applied sciences

Article

Location-Aware Deep Interaction Forest for Web Service QoS Prediction

Shaoyu Zhu [1], Jiaman Ding [1,2,*] and Jingyou Yang [1]

[1] Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China; zhusy@stu.kust.edu.cn (S.Z.); yjy1073260217@163.com (J.Y.)

[2] Artificial Intelligence Key Laboratory of Yunnan Province, Kunming University of Science and Technology, Kunming 650500, China

* Correspondence: jiamanding@kust.edu.cn

Abstract: With the rapid development of the web service market, the number of web services shows explosive growth. QoS is an important factor in the recommendation scene; how to accurately recommend a high-quality service for users among the massive number of web services has become a tough problem. Previous methods usually acquired feature interaction information by network structures like DNN to improve the QoS prediction accuracy, but this generates unnecessary computations. Aiming at addressing the above problem, inspired by the multigrained scanning mechanism in a deep forest, we propose a location-aware deep interaction forest approach for web service QoS prediction (LDIF). This approach offers the following innovations: The model fuses the location similarity of users and services as a latent feature representation of them. In addition, we designed a scanning interaction structure (SIS), which obtains multiple local feature combinations from the interaction between user and service features, uses interactive computing to extract feature interaction information, and concatenates the feature interaction information with original features, which aims to enhance the dimension of the features. Equipped with these, we compose a layer-by-layer cascade by using SIS to fuse low- and high-order feature interaction information, and the early-stop mechanism controls the cascade depth to avoid unnecessary computation. The experiments demonstrate that our model outperforms eight other state-of-the-art methods on MAE and RMSE common metrics on real public datasets.

Keywords: service recommendation; sparse data; feature interaction; deep forest; QoS prediction

check for updates

Citation: Zhu, S.; Ding, J.; Yang, J. Location-Aware Deep Interaction Forest for Web Service QoS Prediction. *Appl. Sci.* **2024**, *14*, 1450. https://doi.org/10.3390/app14041450

Academic Editor: Yutaka Ishibashi

Received: 14 December 2023
Revised: 26 January 2024
Accepted: 6 February 2024
Published: 9 February 2024

1. Introduction

In recent years, most computer software has transformed from native software to web-based software [1]. With the widespread adoption of technologies such as service-oriented architecture (SOA), the Internet of Services (IoS), and cloud computing, an increasing number of enterprises are launching various virtual services to meet user demands. Data, artificial intelligence, and mobile applications also continue to provide new technological support for enterprises to maintain a competitive advantage [2–4]. Nowadays, in the face of the increasing number of services with similar functions, it is difficult for users to make appropriate choices. How to evaluate and screen high-quality services from the massive number of services has become a tricky issue. In this case, investigating and comparing quality of service (QoS) has become a vital reference for service recommendation.

QoS is a key indicator to describe and evaluate web services' quality [5,6]. Many studies focus on the prediction of QoS to achieve service recommendation, such as response time (RT), throughput (TP), and so on. There are many methods for predicting QoS today, most of which are inspired by collaborative filtering for service recommendation [7,8], which completes the prediction of missing QoS by mining historical similarity information of users or services. These methods mainly use information from the original user–service QoS matrix, which is easily affected by data sparsity and ignores some important influencing factors, such as location information [9]. There are certain limitations.

In a real scenario, the QoS value recorded by calling the real service number only accounts for a very small part of the QoS value of all web services, and the extremely sparse data bring difficulties to QoS prediction. With the continuous upgrading and development of deep learning and computing environments, many advanced deep learning methods have been produced, such as those based on deep neural networks (DNN), which have outstanding performance in effectively solving complex problems. Therefore, with the help of deep learning models, which have the ability to automatically learn feature interactive information [10], many scholars solve the problem of feature interaction information fusion and improve the accuracy of QoS prediction.

Traditional research relies on the artificial extraction of low-order feature interaction information, and deep neural networks tend to learn implicit high-order feature interactions, resulting in unnecessary computations. Although existing models have achieved great improvements in prediction accuracy, most models do not structurally enhance the fusion of feature interaction information, and cannot model feature interactions efficiently and explicitly. As an exploratory deep model of non-neural network structures, deep forests [11] have certain advantages in multigranularity scanning to process feature relationships and characterize high-order feature interactions. Therefore, in response to the above problems, this paper proposes a location-aware deep interactive forest method based on the idea of deep forests.

The main contributions of this article are as follows:

- We designed a scanning interactive structure (SIS). It interacts user features and service features, and selects different size combinations of local features to generate more effective feature interaction information and alleviate the effect of sparse data. It also designs a new interactive calculation to express the feature interaction information under different orders.
- We proposed the LDIF model based on deep forests. By fusing the location similarity information of users and services, and using SIS to compose a layer-by-layer cascade, which automatically fuses feature interaction information from low- to high-order, it can find more effective feature information and improve the accuracy of QoS prediction.
- The approach proposed in this paper is experimentally verified under six kinds of data sparsity by using the real public WS-DREAM service dataset. The experiment's result shows that our approach has higher prediction accuracy compared with eight other state-of-the-art approaches.

2. Related Work

Service recommendation algorithms can be divided into traditional collaborative filtering methods and deep learning-based methods. Collaborative filtering methods are further divided into memory-based collaborative filtering and model-based collaborative filtering, but the research focuses are different.

2.1. Collaborative Filtering Methods

Memory-based collaborative filtering methods focus on using historical interactive information of users or services to complete the prediction of QoS. They mainly calculate similarity to improve prediction accuracy. Zheng et al. [12] proposed a new similarity calculation with different weights and a fusion of user- and item-based similarity calculation used to predict missing QoS values. Zheng et al. [13] proposed personalized QoS prediction for cloud services by using two different similarity rank lists. After that, Wu et al. [14] found that users with lower similarity are useless, or even harmful, to the target users, so the two-stage strategy of similarity threshold and similarity fusion was used to replace the traditional TOP-K selection strategy. However, these improvements did not substantially change the mathematical formula of similarity calculation, so Jiang et al. [15] proposed an effective collaborative filtering approach for personalized web service recommendation.

This approach improved the traditional Pearson correlation coefficient (PCC) similarity measurement approach by considering the influence of service personalization, and combined the algorithm based on personalized users and the algorithm based on personalized items. Although these methods are easy to implement, they are easily affected by problems such as sparseness, cold start, and poor scalability.

Model-based collaborative filtering methods focus on using the QoS interactive matrix to learn the model, and are widely used to solve the problems mentioned above. With the introduction of matrix factorization (MF) [16] and factorization machines (FM) [17], a new research idea was derived for how to integrate feature interaction information to achieve higher prediction accuracy. Among them, Zheng et al. [18] believed that the QoS of web services could be predicted according to the characteristics and past experience of the user, so they proposed a personalized web service QoS prediction approach based on neighborhood integration matrix factorization. Tang et al. [19] used the network map to measure the network distance between service and users and identify the neighbors of users by considering the impact of the underlying network on the QoS of web services, and they introduced the above two factors into the matrix factorization as constraints for calculation. And Ngaffo et al. [20] used a flexible matrix factorization technique and a QoS prediction of the future time interval using a time series forecasting approach based on an autoregressive integrated moving average (ARIMA) model. But the latent vectors themselves made by MF are not explanatory, so Chang et al. [21] introduced graph theory into the matrix factorization model to strengthen the interpretability and accuracy; this approach divides the user–service graph into a certain number of subgraphs by dividing the maximum subgraph, and then applies both the subgraph and the user–service graph to the matrix factorization model to perform local and global analysis and prediction. In order to better integrate the contextual information and feature interaction information, Tang et al. [22] directly used the similarity of geographic locations to find user neighbors, and put the features of user neighbors as user features into the factorization machine model to finish QoS prediction. On this basis, Yang et al. [23] proposed a location-based factorization machine model that integrates users, services, similar users, similar services, and geographic location features into a factorization machine. For the reliability of QoS data, Wu et al. [24] proposed a noise-resistant data-characteristic-aware latent factor (DCALF) model to implement highly accurate QoS prediction.

2.2. Deep Learning Methods

In recent years, neural networks have also been gradually applied in QoS prediction. Due to the continuous improvement in numbers of features, current service recommendation algorithms are no longer satisfied with the low-order feature interaction information in FM. More scholars use the characteristics of deep neural networks to automatically complete the fusion of low-order and high-order feature interaction information. For example, Cheng et al. [25] in Google thought that deep neural networks can extend higher-order invisible feature interaction information, so they proposed the Wide and Deep model, which combined polynomial regression and deep neural networks for parallel data processing. On this basis, Wang et al. [26] proposed the Deep&Cross model with a Cross network instead of polynomial regression. The Cross network can be used to explicitly characterize the influence of feature interactions, and the order of feature interaction information will vary with the number of layers, which further improves the interpretability of feature interaction information and the accuracy of result prediction. For the Cross network, Lian et al. [27] thought that it is only a scaled form of the original features, so proposed a new network structure (CIN) and a new recommendation model (xDeepFM), which uses vector-level interactions in different feature domains instead of element-level feature interactions.

