

Received 6 October 2023, accepted 23 October 2023, date of publication 27 October 2023, date of current version 2 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3328217

RESEARCH ARTICLE

Cloud Computing Product Service Scheme Recommendation System Based on a Hierarchical Knowledge Graph

SHULIN XU¹, ZIYANG WU², CHUNYU SHI¹, AND MENGJU SUN¹

¹China Telecom Research Institute, Beijing 102209, China

²Donald Bren School of Information and Computer Science, University of California at Irvine, Irvine, CA 92697, USA


Corresponding author: Shulin Xu (xusl5@chinatelecom.cn)

ABSTRACT It is difficult for users to understand the complex cloud product information for product selection. Using this information to recommend satisfactory cloud products is a challenge. Previous studies focused on similar information of users and products while neglecting relevance; therefore, they could not create recommendation approaches that account for functional dependencies among cloud products. To overcome this challenge, this study proposes a cloud product set recommendation model based on a hierarchical knowledge graph (KG) with a pre-post correlation of product functionality. There are two main contributions: First, we constructed a cloud product functionality and performance KG using the dependency information of layers and entities to represent complicated pre-post logical connections. The KG was designed according to the cloud service model. Second, we designed an improved PageRank algorithm to obtain the importance weight for each functionality and performance, which replaces the original average method with the proportion of connection weight. We considered the release time of the functionality, launch time of the product, and last update time of the product as crucial factors in the recommendation score to reflect the importance of the functionality and current development stage of the product. Finally, our method recommended a product set based on the weighted scores from the above results. In addition, we constructed a cloud product functionality dataset containing 339 functionalities. The experimental results show that the proposed method can generate a closely related set of products, leading to improved accuracy and higher satisfaction compared to mainstream methods.

INDEX TERMS Product set recommendation, knowledge graph, cloud service, PageRank, cloud product functionality dataset.

I. INTRODUCTION

Cloud computing has attracted the attention of both academia and industry worldwide, and is capable of transforming service provision models throughout the current IT industry [1]. Various industries have adopted cloud platforms for the majority of their business operations. Gartner¹ predicted that spending by cloud terminal users will reach \$600 billion by 2023. The growth of cloud computing has led to the

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

¹Gartner is the most authoritative IT research and consulting company in the world. Website: <https://www.gartner.com/en>

development of a variety of cloud products by different companies, which can be complex and difficult for users to understand and select [2]. Therefore, the construction of a cloud product recommendation system that is interpretable and meets user requirements is essential.

Recommendation systems have made significant progress in e-commerce [3]. However, e-commerce recommendation models have limited effectiveness in cloud product recommendation for the following reasons: First, cloud products are high-tech items that are centered on technological innovation [4]. Consequently, they tend to be inherently complex, and cloud companies often use new names or concepts to show their product originality, making product names and concepts

more difficult to understand than in e-commerce. Second, cloud product recommendation methods generally face data sparsity problems owing to insufficient user rating data and cold-start problems caused by unrated users. Thus, the task of mining complex product information to enable efficient cloud product recommendations that maximize user satisfaction is challenging.

Recent research [5], [6], [7] on recommendation methods has focused on collaborative filtering, content-based filtering, and hybrid recommendation methods. These approaches, which rely on the similarity of user information, user reviews, and product descriptions, often generate cloud products that are too similar or irrelevant, because they ignore the potential dependencies between cloud products. On the other hand, graph-based methods have been proposed to better integrate and summarize information to facilitate recommendations [8]. The knowledge graph (KG) expands the limited information of individual projects, provides rich reference value for recommendations, and brings additional diversity and interpretability to the recommendation results [9]. Reference [2] used KG to apply relevance to a cloud product recommendation model. However, it is constructed by labeling entity relationships based on conceptual similarity, which does not fully exploit the interdependence of different cloud products. In conclusion, the KG method can capture rich semantic information for personalized and accurate recommendations and provide the ability to explain recommendation results, which is suitable for application in cloud-computing product recommendation systems with rich product information and less user information. However, existing recommendation systems, including some KG-based approaches, overlook the inherent technical dependencies of cloud computing products. As a result, they lack the ability to recommend sets of cloud products with dependencies, which cannot meet the user requirements for selecting cloud products to build their usage scenarios.

To overcome the aforementioned obstacles, considering the complexity of cloud product names and concepts and the disregard of dependencies between products in the recommended product set, this study, inspired by the hierarchical model of cloud services, proposes a cloud product set recommendation model based on KG to improve user satisfaction and marketing effectiveness. The main contributions of this study are as follows.

- In this study, a multidimensional KG is constructed by delineating “IaaS”, “PaaS”, “Operation and Maintenance (O&M)”, and “Security” in the hierarchical model of cloud products. Furthermore, the functionality and performance (FP) points of the product serve as middleware to establish a connection between user requirements and cloud products, so as to resolve the mismatch between recommended products and user requirements caused by the ambiguity of the product concept. This study excavates the complex pre-post logical relationship between FPs to reflect the interdependence between products.

- Based on this KG, we introduced an improved PageRank [10] algorithm and product set recommendation algorithm. The improved PageRank algorithm was refined based on the FP correlation and product development stage by adding the relationship and time factor to achieve a more reasonable weight distribution. The proposed product recommendation algorithm is based on the FP weight distribution generated by the improved PageRank, in which the variables and their weighted coefficients consider multiple factors of products and functionalities to suggest a significant and related product set.
- We built a dataset of cloud product FP. Experiments showed that our method was superior to the baseline method.

The remainder of this paper is organized as follows. Section II focuses on related studies on cloud product recommendations. Section III introduces the proposed KG of the cloud product’s FP. Section IV presents the details of the algorithms used in the product set recommendation model. Section V introduces the dataset and experimental results. Finally, Section VI provides a summary of the results.

II. RELATED WORK

Recommendation systems for cloud computing products are challenging, but many recommendation methods have been proposed, which can be grouped into filtering approaches for recommendation systems and KG-based methods [11].

A. FILTERING APPROACHES FOR RECOMMENDATION SYSTEMS

The filtering approaches for recommendation systems are statistical or based on machine learning techniques, which can be roughly divided into three categories: (1) content-based (CB), (2) collaborative filtering-based (CF), and (3) hybrid (common content + collaborative + demographic) [12]. CB is learned to recommend items that are similar in terms of content features to those that the user has liked in the past [13]. Most CB methods use simple models such as keyword matching or the vector space model (VSM) with term frequency-inverse document frequency (TF-IDF) weighting [13]. The CB method has some shortcomings such as limited content analysis, serendipity, and new user problems [14]. In CF, the recommendation system suggests a new item to a user that is consumed by similar users [15]. Common algorithms include similarity measures, factorization matrix (MF [16]), Bayesian [17], [18], [19], and clustering algorithms [20], [21]. This method relies on interactive data, therefore, it suffers from data sparsity and cold-start problems. The hybrid method uses a combination of several recommendation methods and exploits the advantages of each method to improve overall performance [22], [23]. These methods often ignore the relationship between valuable information, which can lead to suboptimal performance, particularly when data sparsity exists.

