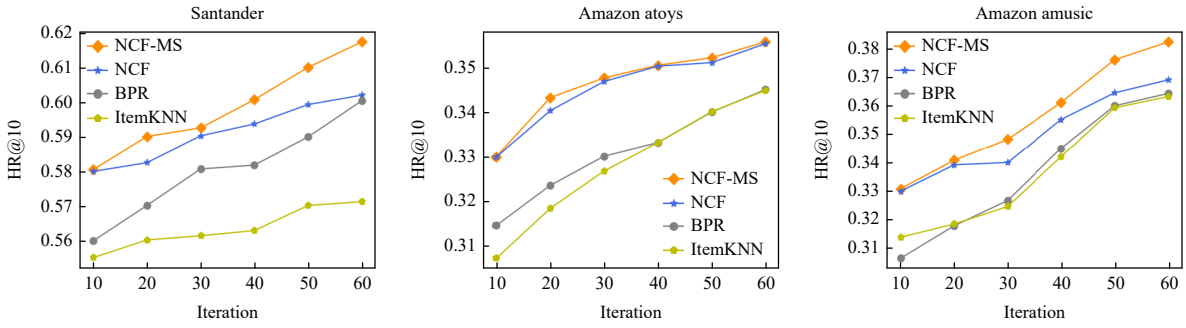
In our experiment, we set the value of K to 10 for both HR and NDCG metrics. The higher value of HR@10 and NDCG@10, the better performance of the recommendation model.

6.3 Comparison analysis

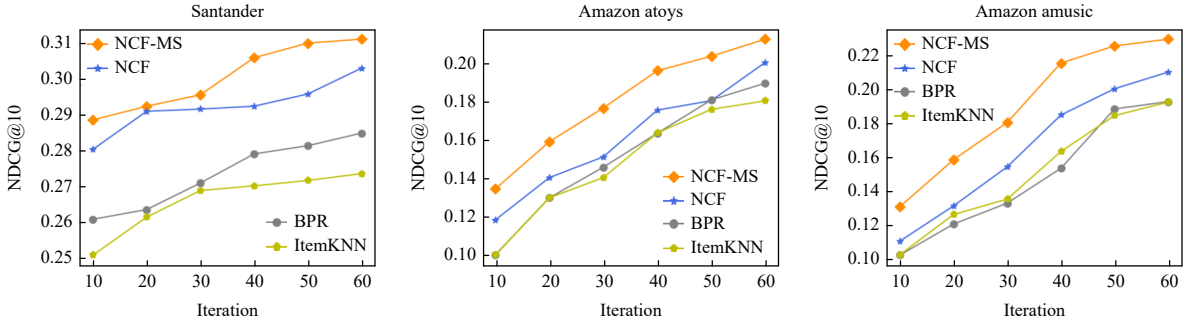
(1) Comparison analysis w.r.t HR@10

Figure 6 depicts the comparison results among the four recommendation models with respective to the performance of HR@10 and NDCG@10. As shown in Fig. 6, we can find that with the increase of iteration rounds, the performance of all the four methods is improving accordingly. Moreover, the NCF-MS method always achieves the best performance on the three public datasets. The reason for this scenario is analysed as follows: the ItemKNN, BPR, and NCF methods only adopt the primitive user-service interaction records as the data source to make recommendation decisions. Based on the aforementioned calculation, we can find all the three public datasets all have sparsity more than 99%. The data sparsity greatly limits the recommendation accuracy. Therefore, by fusing the multi-source user profiles and service attributes data into the collaborative recommendation model could effectively improve the recommendation accuracy.

Furthermore, we could find that even all of the ItemKNN, BPR, and NCF methods are using the primitive user-service interaction records to make the recommendation decision, the NCF method achieves the best performance among the three methods on all the three public datasets. This is because the NCF method adopts the neural network to capture the relationship between the user feature and the service feature. Due to the non-linearity fitting ability of NCF, the interaction function between user/service feature calculated by NCF is more accurate. This also explains



(a) Performance comaprision with respect to HR@10



(b) Performance comaprision with respect to NDCG@10

Fig. 6 Performance comparison of the NCF-MS method and baseline methods.

the better performance of NCF-MS than the BPR and ItemKNN methods.