On this foundation, service recommendation algorithms evolved to integrate geographic location features; Zhang et al. [28] fused geolocation information by combining collaborative filtering and a multilayer perceptron. Li et al. [29] proposed a topological neural network (TAN) by describing the topological structure of the underlying network and the complex interaction between autonomous regions, which combined the path features and end-cross features through an explicit path-modeling layer and an implicit cross-modeling. If only a single type of feature is used, the problem of gradient disappearance in deep network is serious, so Zhang et al. [30] considered user and service historical probability distribution features and location features to reuse the ResNet network, which alleviates the gradient disappearance problem and improves the prediction accuracy. Wang et al. [31] believed that the state of unknown QoS is a hidden state, so they proposed a hidden state initialization and perception approach based on LDA. Different from the above idea, Wang et al. [32] employed information entropy into the combination of FM and DNN to mine more useful features. Recently, Zhang et al. [33] proposed a novel deep neural network model with multistage and multiscale feature fusion. This model extracted global, individual, and local features separately and implemented feature fusion using a multistage neural network architecture. Additionally, Zhu et al. [34] incorporated historical call similarity, constructed neighborhood subgraphs of users and services using similarity relationships, and proposed the double subgraph network (BGCL) method based on graph contrastive learning. Zhang et al. [35] introduced a feature mapping and inference network method to capture the deep features of users and services and made up for the loss of feature information with feature compensation blocks. Lu et al. [36] investigated a Gaussian feature distribution smoothing network to address the issues of noise and label imbalance in the dataset for QoS prediction.

It is worth noting that the previous methods all incorporate geographic location features or other contextual features into the model, but lack the interaction information between geographic location similarity and other features, so our approach integrates geographic location similarity features into feature interaction information based on location awareness. Zhou et al. [11] proposed a new deep learning model: the deep forest. It is different from the neural network structure but achieves better performance than neural networks in most scenarios. As an ensemble model of trees, the deep forest can use the same size parameters as the neural network to achieve higher prediction accuracy in various fields, and can also achieve higher prediction accuracy in the case of sparse data. Therefore, this paper proposes a new service recommendation approach based on the idea of deep forests, so as to alleviate the impact of sparse data and improve the prediction accuracy by fusing feature interaction information.

3. Proposed Approach

3.1. Problem Description

In order to be easily understood, the notation definitions are shown in Table 1:

Table 1. Notation and descriptions.

Notation	Descriptions
U	the set of all web service users
U_i^k	the k-th feature of the i-th user
U_i	the feature of user i $(U_i^1, U_i^2 \ldots U_i^k)$
S	the set of all web services
S_j^k	the k-th feature of the j-th web service
S_j	the feature of service j $(S_1^1, S_2^2 \ldots S_j^n)$
Q	the QoS set
Q_{ij}	the QoS value between user i and service j
X	the form of data $(U_i^1, U_i^2 \ldots U_i^k, S_1^1, S_2^2 \ldots S_j^n, Q_{ij})$
D	the maximum depth of layer
A	the Matrix A
l	the highest order of feature interaction representation in single-dimensional SIS layer

In the real world, the relationship between users and services is a many-to-many type, where a user can invoke multiple services, and a service can also be simultaneously invoked by multiple users. When users invoke services, the current QoS is recorded to represent the performance of the service as observed by the user. QoS is predicted by the known user and service information (service function information, user and service location and network information) and the observed QoS. As defined in Table 1, in X, U_i represents a user, S_j represents a service, and Q_{ij} represents the QoS value that the user feeds back to the service. Ultimately, users can select services that meet their needs based on the predicted QoS list. Figure 1 is the overall flow chart of the service recommendation approach.

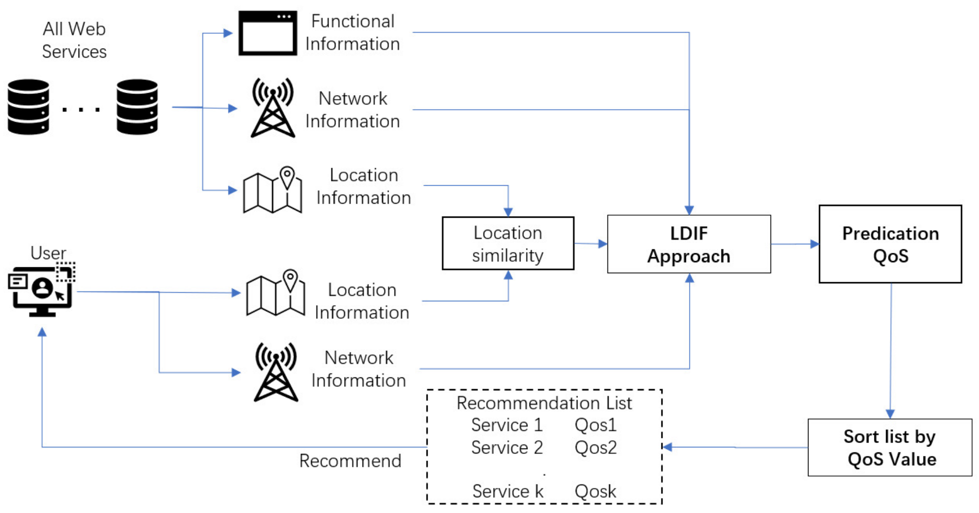


Figure 1. The overall flow of recommendation approach.

3.2. LDIF Approach

For completing high-quality service recommendations, this approach draws on the ideas of the deep forest model. Firstly, the geographical location information of users and services is converted into dense numerical features through the Glove algorithm; the geographical location similarity between users and services is calculated, and the location information between the two is fused; then, using the innovation of the multigrain scanning mechanism in a deep forest, the interaction between user features and service features is automatically scanned to obtain multiple local feature combinations, aiming to generate more effective feature interaction information and alleviate the impact of sparse data; Thirdly, considering that the feature interaction information can be represented numerically, we think that the interaction between local features and connecting this information with the original features is a useful strategy; After that, the high- and low-order feature interaction information is automatically learned and fused using a layer-by-layer structure.

Deep neural networks exhibit the advantage of hierarchical structure, so they have the ability to automatically fuse the interaction information of low- and high-order features, but there is an unchangeable difference between the tree structure and the neuron node structure. The tree structure is used to classify samples according to features rather than reprocess the original feature data by performing a nonlinear operation. So, how to design a layer-by-layer structure while using the tree structure to simultaneously alleviate the impact of sparse data and fuse feature interaction information becomes a tough problem. This paper designs a novel layer-by-layer structure approach based on the deep forest, which can alleviate the impact of sparse data in each layer and fuse feature interaction information though a layer-by-layer structure. Figure 2 is the overall structure of the LDIF approach.

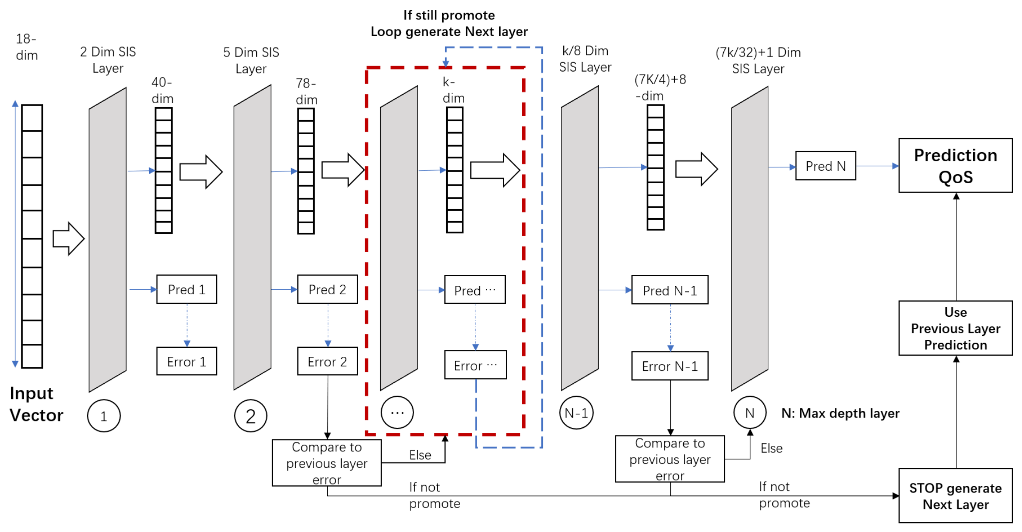


Figure 2. Structure of LDIF Approach. Assuming that the input vector is a *18*-dimensional vector, it will become *40*-dimensional after passing through the first *2*-dimensional SIS layer. We design the size of scanning window in the next layer to be $\lfloor 40/8 \rfloor$, so the input vector will go through *2*-, *5*-, *9*-dimensional scan layers, etc.