B. RECOMMENDATION SYSTEMS BASED ON KG

In 2012, Google introduced the term KG [24]. The KG is a heterogeneous structure that stores knowledge as a machine-readable graph, where nodes represent entities and edges represent relationships between entities [25]. In recent years, several researchers have created different KGs combined with different recommendation methods. KG shows great potential for recommendation owing to its well-defined structure and adequate resources. In combination with various studies, KG recommendation methods can be divided into general recommendation methods in multiple domains and specific recommendation methods in specific domains [26]. The general recommendation method has the advantages of wide application and convenience; however, it has the disadvantages of weak relevance and cannot meet the personalized needs of users in specific fields. A domain-specific recommendation method can effectively use the characteristics of a specific domain, and develop a personalized method according to the personalized needs of users to improve user satisfaction and marketing results [27]. On the other hand, from the perspective of implementation technology, KG-based methods can be divided into ontology-based (OB) [28], [29], path-based (PB) and embedding-based (EB). The OB method uses ontologies to model knowledge of users, context, items, and domains. This method does not experience most of the problems associated with conventional recommendation systems, such as cold start and sparsity of rating data [30]. However, it is difficult to represent the relationships between entities. The PB uses paths and the order in which they pass to recommend projects, which requires sufficient knowledge for construction [31]. In recent years, KG embedding methods have been proposed, which typically involve deep-learning methods. However, deep-learning methods are essentially “black-box” tools, where all intermediate relationships cannot reflect the mechanisms underlying the observed variables [32].

There are many recommendation methods that build user interest models based on KG, however, few have focused on cloud products. In [33], [34], [35], and [36], paths in the KG were extracted, the similarity between items and users on the path was computed and regularized, and user preferences were extracted for recommendation. However, this method still requires a large amount of historical user data to build the rating data, and it still has the disadvantage of collaborative filtering. In [37] and [38], entity embedding and relationship embedding were used to generate the path sequence of a knowledge map. Inspired by NLP, this method embeds nodes in a low-dimensional vector space to learn the implicit characteristics of nodes that are close to each other. However, the constructed path recommendation method does not meet the requirements of cloud product recommendations. In [39], a cloud concept KG was constructed for cloud product recommendation, but only manually built the simplest relationship to connect entities, and did not explore further by building more complex semantics in the relationship.

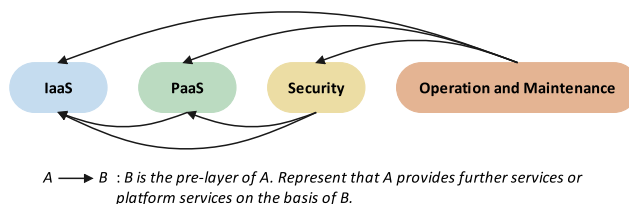


FIGURE 1. Structure of cloud service layer dependency.

The above studies focused on similar characteristics of users, items, and destination paths, which often resulted in the recommended outcome being collected with similar products. These KG-based methods do not fully exploit the pre-post relationships between entities in the KG. In addition, most of the aforementioned recommended methods do not model the relationship between the requirements and products of cloud products; therefore, they cannot meet the personalized requirements recommended by cloud products. Based on the related studies described above, to overcome the poor effectiveness of the recommended product solutions by ignoring the technical relevance, we focus on the specificity of cloud users and cloud products and construct a KG based on the cloud service model [40]. At the same time, we will improve the PageRank algorithm and recommend related product sets based on the complex relationship between before and after cloud product functionalities to improve user satisfaction and marketing effectiveness.

III. CLOUD PRODUCT FUNCTIONALITY AND PERFORMANCE KNOWLEDGE GRAPH

We use the cloud product functionality and performance knowledge graph (CFPKG) to solve the cloud product concept ambiguity problem. We discovered two obvious rules: (1) Although the names and concepts of cloud products are confusing, the FPs of the same cloud products are similar. (2) The essence of product dependency is that the functional logic of a product has a pre-post relationship. Therefore, we built a CFPKG that describes the logical and semantic relationships between the FP entities for use in the recommended cloud product set.

A. THE KNOWLEDGE GRAPH FRAMEWORK

To express the logical relationship between FPs, we constructed a multidimensional KG based on a hierarchical model of cloud services [40]. According to the survey, the relationships between the layers of the cloud service model [41] can be summarized as shown in Fig. 1. A point to B, representing A provides further support services or platform services based on B; B is called the pre-entity of A. Therefore, to achieve a structured organization of this knowledge, we establish a KG that encapsulates the interconnectedness between these layers. Fig. 2 shows the structure of the CFPKG. First, we classify the entities into four categories, ‘IaaS’, ‘PaaS’, ‘O&M’, and ‘Security’. In addition, the CFPKG provides a practical

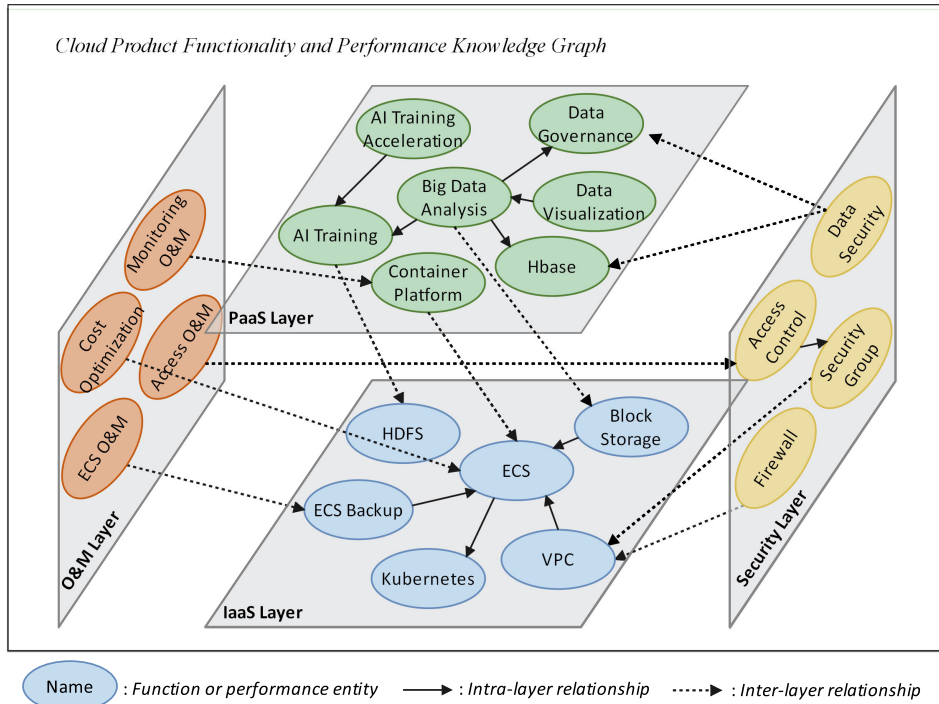


FIGURE 2. Structure of the CFPKG.

overview of an FP repository, including information about pre-post association. Meanwhile, the link between these entities reflects whether there is an inter-layer logical pre-post relationship or an intra-layer dependency relationship, which has a link weight to describe the degree of correlation.

Definition 1(CFPKG): A cloud service KG is simulated as CFPKG = (V, L), where

- V is the vertexes in CFPKG representing FP entities, $V = IAS \cup PAS \cup OAM \cup SCT$.
- IAS is a set of FPs in the IaaS layer, including computing, storage and network. $IAS = \{ias_1, ias_2, \dots, ias_m\}$. *ias* is a functionality or performance in IaaS, and *m* is the number of *ias*.
- PAS is a set of FPs in the PaaS layer, including big data, database, AI, etc. $PAS = \{pas_1, pas_2, \dots, pas_{pn}\}$. *pas* is a functionality or performance in PaaS, and *pn* is the number of *pas*.
- OAM is a set of FPs in the O&M layer, including various O&M and management items. $OAM = \{oam_1, oam_2, \dots, oam_{on}\}$. *oam* is a functionality or performance in O&M, and *on* is the number of *oam*.
- SCT is a set of FPs in the security layer, including data security and network security items. $SCT = \{sct_1, sct_2, \dots, sct_{sn}\}$. where *sct* is a functionality or performance in security, and *sn* is the number of *sct*.
- L is the link of CFPKG, $L \subseteq TER \cup TRA$, TER is the inter-layer link, $TER = (IAS \times PAS) \cup (IAS \times SCT) \cup (IAS \times OAM) \cup (PAS \times SCT) \cup (PAS \times OAM) \cup (SCT \times OAM)$, TRA is the inter-layer

$$link, TRA = (IAS \times IAS) \cup (PAS \times PAS) \cup (SCT \times SCT) \cup (OAM \times OAM)$$

B. COLLECTING PRODUCT FUNCTIONALITY AND PERFORMANCE

We constructed a functionality and performance dataset of cloud products, including 339 items, each of which contained seven factors: functionality or performance name, class, description, release time, product name to which the functionality belongs, product launch time, and last product update time. The classification includes “compute”, “storage”, “network”, “PaaS”, “operation and maintenance” and “security”. The examples of dataset values are listed in Tab. 1.

