2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS) | 978-1-6654-7315-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICPADS56603.2022.00102

Subgraph Sampling for Inductive Sparse Cloud Services QoS Prediction

Jianlong Xu[†]*, Zhiyu Xia[†], Yuhui Li[†], Yuxiang Zeng[†], Zhidan Liu[‡]
[†]College of Engineering, Shantou University, Shantou, 515063, China
[‡]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, 518060, China
Email: {xujianlong, 18zyxia, 20yxzeng}@stu.edu.cn, liyuhui1999@icloud.com, liuzhidan@szu.edu.cn

***Abstract*—Quality-of-Service (QoS) based collaborative prediction models are emerging to select appropriate edge cloud services for users. Nevertheless, there are still challenges in the real-world QoS prediction task. First, existing QoS prediction models are mostly transductive, failing to generalize to unseen users and services. Secondly, an accurate prediction model remains unexplored under the extreme sparse data scenario, where only a few interactions are available for collaborative filtering. To address these problems, we propose Inductive Subgraph Pattern Aware Graph Neural Network (ISPA-GNN), which leverages a novel graph-based collaborative filtering method with a subgraph sampling strategy. We further optimize the embeddings components, replacing the user/service embeddings with compositional context information to enable better generalization to unseen nodes while reducing memory usage. Extensive experiments on a large-scale real-world service QoS dataset demonstrate some decent properties of our model, including high prediction accuracy, memory efficiency, and slight performance degradation even if 25% of users/services are never seen.**

***Index Terms*—Collaborative Filtering, Cloud Service, Graph Neural Network, QoS Prediction**

I. Introduction

Recent years have witnessed the prosperity of edge cloud computing, 5G networks, and Internet-of-Things (IoT) services. Any organization or individual can become a service provider or a service user, which dramatically accelerates the flourishing of the cloud service market and enables the constant emergence of services that offer similar functionality. From users' perspective, applications integrated by cloud services may be impacted by the dynamic nature of the edge cloud network environments, service availability, and service performance. These make it challenging to match the user with good services. Therefore, effective and efficient service selection among large-scale Web services in the edge cloud is essential for facilitating high-performance service-based applications.

Quality of Service (QoS), which describes the non-functional properties of services (e.g., response time, throughput, and failure rate), has been commonly employed to distinguish the functionally equivalent Web services [1]. However, it is nontrivial to acquire pairwise QoS values due to the time and cost constraints. Besides, monitoring many services will introduce prohibitive overhead in computing and bandwidth resources for service providers. Derived from these facts, the collected QoS records are highly sparse. In addition to the restriction of monitoring cost, the ever-increasing size of the service market may aggravate the sparsity problem. Thus, developing an accurate and inductive QoS prediction model under the extremely sparse scenario is necessary for cloud service selection.

Many QoS prediction methods have been proposed in recent studies, among which Collaborative Filtering (CF) algorithms are the most prevalent. CF-based QoS prediction methods mainly exploit historic QoS records from similar users to predict the missing values [2], ensuring the recovery ability to an incomplete QoS value matrix of service recommender systems. However, the technical gaps below are hindering the industrial deployment of these methods:

- **Transduction.** Most of the existing QoS predictive methods [3]–[8] are transductive, indicating a frequently periodic model retraining is required for the latest joined users, services, and updated QoS data. An ideal model should be inductive and serve in an online manner with less update requirement. Current transductive solutions lead to two shortcomings: 1) the latest received QoS data from new users and services can not instantly contribute to the model; 2) the current training data predetermine the predicting QoS values of the new users and services, ignoring the preferences of the new users/services. In short, the transductive characteristic hampers the performance of a QoS prediction model, and low predictive performance for the newly joined users/services imposes frequent retrainings to preserve the model accuracy.
- **Suboptimal Performance in Highly Sparse Prediction.** Based on our extensive experiments, we find that both classic and the state-of-the-art neural network models fail to gain a satisfying performance in the highly sparse scenario. Every QoS prediction system, however, experiences a start-up period when the observed data is highly sparse. This may lead to grave consequences if the predictive model achieves an inferior accuracy. Participants (e.g., users, services) may leave the newly established system, leading to user churn or a business failure. Therefore, there is an urgent need for a QoS prediction model that is still effective under high sparsity.