In the LDIF approach, the *2*-dimensional SIS layer is the layer with SIS and a scanning window size of 2, and the rest of the layers have the same naming rules. First, we specify the window size of $\lfloor the\ dimension\ of\ feature/8 \rfloor$ to address issues such as the number of scans increasing when the scanning window is too small, resulting in size explosion; or when the size is too large, the dimension growing slowly, even in a deep layer. Additionally, we think that if the accuracy of prediction does not improve, it should stop generating the next layer. So, it will be judged by whether the average prediction accuracy in this layer is smaller than the previous layer. If not, it will stop generating the next layer. In the case of looping, this early stop mechanism effectively controls the complexity of the approach and the depth of layers and avoids unnecessary high-order feature interaction calculations. Under the worst case, we believe that we need to specify a maximum number of layers (D) to control the maximum complexity and max depth of this model.

For the problem of fusing feature interaction information and the sparse data, we are rely on the processing of dimensional SIS layer. Through the above layer-by-layer structure, the final dimensional SIS layer will fuse from low-to high-order feature interaction information and make final QoS predictions. It will elaborate and demonstrate the structure of SIS in Section 3.2.2.

3.2.1. Geographic Similarity

Network conditions and geographical location are closely related, QoS is largely dependent on bandwidth and the network distance between the user and cloud server. Considering geographical location factors has a positive effect on QoS prediction. Unlike traditional collaborative filtering methods, we directly consider the geographical similarity between users and services. The previous idea was that similar users may have similar QoS for any service, but our idea is that users have similar network environments for services close to them. Therefore, we convert geographic location information (text features) into dense numerical features for geographic similarity calculation.

Compared with the Word2Vec [37] algorithm, the Glove [38] algorithm considers the global influence of the corpus more. Therefore, we use the Glove word-embedding algorithm to convert the geographical location information into 50-, 200-, and 300-dimensional dense numerical features, respectively. If these numerical features are directly used for feature interaction operating, the computational complexity will be greatly increase. And it is unmeaningful to acquire the feature interaction information in location numerical features. Finally, we concatenate similarity as a feature into the raw feature. This paper uses cosine similarity to simply and quickly calculate the similarity between users' and services' geographic locations. The similarity calculation formula is as follows:

$$\cos\theta = \frac{\overrightarrow{U_{Vector}} \cdot \overrightarrow{S_{Vector}}}{\|\overrightarrow{U_{Vector}}\| \times \|\overrightarrow{S_{Vector}}\|} \tag{1}$$

where U_{Vector} represents the feature vector of user, and S_{Vector} represents the feature vector of service. The value of $\cos\theta$ will be added into the original feature as a new numerical feature.

3.2.2. Scan Interact Structure

Aiming at the problem of data sparsity that is prevalent in service recommendation, it is often effective to consider automatic feature interactions with more fine grains. In the SIS, we interact user features and service features, obtain feature interaction information through scan window sliding and interaction calculation, and adjust the window size adaptively as the dimension grows, thus realizing explicit automatic feature interaction and avoiding complex computation. In addition, unlike the deep model, with the cooperation of the cascade layer, we concatenate the interaction information of each layer into the features of the previous layer, which is used for the scanning interaction calculation of the next layer. For example, the fourth layer can interact with the feature interaction results of the second layer to mine more efficient higher-order feature interactions. The following will take the *two*-dimensional SIS layer as an example to show the structure in detail with Figure 3:

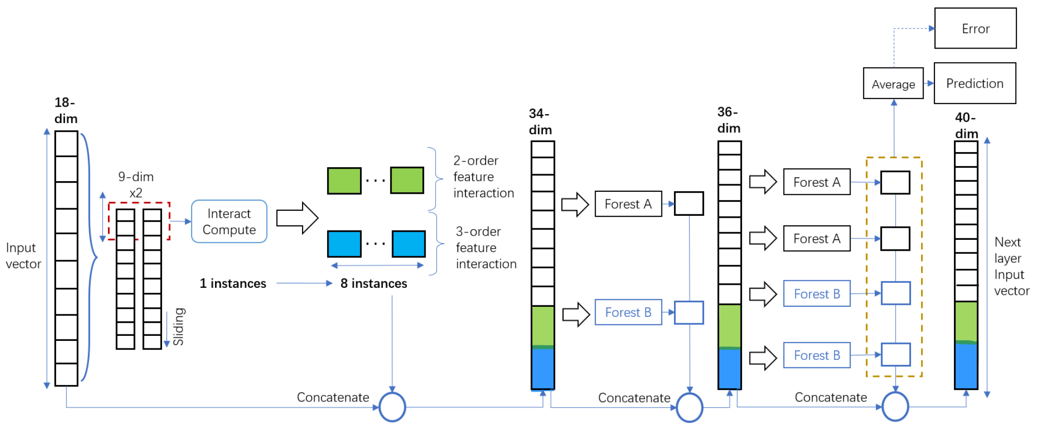


Figure 3. Structure of two-dimensional SIS layer. Assuming that the input vector is *18*-dimensional, of which *9* dimensional vectors are user features and the other *9* are service features. We think that feature interaction is meaningful only when users and features work together, so it can be seen as *9* × 2-dimensional features, representing the features of users and services, respectively.

When the window size of scanning is 2, we define the notation of four numerical features is U_1, U_2 , S_1, and S_2, which can make up matrix A as follows. When the second-order and third-order feature interaction information needs to be calculated, the calculation formulas are as follows:

$$A = \begin{pmatrix} U_1 & U_2 \\ S_1 & S_2 \end{pmatrix} \tag{2}$$

$$A * A^T = \begin{pmatrix} U_1^2 + U_2^2 & U_1S_1 + U_2S_2 \\ U_1S_1 + U_2S_2 & S_1^2 + S_2^2 \end{pmatrix} \tag{3}$$

$$A * A^T * A = \begin{pmatrix} U_1^3 + U_1U_2^2 + U_1S_1^2 + U_2S_2S_1 & U_1^2U_2 + U_2^3 + U_1S_1S_2 + U_2S_2^2 \\ U_1^2S_1 + U_1U_2S_2 + S_1^3 + S_2^2S_1 & U_1U_2S_1 + U_2^2S_2 + S_1^2S_2 + S_2^3 \end{pmatrix} \tag{4}$$

From the above Equations (3) and (4), the four values in the matrix can represent the different combination and its influence, respectively. Therefore, second- and third-order interaction matrices can connect with the original features to fuse feature interaction information. Next, we perform the average sum for the matrix to represent the information in now-order feature interaction. Through this mechanism one can achieve the effect of feature enhancement and the fusion of feature interaction information, as well as controlling the growth rate of feature dimensions. Extending the above situation, the calculation formula for the *n-order* feature interaction matrix is as follows:

$$x_l = \prod_{i=1}^{l} f(i) \tag{5}$$

$$f(i) = \begin{cases} x^T & \text{if } i\%2 = 1 \\ x & \text{if } i\%2 = 0 \end{cases} \tag{6}$$

Since the LDIF approach is a layer-by-layer structure composed by dimensional SIS layers with different scanning window sizes, we take the 2-dim SIS layer as an example to illustrate: Assume that the input vector is *18*-dimensional. First, the input vector is divided into *two* columns. According to the size of the scan window, each sample will be scanned in *8* times. Each time can be seen as a combination of local features to represent a sample. Each sample will compute *two* values to describe second-order and third-order feature interaction information by Equations (3) and (4). So, it will generate *16*-dimensional feature interaction information, and we need to concatenate it with origin feature to fuse information. Until now, the raw input vector has transformed to a *34*-dimensional vector. Next, one random forest and one extreme random forest will generate a *two*-dimensional primary prediction value. After that, a 2-dimensional primary prediction value still needs to be connected to generate a *36*-dimensional vector. For a more accurate prediction value, two-stage prediction is a useful strategy. So, the *36*-dimensional vector will go through two random forests and two extreme random forests, respectively, and generate a *4*-dimensional secondary prediction. Finally, the *36*-dimensional primary prediction value and the *4*-dimensional secondary prediction value are concatenated as the input of the next $\lfloor 40/8 \rfloor$dim SIS layer. As the *4*-dimensional secondary prediction value is $\hat{Q}_1$, $\hat{Q}_2$, $\hat{Q}_3$, $\hat{Q}_4$, and the real QoS value is Q, the prediction and error are computed by Equations (7) and (8). So, each layer describes the feature interaction information for the output of the previous layer. As the depth of layers increases, the order of feature interaction information will continue to increase, which completes and improves the ability to automatically learn and fuse high-order feature interaction information.

$$Prediction = \frac{(\hat{Q}_1 + \hat{Q}_2 + \hat{Q}_3 + \hat{Q}_4)}{4} \tag{7}$$

$$Error = \frac{\sum_{i=1,2,3,4} |Q - \hat{Q}_i|}{4} \tag{8}$$

3.2.3. Algorithm Complexity Analysis

In Algorithm 1 of LDIF, a deep forest is used as a super-integrated ensemble model; we first analyze the time complexity of the random forest and complete random forest. Since the trees in both models are CART trees, the complexity in building the trees is the same. When there are n samples and m attributes, the time complexity of building the tree is $O(mn * \log n)$. In this algorithm, two models are used, so the time complexity is $O(2mn * \log n)$.