The data were collected using the following methods. First, we obtained most of the FPs items according to the product description documents and the functionality dynamic update website published on the official website of the cloud company, including the names of FPs, release times, descriptions and product information. We then referred to the indicators published by Gartner for classification. Because the data obtained from the Internet are not always reliable, expert processing is required. Finally, after the cloud company experts discussed, added, deleted and modified the objects, 339 FP objects were identified with the name, class, description, release time, and product information to which the functionality belongs. Some FPs are summarized by experts, and their attributes are provided by experts based on experience.

Definition 2(FP): The paper denote the FP item as cfp, and is a tuple (id, nm, cl, ot, pn, lt, ut, ds), id is the only

TABLE 1. The examples of functionality and performance data.

FP name	FP class	Release time	Product name	Launch time	Update time	FP description
Elastic Cloud Server	compute	2014	ECS	2014	2023	Users can create ECS, which must support: 1. When creating an ECS, you can configure the ECS's specification, image, network, disk, authentication method, creation quantity and other information. 2. Support the management of the life cycle of ECS, including power-on, power-off, restart and deletion; Support the cloning of ECS, the creation of machine snapshots for ECS, and the ha switch status; It supports modifying the vcpu and memory of ECS. 3. Support the expansion, binding, unbinding and other operations on the disks of ECS, and support the sharing of cloud hard disks. 4. Support switching and reinstalling the operating system of ECS; Support the creation of private images based on existing cloud servers. 5. Support binding and unbinding elastic IP. Support server/disk granularity backup, use policy to backup data, intelligently bind the server, and have backup data management function. Backup data supports intra-regional replication and cross-regional replication. When recovering data, support cross-regional recovery of the replica to the original region or other regions, restore server data based on the backup copy or replica, and create a new virtual machine using the backup copy or replica.
ECS backup	storage	2015	EBS	2014	2023	A service solution based on decoupled sidecar proxy must be provided, which is pre-integrated and deployed as part of one or more container services. The service mesh can be isio or linked with the k8s service
Service Mesh	paas	2020	Service Mesh	2020	2023	

identification, nm is the FP name, cl is the classification, ot is the release time of the FP, pn is the product name which the FP belongs to, lt is the launch time of the pn , ut is the last update time of the pn , ds is the description of items, $ds = \{w_1, w_2, \dots, w_n\}$, w is a word in ds , n is the number of word in description. dk is the keyword in the description, $dk \subseteq ds$, $dk = TF - IDF(ds) = \{e_1, e_2, \dots, e_m\}$, e is a keyword in dk , m is the number of keywords

C. KNOWLEDGE GRAPH RELATIONSHIP INFORMATION MINING

To describe the complex logical relationship between entities, CFPKG entity links are divided into inter-layer and intra-layer links. First, the inter-layer link shows the relationship between the layers, including "IaaS", "PaaS", "O&M", and "Security", which is inspired by the cloud service model. We specify the direction based on the dependencies between the layers of the cloud service. The inter-layer direction is illustrated in Fig. 1. Next, the intra-layer links are entity links within the same layer, and their direction is determined by name inclusion relationships and description coverage relationships, with name factors taking precedence over description. The name factor indicates whether the name of an FP entity includes the name of another entity, where the entity whose name is included is the base entity, and is therefore pointed to. The description factors are shown in Fig. 3. The main keyword in a description is the core content of the functionality, and the proportion of a keyword in a description represents the thematic bias of the functionality. The words in a Description B contain the keywords of another Description A, reflecting that the content of B covers the topic of A and the degree of coverage can be expressed by the frequency of occurrence of intersection words. We use this coverage relationship to determine the direction between the two FP entities. The formula is as follows. If $Pro_j > Pro_i$, the edge of i points to j .

$$Pro_i = \frac{C_i(\{x | x \in ds_i \cap x \in dk_j\})}{N_i} \quad (1)$$

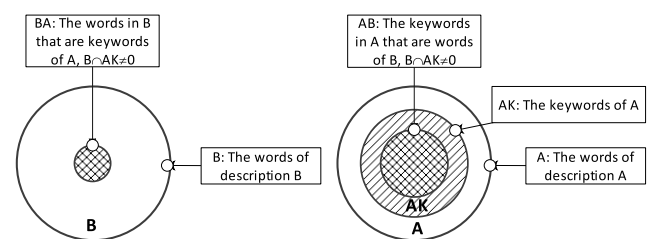


FIGURE 3. Examples of description covered factors. ($Pro_b = BA/B < Pro_a = AB/A$, B points to A).

$$Pro_j = \frac{C_j(\{x | x \in ds_i \cap x \in dk_j\})}{N_j} \quad (2)$$

where i is an entity, j is an entity that needs to be judged whether it is the pre-entity of i . Pro denotes the keyword coverage factor. ds_i is the set of words for description of the i th entity. dk_j is the keywords for the description of the j th entity. $C_i(\varphi)$ is the sum of the times of words in φ among words in description of entity i , $C_j(\varphi)$ is the sum in entity j . N_i is the number of words of description of entity i .

The process of creating a link consists of two steps: calculating link weight and pruning. The link weights were calculated using three factors: name, description, and class. First, two functionality entities with strong associations are frequently have a name-inclusion relationship. Second, the degree of keyword coverage represents the compactness of a relationship. Third, the class factor is used to determine whether a relationship is inter-layer or intra-layer. When links are built based on these factors, redundant edges are created, and the links must be further trimmed. An example of pruning is shown in Fig. 4, where the pointed node is called a child node and the intermediate child node must be cut off if the similarity between child nodes exceeds the threshold δ or if the name has an inclusion relationship. The link algorithm between the entities is given in Alg. 1. The formula for this relationship is shown in Equation 3.

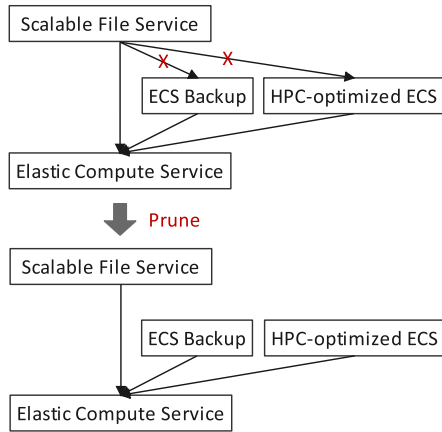


FIGURE 4. Example of prune.

Definition 3(link): The link between entities is a tuple (cfp, r, cfp) , r is the link weight between entities.

$$r = Link(cfp_i, cfp_j) > \theta, i \neq j \& Prune_{ij} \neq 1 \quad (3)$$

$$Link(cfp_i, cfp_j) = \alpha Key_{ij} + \beta Name_{ij} \quad (4)$$

where $Link(\varphi)$ is the weight of the edges, cfp is the FP entity, i and j are the indices of the FP entity. θ is a threshold. $Prune$ is the judgment of pruning. The Key factor was the descriptive keywords factor. The $Name$ is the name factor. The α and β indicate the importance of each influencing factor, respectively.

