Collaborative Learning-Based Industrial IoT API Recommendation for Software-Defined Devices: The Implicit Knowledge Discovery Perspective

Honghao Gao, *Senior Member, IEEE*, Xi Qin, Ramón J. Durán Barroso, Walayat Hussain, Yueshen Xu, *Senior Member, IEEE*, and Yuyu Yin

***Abstract*—The industrial Internet of things (IIoT), a new computing mode in Industry 4.0, is deployed to connect IoT devices and use communication technology to respond to control commands and handle industrial data. IIoT is typically employed to improve the efficiency of computing and sensing and can be used in many scenarios, such as intelligent manufacturing and video surveillance. To build an IIoT system, we need a collection of software to manage and monitor each system component when there are large-scale devices. Application programming interface (API) is an effective way to invoke public services provided by different platforms. Developers can invoke different APIs to operate IoT devices without knowing the implementation process. We can design a workflow to configure how and when to invoke target APIs. Thus, APIs are a powerful tool for rapidly developing industrial systems. However, the increasing number of APIs exacerbates the problem of finding suitable APIs. Current related recommendation methods have defects. For example, most existing methods focus on the relation between users and APIs but neglect the valuable relations among the users or APIs themselves. To address these problems, this article studies implicit knowledge in IIoT by using collaborative learning techniques. Considering the increased dimensions and dynamics of IoT devices, we explore the possible relationships between users and between APIs. We enhance the matrix factorization (MF) model with the mined implicit knowledge that are implicit relationships on both sides. We build an ensemble model by using all implicit knowledge. We conduct experiments on a collected real-world dataset and simulate industrial system scenarios. The experimental results verify the effectiveness and superiority of the proposed models.**

***Index Terms*—API recommendation, collaborative learning, implicit relationship mining, industrial internet of things, matrix factorization.**

I. Introduction

THE industrial Internet of things (IIoT) system is distributed and heterogeneous and works collaboratively to bridge the application and device domains. With the explosive growth in IIoT devices, applications have also substantially expanded in recent years. The cooperation between large-scale IIoT devices can be considered to be *software-defined devices* (SDDs), where industrial workflow predefines tasks on when and how to invoke or operate devices. When software features cross, developers begin to think about whether the same data or functionality can be just as easily consumed by another piece of software [1], [2]. As one possible solution, application programming interfaces (APIs) are developed in this context. An API is geared for consumption by software and allows applications, software programs and hardware to interact [3], [4]. Figure 1 shows that application developers can leverage APIs offered by local software systems or IIoT devices where their application runs. For example, applications can discover the current location of a smart phone by calling the API associated with the Global Positioning System (GPS) receiver of the mobile phone. Intelligent manufacturing can order the operation of different types of machine tool processing, such as metal cutting and stamping, to work with the API channel. In Fig. 1, ERP is short for enterprise resource planning, SCADA is short for supervisory control and data acquisition, and DNC is short for direct numerical control.

APIs have become popular and have gained enormous support from multiple platforms. For instance, there are more than 22914 publicly accessible APIs on ProgrammableWeb.[1] The task of finding the best API among a large number of candidates is challenging. In this paper, we aim to solve the IIoT API recommendation task by studying implicit knowledge to improve function integration and device collaboration.

To recommend appropriate items to a user, traditional recommendation algorithms focus on historical user data. 1) Content-based recommendations use the correlation between the information content of items and preferences to filter information,

Manuscript received April 24, 2020; revised August 17, 2020; accepted September 6, 2020. This work is supported by the National Natural Science Foundation of China (NSFC) under Grant 61902236. *(Xi Qin is co-first author.) (Corresponding author: Yueshen Xu.)*

Honghao Gao is with the School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China (e-mail: gaohonghao@shu.edu.cn).

Xi Qin is with the School of Computer Science and Technology, Xidian University, Xi'an 710126, China (e-mail: qinxics@hotmail.com).

Ramón J. Durán Barroso is with the Faculty of Telecommunication Engineering, University of Valladolid, 47002 Valladolid, Spain (e-mail: rduran@tel.uva.es).

Walayat Hussain is with the Faculty of Engineering and IT, University of Technology Sydney, Sydney 2007, Australia (e-mail: Walayat.Hussain@uts.edu.au).

Yueshen Xu is with the School of Computer Science and Technology, Xidian University, Xi'an 710126, China (e-mail: ysxu@xidian.edu.cn).

Yuyu Yin is with the School of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, China, and also with the Key Laboratory of Complex Systems Modeling and Simulation, Ministry of Education, Hangzhou 310018, China (e-mail: yinyuyu@hdu.edu.cn).

Digital Object Identifier 10.1109/TETCI.2020.3023155

[1] https://www.programmableweb.com

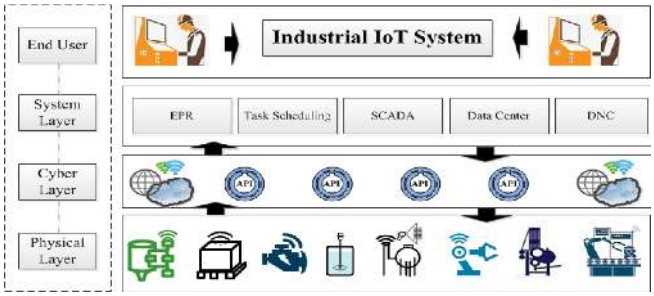


Fig. 1. The API-empowered IIoT framework.

thereby generating recommendations for a user [5]. The shortcoming of this approach is clear. If there is insufficient information to distinguish between items that users like and items that users dislike, a content-based recommendation system cannot provide good recommendations [1]. 2) Collaborative filtering (CF) recommendation systems filter information based on the preferences of others and historical records. In general, CF can be divided into two types: neighborhood-based methods and model-based methods [6].

Neighborhood-based methods include user-based CF algorithms and item-based CF algorithms. User-based CF algorithms identify a user group that is similar to the target user and predict the user rating of an item based on user group ratings of the item [7]. User-based CF algorithms are dependent on user rating data, so if two users have almost no ratings, bias can easily occur. Item-based CF algorithms utilize user preferences for items, find similarities between items and recommend similar items to the target user based on learned historical preference [8]. A problem with user-based and item-based CF algorithms is the cold-start problem; i.e., it is difficult to provide recommendations without historical data. Model-based methods train predefined models to predict ratings [9]. Representative model-based CF algorithms include clustering methods [10], aspect models [11] and latent factor models (LFMs) [12]. Model-based methods partially overcome the problem of data sparsity and improve prediction accuracy but still ignore the relationships that exist in the real world. In this paper, we use CF to develop API recommendation for IIoT systems.

Current API recommendation approaches can be classified into two categories, i.e., one approach for code and the other for APIs. For code recommendation, some researchers try to use statistical learning models trained for fine-grained code changes [13], [14], while other researchers apply neural network models [15]. Nearly one million APIs are available for query on the website ProgrammableWeb. In [16]–[19], different methods are used to recommend suitable APIs for mashups. Some researchers combine knowledge graphs and random walks with restart to find the potential relevance between the mashup and API [20]. Some researchers present a knowledge graph framework that uses side information to improve API recommendation [21]. Other researchers focus on the popularity of APIs and predict popularity by incorporating heterogeneous information [22]. The approaches used in all of these studies have performed well, but for APIs and users, many features and the potential connections between users, respectively, have not been explored.