Algorithm 1 The algorithm of LDIF

```
Input:
    User Service QoS dataset $X$,
    User location information $U_{loc}$,
    Service location information $S_{loc}$,
    Word Table $E_{table}$
Output:
    The prediction value $\hat{Q}_{u,s}$
 1: X ← $(U_i^1, U_i^2 \ldots U_i^k, S_1^1, S_2^2 \ldots S_j^n, Q_{ij})$
 2: if $E_{table}[U_{loc}]$ and $E_{table}[S_{loc}]$ then
 3:     $U_{loc} \leftarrow E_{table}[U_{loc}]$
 4:     $S_{loc} \leftarrow E_{table}[S_{loc}]$
 5: end if
 6: $Sim_{us} \leftarrow$ Equation (1)
 7: $U_{loc}, S_{loc} \leftarrow Sim_{us}, Sim_{us}$
 8: $Previous_{Loss} \leftarrow \infty$
 9: $Now_{Loss} \leftarrow$ SIS Structure
10: while $Now_{Loss} < Previous_{Loss}$ do
11:     $Temp_{Loss} \leftarrow$ SIS Structure
12:     $Previous_{Loss} \leftarrow Now_{Loss}$
13:     $Now_{Loss} \leftarrow Temp_{Loss}$
14: end while
15: $\hat{Q}_{u,s} = prediction$
16: return $\hat{Q}_{u,s}$
```

In step 1, it is necessary to use Glove to replace the text category features. If there are v text words in total, the time complexity of building a corpus is $O(v^2)$. Assuming that the feature length of users and services is d_1, the local feature size is z_1, the sliding step size is 1, and the highest order of feature interaction calculation is P, then it takes $d_1 - z_1 + 1$ times to complete one scan, denoted as Q. So, the computational complexity of the feature interaction module is $O(PQz_1^3)$, Then the time complexity of completing a single-dimensional scan layer is $O(6PQz_1^3 mn * logn)$. In step 14, assuming that the max depth of layers is k, the time complexity is $O(6kPQz_1^3 mn * logn)$. The overall time complexity is $O(6kPQz_1^3 mn * logn + v^2)$.

4. Experiment

4.1. Dataset Description

To verify the effectiveness of our approach, we use a real web service QoS dataset WS-DREAM obtained from the https://wsdream.github.io website for experimental evaluation. The dataset contains two QoS attribute subdatasets, the response time (RT) and the throughput (TP) dataset. Each dataset includes 1,974,675 records of calls made by 339 users to 5825 services. The specific data information is as follows:

- User Information: User ID, IP Address, Country, IP No., AS (Autonomous System), Latitude, and Longitude.
- Service Information: Service ID, WSDL Address, Service Provider, IP Address, Country, IP No., AS, Latitude, and Longitude.

Under further analysis, the 339 users are distributed in 137 autonomous regions and 31 countries, and the 5825 services are distributed in 1021 autonomous regions and 74 countries. QoS values include response time (RT), with the attribute ranging from 0–20 s; and throughput (TP), with the attribute ranging from 0–1000 kbps.

In the comparison experiment, we randomly remove a part of the data to sparse the data, so we set the matrix density to 5%, 10%, 15%, 20%, 25%, and 30%; the randomly removed QoS data are used as the test set and the rest of the QoS data are used as the train set. For example, when the QoS matrix density is 5%, 5% of the data are used as the training dataset and the remaining 95% are used as the test set. We perform multiple experiments, averaging all the experimental results as the final prediction.

4.2. Evaluation Metric

We specify mean absolute error (MAE) and root mean square error (RMSE) to measure the performance of the proposed approach and other baselines. Lower values for these two metrics suggest a more accurate prediction. They can be defined as follows:

- Mean Absolute Error: MAE is the average of the absolute values of the deviations of all individual observations from the arithmetic mean.

$$MAE = \frac{\sum_{u,s} |Q_{u,s} - \hat{Q}_{u,s}|}{N} \tag{9}$$

- Root Mean Square Error: RMSE indicates the deviation between the observed value and the truth.

$$RMSE = \frac{\sum_{u,s} (Q_{u,s} - \hat{Q}_{u,s})^2}{N} \tag{10}$$

where $Q_{u,s}$ is the QoS value actually recorded by user u calling service s, $\hat{Q}_{u,s}$ is the predicted QoS value when user u calls service s. N represents the size of the number in the test set.

4.3. Baselines

In order to verify the performance of the LDIF approach, we compare the eight most classic recommendation methods, including the traditional collaborative filtering approaches and the state-of-the-art deep learning combined approaches.

1. UPCC [39]: User-based collaborative filtering, using Pearson's correlation coefficient (PCC) to calculate the similarity between users, then using the real values of the top k similar user neighbors on the service to predict the missing values, so as to provide web service recommendations.
2. IPCC [40]: An item-based collaborative filtering algorithm, which first uses PCC to calculate the similarity between web services, and then uses the real values of its top k similar service neighbors on users to predict missing values, so as to provide web service recommendations.
3. UIPCC: This approach is a combination of UPCC and IPCC methods, and the values obtained by the two methods are weighted and summed to recommend web services.
4. PMF [41]: Probabilistic matrix factorization is a approach that adds a probability distribution to the traditional matrix decomposition. It uses Bayes to derive the posterior probability of the implicit features of users and items, so as to analyze web services and make a recommendation.
5. DeepFM [42]: This approach uses a deep neural network combined with a classical factorization machine, and comprehensively considers the interaction effects of low-order features and high-order features to predict the QoS value of web services.
6. DCN: This approach is the deep and cross network, which combines a cross network with a deep neural network to achieve more efficiency and comprehensively consider the interaction information from low- and high-order features, so as to predict the QoS value of web services.
7. WDL: This approach is the wide and deep network, which combines polynomial regression and deep neural networks to describe linear and nonlinear relationships, respectively.
8. DCN-V2 [43]: This approach is the improved deep and cross network, which is a mixture of low-rank DCN (DCN-Mix) to achieve a healthier trade-off between model performance and latency.

Among the above methods, 1–3 are the traditional memory-based collaborative filtering benchmark methods, 4 is the traditional model-based collaborative filtering benchmark method, and 5–8 are the deep learning-based benchmarking methods.

4.4. Experimental Results and Analysis

In this experiment, the MAE and RMSE of nine methods are compared under six different matrix densities (5%, 10%, 15%, 20%, 25%, and 30%). In the UPCC, IPCC, and UIPCC methods, we set the maximum similarity neighborhood number k to 10; in the PMF method, the L1 regularity coefficient is set as 0.01 and the dimension of the latent vector is 10; the DeepFM, DCN, WDL, and DCN-V2 methods are implemented through the deepctr library; in the LDIF method, we set the number of decision trees to 100 and the max depth to 10. Each experiment is randomly divided into a training set and a test set, and a total of 10 replicates are performed. The average of all experimental results is taken as the final prediction result.

Tables 2 and 3 show the prediction performance of LDIF and various comparison methods under different data sparsities. It can be clearly seen that under the six different matrix densities, the MAE and RMSE of LDIF are smaller than other methods, indicating that the prediction accuracy of this approach is higher than the current state-of-the-art methods.

Table 2. Comparison of TP prediction performance.

Approach	Density = 0.05		Density = 0.10		Density = 0.15		Density = 0.20		Density = 0.25		Density = 0.30	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
UPCC	27.26	63.80	24.87	59.87	23.97	58.59	23.63	58.01	23.44	57.71	22.97	57.30
IPCC	48.41	123.27	27.92	84.63	25.90	74.24	26.49	75.92	25.72	75.05	25.13	74.62
UIPCC	37.82	93.53	26.40	72.25	24.93	66.42	25.06	66.96	24.58	66.37	24.05	65.96
PMF	49.77	123.81	49.77	123.82	49.78	123.80	49.85	123.98	49.79	123.87	49.83	123.94
DeepFM	25.37	57.85	21.83	55.16	18.16	48.17	18.85	50.66	18.39	47.82	18.04	48.64
DCN	24.64	55.44	22.37	51.82	20.57	51.03	19.41	50.72	19.09	48.96	18.94	49.21
WDL	23.82	57.19	22.43	56.67	22.72	52.13	19.10	48.11	18.51	48.42	18.71	48.97
DCN-V2	22.53	49.02	20.91	46.47	17.64	41.78	17.23	41.94	15.95	41.50	16.28	41.31
LDIF	**17.20**	**48.14**	**15.81**	**45.50**	**15.40**	**40.84**	**15.10**	**39.65**	**14.91**	**40.71**	**14.71**	**39.70**
Gains	**23.7%**	**1.8%**	**24.4%**	**2.1%**	**12.7%**	**2.2%**	**12.4%**	**5.5%**	**6.5%**	**1.9%**	**9.6%**	**3.9%**

The best results and Gains are highlighted in bold. The following tables are the same.