If the description of the entity cfp_i contains the keywords of the description of cfp_j , Key_{ij} consists of two parts: the proportion of the keyword weight of B to the total keyword weight, Coe_j . This reflects the importance of overlapping keywords in all keywords. The second factor is the coverage factor, Pro_i . This is the proportion of keywords in the descriptions. These two parts synthesize the relationship between these two descriptions.

$$Key_{ij} = Coe_j \times Pro_i \quad (5)$$

$$Coe_j = \frac{W(\{x|x \in ds_i \cap x \in dk_j\})}{W(x \in dk_j)} \quad (6)$$

where $W(\varphi)$ is the sum of the weights of the keywords in φ .

If cfp_i 's name contains cfp_j 's name, cfp_i provides support services or platform services based on cfp_j .

$$Name_{ij} = \begin{cases} 1 & nm_i \supseteq nm_j \\ 0 & others \end{cases} \quad (7)$$

where nm_i is the name of cfp_i . nm_j is the name of cfp_j .

Pruning occurs when two child nodes are too similar or the name has an inclusion relationship.

$$Prune_{ck} = \begin{cases} 1 & (Link_{kl} > \theta \& Sim_{kl} > \delta) \& Name_{kl} > 0 \\ 0 & others \end{cases} \quad (8)$$

Algorithm 1 The Entity Link Algorithms for CFPKG

input : The functionality and performance entity,

$$cfp = (id, nm, cl, lt, ut, ds);$$

output: The entities link set, $link$;

begin

```

link[][] = 0;
foreach cfp_i, cfp_j ∈ cfp and cfp_i ≠ cfp_j do
    P ← Pro; N ← Name; K ← Key;
    S ← Sim; L ← Layer = sort(cl_i, cl_j);
    // inter-layer direction
    if L_ij > 0 then
        link[i][j] += α × K_ij + β × N_ij;
    // intra-layer direction
    else if L_ij == 0 then
        if N_ij > 0 then
            link[i][j] += α × K_ij + β × N_ij;
            continue;
        if K_ij ≠ 0 then
            if P_i < P_j then
                link[i][j] += α × K_ij + β × N_ij;
    // child node pruning
    foreach cfp_m, cfp_n ∈ cfp and link[i][m] > θ and
        link[i][n] > θ do
        if N_mn > 0 then
            link[i][m] = 0;
        if S_mn > δ then
            if link[m][n] > link[n][m] then
                link[i][m] = 0;
return link;

```

$$Sim_{kl} = Sim(ds_k, ds_l) = \frac{\sum_{i=1}^n (x_i \times y_i)}{\sqrt{\sum_{i=1}^n x_i^2} \times \sqrt{\sum_{i=1}^n y_i^2}} \quad (9)$$

where c is an indices of the FP entity, k and l are the indices of the FP entity to which c points. where Sim is the cosine similarity of word vector. x_i is the i th feature of the word vector of ds_k . y_i is the i th feature of the word vector of ds_l .

IV. CLOUD PRODUCT SET RECOMMENDATION BASED ON CFPKG

The recommendation strategy based on CFPKG is shown in Fig. 5. For user query Q, we first matched the functionality and performance F of the product closest to the semantics of the user's requirements using the natural language processing method SimCSE [42]. The weights for each node were then obtained using the KGPageRank algorithm, which considers the link relationships and time factors of the products and functionalities. Next, we obtain the high-weighted FP set

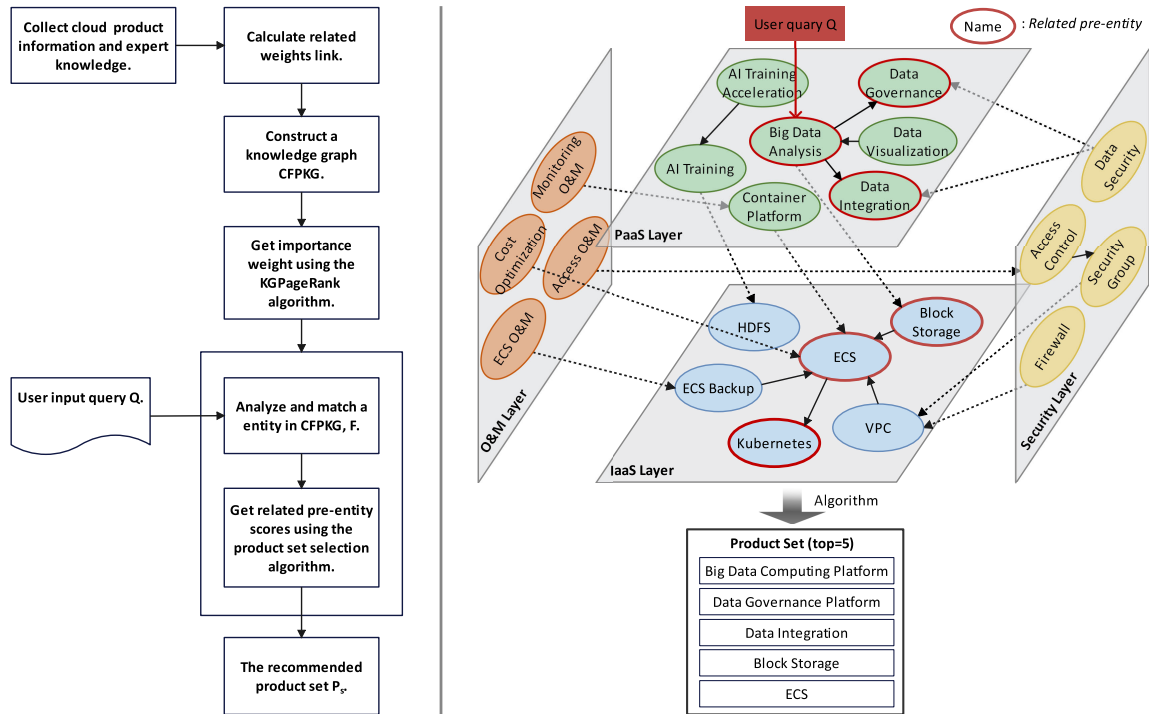


FIGURE 5. Structure of recommendation model based CFPKG.

FS associated with F. Finally, we recommend a set of cloud products that covers the most important functionalities and performances. Cloud product recommendation based on the CFPKG algorithm is presented in Alg. 2.

Algorithm 2 Cloud Product Recommendation Based on CFPKG

```

input : The knowledge graph,  $CFPKG = (V, L)$ ;
         The user's query,  $Q$ ;
output: The recommended cloud product set,  $P_s$ ;
begin
    // analyze users' needs to obtain
    // the most desired product
    // functionalities or performance
     $F$ .
     $F = SimCSE(Q, V)$ ;
    // obtain the node weight KGPR by
    // KGPageRank algorithm.
     $KGPR = KGPageRank(CFPKG)$ ;
    // select recommended cloud
    // product set  $P_s$ .
     $P_s = GetCloudProducts(CFPKG, KGPR, F)$ ;
return  $P_s$ ;
    
```

A. IMPROVED PAGERANK ALGORITHM

In graph theory, indicators for measuring importance include degree centrality, betweenness centrality, presence centrality, eigenvector centrality, and PageRank. Degree centrality, betweenness centrality and presence centrality depend on the

number of other nodes directly connected to the node [2]. This is clearly undesirable for the content of this study, as the importance of functionality should be proportional to the importance of other functionalities that rely on it. Eigenvector centrality solves this problem by allowing centrality to “diffuse” along the edge. PageRank builds on this and proposes two important changes to make the diffuse centrality more reasonable.