In IIoT, we consider the function entity to be an IIoT device public interface that is encapsulated as an API to support the linkage between hardware and software. To address the existing problems, we propose an ensemble model that combines an LFM and the relationship between the user and the features of the IIoT API. That is, by adding a regularization constraint to the LFM, we support IIoT API recommendation by discovering more implicit knowledge. The main contributions of this paper are as follows:

1) We propose discovering and leveraging potential implicit relations between users and between APIs. We propose a solution to developing such a task by designing a similarity computation between two users and between two APIs. The computed similarities are to be implicit knowledge.
2) We propose two novel APIs recommendation models, which are built by using matrix factorization (MF), similarity computation and new regularization terms. One model is built with the new designed regularization term on the user side, another model is built with the new designed regularization term on the API side and, finally, another model is an ensemble model combining mined implicit knowledge on both sides.
3) We crawled a new large-scale real-world dataset and evaluated our models under different experimental settings. We also studied the sensitivity of our model to parameters.

The rest of this paper is structured as follows: Section II discusses current API recommendation methods. Section III describes the details of the framework and our method. The experimental results are presented in Section IV. Finally, Section V concludes this paper and discusses future research.

II. Related Work

In this section, we review the algorithms that are currently applied to APIs recommendation and service recommendation.

CF algorithms are widely used to make service recommendations [19], [23], [24]. To address the shortcomings of traditional CF algorithms, in paper [23], the authors proposed a hybrid CF algorithm. Specifically, the authors first proposed a mechanism for collecting web service information and combined a user-based and an item-based CF algorithm with different weights by using a new weight calculation method, which improved the algorithm's recommendation accuracy. In [19], the author proposed a mechanism for collecting web service information and then combined a user-based and an item-based CF algorithm with different weights by using a new weight calculation method.

As a new recommendation problem, APIs recommendation has attracted much attention in recent years. Algorithms in this field can be divided into APIs recommendation and mashup recommendation.

For mashup recommendation, in paper [16], the authors recommended mashups to users based on user interests and service social networks. User interest was extracted from the mashup historical usage data. The service social network was constructed based on the information related to the mashup service, Web

API, and API tags. The mined interests of users were combined with the network to make recommendations. To determine the APIs that constituted a mashup, in [19], the author focused on mining the potential associations between APIs and the mined API joint invocation pattern in mashups and applied the mined association to a recommendation model, which was a probabilistic model enhanced with implicit correlation regularization.

For APIs recommendation, in paper [1], the authors presented a knowledge graph-based framework. The framework first identifies the API invocation relationship and extracts key information, such as labels and categories, and uses the extracted information to generate the relationship component between entities. The component is input into the knowledge graph for recommendation. The authors in [20] proposed a recommendation framework based on random walks on a knowledge graph. The authors first captured the most useful information of APIs by using a knowledge graph and filtered the nodes that have rich semantic information. The authors used random walks and reestimated the relatedness between the recommender and the candidate. Heterogeneous networks were also employed to resolve multiple types of objects and multiple links representing different relationships. In paper [25], the authors combined API information with mashup information and related attributes to construct a heterogeneous information network and clustered the information in the network to obtain a mashup group preference. The model was trained to obtain personalized recommendation results. In [26], the authors solved the problem that current API search engines allow only representative services to be searched. The authors proposed a new retrieval approach that is based on service cooperative network and provides a visualization technique for easy browsing.

Traditional CF algorithms are also used to recommend APIs. In paper [27], the author proposed an item-based CF algorithm for API recommendation; the algorithm computes the item similarity based on a user group that contains users with similar preferences and develops an item-based CF algorithm specific to each user group. The performance of the MF model has been verified to be superior to neighborhood-based CF, especially in cases where the data are sparse. In paper [28], the authors tried to learn the preferences that are displayed by users during the invocation of APIs, mashups, and users. The authors created a new recommendation model by combining embedded user preference and the MF model.

Current methods have achieved good recommendation accuracy, but the information in the data has not been fully utilized. In IoT, Chen et.al. [29] considered that temporal context plays an important role in smart object recommendation, as most users tend to utilize different objects at different time slots in a day. The authors proposed a time-aware smart object recommendation model by jointly considering user preference over time and the smart object similarity. Duan *et al.* [30] proposed JointRec, a deep learning-based joint cloud video recommendation framework for IoT; JointRec aims to provide an accurate video recommendation service to a minority of users. Huang *et al.* [31] considered that in IoT, the description information of items is typically heterogeneous and multimodal. The authors proposed a multimodal representation learning-based model (MRLM) to improve the recommendation accuracy in IoT. Hu *et al.* [32] proposed vehicular ad-hoc network representation learning for recommendations in IoT because trajectory records enable a better understanding of human mobility patterns. In contrast to these methods, in this paper, we explore the potential connections between users and analyze the characteristics of APIs. We apply collaborative learning to implicit knowledge discovery. Finally, we propose several models based on the relationship between users and the relationship between APIs for IIoT devices and API recommendation.

III. Collaborative Learning for IIoT API Recommendation

A. Framework Overview

Figure 2 shows an overview of our proposed method of collaborative learning for IIoT API recommendation; this method includes a user-oriented recommendation model, an API-oriented recommendation model, and an ensemble recommendation model. The core problem is how to implement implicit knowledge discovery to improve IIoT API recommendations.

Module 1 (User-oriented recommendation). The similarity between a user and a set of friends of the user is computed and inserted into the reference matrix as the regularization term. Next, the user matrix with social regularization is obtained as the result of user-oriented recommendation.

Module 2 (API-oriented recommendation). The similarity between the IIoT API and the regularization terms in the reference matrix is computed from the new auxiliary angle of IIoT API records. In contrast to Module 1, we focus on the IIoT API to calculate the similarity.

Module 3 (Ensemble recommendation). In this step, when the user social matrix and the API social matrix are ready, the two matrices have a different significance for the ensemble model; therefore, we assign different weights to the two matrices to obtain the minimum loss value.

B. MF

Now, we discuss the basic user-API matrix. For m users and n APIs, we build a matrix in which the elements indicate whether the user is following the API. Thus, the values in the matrix can be 1 or 0, thus indicating following or not following, respectively. Let $R^{m\times n}$ denote the user-API matrix, R_{ij} represent the value of user i for API j, $U^{m\times l}$ be the user latent feature matrix, and $A^{n\times l}$ be the API latent feature matrix. U_i denotes the user latent feature vector, and A_j denotes the API latent feature vector. The goal is to minimize the error between the predicted value and the real value; thus, the basic objective function is to minimize the squared error as follows:

$$\min_{U,V} L_{basic}(R,U,V) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i^T A_j)^2 \quad (1)$$

To alleviate the overfitting problem that may occur during training, especially in cases where the training data are sparse, several regularization terms can be added to the basic objective function. Therefore, the final objective function of MF is constructed as

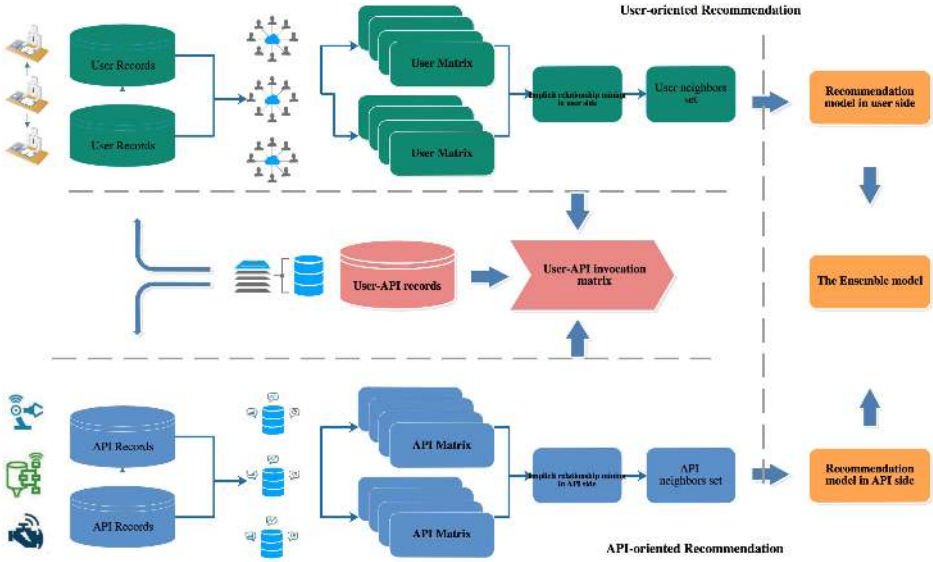


Fig. 2. The overall framework of our proposed model.