(2) Comparison analysis with the native NCF method

Based on aforementioned analysis, we can find that both of the NCF and NCF-MS methods are adopting the deep neural network to fit the interaction between user and service latent features. NCF-MS optimizes the native NCF method by fusing multi-source data (i.e., user profiles and service attributes) to the collaborative recommendation model. Here, to further compare the two methods, we summarize the performance comparison between the two methods in Table 3.

As shown in Table 3, the NCF-MS method achieves better performance than the NCF method on all the three public datasets. This illustrates the multi-source data could help to capture more accurate user preference to improve recommendation accuracy. However, from Table 3, we could also find that the NCF-MS does not have outstanding advantages than the NCF method. Therefore, the user profiles and service attributes may contain limit latent user preference information. In the future work, we can try to fuse more accurate auxiliary information such as service comments to extract more accurate user/service latent features to improve recommendation accuracy.

7 Conclusion

In this work, we proposes a deep neural collaborative filtering based service recommendation model NCF-MS, which takes the popular cloud-edge computing paradigm to implement recommendation model over the distributed big data resources. Moreover, the NCF-MS model fuses multi-source data (i.e., user profiles and service attributes) into the native NCF recommendation model to improve recommendation accuracy accordingly. Briefly, the SDAE module is adopted to extract latent user/service features from the

Table 3 Performance comparison of NCF-MS vs. NCF.

Dataset	Metric	NCF	NCF-MS	Improvement (%)
Santander	HR@10	0.5831	0.6110	2.79
	NDCG@10	0.3054	0.3161	2.36
Amazon AToys	HR@10	0.3422	0.3599	1.77
	NDCG@10	0.1998	0.2054	0.56
Amazon AMusic	HR@10	0.3697	0.3855	1.58
	NDCG@10	0.2151	0.2269	1.18

multi-source auxiliary information, which are transferred to train the collaborative recommendation model via the MLP module. A set of simulation experiments are conducted on three public datasets to evaluate the effectiveness of our proposal. In the future work, we will enhance the recommendation model by fusing more auxiliary information such as user-service comments into the NCF-MS model, to further improve the flexibility and accuracy of recommendation results.

Acknowledgment

This work was supported by the Natural Science Foundation of Zhejiang Province (Nos. LQ21F020021 and LZ21F020008), Zhejiang Provincial Natural Science Foundation of China (No. LZ22F020002), and the Research Start-up Project funded by Hangzhou Normal University (No. 2020QD2035）.

References

[1] X. Zhou, Y. Hu, J. Wu, W. Liang, J. Ma, and Q. Jin, Distribution bias aware collaborative generative adversarial network for imbalanced deep learning in industrial IoT, *IEEE Trans. Ind. Inf.*, vol. 19, no. 1, pp. 570–580, 2023.

[2] L. Qi, Q. He, F. Chen, X. Zhang, W. Dou, and Q. Ni, Data-driven web APIs recommendation for building web applications, *IEEE Trans. Big Data*, vol. 8, no. 3, pp. 685–698, 2022.

[3] F. Wang, H. Zhu, G. Srivastava, S. Li, M. R. Khosravi, and L. Qi, Robust collaborative filtering recommendation with user-item-trust records, *IEEE Trans. Comput. Soc. Syst.*, vol. 9, no. 4, pp. 986–996, 2022.

[4] Y. Liu, A. Pei, F. Wang, Y. Yang, X. Zhang, H. Wang, H. Dai, L. Qi, and R. Ma, An attention-based category-aware GRU model for the next POI recommendation, *Int. J. Intell. Syst.*, vol. 36, no. 7, pp. 3174–3189, 2021.

[5] W. Lin, X. Zhang, L. Qi, W. Li, S. Li, V. S. Sheng, and S. Nepal, Location-aware service recommendations with privacy-preservation in the Internet of Things, *IEEE Trans. Comput. Soc. Syst.*, vol. 8, no. 1, pp. 227–235, 2021.

[6] C. Wu, F. Wu, Y. Huang, and X. Xie, Personalized news recommendation: Methods and challenges, arXiv preprint arXiv: 2106.08934, 2021.