FP weights must obey the following rules: 1) the more other FPs depend on an FP, the more important it is; 2) the more an FP depends on an important FP, the more important it is. These rules are similar to those of PageRank recommended by web pages; however, they do not consider the close relationship between FP entities, FP release time and product development stage. Therefore, PageRank is not fully suitable for cloud product recommendation scenarios in which the PageRank algorithm is improved. Our contributions to the PageRank improvement are as follows:

- In this study, the weight fraction of the FP relationship is used instead of the average assignment of the original PageRank. Because the weights of the FP links represent the closeness of the relationship, high-weight links reflect a high dependency. The algorithm is shown in Equation 10.
- We considered FP release time, product launch time, and product update time as important factors of FP, so we added an FP importance weight. First, the importance of an FP is closely related to its importance. The FP release time reflects the importance of functionality in products,

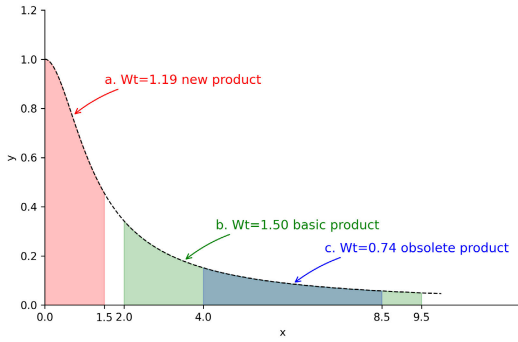


FIGURE 6. Example of W_t of 3 products.

where early release is often an essential functionality, and late release is the secondary complementary functionality. Therefore, as shown in Equation 11, this study uses the proportion of FP release time to product launch time multiplied by product importance as the FP importance. Another, the importance of the product is shown in Fig. 6, the x-axis represents the distance from the previous time to the current time, $x=0$ is the current time, and the y-axis is in the interval $(0,1)$. In particular, the gradient of the function represents the degree of product innovation, which decreases as x increases. The product launch time represents the degree of its foundation and the product update time represents the degree of attention. The square of the function from the launch time to the update time can be considered the product importance. For example, product B was released early and last updated late, which makes it an important base product and therefore has a high score.

$$KGPR(A) = \frac{1-d}{C_{total}-1} + d \sum_{i=1}^n \frac{KGPR(Post_i) \times r_{iA}}{\sum r_i} + W \quad (10)$$

$$W = \frac{ut-rt}{ut-lt} W_t = \frac{ut-rt}{ut-lt} \int_{t_{now}-ut}^{t_{now}-lt+\tau} \frac{1}{\sqrt[3]{(x^2+1)^2}} dx$$

$$\approx \frac{(ut-rt)(t_b-t_a)}{2ut-2lt} \left[\frac{1}{\sqrt[3]{(t_a^2+1)^2}} + \frac{1}{\sqrt[3]{(t_b^2+1)^2}} \right],$$

$$t_a = t_{now} - ut, t_b = t_{now} - lt + \tau \quad (11)$$

where A is the pre-entity. $KGPR(A)$ is the score of A . d is the damping factor. C_{total} is the total number of nodes. $Post_i$ represents the i th entity pointing to A . $KGPR(Post_i)$ is the score of the i th node pointing to A . r_{iA} is the link weight between i and A . $\sum r_i$ is the sum of all link weights output by the i th node. W is the importance weight of FP. W_t is the time factor of the product development stage. t_{now} is the current time. rt is the FP release time. lt is the launch time(LT) for the product. Where ut denotes the final update time (UT) of product. τ is a non-zero factor representing that the minimum interval between LT and UT is τ years and $\tau = 0.5$. For ease of calculation, Equation 11 is approximated using

the Newton-Cotes method, where the Cotes coefficient is set to 1.

Algorithm 3 The Algorithm to Recommend Product Set, $P_s = GetCloudProducts(CFPKG, KGPR, F)$

input : The knowledge graph, $CFPKG = (V, L)$;
The result of KGPageRank, KGPR; The FP Users Need Most, F

output: The recommended cloud product set, FS;
begin

```

P ← GetProduct; // Get the product
that the FP entity belongs to.
O ← GetOut; // Get the pre-entities
for an FP entity.
PS = []; PS.append(PF);
queue<int>level; level.push(F);
visited = []; key = F;
while count < Top do
    foreach u ∈ Okey do
        scoreu = γKGPRu + link[key][u];
    U = SortByScore(Okey);
    foreach u ∈ U do
        if count < Top and Pu ∉ PS then
            PS.append(Pu);
            count ++;
            if visited[u] = 1 then
                level.push(u);
                visited[u] = 1;
            level.pop();
        if !level.empty() then
            key = level.front();
        else
            break;
    return PS;

```

B. PRODUCT SET SELECTION ALGORITHM

The recommended products are selected based on FP entities that match the user requirements. To make more efficient recommendations using functionality association information, this study selected the functionality of requirement matching and other important functionalities closely related to this functionality to recommend the product to which they belong.

This algorithm is described in Alg. 3. First, this method obtains the pre-entity O related to the requirement functionality and scores, and sorts them. The scoring method balances the tightness of the connections and importance of the nodes using the following formulas. The products associated with the top FP were then added to the product set. If the first level of related nodes is connected and the number of products added is insufficient, the products belonging to the other nodes associated with the highest-scoring FP are added to the recommended product set and iterated

TABLE 2. The data processing methods.

Step	Method
1	Segment vocabulary with cloud computing terminology.
2	Replace synonyms of cloud computing.
3	Delete stop word.
4	Extract keywords using TF-IDF.
5	Build relationship using the method in Section 3.3.
6	Save data in graph database Neo4j[9].

hierarchically.

$$Score = \gamma KGPR_U + Link_{FU} \quad (12)$$

where γ is a factor that balances compactness and importance.

V. EXPERIMENTS

This section presents the evaluation of the proposed method. Experiments were conducted using the cloud product service functionality dataset. It was performed on a computer with an Intel Core i7-1165G7 processor at 2.8 GHz. The dataset, measurement metrics, variables, and experimental results of the proposed technique are presented in the following subsections.

A. THE CLOUD PRODUCT SERVICE FUNCTIONALITY DATASET

The data source for the dataset was the official cloud website, including Alibaba Cloud Computing (Alicloud, www.aliyun.com), Amazon Web Services (AWS, aws.amazon.com), and data supplemented by cloud computing experts. The validation results for the product recommendations were obtained from the experience of five cloud computing experts. The data collection methods and samples are described in Section III-B. The data processing methods used in this study are listed in Tab. 2.

B. MEASUREMENT METRICS

Precision(P), recall(R) and F-score(F1) were the evaluation metrics used in this experiment.

$$P = \frac{|Set_{exp} \cap Set_{rec}|}{|Set_{rec}|} \quad (13)$$

$$R = \frac{|Set_{exp} \cap Set_{rec}|}{|Set_{exp}|} \quad (14)$$

$$F1 = \frac{2PR}{P+R} \quad (15)$$

where Set_{exp} is the expected product set for a query, and Set_{rec} is a recommended product set.

C. VARIABLES AFFECTING RECOMMENDATION EFFICIENCY EVALUATION

In this section, three variables were considered to evaluate the influence of the proposed algorithms.

- θ_n : link threshold of the two FP entities. Fig. 7 shows the P, R and F1 results of the recommended algorithm for

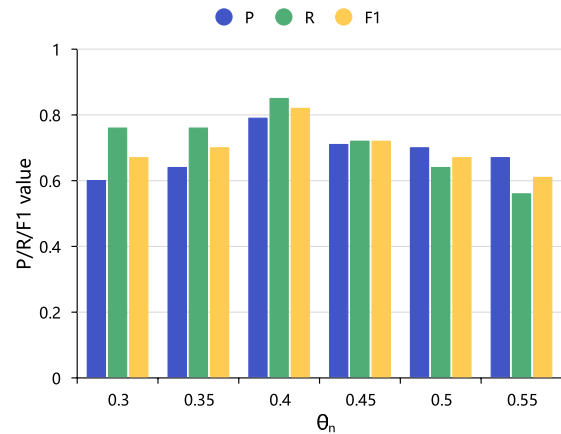


FIGURE 7. The P R and F1 result for our method when the link threshold θ is set to 0.3, 0.35, 0.4, 0.45, 0.5, and 0.55.