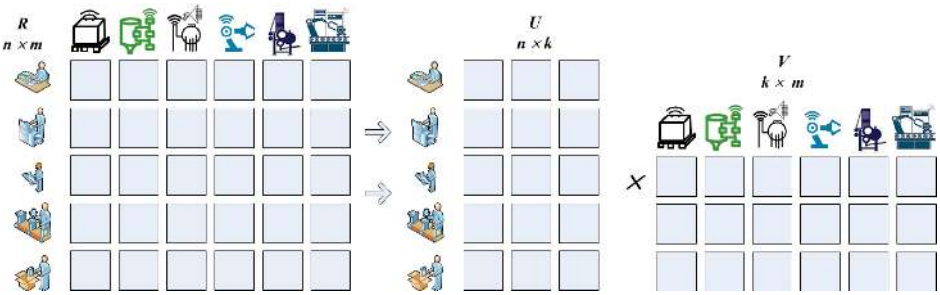


Fig. 3. The IIoT example for MF.

follows:

$$\min_{U,V} L_{basic}(R,U,V) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i^T A_j)^2 + \frac{\lambda}{2}\|U_i\|_F^2 + \frac{\lambda}{2}\|A_j\|_F^2 \quad (2)$$

Figure 3 shows an example of applying the MF method to IIoT data. The local minimum of the objective function given by Eq. 2 can be obtained by using gradient descent in the feature vectors of U and A:

$$\frac{\partial L_{basic}}{\partial U_i} = \sum_{j=1}^{n}(R_{ij} - U_i^T A_j)A_j + \lambda U_i$$
$$\frac{\partial L_{basic}}{\partial A_j} = \sum_{i=1}^{m}(R_{ij} - U_i^T A_j)U_i + \lambda A_j \quad (3)$$

C. API Recommendation With User Relations Mining

1) User Relations Mining: We assume that there is a relationship between two users if and only if the two users follow the same API. For user-API item pairs, we can obtain the similarity of user interests by calculating the similarity between user behaviors. Given user i and user f, let $N_(i)$ denote the set of APIs that user i has acted on, and let $N_(f)$ be the set of APIs on which user f has acted. Then, we can simply calculate the interest similarity of i and f by using the Jaccard formula:

$$Sim(i,f) = \frac{|N_{(i)} \cap N_{(f)}|}{|N_{(i)} \cup N_{(f)}|} \quad (4)$$

In the real word, there are likely to be hundreds or thousands of similar users with different degrees of similarity. Usually people are more willing to trust recommendations from highly similar users. On the other hand, decisions can be made based on the opinions of different users. Thus, we propose a regularization term for user relationships, defined as Eq. 5:

$$\frac{\lambda_u}{2}\sum_{i=1}^{m}\left\|U_i - \frac{\sum_{f\in F^+(i)} Sim(i,f)\times U_f}{\sum_{f\in F^+(i)} Sim(i,f)}\right\|_F^2 \quad (5)$$

where $F^+(i)$ denotes the set of similar users (each denoted as user U_i) and the number of users in the set is $|F^+(i)|$. User U_i represents the latent feature vector of user i. The regularization term aims to minimize the difference between the user interest and the average value of the user interest of the similar user set. Moreover, in the similar user set, users with high similarity should contribute more to the interest orientation. Figure 4 shows

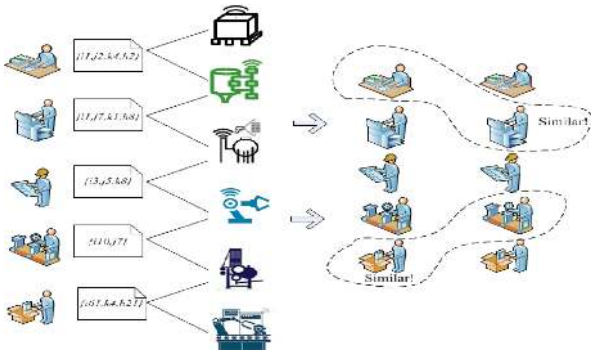


Fig. 4. An example of a similarity process among users.

an example of a similarity process in which similar users can be identified from among users.

2) MF With Users' Implicit Knowledge: We extend the basic MF model by adding user relationships to the MF objective function in Eq. 2 in the form of regularization terms. We obtain the user relation-based MF objective function:

$$\begin{aligned}\min_{U,V} L_{user}(R,U,V) &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij}-U_i^T A_j)^2 \\ &+ \frac{\lambda_u}{2}\sum_{i=1}^{m}\left\|U_i - \frac{\sum_{f\in F^+(i)} Sim(i,f)\times U_f}{\sum_{f\in F^+(i)} Sim(i,f)}\right\|_F^2 \\ &+ \frac{\lambda}{2}\left\|U_i\right\|_F^2 + \frac{\lambda}{2}\left\|A_j\right\|_F^2 \end{aligned} \tag{6}$$

Obtaining the global minimum of Eq. 6 by optimizing U_i and A_j is an NP-hard (nondeterministic-polynomial-time-hard) task. Therefore, we try to generate the local minimum of the objective function in Eq. 6 by using gradient descent to learn all vectors U_i and A_j. The detailed partial derivatives over U_i and A_j are as follows:

$$\begin{aligned}\frac{\partial L_{user}}{\partial U_i} &= \sum_{j=1}^{n}(R_{ij}-U_i^T A_j)A_j + \lambda U_i \\ &+ \lambda_u\left(U_i - \frac{\sum_{f\in F^+(i)} Sim(i,f)\times U_f}{\sum_{f\in F^+(i)} Sim(i,f)}\right) \\ &+ \lambda_u \sum_{g\in F^-(i)} \frac{-Sim(i,g)(U_g - \frac{\sum_{f\in F^+(g)} Sim(g,f)\times U_f}{\sum_{f\in F^+(g)} Sim(g,f)})}{\sum_{f\in F^+(g)} Sim(g,f)} \\ \frac{\partial L_{user}}{\partial A_j} &= \sum_{i=1}^{m}(R_{ij}-U_i^T A_j)U_i + \lambda A_j \end{aligned} \tag{7}$$

where U_g represents those users whose similar neighbors' set includes user i. Using the gradient descent algorithm, we can obtain the local optima of U_i and A_j, and the preference of user i for API j can be predicted to be $R_{ij} \approx U_i^T A_j$.