Table 3. Comparison of RT prediction performance.

Approach	Density = 0.05		Density = 0.10		Density = 0.15		Density = 0.20		Density = 0.25		Density = 0.30	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
UPCC	2.221	3.337	1.959	3.058	1.581	2.599	1.691	2.558	1.709	1.846	1.633	2.019
IPCC	3.539	4.971	2.846	4.216	2.083	3.372	1.618	2.806	1.438	2.588	1.352	2.473
UIPCC	2.879	4.153	2.402	3.637	1.831	2.985	1.654	2.682	1.573	2.216	1.492	2.246
PMF	3.557	4.990	3.555	4.989	3.555	4.99	3.555	4.991	3.558	4.994	3.558	4.990
DeepFM	1.385	2.066	1.428	1.948	0.894	1.590	1.058	1.664	0.874	1.499	0.710	1.396
DCN	2.639	4.004	1.816	2.719	1.565	2.323	1.462	2.108	1.092	1.709	1.012	1.626
WDL	1.382	2.128	1.013	1.724	1.362	1.944	0.818	1.513	0.907	1.607	0.939	1.536
DCN-V2	0.980	1.660	0.490	1.381	0.433	1.362	0.381	1.390	0.421	1.382	0.410	1.373
LDIF	**0.403**	**1.305**	**0.402**	**1.303**	**0.303**	**1.301**	**0.301**	**1.300**	**0.300**	**1.300**	**0.300**	**1.300**
Gains	**58.8%**	**21.4%**	**18.0%**	**5.6%**	**30.0%**	**4.8%**	**21.0%**	**6.5%**	**28.7%**	**6.0%**	**26.8%**	**5.3%**

As can be seen from the table, traditional memory-based collaborative filtering methods of UPCC, IPCC, and UIPCC do not perform well. They are greatly affected by the matrix density. When the matrix density is 5%, the performance of other matrix densities will be greatly reduced. At the same time, since the number of services in the dataset is much larger than the number of users, the running time of IPCC will be greatly increased compared with UPCC. The above shows that the memory-based collaborative filtering approach is greatly affected by the size of data sparsity and the size of the data volume. Surprisingly, the PMF has the lowest prediction performance among the two indicators. The main reason is that when the data sparsity is too small and the data volume is too large, the low-dimensional matrix decomposed by the scoring matrix cannot finally converge. This leads to an increase in the error, and it also shows that the model-based collaborative filtering approach does not perform well when the data sparsity is too small. The following DeepFM, DCN, and WDL combine the deep neural network, relatively fully utilizing the feature of the data and the influence of the feature interaction information from high- to low-order, so the prediction performance on the two QoS indicators of TP and RP is better than those traditional models. However, due to the fact that sufficient training of deep neural networks requires a sufficient amount of data, the prediction performance is not optimal in sparse data, while LDIF uses the multi-size of the local feature scan mechanism in the deep forest to effectively alleviate the problem of sparse data, and special interactive computing improves its ability to represent different-order feature interaction information. Thereby, especially in the case of more sparse data, its prediction performance is significantly better than other methods.

Looking at the data in Tables 2 and 3 from the perspective of horizontal comparison, it can be observed that when the matrix density becomes smaller, the MAE and RMSE will still increase regardless of the approach. This phenomenon shows that although data sparsity will have different influences on various web service recommendation methods, its influence cannot be eliminated. This paper will evaluate the impact of data sparsity on LDIF models in detail in Section 4.5.

4.5. The Impact of Data Sparsity

This experiment evaluates the impact of data sparsity on prediction performance by setting different matrix densities to represent the sparsity of the data. By observing Figure 4, it can be seen that the prediction performance of the four traditional collaborative filtering algorithms gradually decline when the data sparsity gradually becomes smaller. The three methods combined with deep neural networks are less affected by data sparsity than traditional algorithms. Regardless of the sparsity, the prediction performance is always higher than the traditional methods. However, the LDIF approach proposed in this paper has the best performance regardless of the data sparsity, and the prediction performance is relatively stable. This phenomenon shows that this approach is relatively less affected by the data sparsity and can effectively alleviate the data sparsity problem.

In order to view the impact of data sparsity in more detail, we compare this approach under different data sparsity conditions separately. As can be seen from Figure 5, both in MAE and RMSE indicators, the prediction accuracy shows a trend of improvement as the data sparsity increases. But at the same time, we can also see that in the RMSE indicator on TP, the prediction accuracy fluctuates slightly when the data sparsity is 25%. After many experimental analyses, we can conclude that prediction performance may fluctuate within a controllable range when the data sparsity increases, but the overall performance is improving.

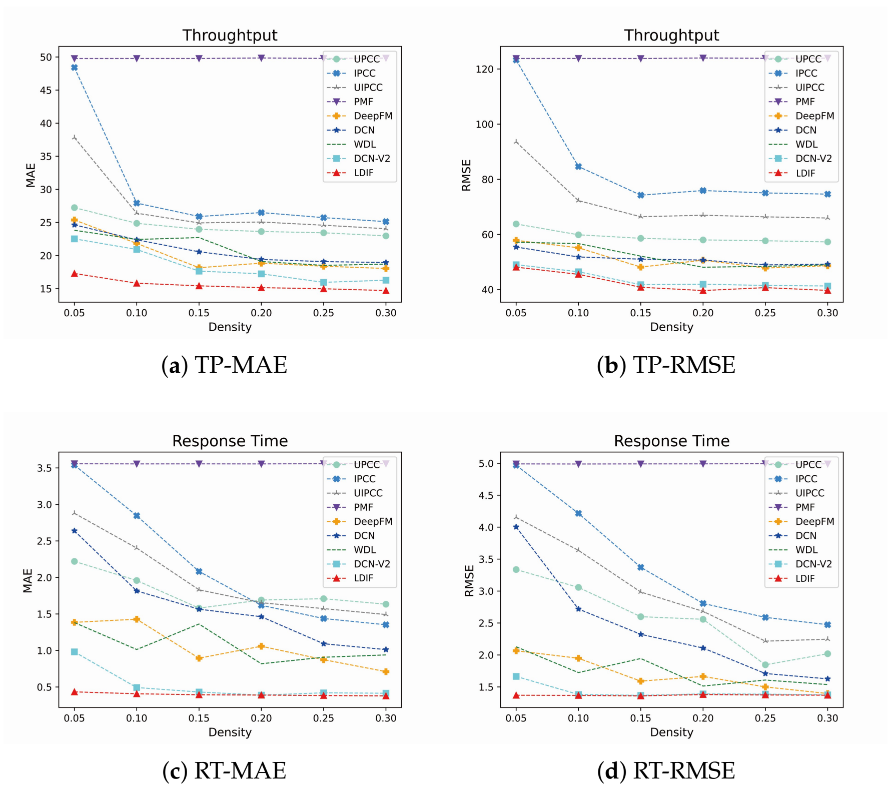


Figure 4. Comparison of prediction performance. (**a**–**d**) represent the prediction performance of all methods on MAE and RMSE metrics in TP and RT, respectively.

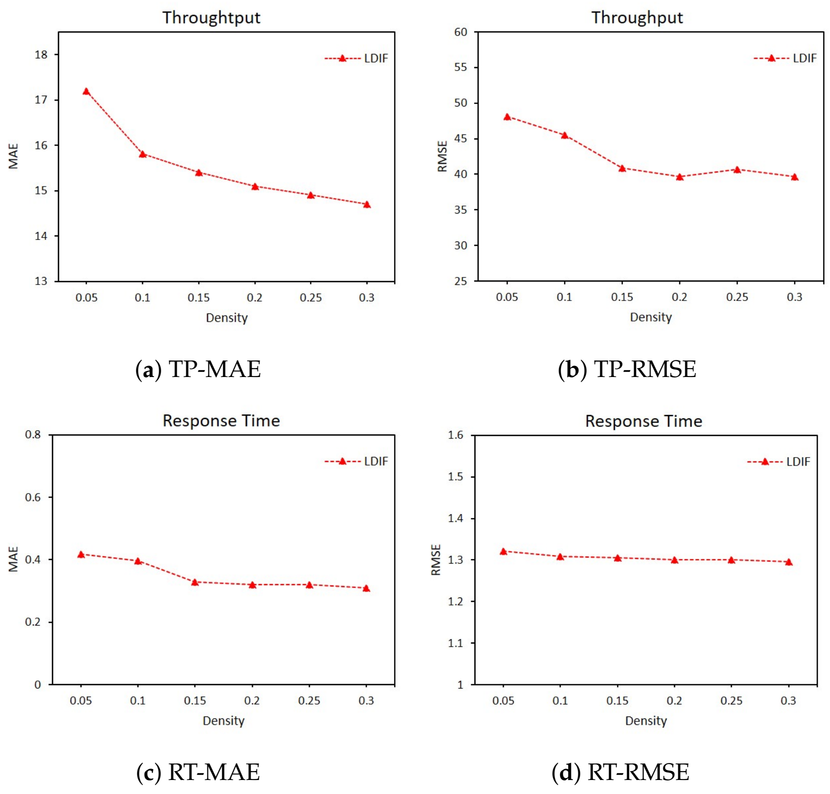


Figure 5. Show impact of data sparsity. (**a**–**d**) represent the prediction performance of LDIF methods on MAE and RMSE metrics in TP and RT, respectively.