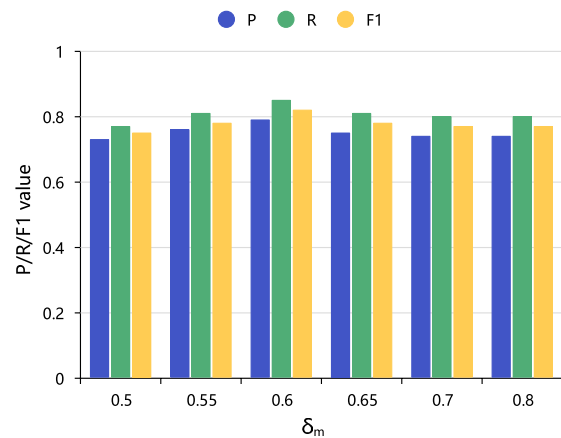


FIGURE 8. The P R and F1 result for our method when the similarity threshold θ is set to 0.5, 0.55, 0.6, 0.65, 0.7, and 0.8.

$\theta_n = 0.3, 0.35, 0.4, 0.45, 0.5$ and 0.55 , when $top = 5$ and $\delta = 0.6$. If the link threshold is too large or too small, the recommendation effect is poor, and an appropriate threshold must be set to obtain the best results. When the connection threshold was large, the P value was larger than the R value because as the connection threshold increased, the edges of the knowledge map decreased and the recommended products decreased.

- δ_m : Similarity threshold for pruning. Fig. 8 shows the P, R and F1 results of the recommended algorithm for $\delta_n = 0.5, 0.55, 0.6, 0.65, 0.7$ and 0.8 , when $top = 5$ and $\theta = 0.4$. The degree of similarity in pruning affects the pruning count, and excessive pruning occurs when the threshold is too low. When the similarity is higher than 0.7, the recommended result tends to be stable because the number of clippings by judging the similarity is 0.
- Top_k : Maximum number of recommended products in the scheme. When $\theta = 0.4$ and $\delta = 0.5$, Fig. 9 shows the P, R and F1 results of the recommendation algorithm for $top_K = 1, 2, 3, 4, 5$ and 6. The figure shows that the P value decreases as the top increases, R first

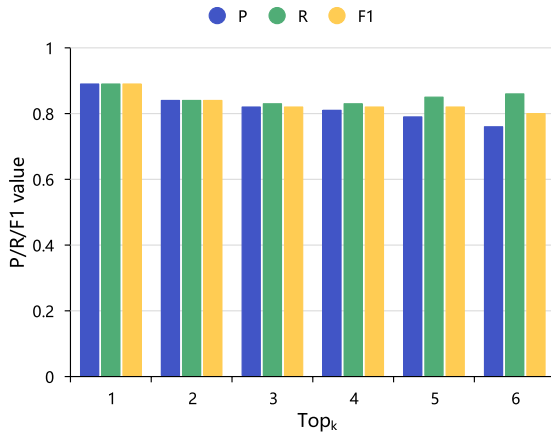


FIGURE 9. The P R and F1 result for our method when the maximum number of recommended products top is set to 1, 2, 3, 4, 5, and 6.

decreases and then increases because the difficulty of the recommendation increases as the number of recommendations increases, so the P value decreases. As the number of recommendations increased, the number of satisfied test results also increased, leading to an increase in R values. Based on the actual needs, top = 5 is usually chosen as the expected number of recommendations.

D. THE RESULT OF EXPERIMENTAL

To demonstrate the effectiveness of our method, we compared the proposed approach with TF-IDF [15], PageRank [10] and Personalized PageRank [43]. And we conducted a questionnaire survey on user satisfaction. The comparison results for different scenarios are shown in Fig. 10. Our method achieved the best F1 results under the different Top requirements. As Top_k increases, the recommendation becomes more difficult and the accuracy of each method decreases. The TF-IDF based recommendation method is the least accurate, and the accuracy decreases rapidly as the number of recommendations increases. Because it uses word frequency and similar information about the content and ignores relevant information, the results are similar. Tab. 3 details the subject P R and F1 of the proposed framework and baseline methods. When top = 5, the method with CFPKG and KGPageRank in this study achieved the highest 79% P, 85% R, and 82% F1 compared to the mainstream product scheme recommendation methods. The accuracy of the improved KGPageRank was 5% higher than that of the PageRank. The PageRank method based on CFPKG provides better recommendations than TF-IDF using information about the functionalities in the knowledge map. However, PageRank does not consider the importance of the functionality itself and the stage of product development. New products tend to have less functionality owing to imperfections and are at a disadvantage in random roaming, resulting in poorer results that are less likely to be recommended than products on the verge of elimination. Personalized PageRank has limitations compared to PageRank’s global walk, with similar

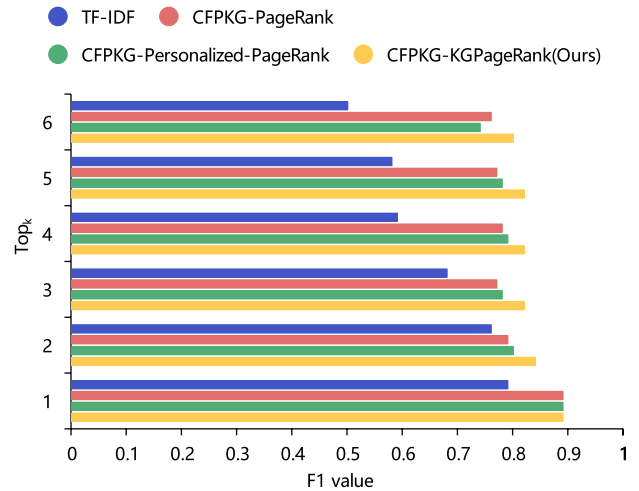


FIGURE 10. Results of the proposed method and baseline methods in the multiple scenarios.

TABLE 3. The results of different methods in different scenarios.

	Method	P	R	F1
Top=1	TF-IDF	0.78	0.80	0.79
	CFPKG+PageRank	0.89	0.89	0.89
	CFPKG+Personalized PageRank	0.89	0.89	0.89
	CFPKG+KGPageRank	0.89	0.89	0.89
Top=2	TF-IDF	0.74	0.78	0.76
	CFPKG+PageRank	0.78	0.80	0.79
	CFPKG+Personalized PageRank	0.78	0.83	0.80
	CFPKG+KGPageRank	0.84	0.84	0.84
Top=3	TF-IDF	0.66	0.70	0.68
	CFPKG+PageRank	0.75	0.78	0.77
	CFPKG+Personalized PageRank	0.78	0.81	0.78
	CFPKG+KGPageRank	0.82	0.83	0.82
Top=4	TF-IDF	0.57	0.62	0.59
	CFPKG+PageRank	0.76	0.81	0.78
	CFPKG+Personalized PageRank	0.79	0.80	0.79
	CFPKG+KGPageRank	0.81	0.83	0.82
Top=5	TF-IDF	0.56	0.60	0.58
	CFPKG+PageRank	0.75	0.80	0.77
	CFPKG+Personalized PageRank	0.78	0.81	0.78
	CFPKG+KGPageRank	0.79	0.85	0.82
Top=6	TF-IDF	0.48	0.52	0.50
	CFPKG+PageRank	0.74	0.80	0.76
	CFPKG+Personalized PageRank	0.74	0.75	0.74
	CFPKG+KGPageRank	0.76	0.86	0.80

results to PageRank, mainly due to the contingency and randomness of these algorithms. Our approach not only considers the tightness of the CFPKG relationship, but also adds time factors that reflect functional importance and the stage of product development to achieve the best results.