D. API Recommendation With API Relations Mining

1) API Relations Mining: For API relations, we propose computing the similarity between APIs by using their content

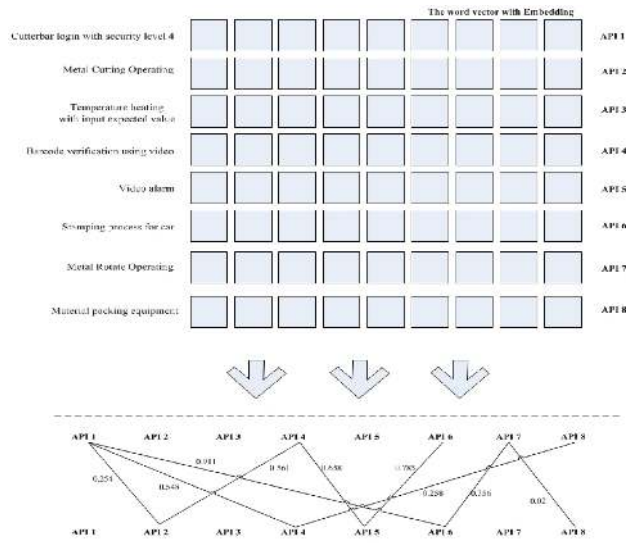


Fig. 5. An example of a similarity process among API word vectors.

information. As the content's introduction contains most of the information on the API, such as the API functionality, category and attributes, we use the introduction to compute the correlation between APIs. As shown in Fig. 5, we first convert the API introduction to a word vector in which these words are embedded into the feature vector. We then calculate the similarity of the API word vector to obtain the API similarity calculation formula, as shown in Eq. 8:

$$Sim(j,s) = \frac{\left|W_{(j)} \cap W_{(s)}\right|}{\left|W_{(j)} \cup W_{(s)}\right|} \tag{8}$$

Given API j and API s, $W_(j)$ denotes the word vector of API j, and $W_(s)$ is the word vector of API s. The degree of similarity between the two APIs reflects their functional similarity to some extent. Then, we propose a regularization term for API relationships, as shown in Eq. 9.

$$\frac{\lambda_a}{2}\sum_{j=1}^{n}\left\|A_j - \frac{\sum_{s\in S^+(j)} Sim(j,s)\times A_s}{\sum_{s\in S^+(j)} Sim(j,s)}\right\|_F^2 \tag{9}$$

where $S^+(j)$ denotes the set of similar APIs of API A_j and the number of APIs in the set is $|S^+(j)|$. Furthermore, API A_j represents the latent feature vector of API j.

2) MF With APIs' Implicit Knowledge: We elaborate the design of the API relation-based MF model.

$$\begin{aligned}\min_{U,V} L_{API}(R,U,V) &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}\left(R_{ij}-U_i^T A_j\right)^2 \\ &+ \frac{\lambda_a}{2}\sum_{j=1}^{n}\left\|A_j - \frac{\sum_{s\in S^+(j)} Sim(j,s)\times A_s}{\sum_{s\in S^+(j)} Sim(j,s)}\right\|_F^2 \\ &+ \frac{\lambda}{2}\left\|U_i\right\|_F^2 + \frac{\lambda}{2}\left\|A_j\right\|_F^2 \end{aligned} \tag{10}$$

Similar to Eq. 7, obtaining the global minimum of Eq. 10 is an NP-hard problem; we try to obtain the local minimum of Eq. 10

by using gradient descent. The detailed partial derivatives over U_i and A_j are as follows.

$$\frac{\partial L_{API}}{\partial U_i} = \sum_{j=1}^{n} (R_{ij} - U_i^T A_j) A_j + \lambda U_i$$

$$\frac{\partial L_{API}}{\partial A_j} = \sum_{i=1}^{m} (R_{ij} - U_i^T A_j) U_i + \lambda A_j + \lambda_a \left(A_j - \frac{\sum_{s \in S^+(j)} Sim(j,s) \times A_s}{\sum_{s \in S^+(j)} Sim(j,s)} \right) + \lambda_a \sum_{h \in S^-(j)} \frac{-Sim(j,h) \left(A_j - \frac{\sum_{s \in S^+(j)} Sim(h,s) \times A_s}{\sum_{s \in S^+(j)} Sim(h,s)} \right)}{\sum_{s \in S^+(j)} Sim(h,s)} \tag{11}$$

where A_h represents the APIs whose similar neighbors sets include API j. With the generated local optima U_i and A_j, the user preference can be predicted to be $R_{ij} \approx U_i^T A_j$.

E. The Ensemble Recommendation Model

After obtaining two independent models, we consider whether we can improve the recommendation accuracy by combining the two models to take advantage of their respective strengths. Therefore, based on the user-based and API-based model, we obtain the final ensemble model, which is a linear combination of the two individual models:

$$L_{ensemble}(R,U,V) = \beta \times L_{user}(R,U,V) + (1-\beta) L_{API}(R,U,V) \tag{12}$$

where β represents the weight of the model and the value of β is obtained experimentally. Algorithm 1 shows the process of the ensemble recommendation model, where the first two steps aim to obtain the user-model and API model and the latter step generates the hybrid model as the ensemble model. In Algorithm 1, line 2 computes the similarity between two users, and line 5 computes the similarity between two APIs. Line 9 computes the local minimum of the proposed user-oriented MF model and generates the local optima of U_i and A_j. Line 15 computes the local minimum of the proposed API-oriented MF model. Line 20 denotes the ensemble model.

Algorithm 1: The Algorithm For Ensemble Model Recommendation (EMR).

Input: List $R^{m \times n}$ of userâĂŞIoT API pairs and a set W of API introductions.
Output: The ensemble model $L_{ensemble}(R,U,V)$ for IoT API recommendation

1 **for** $u_i \in U$ **do**
2 $\quad us[i] \leftarrow \text{sim}(i,f) = \frac{|N_{(i)} \cap N_{(f)}|}{|N_{(i)} \cup N_{(f)}|}$;
3 **end**
4 **for** $w_j \in W$ **do**
5 $\quad ws[j] \leftarrow \text{sim}(j,s) = \frac{|N_{(j)} \cap N_{(s)}|}{|N_{(j)} \cup N_{(s)}|}$;
6 **end**
7 **for** *MF model* $\min_{U,V} L_{user}(R,U,V)$ *with set us* **do**
8 $\quad$ **if** $\frac{\partial L_{user}}{\partial U_i}$ *is minimal and* $\frac{\partial L_{user}}{\partial A_j}$ *is minimal* **then**
9 $\quad\quad L_{user}(R,U,V) \leftarrow \min_{U,V} L_{user}(R,U,V)$;
10 $\quad\quad$ break;
11 $\quad$ **end**
12 **end**
13 **for** *MF model* $\min_{U,V} L_{API}(R,U,V)$ *with set ws* **do**
14 $\quad$ **if** $\frac{\partial L_{API}}{\partial U_i}$ *is minimal and* $\frac{\partial L_{API}}{\partial A_j}$ *is minimal* **then**
15 $\quad\quad L_{API}(R,U,V) \leftarrow \min_{U,V} L_{API}(R,U,V)$;
16 $\quad\quad$ break;
17 $\quad$ **end**
18 **end**
19 **for** $\beta \in [0,1]$ **do**
20 $\quad L_{\text{ensemble}}(R,U,V) = \beta \times L_{\text{user}}(R,U,V) + (1-\beta) L_{API}(R,U,V)$;
21 $\quad$ **if** $L_{ensemble}(R,U,V)$ *is minimal* **then**
22 $\quad\quad \text{res} \leftarrow L_{\text{ensemble}}(R,U,V)$
23 $\quad$ **end**
24 **end**
25 return res

IV. Experiments and Evaluation

To evaluate the performance of our proposed method, we conducted a series of experiments to compare 9 other traditional recommendation methods. In these experiments, the value of each matrix is initially defined as 0 or 1; that is, a user following the IIoT API is denoted as 1; otherwise, the value is set to 0. In Industry 4.0, factory components/devices can virtually "talk" and "negotiate" with each other to decide for themselves how to best optimize production. Thus, we conduct IIoT API recommendation experiments.

A. Dataset

To simulate an IIoT API dataset, we crawled API data from Programmable Web. This website contains detailed information about almost all current APIs. We crawled data for 17412 APIs, with each API having 5 attributes: the date the API was posted on the site, a group of tags, a short description, a list of followers and a list of developers. We cannot crawl users of the site, but for each API, there is a property called follow, which is similar to the collection feature when we browse an item, just like invoking or operating devices. Therefore, we obtained 140,000 users after collecting and sorting 17412 API followers. The data statistics are given in Table I.