*Corresponding author: Jianlong Xu (email: xujianlong@stu.edu.cn).

978-1-6654-7315-6/23/$31.00 ©2023 IEEE
DOI 10.1109/ICPADS56603.2022.00102

- **Massive Storage.** Previous model-based QoS prediction models served in service recommender systems create unique embeddings for each user, service, and sometimes, even their context feature. This will linearly scale up the model storage and thus occupy a massive space, which is incompatible with the edge cloud scenario. Therefore, an efficient embedding technique is yet to explore.

To address the limitations of existing QoS prediction models, this paper aims to introduce an inductive solution for highly sparse QoS prediction, which can be served in a dynamic, online cloud service recommender system without frequent retraining. We propose a novel model named Inductive Subgraph Pattern Aware Graph Neural Network (ISPA-GNN), which mainly learns local subgraph patterns generated from the QoS matrix. To improve the performance under high sparsity, we then design another context-guided neighborhood subgraph sampling strategy to extract coarse-grain network environments. We simplify the unique user/service embedding as the composition of context embeddings to generalize to unseen users and services to support inductive inference. Based on the experiments, our model can gain an excellent QoS predictive performance in the extreme sparse scenario. The main technical contributions are summarized as follows:

1) **New Approach**. ISPA-GNN is an inductive and memory-efficient approach based on GNN for QoS prediction, which can generalize to unseen users and edge services without retraining. Instead of assigning embedding for each user/service, the presented model uses compositional context embeddings to generate embeddings, effectively reducing memory usage.
2) **Decent Sampling Technique**. We present two sampling strategies, BFS-based subtree sampling, and context-guided neighborhood subgraph sampling, facilitating accurate QoS prediction under extreme data sparsity. Unlike current models, our model learns the patterns of reported QoS data directly rather than a low-rank approximation, enabling inductive inference.
3) **Extensive Experiment**. We conduct a wide range of experiments on a public QoS dataset and compared ours to well-known existing methods under extreme sparse matrix density settings, demonstrating the effectiveness of our proposed model in addressing sparse prediction problems. We then hold out up to 25% of the users and services in the training process to simulate the unseen users and services, ISPA-GNN gains a competitive performance under the sparse scenario (density = 1%), with only incurring up to 3.52% performance loss on RMSE metrics.

The rest of our paper is organized as follows: Section II illustrates a motivating example. Section III elaborates our model design. We then presents the experimental results in Section IV. Related work is summarized in Section V. Finally, Section VI draws the conclusion and discusses future work.

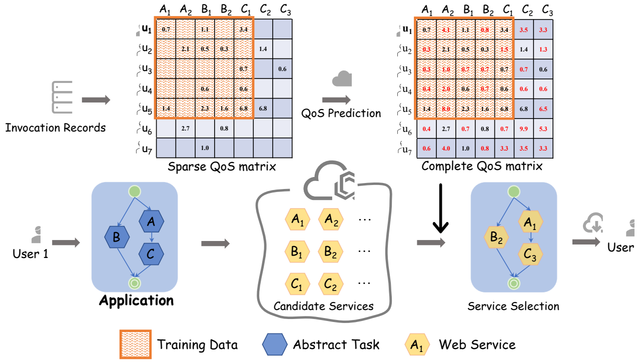

Fig. 1. Illustration of the inductive QoS prediction problem.

TABLE I
MAIN NOTATIONS

Notation	Meaning
Q, G	QoS record set and the global invocation graph
u, s	a user node and a service node
(u, s)	the edge between the user u and the service s
n_u^c, n_s^c	the context node of the user u (the service s)
$\mathbf{e}_u^{(L)}$, $\mathbf{e}_s^{(L)}$	embedding of the user u (the service s) after L layers message propagation
q_{us}	QoS value of the user u invoking the service s
$\mathbf{m}_{u\leftarrow s}^{(L)}$, $\mathbf{m}_{s\leftarrow u}^{(L)}$	message received by the user u (the service s) in the L propagation layer
$\mathbf{p}_u^{(L)}$, $\mathbf{p}_s^{(L)}$	the context representation of the user u (the service s) after L layers propagation
$f(\cdot)$	an MLP function
$\hat{r}_{us}$, r_{us}	the predicted value and the ground truth QoS value