Some of the results of CFPKG wandering through KGPageRank are shown in Fig. 11, where most of the nodes in the center of the knowledge map are IaaS layer functionalities. Because the security layer and IaaS layer are

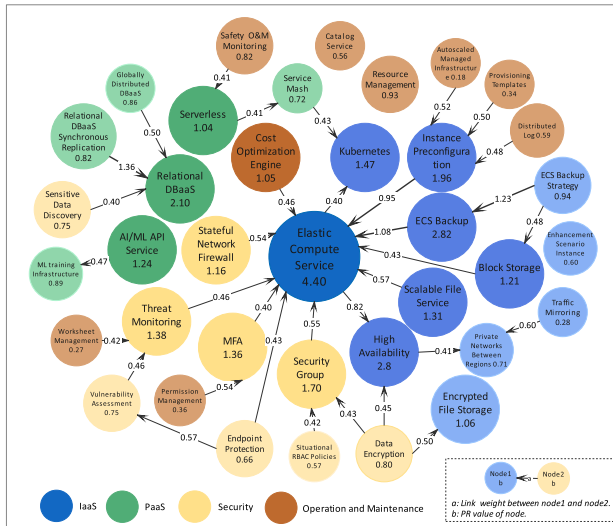


FIGURE 11. The partial results of CFPKG wandering through KGPageRank.

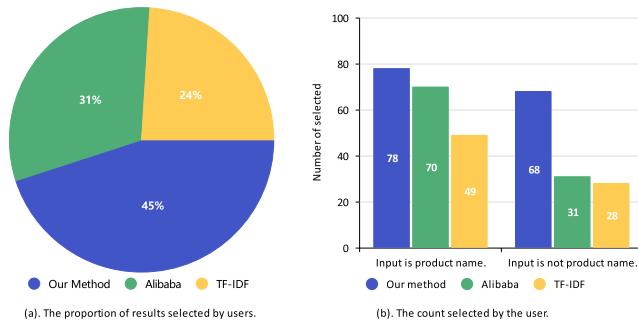


FIGURE 12. The results of the user satisfaction experiment.

more closely connected, most of the security layer functionalities are closer to the center of the diagram. The PaaS layer has fewer connections to the other layers because of its independent functionality. Operations and management functionalities are often at the edge of the diagram, which is consistent with the service model and rules of cloud computing.

Because the validation set is determined by cloud computing experts and does not fully represent user satisfaction, we were inspired by the user satisfaction experimental method in [44], [45], and [46] to design a test web link and randomly sent it to 300 cloud service users, asked them to input their needs, recommended product set schemes, and then let the users choose a satisfactory recommendation result. To reduce the user burden, this study chooses three product set schemes: Alibaba website recommendation results, the TF-IDF method and our method. Among the three methods, the Top was set to 5. To prevent individual users from testing too much data, the subjective impact is greater, and each user has up to 10 valid data points. A total of 324 valid data were collected, and the results are shown in Fig. 12, (a). Our method was selected by users in

45% of cases, 14% more than Alibaba’s 31%, which indicates that our product-related method better meets users’ needs than the similarity-based recommendation method and the current cloud official website method. In addition, we found that when the user’s input requirement was a cloud product name, all three methods yielded better results. However, when the user’s input requirements were other descriptions, our method was selected significantly more frequently than the other methods, indicating that our method can achieve higher user satisfaction in the case of ambiguous user requirements. It should be noted that to obtain the cooperation of users to provide data, the web link is emailed to users who are already using the cloud service. As a result they have more knowledge about cloud products. Thus, so 61% of the input requirements are the name of the cloud product, a value that should be a smaller percentage in the real world, and the advantages of our method should be more obvious.

VI. CONCLUSION

In this study, we systematically investigated the problem of cloud service product scheme recommendation. To improve the accuracy and user satisfaction of cloud product scenario recommendation, we propose a cloud product recommendation model based on CFPKG. Our approach first builds a hierarchical functionality KG based on functionality correlation and cloud service hierarchical model, and then improves PageRank by introducing entity connectivity tightness and time factors that reflect functionality importance and product development stages. Finally, based on the above results, a product set recommendation algorithm was designed to achieve a good product set. Experimental results show that our method can effectively improve recommendation accuracy and achieve better user satisfaction, and also validate the advantage of product dependency recommendation on the cloud product service functionality dataset.

This paper presents an efficient recommendation method for cloud computing products. However, its applicability to other domains is limited because of its reliance on the professional service model and mining of technology dependencies specific to cloud computing. In the future, we will explore the integration of domain knowledge and deep learning methods inspired by emerging recommendation methods based on deep learning. Our goal is to develop a universal method that maintains interpretability within the domain, similar to the method in [32] and [47]. In addition, as noted by [48], the construction of the KG is time consuming, and we will strive to optimize the automated process.

REFERENCES

- [1] M. Zhou, R. Zhang, D. Zeng, and W. Qian, “Services in the cloud computing era: A survey,” in *Proc. 4th Int. Universal Commun. Symp.*, Oct. 2010, pp. 40–46.
- [2] C. Luo, X. Liu, K. Zhang, and Q. Chang, “A recommendation system for cloud services based on knowledge graph,” in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 1–4.