For training sets, validation sets and test sets, we evaluated our models on five data settings, and in so doing, we tried to test the performance comprehensively. We randomly select 90%, 80%, 70%, 60% and 10% data from the whole dataset as training sets, and the remaining data are separated into validation sets and test sets. In detail, in cases where the training set density is 90%, the validation set density is 5% and test set density is also 5%.

TABLE I
DATASET STATISTICS

Dataset	Count
Number of users	17412
Number of APIs	22032
Number of invocation records	248530

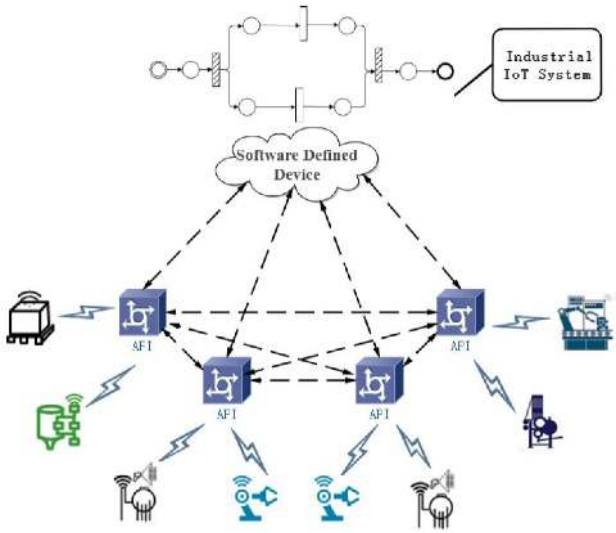


Fig. 6. The SDD industrial system scenario using IoT APIs.

For the rest of the four settings, the validation set accounts for 10%, and the remaining data compose the test set. For example, when the training set density is 10%, the validation set density is 10%, and the test set density is 80%. We performed each group of experiments 10 times and obtained the mean absolute error (MAE) and root-mean-square error (RMSE) for all groups.

B. Industrial Scenario Example of an Experiment

Figure 6 shows an example of an industrial scenario in which an SDD calls for a workflow in which Agents organize different industrial components to work together. We simulate this scenario and support IIoT API recommendation. Suppose that each API collected from the real-word dataset is considered to be an IIoT API that will have a corresponding IIoT device supporting specified industrial services. Thus, the preprocessed data are manually marked and used in this simulation experiment for an IIoT system.

C. Evaluation Metrics

Recommendation is a kind of prediction method. Thus, our IIoT API recommendation experiments aim to assess prediction accuracy. The MAE and RMSE are employed as the evaluation metrics:

$$MAE = \frac{1}{N}\sum_{i,j}\left|R_{i,j} - \hat{R}_{ij}\right| \tag{13}$$

$$RMSE = \sqrt{\frac{1}{N}\sum\nolimits_{i,j}\left(R_{ij} - \hat{R}_{ij}\right)^2} \tag{14}$$

where R_{ij} denotes the rating user i gave to item j, $\hat{R}_{ij}$ denotes the rating user i gave to item j as predicted by the specified method, and N denotes the number of tests.

D. Parameter Setting

Our model is based on the LFM, and the main parameters affecting the model include the following: The first parameter is the number of latent features: if this value is too large, the model is not interpretable, and if the value is too small, the model training will be negatively affected. Therefore, we set the number of latent features to 10. Moreover, a regularization term is introduced to prevent overfitting. The regularization parameter λ adjusts the weight of the regularization term. In our proposed model, there are three regularization parameters: λ is set to 0.1, and the user regularization term λ_u and API regularization term λ_a are each set to 0.01.

E. Performance Comparison

We implemented the following methods to evaluate the performance of our method; the experimental results are given in Table II.

1) UPCC (user-based Pearson correlation coefficient). User-based CF using the PCC to calculate users similarity [26].

2) IPCC (item-based PCC). Item-based CF using the PCC to calculate user similarity [27].

3) WSRec (web service recommendation). A linear combination of the prediction results of the UPCC and IPCC [23].

4) NMF (nonnegative MF). Approximation of a nonnegative matrix as the product of two nonnegative matrices [33].

5) LR (logistic regression). Used to estimate the probability that an instance belongs to a particular category.

6) Autoencoder. A multilayer neural network that is often used for dimensionality reduction in feature learning.

7) MF (matrix factorization). Connects user interests and items through hidden features and adopts automatic clustering based on user behavior statistics [34].

8) AR-URM (API recommendation with user relations mining). Latent factor model using user similarity as the regularization term.

9) AR-ARM (API recommendation with API relations mining). Latent factor model using API similarity as the regularization term.

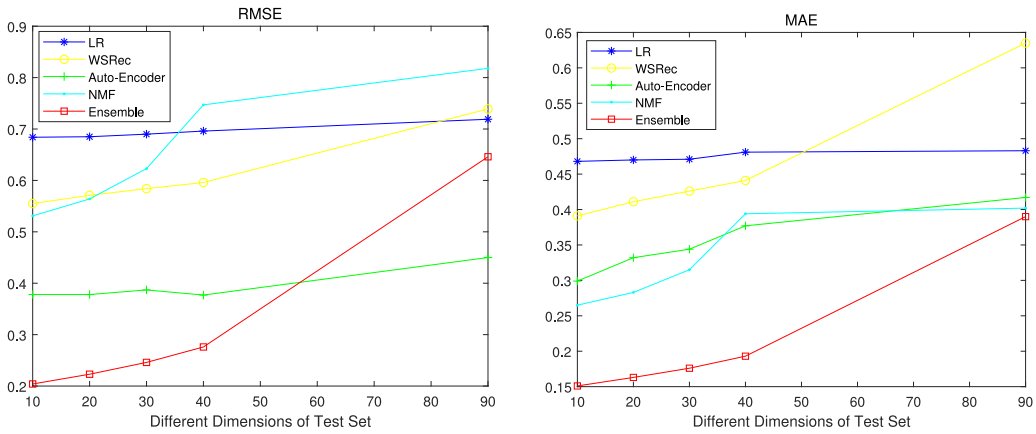
10) Ensemble. Our method, which combines the user-oriented model and API-oriented model into a linear combination.

Table II presents the experimental results, and we have the following observations and findings:

1) The recommendation error of all methods is evaluated at different training set densities, which are 90%, 80%, 70%, 60% and 10%. Our three proposed models achieve the lowest errors at all data densities.
2) The training set density significantly affects the error. For example, when the training set density is 70% as compared to when the training set density is 90%, the errors of all methods are higher; the higher errors indicate that the model is not trained properly and that the data do not fit the model well, thus resulting in large errors.

TABLE II
A PREDICTION ACCURACY COMPARISON GROUPED BY TEST SET DENSITY (A SMALLER VALUE INDICATES BETTER PERFORMANCE)

Model	Training set density (T)									
	T = 90%		T = 80%		T = 70%		T = 60%		T=10%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	0.488	0.656	0.501	0.665	0.515	0.679	0.529	0.691	0.727	0.835
IPCC	0.958	0.968	0.960	0.970	0.961	0.971	0.965	0.974	0.987	0.991
WSRec	0.391	0.555	0.411	0.571	0.426	0.584	0.441	0.596	0.635	0.739
NMF	0.265	0.531	0.283	0.564	0.315	0.623	0.395	0.747	0.402	0.818
LR	0.468	0.684	0.469	0.685	0.471	0.690	0.481	0.696	0.483	0.719
Autoencoder	0.299	0.378	0.332	0.378	0.344	0.387	0.377	0.377	0.417	0.450
LFM	0.224	0.320	0.235	0.341	0.253	0.374	0.266	0.395	0.510	0.763
AR-URM	0.159	0.216	0.173	0.241	0.187	0.268	0.203	0.295	0.447	0.686
AR-ARM	0.157	0.213	0.171	0.236	0.185	0.262	0.202	0.293	0.448	0.688
Ensemble	**0.151**	**0.204**	**0.163**	**0.223**	**0.176**	**0.246**	**0.193**	**0.276**	**0.424**	**0.646**



(a) Impact on RMSE (b) Impact on MAE

Fig. 7. The impact of the data density of the training set. (a)Impact on RMSE. (b)Impact on MAE.