[7] S. Jiang, X. Qian, T. Mei, and Y. Fu, Personalized travel sequence recommendation on multi-source big social media, *IEEE Trans. Big Data*, vol. 2, no. 1, pp. 43–56, 2016.

[8] H. Liu, Y. Tong, J. Han, P. Zhang, X. Lu, and H. Xiong, Incorporating multi-source urban data for personalized and context-aware multi-modal transportation recommendation, *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 723–735, 2022.

[9] G. Zhao, T. Liu, X. Qian, T. Hou, H. Wang, X. Hou, and Z. Li, Location recommendation for enterprises by multi-source urban big data analysis, *IEEE Trans. Serv. Comput.*, vol. 13, no. 6, pp. 1115–1127, 2020.

[10] L. Qi, W. Lin, X. Zhang, W. Dou, X. Xu, and J. Chen, A correlation graph based approach for personalized and compatible web APIs recommendation in mobile APP development, *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 5444–5457, 2023.

[11] S. Wu, S. Shen, X. Xu, Y. Chen, X. Zhou, D. Liu, X. Xue, and L. Qi, Popularity-aware and diverse web APIs recommendation based on correlation graph, *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 2, pp. 771–782, 2023.

[12] X. Zhou, W. Liang, K. Yan, W. Li, K. I. K. Wang, J. Ma, and Q. Jin, Edge-enabled two-stage scheduling based on deep reinforcement learning for Internet of Everything, *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3295–3304, 2022.

[13] X. Xu, J. Gu, H. Yan, W. Liu, L. Qi, and X. Zhou, Reputation-aware supplier assessment for blockchain-enabled supply chain in industry 4.0, *IEEE Trans. Ind. Inform.*, vol. 19, no. 4, pp. 5485–5494, 2023.

[14] L. Qi, Y. Yang, X. Zhou, W. Rafique, and J. Ma, Fast anomaly identification based on multiaspect data streams for intelligent intrusion detection toward secure industry 4.0, *IEEE Trans. Ind. Inform.*, vol. 18, no. 9, pp. 6503–6511, 2022.

[15] Z. Li, X. Xu, T. Hang, H. Xiang, Y. Cui, L. Qi, and X. Zhou, A knowledge-driven anomaly detection framework for social production system, *IEEE Trans. Comput. Soc. Syst.*, vol. 1, no. 99, pp. 1–14, 2022.

[16] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, Deep-learning-enhanced multitarget detection for end–edge–cloud surveillance in smart IoT, *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12588–12596, 2021.

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

[18] L. Qi, Z. Zhou, J. Yu, and Q. Liu, Data-sparsity tolerant web service recommendation approach based on improved collaborative filtering, *IEICE Trans. Inf. & Syst.*, vol. E100.D, no. 9, pp. 2092–2099, 2017.

[19] S. Meng, L. Qi, Q. Li, W. Lin, X. Xu, and S. Wan, Privacy-preserving and sparsity-aware location-based prediction method for collaborative recommender systems, *Future Gener. Comput. Syst.*, vol. 96, pp. 324–335, 2019.

[20] X. Zhou, X. Yang, J. Ma, and K. I. K. Wang, Energy-efficient smart routing based on link correlation mining for wireless edge computing in IoT, *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14988–14997, 2022.

[21] Y. Jia, B. Liu, W. Dou, X. Xu, X. Zhou, L. Qi, and Z. Yan, CroApp: A CNN-based resource optimization approach in edge computing environment, *IEEE Trans. Ind. Inform.*, vol. 18, no. 9, pp. 6300–6307, 2022.

[22] X. Xu, H. Tian, X. Zhang, L. Qi, Q. He, and W. Dou, DisCOV: Distributed COVID-19 detection on X-ray images with edge-cloud collaboration, *IEEE Trans. Serv. Comput.*, vol. 15, no. 3, pp. 1206–1219, 2022.