4.6. The Effectiveness of the Scan Interaction Structure

In order to verify the effectiveness of scan interaction structure (SIS) in alleviating the impact of sparse data and extracting feature interaction information, we design control experiments based on the above experimental preparation and work: select the DCN model specifically for the sparse data environment, and use the original features and the features output through single-layer SIS to predict RT and TP. We perform a total of 10 replicates on each different piece of sparse data. The average of all experimental results is used as the final prediction. The experimental results are shown in Tables 4 and 5.

Table 4. Comparison of TP prediction performance between DCN and SIS + DCN.

Approach	Density = 0.05		Density = 0.10		Density = 0.15		Density = 0.20		Density = 0.25		Density = 0.30	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
DCN	24.64	55.44	22.37	51.82	20.57	51.03	19.41	50.72	19.09	**48.96**	18.94	**49.21**
SIS + DCN	**17.63**	**51.01**	**17.41**	**49.27**	**15.75**	**48.70**	**15.39**	**48.29**	**15.64**	49.67	**15.31**	49.55

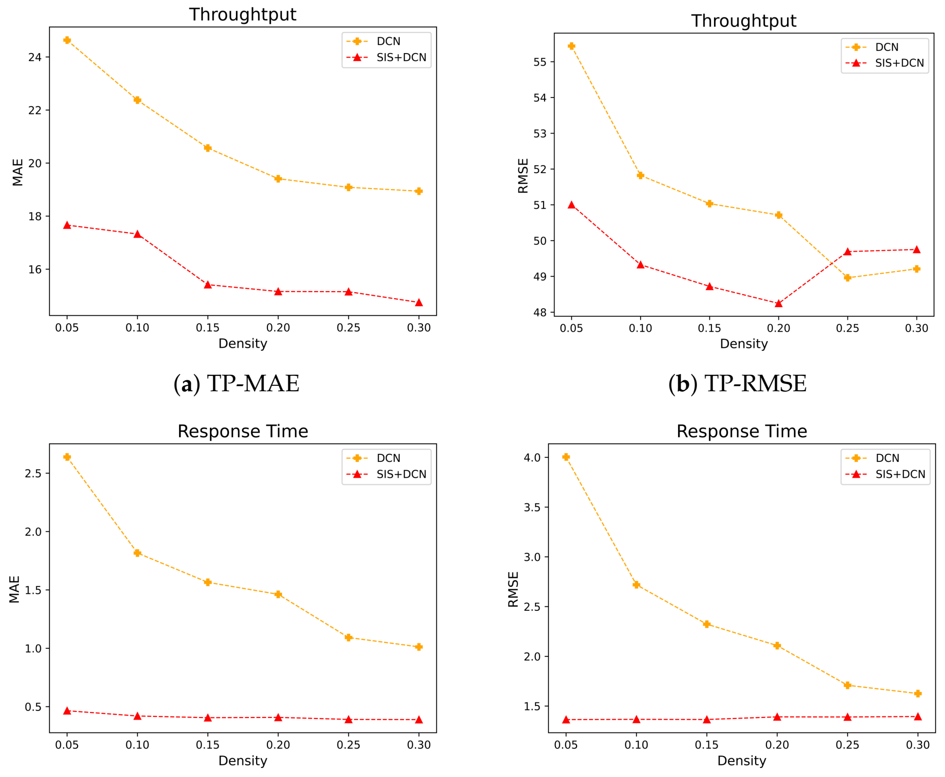
Table 5. Comparison of RT prediction performance between DCN and SIS + DCN.

Approach	Density = 0.05		Density = 0.10		Density = 0.15		Density = 0.20		Density = 0.25		Density = 0.30	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
DCN	2.639	4.004	1.816	2.719	1.565	2.323	1.462	2.108	1.092	1.709	1.012	1.626
SIS + DCN	**0.498**	**1.375**	**0.438**	**1.376**	**0.405**	**1.379**	**0.385**	**1.451**	**0.356**	**1.470**	**0.335**	**1.479**

From the above results, it can be clearly seen that when the scanning interaction structure is fused, the prediction performance of the DCN model for RT and TP is greatly improved when the data are more sparse. In addition, carefully looking at Figure 6, it can be found that the original DCN model will show higher error with the higher data sparsity, and its rising slope is larger. When the DCN model fuses the scanning interaction structure, although there is still a rising error, the rising slope is obviously stable. The above two phenomena strongly illustrate that the SIS structure has an obvious mitigation effect on alleviating the impact of sparse data. And the SIS structure can significantly improve the prediction performance of the model for QoS values.

4.7. The Impact of Word-Embedding Dimensions

This experiment evaluates the effect of a word-embedding dimension on prediction performance by comparing this algorithm prediction performance under different word-embedding dimensions. We separately set the embedding dimension as 50, 200, and 300 dimensions to simulate most situations in reality. Since the effect of expressing the prediction performance of the QoS value is the same whether it is TP or RP, this experiment only uses TP as the target QoS value. The same as the six data sparsities taken in the above experiments, Table 6 uses the MAE metric, and Table 7 uses the RMSE metric, and the average values under the six data sparsities are calculated. Experiments show in Figure 7 that with the increase in the word-embedding dimension, MAE and RMSE are relatively stable and the relative improvement effect is not significant. Besides that, we found that when the word-embedding dimension increases, it increases the time overhead of data preprocessing, which will increase the overall running time of the experiment. Therefore, this paper suggests that it is more reasonable to set the word-embedding dimension as 50.



(**a**) TP-MAE (**b**) TP-RMSE

(**c**) RT-MAE (**d**) RT-RMSE

Figure 6. Comparison of prediction performance between DCN and (SIS + DCN). (**a**–**d**) represent the prediction performance of DCN and (SIS + DCN) on MAE and RMSE metrics in TP and RT, respectively.

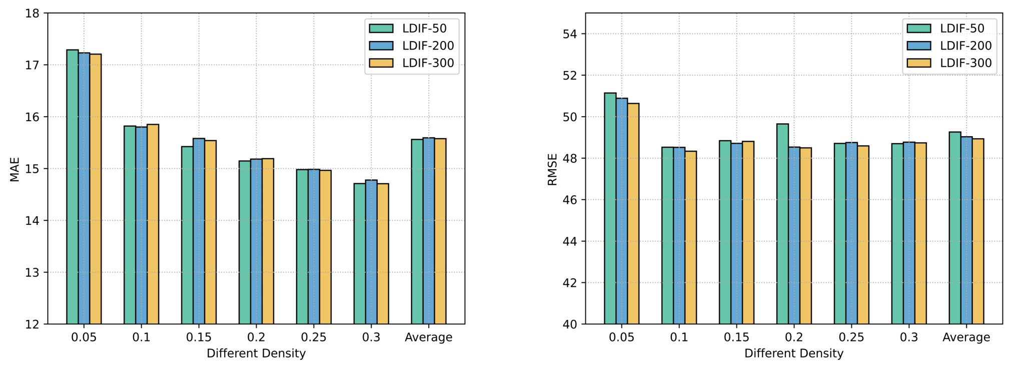
Table 6. Comparison of TP prediction performance in different word-embedding dimensions on MAE.

Approach	**Density = 0.05 MAE**	**Density = 0.10 MAE**	**Density = 0.15 MAE**	**Density = 0.20 MAE**	**Density = 0.25 MAE**	**Density = 0.30 MAE**	**Average MAE**
LDIF-50	17.29	15.82	15.42	15.15	14.98	14.71	**15.56**
LDIF-200	17.23	15.80	15.58	15.18	14.98	14.78	**15.59**
LDIF-300	17.21	15.85	15.54	15.19	14.97	14.71	**15.58**

The average values are highlighted in bold. The following tables are the same.

Table 7. Comparison of TP prediction performance in different word-embedding dimensions on RMSE.