3) The ensemble recommendation model is consistently superior to the two individual models because the ensemble model utilizes all implicit knowledge among users and APIs.
4) The last case (training set density of 10%) evaluates the performance of different methods on few training data, i.e., the cold-start case. Our models also perform best in this scenario.

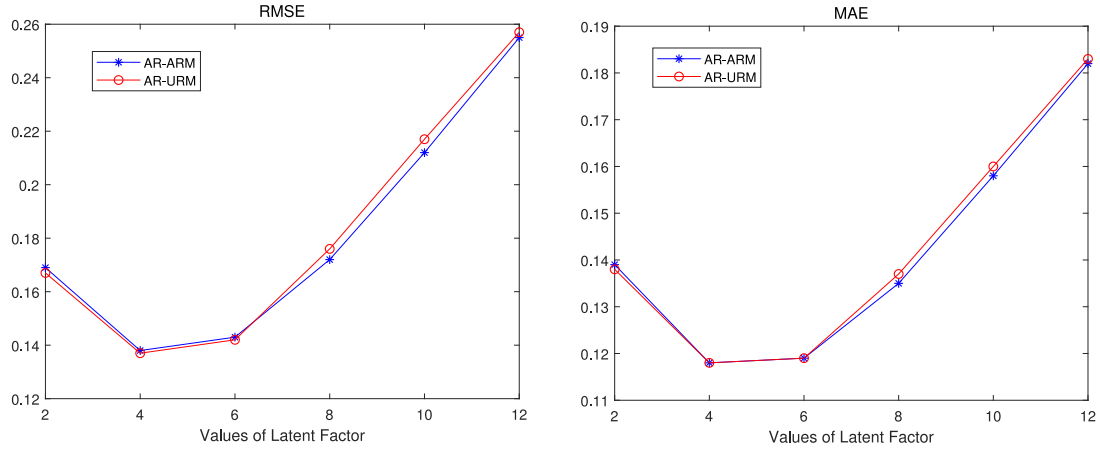
F. Sensitivity Analysis of Parameters

1) Impact of the Data Density of the Training Set: To more intuitively illustrate the performance of different methods under different training set densities, we drew a line chart showing the performance of several typical methods, as shown in Figure 7. Figure 7 shows the RMSE and MAE values of the five methods we selected for different training set densities. The trend of NMF is relatively stable, and relatively stable performance is maintained under different test set densities. By contrast, although logistic regression is also relatively stable, the error is relatively large. When the test set density is 90%, the RMSE of the autoencoder and the MAE of WSRec are the highest. The error of our proposed method is generally lower than that of the other algorithms, and our method performs well. As a result, our method can provide better IIoT API recommendation not only in the data sparsity scenario but also in the big data scenario in which the IIoT system has the features of big data.

2) Impact of the Number of Latent factors: The value of the latent factor determines the dimension of the matrix after factorization. To identify the optimal value, we experimented with the conventional range of latent factors. Figure 8 shows the RMSE and MAE values of AR-URM and AR-ARM under different latent factor values. When the value of the latent factor is 4, the error of the two models reaches the minimum; however, in papers on MF, the most common latent factor value is usually 10, so we also use 10 as the value of the latent factor.

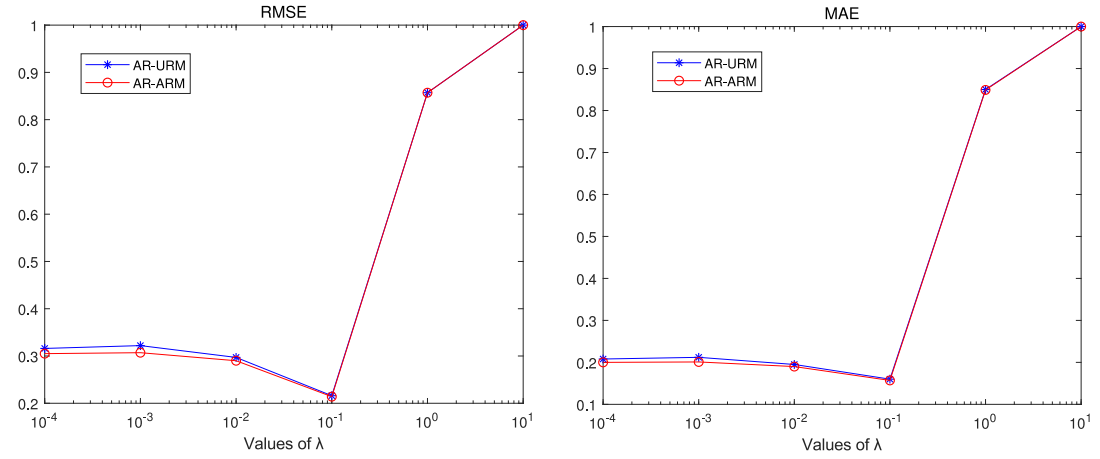
3) Impact of Parameter λ: The regularization parameter regulates the influence of the regularization term on the model. Figure 9 shows the results of the AR-URM and AR-ARM under different values of λ: the error is smallest when λ is 0.1.

4) Impact of Parameter β: In the combination of AR-ARM and AR-URM, how large a role each should play individually to achieve the best effect of the ensemble model can be determined experimentally. Figure 10 shows the RMSE and MAE values of the ensemble model for different weights. When the weight is 0.7, that is, when the AR-URM accounts for 70% and the AR-ARM accounts for 30%, the RMSE and MAE of the ensemble model reach the minimum values.



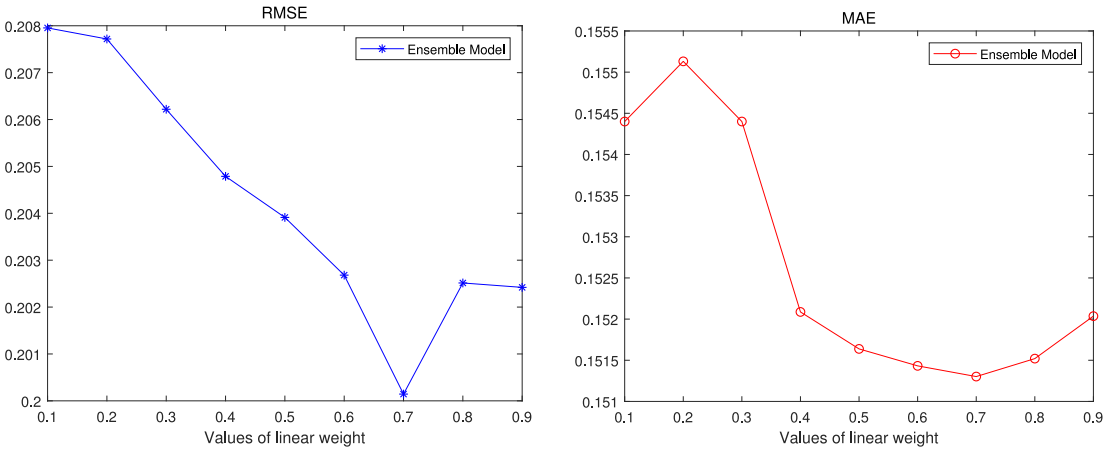
(a) Impact on RMSE (b) Impact on MAE

Fig. 8. The impact of the latent factor. (a)Impact on RMSE. (b)Impact on MAE.