II. Motivating Example

In this section, we provide a motivating example to demonstrate the need for an inductive QoS prediction model as shown in Figure 1. Assume that a user is running a service-based application. The application function is achieved by several abstract tasks. And the application should choose the best service from candidates for each task according to QoS values. The application will upload QoS values to the service recommender system after invocation. With collected QoS records, a QoS matrix can be built for service recommendation and service selection via matrix completion. However, new users and services (e.g., u_6, C_2) are not included in the previous training set. Therefore, the transductive model cannot predict QoS for new users/services. Besides, the real-world QoS matrix may be highly sparse. According to the business recommendation dataset, the sparsity is usually below 1% [9]. However, the current research mainly conducts experiments on 2.5%-10% or 10%-30%, we argue that this setting may fail to simulate the real-world scenario.

To achieve every missing value in the QoS matrix, we convert the matrix completion task into a link prediction problem and propose our approach, ISPA-GNN, which will be elaborated in the following section. The main notations used throughout the paper are described in Table I.

III. Our Solution

In this section, we elaborate our model design step by step, which consists of four stages: graph construction, subgraph extraction, graph neural network, and link prediction. Before

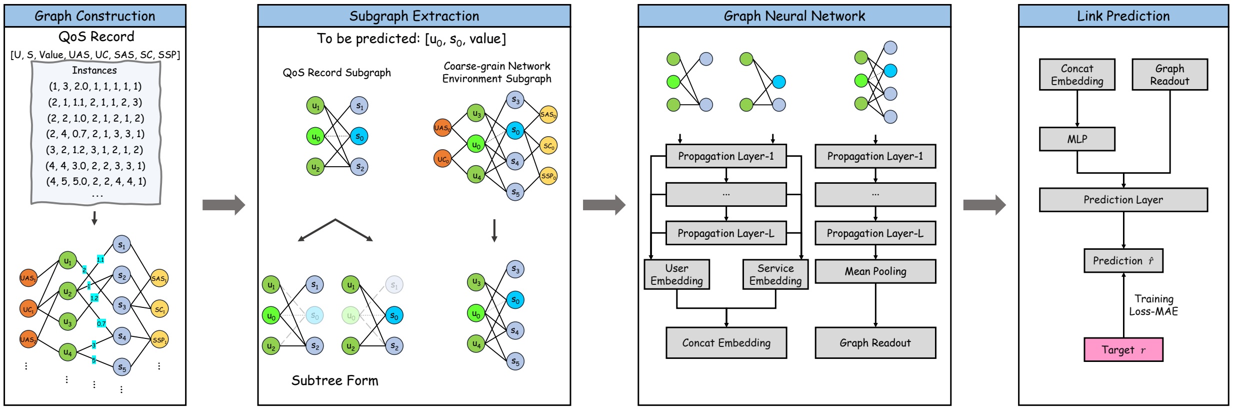

Fig. 2. ISPA-GNN framework for sparse QoS prediction. The framework consists of four procedures: 1) **Graph Construction.** We construct bipartite graph from QoS data matrix and user/service context graph from reported records. 2) **Subgraph Extraction.** We utilize two sampling strategies to extract subgraph for furthur GNN-based feature extraction. 3) **Graph Neural Network.** GNNs are applied to learn patterns inside the subgraph. 4) **Link Prediction.** Based on the extracted subgraph patterns, we can predict the QoS value via link prediction component.

we go into detail, we first present the whole framework of ISPA-GNN, as illustrated in Figure 2.

A. Graph Construction

Given a sparse QoS record set Q, we use undirected heterogeneous graph G to turn the invocation QoS record set Q into a graph-structured format. Initially, a node is either a user u or a service s in the graph G, and each edge (u, s) carries invocation details (e.g., Response time, Throughput) that user u experienced after invoking service s. Next, context nodes of the users/services are added into the graph. Users/services under the same context are linked to a specific context node. Then, the QoS prediction task can be considered as a link prediction problem, which is one of the most common formulations in the recommender system domain. We will detail the entire process of performing QoS prediction as a link prediction task in the following subsections.