Approach	**Density = 0.05 RMSE**	**Density = 0.10 RMSE**	**Density = 0.15 RMSE**	**Density = 0.20 RMSE**	**Density = 0.25 RMSE**	**Density = 0.30 RMSE**	**Average RMSE**
LDIF-50	51.14	48.53	48.84	49.65	48.71	48.70	**49.26**
LDIF-200	50.89	48.52	48.71	48.53	48.75	48.77	**49.03**
LDIF-300	50.64	48.33	48.81	48.50	48.59	48.74	**48.93**



(**a**) TP-MAE (**b**) TP-RMSE

Figure 7. Comparison in different word-embedding dimensions. (**a**,**b**) represent the prediction performance of different word-embedding dimensions on MAE and RMSE metrics in TP, respectively.

4.8. The Impact of Highest Feature Interaction Order

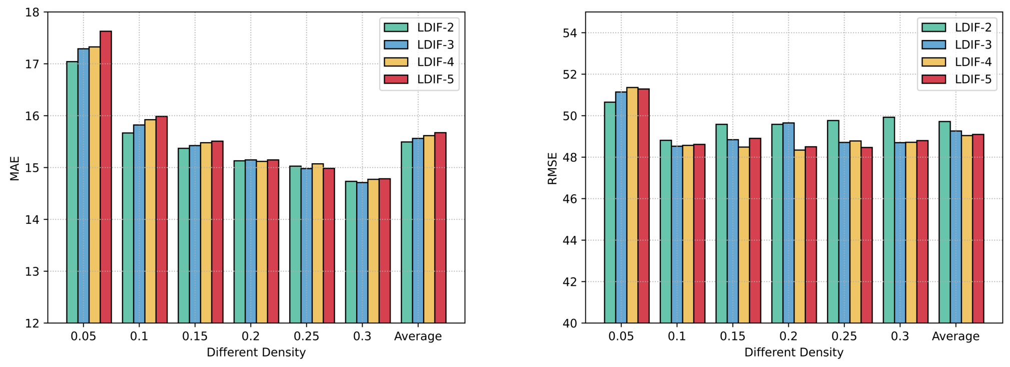
By comparing the prediction performance under the different highest feature interaction order, this experiment evaluates the effect of the highest feature interaction order in the scan mechanism on the prediction performance. Here, the word-embedding dimension in our experiment is set as 50 dimensions, then the highest interaction order in local features will be set as 2, 3, 4, and 5, respectively. It will greatly increase the amount of computation in the model and improve the running time of the model, so only the comparison experiments of order 2–5 are selected. LDIF-2 in the methods in Tables 8 and 9 represents the LDIF approach with the highest order of 2. Through the MAE and RMSE metrics under the TP value in Figure 8, it can be found that with the increase in the highest interaction order, the MAE does not improve obviously, but the RMSE has a trend of improvement, which shows that with the increase in the highest interaction order in the local features, the prediction accuracy has not improved, but the prediction performance has become more stable. Similarly, since the increase in the highest interaction order will increase the running time of the model, this paper proposes to set the maximum interaction order to 2, while ensuring a certain stability and accuracy, and the running time of the model can be applied to many practical situations.

Table 8. Comparison of TP prediction performance in different highest orders on MAE.

Approach	Density = 0.05 MAE	Density = 0.10 MAE	Density = 0.15 MAE	Density = 0.20 MAE	Density = 0.25 MAE	Density = 0.30 MAE	Average MAE
LDIF-2	17.04	15.67	15.37	15.13	15.03	14.73	**15.49**
LDIF-3	17.29	15.82	15.42	15.15	14.98	14.71	**15.56**
LDIF-4	17.33	15.92	15.48	15.12	15.07	14.77	**15.62**
LDIF-5	17.63	15.99	15.51	15.15	14.98	14.78	**15.67**

Table 9. Comparison of TP prediction performance in different highest orders on RMSE.

Approach	Density = 0.05 RMSE	Density = 0.10 RMSE	Density = 0.15 RMSE	Density = 0.20 RMSE	Density = 0.25 RMSE	Density = 0.30 RMSE	Average RMSE
LDIF-2	50.65	48.81	49.58	49.58	49.76	49.92	**49.72**
LDIF-3	51.14	48.53	48.84	49.70	48.71	48.70	**49.26**
LDIF-4	51.36	48.57	48.49	48.34	48.78	48.72	**49.04**
LDIF-5	51.29	48.62	48.91	48.50	48.47	48.80	**49.10**

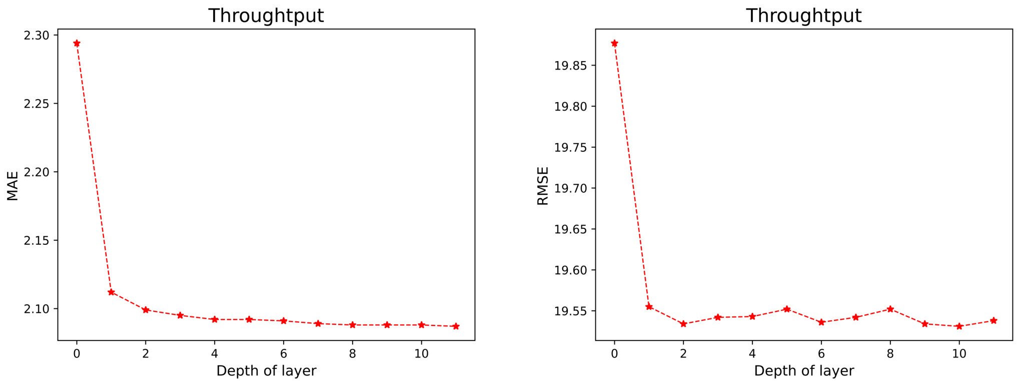


(**a**) TP-MAE (**b**) TP-RMSE

Figure 8. Comparison in different highest orders. (**a**,**b**) represent the prediction performance of different highest orders on MAE and RMSE metrics in TP, respectively.

4.9. The Impact of Layer Depth

By comparing the prediction performance under different depths of layers, this experiment evaluates the effect of the depths of layers on the prediction performance. Here, the word-embedding dimension in our experiment is set as 50 dimensions, and the highest interaction order in local features is set as 2. The result is shown in Figure 9. Through the MAE and RMSE metrics under the TP value, it can be found that with the increase in the depth of the layer, both of the metrics decrease. But if it still deepens the depth of the layer, it shows no obvious decreasing trend, which means that stopping generating the next layer in time can reduce the time cost and model complexity while keeping the prediction accuracy. Experimental results demonstrate that the stop mechanism is effective in our approach.



(**a**) TP-MAE (**b**) TP-RMSE

Figure 9. Comparison in different depths of layers. (**a**,**b**) represent the prediction performance of layer depths on MAE and RMSE metrics in TP, respectively.

5. Conclusions

The location-aware deep interaction forest for web service QoS prediction approach (LDIF) has the greatest performance regardless of the data sparsity. Firstly, we use word vector embedding technology to enrich the feature information of users and services, and consider the geographical similarity as the potential feature representation between them. In this paper, a scanning interaction structure (SIS) is proposed based on a multigranularity scanning mechanism. We extract local feature combinations of different sizes from the interaction between user features and service features and use interaction calculation to obtain feature interaction information. The dimension enhancement of the feature vector is realized through feature fusion, which effectively alleviates the impact of sparse data. Furthermore, we demonstrate the effectiveness of the scanning interaction structure (SIS). The layer-by-layer cascade composed by SIS effectively achieves the purpose of fusing low- to high-order feature interaction information. Experiments demonstrated that the performance of this approach has a certain improvement compared with traditional approaches and deep learning approaches. In this paper, we fuse the feature interaction information generated by each layer in a concatenation manner. Considering the connection between layers may further improve the performance. Therefore, in the next step, we will focus on mining the feature interaction information between each layer in the forest through group convolution to further improve the performance and efficiency of the approach.