(a) Impact on RMSE (b) Impact on MAE

Fig. 9. The impact of λ. (a)Impact on RMSE. (b)Impact on MAE.



(a) Impact on RMSE (b) Impact on MAE

Fig. 10. The impact of the linear weight β. (a)Impact on RMSE. (b)Impact on MAE.

The experimental results in Fig. 10 demonstrate that our method with appropriate parameters can be used for IIoT API recommendation. We believe that the API-empowered IIoT framework will continue to flourish with the establishment of ubiquitous connections among IoT devices.

V. Conclusion

The proliferation of IoT devices has led to revolutionary changes in industrial systems that have transformed how humans interact with and control the physical world. However, these heterogeneous systems and platforms need a unified management application to operate and control IIoT devices, and how to interact with IIoT devices has become an important problem. APIs represent one solution for supporting SDDs and are a ubiquitous operating mode for IIoT systems. Considering API recommendation in the IIoT environment, we propose an approach that combines collaborative learning with implicit knowledge discovery. We use MF and add regularization terms to fuse user similarity and item similarity. Then, these two models are combined via linear combination to generate the final recommendation model. The experimental results show that our approach is superior to other CF methods.

In future research, we will continue to further optimize our models and explore the possible effects of different parameter ratios for different application scenarios. We will also employ machine learning or transfer learning [35] in our method to implement real-time safety checking [36]–[38] and exception prediction for IIoT systems. The safety and reliability of IIoT challenges should be verified by using formal methods, such as model checking, theorem-proving and formal testing. Verification is another direction of our research on SDD-oriented IIoT systems.

References

[1] H. Xue and D. Zhang, "A recommendation model based on content and social network," in *IEEE Joint Int. Inform. Technol. and Artif. Intell. Conf. (ITAIC)*, 2019, pp. 477–481.

[2] "Toward service selection for workflow reconfiguration: An interface-based computing solution," *Future Gener. Comput. Syst.*, vol. 87, pp. 298–311, 2018.

[3] B. Xia, Y. Fan, W. Tan, K. Huang, J. Zhang, and C. Wu, "Category-aware API clustering and distributed recommendation for automatic mashup creation," *IEEE Trans. Serv. Comput.*, vol. 8, no. 5, pp. 674–687, Sep./Oct. 2015.

[4] F. Thung, R. J. Oentaryo, D. Lo, and Y. Tian, "WebAPIRec: Recommending web APIS to software projects via personalized ranking," *IEEE Trans. Emerging Topics Comput. Intell.*, vol. 1, no. 3, pp. 145–156, 2017.

[5] M. J. Pazzani, and D. Billsus, *Content-Based Recommendation Syst.*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 325–341.

[6] H. Liang, Y. Xu, Y. Li, and R. Nayak, "Collaborative filtering recommender systems using tag information," in *Int. Conf. Web Intell. and Intell. Agent Technol.*, vol. 3, 2008, pp. 59–62.

[7] Z. Jia, Y. Yang, W. Gao, and X. Chen, "User-based collaborative filtering for tourist attraction recommendations," in *IEEE Int. Conf. Comput. Intell. Commun. Technol.*, 2015, pp. 22–25.

[8] X. Shi, H. Ye, and S. Gong, "A personalized recommender integrating item-based and user-based collaborative filtering," in *Int. Semin. Bus. and Inform. Manag.*, 2008, pp. 264–267.

[9] N. N. Liu and Q. Yang, "Eigenrank: A ranking-oriented approach to collaborative filtering," in *Proc. Annu. Int. ACM SIGIR Conf. Res. and Develop. Inform. Retrieval (SIGIR)*, 2008, pp. 83–90.

[10] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen, "Scalable collaborative filtering using cluster-based smoothing," in *Proc. Annu. Int. ACM SIGIR Conf. Res. and Develop. Inform. Retrieval (SIGIR)*, 2005, p. 114âĂŞ121.

[11] T. Hofmann, "Collaborative filtering via Gaussian probabilistic latent semantic analysis," in *Proc. Annu. Int. ACM SIGIR Conf. Res. and Develop. Inform. Retrieval (SIGIR)*, 2003, pp. 259–266.

[12] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proc. Annu. Int. ACM SIGIR Conf. Res. and Develop. Inform. Retrieval (SIGIR)*, 2002, pp. 238–245.

[13] A. T. Nguyen, *et al*, "API code recommendation using statistical learning from fine-grained changes," in *Proc. ACM SIGSOFT Int. Symp. Foundations Softw. Eng.*, 2016, pp. 511–522.

[14] P. T. Nguyen, J. Di Rocco, D. Di Ruscio, L. Ochoa, T. Degueule, and M. Di Penta, "FOCUS: A recommender system for mining API function calls and usage patterns," in *Proc. 41st Int. Conf. Softw. Eng.*, 2019, pp. 1050–1060.

[15] J. Liu, Y. Qiu, Z. Ma, and Z. Wu, "Autoencoder based API recommendation system for android programming," in *The 14th Int. Conf. Comput. Sci. Edu. (ICCSE)*, 2019, pp. 273–277.

[16] B. Cao, J. Liu, M. Tang, Z. Zheng, and G. Wang, "Mashup service recommendation based on user interest and social network," in *IEEE Int. Conf. Web Serv. (ICWS)*, 2013, pp. 99–106.

[17] X. Liu and I. Fulia, "Incorporating user, topic, and service related latent factors into web service recommendation," in *Int. Conf. Web Serv. (ICWS)*, 2015, pp. 185–192.

[18] Y. Wan, L. Chen, Q. Yu, T. Liang, and J. Wu, "Incorporating heterogeneous information for mashup discovery with consistent regularization," in *Adv. Knowl. Discovery and Data Mining*, 2016, pp. 436–448.

[19] L. Yao, X. Wang, Q. Z. Sheng, W. Ruan, and W. Zhang, "Service recommendation for mashup composition with implicit correlation regularization," in *Int. Conf. Web Serv. (ICWS)*, 2015, pp. 217–224.

[20] X. Wang, H. Wu, and C. Hsu, "Mashup-oriented API recommendation via random walk on knowledge graph," *IEEE Access*, vol. 7, pp. 7651–7662, 2019.

[21] B. Kwapong and K. Fletcher, "A knowledge graph based framework for web API recommendation," in *IEEE World Congr. Serv. (SERVICES)*, 2019, pp. 115–120.

[22] Y. Wan, L. Chen, J. Wu, and Q. Yu, "Time-aware API popularity prediction via heterogeneous features," in *IEEE Int. Conf. Web Serv.*, 2015, pp. 424–431.

[23] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "WSRec: A collaborative filtering based web service recommender system," in *IEEE Int. Conf. Web Serv.*, 2009, pp. 437–444.

[24] Z. Zheng, H. Ma, M. Lyu, and I. King, "Collaborative web service QOS prediction via neighborhood integrated matrix factorization," *IEEE Trans. Serv. Comput.*, vol. 6, no. 3, pp. 289–299, 2013.

[25] F. Xie, L. Chen, D. Lin, C. Chen, Z. Zheng, and X. Lin, "Poster: Group preference based API recommendation via heterogeneous information network," in *IEEE/ACM 40th Int. Conf. Softw. Eng. (ICSE)*, 2018, pp. 362–363.

[26] S. Ma, H. Lin, C. Yu, and C. Lee, "Web API recommendation based on service cooperative network," in *Int. Conf. Appl. System Innovation (ICASI)*, 2017, pp. 1922–1925.

[27] H. Sun, Z. Zheng, J. Chen, W. Pan, C. Liu, and W. Ma, "Personalized open API recommendation in clouds via item-based collaborative filtering," in *Int. Conf. Utility and Cloud Comput.*, 2011, pp. 237–244.

[28] K. Fletcher, "Regularizing matrix factorization with implicit user preference embeddings for web API recommendation," in *Int. Conf. Serv. Comput. (SCC)*, 2019, pp. 1–8.