- [3] S. Li and E. Karahanna, "Online recommendation systems in a B2C e-commerce context: A review and future directions," *J. Assoc. Inf. Syst.*, vol. 16, no. 2, pp. 72–107, Feb. 2015.
- [4] C. Low, Y. Chen, and M. Wu, "Understanding the determinants of cloud computing adoption," *Ind. Manage. Data Syst.*, vol. 111, no. 7, pp. 1006–1023, Aug. 2011.
- [5] Z. Cui, X. Xu, F. XUE, X. Cai, Y. Cao, W. Zhang, and J. Chen, "Personalized recommendation system based on collaborative filtering for IoT scenarios," *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 685–695, Jul. 2020.
- [6] D. Wang, Y. Liang, D. Xu, X. Feng, and R. Guan, "A content-based recommender system for computer science publications," *Knowl.-Based Syst.*, vol. 157, pp. 1–9, Oct. 2018.
- [7] Y. Tian, B. Zheng, Y. Wang, Y. Zhang, and Q. Wu, "College Library personalized recommendation system based on hybrid recommendation algorithm," *Proc. CIRP*, vol. 83, pp. 490–494, Jan. 2019.
- [8] A. Kalaï, C. A. Zayani, I. Amous, W. Abdelghani, and F. Sèdes, "Social collaborative service recommendation approach based on user's trust and domain-specific expertise," *Future Gener. Comput. Syst.*, vol. 80, pp. 355–367, Mar. 2018.
- [9] X. Geng and T. Deng, "Research on intelligent recommendation model based on knowledge map," *J. Phys., Conf. Ser.*, vol. 1915, no. 3, May 2021, Art. no. 032006.
- [10] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bring order to the web," Stanford Univ., Stanford, CA, USA, Tech. Rep., 1998.
- [11] J. Chicaiza and P. Valdiviezo-Diaz, "A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions," *Information*, vol. 12, no. 6, p. 232, May 2021.
- [12] X. Du, S.-L. Ge, N.-X. Wang, and Z. Yang, "Personalized product service scheme recommendation based on trust and cloud model," *IEEE Access*, vol. 8, pp. 82581–82591, 2020.
- [13] J. K. Tarus, Z. Niu, and G. Mustafa, "Knowledge-based recommendation: A review of ontology-based recommender systems for e-learning," *Artif. Intell. Rev.*, vol. 50, no. 1, pp. 21–48, Jun. 2018.
- [14] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: Introduction and challenges," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2015, pp. 1–34.
- [15] S. Kanwal, S. Nawaz, M. K. Malik, and Z. Nawaz, "A review of text-based recommendation systems. A review of text-based recommendation systems," *IEEE Access*, vol. 9, pp. 31638–31661, 2021.
- [16] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
- [17] A. Hernando, J. Bobadilla, and F. Ortega, "A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model," *Knowl.-Based Syst.*, vol. 97, pp. 188–202, Apr. 2016.
- [18] S. Wen, C. Wang, and H. Li, "Naïve Bayes regression model and its application in collaborative filtering recommendation algorithm," *Int. J. Internet Manuf. Services*, vol. 5, no. 1, pp. 85–99, 2018.
- [19] P. Valdiviezo-Diaz, F. Ortega, E. Cobos, and R. Lara-Cabrera, "A collaborative filtering approach based on Naïve Bayes classifier," *IEEE Access*, vol. 7, pp. 108581–108592, 2019.
- [20] M. Wasid and R. Ali, "An improved recommender system based on multi-criteria clustering approach," *Proc. Comput. Sci.*, vol. 131, pp. 93–101, Jan. 2018.
- [21] J. Bobadilla, R. Bojorque, A. H. Esteban, and R. Hurtado, "Recommender systems clustering using Bayesian non negative matrix factorization," *IEEE Access*, vol. 6, pp. 3549–3564, 2018.
- [22] Z. Z. Darban and M. H. Valipour, "GHRs: Graph-based hybrid recommendation system with application to movie recommendation," *Expert Syst. Appl.*, vol. 200, Aug. 2022, Art. no. 116850.
- [23] B. Walek and V. Fojtik, "A hybrid recommender system for recommending relevant movies using an expert system," *Expert Syst. Appl.*, vol. 158, Nov. 2020, Art. no. 113452.
- [24] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 494–514, Feb. 2022.
- [25] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *Proc. World Wide Web Conf.*, May 2019, pp. 151–161.
- [26] Z. Yehui, L. Lin, W. Hailong, H. Haiyan, and P. Dongmei, "Survey of knowledge graph recommendation system research," *J. Frontiers Comput. Sci. Technol.*, vol. 17, no. 4, p. 771, 2023.
- [27] B. Jia, X. Huang, and S. Jiao, "Application of semantic similarity calculation based on knowledge graph for personalized study recommendation service," *Kuram ve Uygulamada Egitim Bilimleri*, vol. 18, no. 6, pp. 2958–2966, 2018.
- [28] M. Nilashi, O. Ibrahim, and K. Bagherifard, "A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques," *Expert Syst. Appl.*, vol. 92, pp. 507–520, Feb. 2018.
- [29] C. Obeid, I. Lahoud, H. El Khoury, and P.-A. Champin, "Ontology-based recommender system in higher education," in *Proc. Companion Web Conf. Web Conf. (WWW)*, 2018, pp. 1031–1034.
- [30] G. George and A. M. Lal, "Review of ontology-based recommender systems in e-learning," *Comput. Educ.*, vol. 142, Dec. 2019, Art. no. 103642.
- [31] X. Zou, "A survey on application of knowledge graph," *J. Phys., Conf. Ser.*, vol. 1487, no. 1, Mar. 2020, Art. no. 012016.
- [32] J. Zhang, Y. Zhao, F. Shone, Z. Li, A. F. Frangi, S. Q. Xie, and Z.-Q. Zhang, "Physics-informed deep learning for musculoskeletal modeling: Predicting muscle forces and joint kinematics from surface EMG," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 484–493, 2023.
- [33] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han, "Collaborative filtering with entity similarity regularization in heterogeneous information networks," in *Proc. 1st Int. Joint Conf. Artif. Intell. Workshop Heterogeneous Inf. Netw. Anal.*, 2013.
- [34] C. Luo, W. Pang, Z. Wang, and C. Lin, "Hete-CF: Social-based collaborative filtering recommendation using heterogeneous relations," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2014, pp. 917–922.
- [35] X. Yu, X. Ren, Y. Sun, B. Sturt, U. Khandelwal, Q. Gu, B. Norick, and J. Han, "Recommendation in heterogeneous information networks with implicit user feedback," in *Proc. 7th ACM Conf. Recommender Syst.*, Oct. 2013, pp. 347–350.
- [36] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proc. 7th ACM Int. Conf. Web Search Data Mining*, Feb. 2014, pp. 283–292.
- [37] X. Wang, D. Wang, and C. Xu, "Explainable reasoning over knowledge graphs for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Aug. 2019, pp. 5329–5336.
- [38] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1531–1540.
- [39] M. Godse, R. Sonar, and A. Jadhav, "A hybrid approach for knowledge-based product recommendation," in *Proc. Int. Conf. Inf. Syst., Technol. Manag.* Ghaziabad, India: Springer, Mar. 2009, pp. 268–279.
- [40] Y. Jadeja and K. Modi, "Cloud computing-concepts, architecture and challenges," in *Proc. Int. Conf. Comput., Electron. Electr. Technol. (ICCEET)*, Mar. 2012, pp. 877–880.
- [41] G. Ramachandra, M. Iftikhar, and F. A. Khan, "A comprehensive survey on security in cloud computing," *Proc. Comput. Sci.*, vol. 110, pp. 465–472, Jan. 2017.
- [42] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," 2021, *arXiv:2104.08821*.
- [43] B. Bahmani, A. Chowdhury, and A. Goel, "Fast incremental and personalized PageRank," 2010, *arXiv:1006.2880*.
- [44] T. T. Nguyen, F. M. Harper, L. Terveen, and J. A. Konstan, "User personality and user satisfaction with recommender systems," *Inf. Syst. Frontiers*, vol. 20, no. 6, pp. 1173–1189, Dec. 2018.
- [45] X. Niu and A. Al-Doulat, "LuckyFind: Leveraging surprise to improve user satisfaction and inspire curiosity in a recommender system," in *Proc. Conf. Hum. Inf. Interact. Retr.*, Mar. 2021, pp. 163–172.
- [46] H. Zhu, F. Tian, K. Wu, N. Shah, Y. Chen, Y. Ni, X. Zhang, K.-M. Chao, and Q. Zheng, "A multi-constraint learning path recommendation algorithm based on knowledge map," *Knowl.-Based Syst.*, vol. 143, pp. 102–114, Mar. 2018.

- [47] J. Zhang, Y. Zhao, T. Bao, Z. Li, K. Qian, A. F. Frangi, S. Q. Xie, and Z.-Q. Zhang, "Boosting personalized musculoskeletal modeling with physics-informed knowledge transfer," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–11, 2023.
- [48] Y. Peng, "Query-driven knowledge graph construction using question answering and multimodal fusion," in *Proc. Companion ACM Web Conf.*, Apr. 2023, pp. 1119–1126.



CHUNYU SHI received the B.S. degree in software engineering and the M.S. degree in computer technology from North China Electric Power University, Beijing, China, in 2015 and 2018, respectively. He is currently a Project Leader with China Telecom Research Institute, Guangzhou Campus, Guangzhou, China. His current research interests include cloud computing and network functions virtualization.



SHULIN XU received the B.S. degree in computer science and technology from the Shandong University of Technology, Shandong, China, in 2019, and the M.S. degree in computer technology from the Beijing University of Technology, Beijing, China, in 2022. She is currently an Engineer with China Telecom Research Institute, Beijing. Her current research interests include data mining, knowledge graph, and cloud computing.



ZIYANG WU received the B.S. degree in computer science and engineering from the University of California, Irvine, USA, in 2022. He has previously worked at TikTok Inc., as a Backend Software Engineer and he is currently contributing to research works at UC Irvine's Donald Bren School of Information and Computer Science. His research interests include data mining, databases, mobile and infrastructure development.



MENGYU SUN received the B.S. degree from the Beijing University of Technology, Beijing, China, in 2016, and the Ph.D. degree from the China University of Geosciences, Beijing, in 2022. She is currently an Engineer with China Telecom Research Institute, Beijing. Her current research interests include cloud computing, edge computing, and service computing.

...