Author Contributions: Conceptualization, S.Z., J.D. and J.Y.; methodology, S.Z.; software, J.Y.; validation, S.Z., J.D. and J.Y.; formal analysis, S.Z. and J.Y.; investigation, J.Y.; resources, S.Z.; data curation, J.Y.; writing—original draft preparation, S.Z.; writing—review and editing, S.Z. and J.D.; visualization, J.Y.; supervision, J.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by National Natural Science Foundation of China (No. 62262034, No. 62262035).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://wsdream.github.io.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ghafouri, S.H.; Hashemi, S.M.; Hung, P.C. A survey on web service QoS prediction methods. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2439–2454. [CrossRef]
2. Goudarzi, A.; Ghayoor, F.; Waseem, M.; Fahad, S.; Traore, I. A Survey on IoT-Enabled Smart Grids: Emerging, Applications, Challenges, and Outlook. *Energies* **2022**, *15*, 6984. [CrossRef]
3. Waseem, M.; Adnan Khan, M.; Goudarzi, A.; Fahad, S.; Sajjad, I.A.; Siano, P. Incorporation of blockchain technology for different smart grid applications: Architecture, prospects, and challenges. *Energies* **2023**, *16*, 820. [CrossRef]
4. Ma, Y.; Yu, C.; Yan, M.; Sangaiah, A.K.; Wu, Y. Dark-Side Avoidance of Mobile Applications with Data Biases Elimination in Socio-Cyber World. *IEEE Trans. Comput. Soc. Syst.* **2023**, 1–10. [CrossRef]
5. Zheng, Z.; Zhang, Y.; Lyu, M.R. Investigating QoS of Real-World Web Services. *IEEE Trans. Serv. Comput.* **2014**, *7*, 32–39. [CrossRef]
6. Zheng, Z.; Zhang, Y.; Lyu, M.R. Distributed QoS Evaluation for Real-World Web Services. In Proceedings of the 2010 IEEE International Conference on Web Services, Miami, FL, USA, 5–10 July 2010; pp. 83–90. [CrossRef]
7. Shao, L.; Zhang, J.; Wei, Y.; Zhao, J.; Xie, B.; Mei, H. Personalized QoS Prediction forWeb Services via Collaborative Filtering. In Proceedings of the IEEE International Conference on Web Services, Salt Lake City, UT, USA, 9–13 July 2007; pp. 439–446. [CrossRef]
8. Lo, W.; Yin, J.; Deng, S.; Li, Y.; Wu, Z. An Extended Matrix Factorization Approach for QoS Prediction in Service Selection. In Proceedings of the 2012 IEEE Ninth International Conference on Services Computing, Honolulu, HI, USA, 24–29 June 2012; pp. 162–169. [CrossRef]
9. Zheng, Z.; Li, X.; Tang, M.; Xie, F.; Lyu, M.R. Web service QoS prediction via collaborative filtering: A survey. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2455–2472. [CrossRef]
10. Cao, B.; Xiao, Q.; Zhang, X.; Liu, J. An API service recommendation method incorporating SOM functional clustering and DeepFM quality prediction. *J. Comput. Sci.* **2019**, *6*, 1367-1383. [CrossRef]
11. Zhou, Z.H.; Feng, J. Deep forest. *Natl. Sci. Rev.* **2019**, *6*, 74–86. [CrossRef] [PubMed]
12. Zheng, Z.; Ma, H.; Lyu, M.R.; King, I. QoS-Aware Web Service Recommendation by Collaborative Filtering. *IEEE Trans. Serv. Comput.* **2011**, *4*, 140–152. [CrossRef]
13. Zheng, Z.; Wu, X.; Zhang, Y.; Lyu, M.R.; Wang, J. QoS Ranking Prediction for Cloud Services. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 1213–1222. [CrossRef]
14. Wu, J.; Chen, L.; Feng, Y.; Zheng, Z.; Zhou, M.C.; Wu, Z. Predicting Quality of Service for Selection by Neighborhood-Based Collaborative Filtering. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 428–439. [CrossRef]

15. Jiang, Y.; Liu, J.; Tang, M.; Liu, X. An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering. In Proceedings of the 2011 IEEE International Conference on Web Services, Washington, DC, USA, 4–9 July 2011; pp. 211–218. [CrossRef]
16. Koren, Y.; Bell, R.; Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
17. Rendle, S. Factorization Machines. In Proceedings of the 2010 IEEE International Conference on Data Mining, Sydney, NSW, Australia, 13–17 December 2010; pp. 995–1000. [CrossRef]
18. Zheng, Z.; Ma, H.; Lyu, M.R.; King, I. Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization. *IEEE Trans. Serv. Comput.* **2013**, *6*, 289–299. [CrossRef]
19. Tang, M.; Zheng, Z.; Kang, G.; Liu, J.; Yang, Y.; Zhang, T. Collaborative Web Service Quality Prediction via Exploiting Matrix Factorization and Network Map. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 126–137. [CrossRef]
20. Ngaffo, A.N.; El Ayeb, W.; Choukair, Z. Service recommendation driven by a matrix factorization model and time series forecasting. *Appl. Intell.* **2022**, *52*, 1110–1125. [CrossRef] [PubMed]
21. Chang, Z.; Ding, D.; Xia, Y. A graph-based QoS prediction approach for web service recommendation. *Appl. Intell.* **2021**, *51*, 6728–6742. [CrossRef]
22. Tang, M.; Zhang, T.; Yang, Y.; Zheng, Z.; Cao, B. Quality-aware Web Service Recommendation Method Based on Factor Decomposer. *J. Comput. Sci.* **2018**, *41*, 14.
23. Yang, Y.; Zheng, Z.; Niu, X.; Tang, M.; Lu, Y.; Liao, X. A Location-Based Factorization Machine Model for Web Service QoS Prediction. *IEEE Trans. Serv. Comput.* **2021**, *14*, 1264–1277. [CrossRef]
24. Wu, D.; Luo, X.; Shang, M.; He, Y.; Wang, G.; Wu, X. A data-characteristic-aware latent factor model for web services QoS prediction. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 2525–2538. [CrossRef]
25. Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & Deep Learning for Recommender Systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 7–10. [CrossRef]
26. Wang, R.; Fu, B.; Fu, G.; Wang, M. Deep & Cross Network for Ad Click Predictions. In Proceedings of the ADKDD'17, Halifax, NS, Canada, 13–17 August 2017; pp. 1–7. [CrossRef]
27. Lian, J.; Zhou, X.; Zhang, F.; Chen, Z.; Xie, X.; Sun, G. XDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1754–1763. [CrossRef]
28. Zhang, Y.; Yin, C.; Wu, Q.; He, Q.; Zhu, H. Location-Aware Deep Collaborative Filtering for Service Recommendation. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 3796–3807. [CrossRef]
29. Li, J.; Wu, H.; Chen, J.; He, Q.; Hsu, C.H. Topology-Aware Neural Model for Highly Accurate QoS Prediction. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 1538–1552. [CrossRef]
30. Zhang, W.; Xu, L.; Yan, M.; Wang, Z.; Fu, C. A Probability Distribution and Location-aware ResNet Approach for QoS Prediction. *J. Web Eng.* **2021**, *20*, 1189–1227. [CrossRef]
31. Wang, Z.; Zhang, X.; Yan, M.; Xu, L.; Yang, D. HSA-Net: Hidden-State-Aware Networks for High-Precision QoS Prediction. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 1421–1435. [CrossRef]
32. Wang, Q.; Zhang, M.; Zhang, Y.; Zhong, J.; Sheng, V.S. Location-based deep factorization machine model for service recommendation. *Appl. Intell.* **2022**, *52*, 9899–9918. [CrossRef]
33. Zhang, P.; Huang, W.; Chen, Y.; Zhou, M.; Al-Turki, Y. A Novel Deep-Learning-Based QoS Prediction Model for Service Recommendation Utilizing Multi-Stage Multi-Scale Feature Fusion with Individual Evaluations. *IEEE Trans. Autom. Sci. Eng.* **2023**, 1–14 . [CrossRef]
34. Zhu, J.; Li, B.; Wang, J.; Li, D.; Liu, Y.; Zhang, Z. BGCL: Bi-subgraph network based on graph contrastive learning for cold-start QoS prediction. *Knowl.-Based Syst.* **2023**, *263*, 110296. [CrossRef]
35. Zhang, P.; Ren, J.; Huang, W.; Chen, Y.; Zhao, Q.; Zhu, H. A Deep-Learning Model for Service QoS Prediction Based on Feature Mapping and Inference. *IEEE Trans. Serv. Comput.* **2023**, 1–14 . [CrossRef]
36. Lu, T.; Zhang, X.; Wang, Z.; Yan, M. A feature distribution smoothing network based on gaussian distribution for qos prediction. In Proceedings of the 2023 IEEE International Conference on Web Services (ICWS), IEEE Computer Society, Chicago, IL, USA, 2–8 July 2023; pp. 687–694. [CrossRef]
37. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781. [CrossRef]
38. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543. [CrossRef]
39. Herlocker, J.L.; Konstan, J.A.; Borchers, A.; Riedl, J. An Algorithmic Framework for Performing Collaborative Filtering. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, USA, 15–19 August 1999; pp. 230–237. [CrossRef]
40. Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. Item-Based Collaborative Filtering Recommendation Algorithms. In Proceedings of the 10th International Conference on World Wide Web, Hong Kong, China, 1–5 May 2001; pp. 285–295. [CrossRef]
41. Mnih, A.; Salakhutdinov, R.R. Probabilistic matrix factorization. *Adv. Neural Inf. Process. Syst.* **2007**, *20*, 1257–1264 .

42. Guo, H.; Tang, R.; Ye, Y.; Li, Z.; He, X. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. *arXiv* **2017**, arXiv:1703.04247. [CrossRef]
43. Wang, R.; Shivanna, R.; Cheng, D.; Jain, S.; Lin, D.; Hong, L.; Chi, E. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 1785–1797. [CrossRef]