[29] Y. Chen, M. Zhou, Z. Zheng, and D. Chen, "Time-aware smart object recommendation in social internet of things," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2014–2027, Mar. 2020.

[30] S. Duan, D. Zhang, Y. Wang, L. Li, and Y. Zhang, "JointRec: A deep-learning-based joint cloud video recommendation framework for mobile IoT," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1655–1666, Mar. 2020.

[31] Z. Huang, X. Xu, J. Ni, H. Zhu, and C. Wang, "Multimodal representation learning for recommendation in internet of things," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 675–10 685, Dec. 2019.

[32] H. Hu, B. Tang, Y. Zhang, and W. Wang, "Vehicular ad hoc network representation learning for recommendations in internet of things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2583–2591, Apr. 2020.

[33] M. N. Schmidt, O. Winther, and L. K. Hansen, "Bayesian non-negative matrix factorization," in *Independent Component Anal. and Signal Separation*, T. Adali, C. Jutten, J. M. T. Romano, and A. K. Barros, Eds., 2009, pp. 540–547.

[34] P. Symeonidis and A. Zioupos, *Related Work Matrix Factorization*. Springer International Publishing, 2016, pp. 19–31.
[35] S. Deng, Z. Xiang, P. Zhao, J. Taheri, H. Gao, J. Yin, and A. Y. Zomaya, "Dynamical resource allocation in edge for trustable internet-of-things systems: A reinforcement learning method," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6103–6113, Sep. 2020.
[36] H. Gao, W. Huang, Y. Duan., X. Yang, and Q. Zou, "Research on cost-driven services composition in an uncertain environment," *J. Internet Technol.*, vol. 20, no. 3, pp. 755–769, 2019.
[37] H. Gao, D. Chu, Y. Duan, and Y. Yin, "Probabilistic model checking-based service selection method for business process modeling," *Int. J. Softw. Eng. and Knowl. Eng.*, vol. 27, no. 6, pp. 897–924, 2017.
[38] 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. and Knowl. Eng.*, vol. 28, no. 10, pp. 1369–1397, 2018.

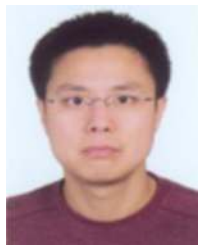
Honghao Gao (Senior Member, IEEE) received the Ph.D. degree in computer science and started his academic career at Shanghai University, Shanghai, China, in 2012. He is currently with the School of Computer Engineering and Science, Shanghai University. He is also a Professor with Gachon University, Seongnam, South Korea. He was a Research Fellow with the Software Engineering Information Technology Institute, Central Michigan University, Mount Pleasant, MI, USA, and was also an Adjunct Professor with Hangzhou Dianzi University, Hangzhou, China. His research interests include service computing, model checking based software verification, wireless networks, and intelligent medical image processing. He has publications in IEEE TII, IEEE T-ITS, IEEE IoT-J, IEEE TNSE, IEEE TCCN, IEEE/ACM TCBB, ACM TOIT, ACM TOMM, IEEE TCSS, IEEE TETCI, IEEE NETWORK, and IEEE JBHI. He is a Fellow of IET, BCS, and EAI, and a Senior Member of CCF and CAAI. He is the Editor-in-Chief for the *International Journal of Intelligent Internet of Things Computing*, and an Associate Editor for the *IET Software*, *Wireless Network*, *International Journal of Communication Systems*, *IET Wireless Sensor Systems*, IET *The Journal of Engineering*, and *Journal of Medical Imaging and Health Information*. Moreover, he has broad working experiences in industry-university-research cooperation. He is a European Union Institutions appoint external expert for reviewing and monitoring EU Project, is a member of the EPSRC Peer Review Associate College for UK Research and Innovation in the U.K., and is also a founding member of the IEEE Computer Society Smart Manufacturing Standards Committee.

Xi Qin received the bachelor's degree from the North University of China, Taiyuan, China. She is currently working toward the master's degree with the School of Computer Science and Technology, Xidian University, Xi'an, China. Her current research interests include recommender systems and service computing.

Ramón J. Durán Barroso received the telecommunication engineering and Ph.D. degrees from the University of Valladolid, Valladolid, Spain, in 2002 and 2008, respectively. He is currently an Associate Professor with the University of Valladolid, where he serves as a Deputy Director of the Faculty of Telecommunication Engineering. He has authored more than 100 articles published in international journals and conferences. His current research focuses on the use of artificial intelligence techniques for the design, optimization, and operation of future heterogeneous networks, mobile edge computing, and network function virtualization. He is the Coordinator of the Spanish Research Thematic Network (Go2Edge: Engineering Future Secure Edge Computing Networks, Systems and Services), which comprises 15 entities.

Walayat Hussain received the Ph.D. degree from the University of Technology Sydney, Ultimo, NSW, Australia. He was a Lecturer and an Assistant Professor with the Balochistan University of Information Technology, Engineering, and Management Sciences for many years. He is currently a Lecturer with the Faculty of Engineering and Information Technology, University of Technology Sydney. He has authored or coauthored articles in various top-ranked reputable journals and conferences, such as the *Computer Journal*, *Information Systems*, IEEE ACCESS, *Future Generation Computer Systems*, *Computers and Industrial Engineering*, *Mobile Networks and Applications*, *Journal of Ambient Intelligence and Humanized Computing*, IEEE International Conference on Fuzzy Systems, and International Conference on Neural Information Processing. His research areas include business intelligence, cloud computing, and usability engineering by focusing on providing an informed decision to different stakeholders. He was the recipient of one national and three international and research awards and recognitions for his research. He was also the recipient of a 2016 Faculty of Engineering and IT Higher Degree by Research Publication Award from the University of Technology Sydney.

Yueshen Xu (Senior Member, IEEE) received the Doctorate degree from Zhejiang University, Hangzhou, China. He was a co-trained Ph.D. Student with the University of Illinois at Chicago, Chicago, IL, USA. He is currently a Lecturer with the School of Computer Science and Technology, Xidian University, Xi'an, China. He has authored/coauthored more than 30 papers in international journals or conferences, such as *Expert Systems with Applications*, *Educational Advances in Artificial Intelligence*, *Web Information Systems Engineering*, *Web-Age Information Management*, and *Sensors*. His research interests include recommender systems, text mining, and service computing. He is a member of ACM and CCF. He is a Reviewer for several journals and conferences, including IEEE TRANSACTIONS ON SERVICES COMPUTING, *Knowledge Based Systems*, IEEE ACCESS, *Journal of Network and Computer Applications*, and Neurocomputing and International Conference on Distributed Computing Systems.



Yuyu Yin received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2010. He is currently an Associate Professor with the College of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China, and is engaged as a Supervisor of master's students with the School of Computer Engineering and Science, Shanghai University, Shanghai, China. During the past ten years, he has authored or coauthored more than 40 papers in journals and has served as a referee in conferences, such as *Sensors*, *Entropy*, *International Journal of Software Engineering and Knowledge Engineering* (IJSEKE), *Mobile Information Systems*, International Conference on Web Services, and Software Engineering and Knowledge Engineering. His research interests include service computing, cloud computing, and business process management. He is a member of the CCF and a CCF Service Computing Technical Committee member. He has organized more than ten international conferences and workshops, such as Formal Methods in Services and Cloud Computing from 2011 to 2017 and Digital Intelligence for Systems and Machines from 2012 to 2018. Additionally, he worked as a Guest Editor for the *Journal of Information Science and Engineering* and IJSEKE and as a Reviewer for the IEEE TRANSACTION ON INDUSTRIAL INFORMATICS, *Journal of Database Management*, *Future Generation Computer Systems*, etc.