[23] T. T. Nguyen, On the edge and cloud: Recommendation systems with distributed machine learning, in *Proc. 2021 Int. Conf. Information Technology* (*ICIT*), Amman,

Jordan, 2021, pp. 929–934.
[24] C. Li and K. He, CBMR: An optimized MapReduce for item-based collaborative filtering recommendation algorithm with empirical analysis, *Concurr. Comput. Pract. Exp.*, vol. 29, no. 10, p. e4092, 2017.
[25] F. Ortega, A. Hernando, J. Bobadilla, and J. H. Kang, Recommending items to group of users using matrix factorization based collaborative filtering, *Inf. Sci.*, vol. 345, pp. 313–324, 2016.
[26] R. Salakhutdinov, A. Mnih, and G. Hinton, Restricted Boltzmann machines for collaborative filtering, in *Proc. 24th Int. Conf. Machine learning*, Corvalis, OR, USA, 2007, pp. 791–798.
[27] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.
[28] W. J. Li, D. Y. Yeung, and Z. Zhang, Generalized latent factor models for social network analysis, in *Proc. Twenty-Second Int. joint Conf. Artificial Intelligence*, Barcelona, Spain, 2011, pp. 1705–1710.
[29] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu, Personalized recommendation via cross-domain triadic factorization, in *Proc. 22nd Int. Conf. World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 595–606.
[30] Z. H. Deng, L. Huang, C. D. Wang, J. H. Lai, and P. S. Yu, DeepCF: A unified framework of representation learning and matching function learning in recommender system, arXiv preprint arXiv: 1901.04704, 2019.
[31] 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.
[32] L. Zheng, V. Noroozi, and P. S. Yu, Joint deep modeling of users and items using reviews for recommendation, in *Proc. Tenth ACM Int. Conf. Web Search and Data Mining*, Cambridge, United Kingdom, 2017, pp. 425–434.
[33] C. Chen, M. Zhang, Y. Liu, and S. Ma, Neural attentional rating regression with review-level explanations, *in Proc. 2018 World Wide Web Conference Lyon, France*, 2018, pp. 1583–1592.
[34] C. Wu, F. Wu, T. Qi, and Y. Huang, SentiRec: Sentiment diversity-aware neural news recommendation, in *Proc. 1st Conf. Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th Int. Joint Conf. on Natural Language Processing*, Suzhou, China, 2020, pp. 44–53.
[35] One-hot Encoding, https://www.cs.toronto.edu/guerzhoy/321/lec/W04/onehot.pdf, 2023.
[36] Banco Santander, https://www.santander.com/en/landing-pages/dataprotection-policy, 2023.
[37] Amazon Review Data, https://nijianmo.github.io/amazon/index.html, 2018.

Wenmin Lin received the PhD degree in computer science and technology from Nanjing University, Nanjing, China in 2015. She is currently a lecturer with Alibaba Business School, Hangzhou Normal University, Hangzhou, China. Her research interests include service computing, big data processing techniques, and recommender systems.

Xinyi Zhou is currently an undergraduate student with Hangzhou Normal University, China. Her research interests include recommender systems and data mining

Ruowei Zhang received the BS degree from Qufu Normal University, Rizhao, China in 2021. She is pursuing the MSc degree from Qufu Normal University, China. Her research interest includes recommender systems.

Min Zhu received the BS degree from Liaocheng University, China in 2008. She obtained the MS degree in electronic and communication engineering from Ocean University of China in 2012. She is currently a lecturer in Weifang University of Science and Technology, China. Her research interests include big data and information systems.

Xiaoran Zhao received the BS degree from Qufu Normal University, Rizhao, China in 2022. She is pursuing the MSc degree from Qufu Normal University, China. Her research interests include recommender systems and cluster analysis.

Lu Sun is an associate professor at Alibaba Business School of Hangzhou Normal University, China. She received the PhD degree from Harbin Institute of Technology, China in 2016. Her research focuses on information interaction and information management.

Shigen Shen received the BS degree in fundamental mathematics from Zhejiang Normal University, Jinhua, China in 1995, the MS degree in computer science and technology from Zhejiang University, Hangzhou, China in 2005, and the PhD degree from Donghua University, Shanghai, China in 2013. He is a professor with School of Information Engineering, Huzhou University, Huzhou, China. He has published more than 100 technical papers, including respected journals such as IEEE TIFS, TDSC, TII, TVT, and IoT-J. His current research interests include Internet of Things, cyber security, edge computing, and game theory. He is currently serving as a member of editorial boards of *CMC-Computers*, *Materials Continua*, *and Intelligent Automation Soft Computing*, as well as the reviewer of *Journal of Organizational and End User Computing*.