Algorithm 1: QoS Subtree Extraction

```
Input: user $u$, service $s$, depth $h$
Output: User-, Service-QoS Subtree $T_u$ and $T_s$
Initialize set $U$ = $N_u$ = {u}, $S$ = $N_s$ = {s} ;
for $i$=$1$:$h$ do
    $N_u^{'} = \{u_i | \; if \; edges \; (u_i, N_s) \; exist\} \backslash U;$
    $N_v^{'} = \{s_i | \; if \; edges \; (s_i, N_u) \; exist\} \backslash S;$
    $N_s = N_s^{'}, N_u = N_u^{'}$ ;
    $U = U \cup N_u, S = S \cup N_s$ ;
end
Let $T_u$, $T_s$ be the node-induced subtree extracted from
    $G$ using $U$ and $S$ ;
Remove any edge (u,s) from $T_u$, $T_s$ ;
return $T_u$, $T_s$ ;
```

B. Subgraph Extraction

Based on the graph construction mentioned above, we extract two different subgraph (tree is a kind of special graph), namely, the QoS subtree and coarse-grain network environment subgraph.

1) QoS Subtree Extraction: First we introduce QoS subtree extraction. For a link prediction task for (u, s) pair, we extract subtree for u and s from G, respectively. The subtree contains local collaborative signals for link prediction. We present a Breadth First Search (BFS) based algorithm to extract subtree of height h in algorithm 1. Noted that, in the training process, the link between (u, s) is our label, therefore, we should remove target link (u, s) before running subgraph extraction algorithm.

Algorithm 2: Coarse-grain Network Environment Subgraph

```
Input: user $u$, service $s$, context $c$
Output: Context Neighbors Subgraph $G_{us}^{c}$
Initialize set $U$ = $N_u$ = {u}, $S$ = $N_s$ = {s} ;
$N_u^c = \{n_u^c \mid if \; edge \; (u, n_u^c) \; exist\}$;
$N_s^c = \{n_s^c \mid if \; edge \; (s, n_s^c) \; exist\}$;
for each $n \in N_u^c$ do
    $N_u^{'} = \{u_n | \; if \; edges \; (n, u_n) \; exist\} \backslash U;$
    $N_u = N_u^{'}$;
    $U = U \cup N_u$ ;
end
for each $n \in N_s^c$ do
    $N_s^{'} = \{s_n | \; if \; edges \; (n, s_n) \; exist\} \backslash S;$
    $N_s = N_s^{'}$;
    $S = S \cup N_s$ ;
end
Let $G_{us}^c$ be the node-induced subgraph constructed
    from $G$ using $U$ and $S$ ;
Remove any edge (u,s) from $G_{us}^c$ ;
return $G_{us}^c$ ;
```

2) Coarse-grain Network Environment Subgraph: If the QoS matrix is not highly sparse, then the extracted QoS subtree contains rich semantics for collaborative filtering. However, when it comes to the extreme sparse scenario, the subtree only contain few edges. This may degrade the prediction performance due to the lack of collaborative signals. To cope with this problem, we propose another subgraph extraction strategy to describe coarse-grain network environment between target user-service pair (u, s). The core idea behind is that the users and services in the same region may have very similar QoS values. We try to exploit other users' QoS record who are in the same region (e.g., AS, Country) with target user u, so do the service, to provide more information for QoS prediction. As mentioned in subsection III-A, we first joint the existing users and services under different context environments with respectively different nodes. In this paper, these context nodes are denoted as n_u^c (for user context) and n_s^c (for service context). With different context nodes, we can filter out neighbors within the same context environment as the target user or service does, respectively. Let n_u^c denotes the set of users share the same context c with user u, similarly n_s^c. The process of the coarse-grain network environment subgraph extraction is described in algorithm 2.

C. Graph Neural Network for Subgraph Pattern Learning

Based on the extracted subgraphs (QoS subtree and network environment subgraph), we utilize graph neural networks to extract features for link prediction. Before we introduce the message passing in GNN, let look at the embeddings components.

1) Compositional Embeddings: Traditional model-based Web service recommender system commonly uses ID as input features to get user/service embeddings. However, this widely adopted design has two disadvantages:

- **Embedding Extrapolation Failure**. The ID-based embeddings can not instantly adapt to the new user. Newly assign embeddings need to be trained. Therefore, ID-based embedding technique is not suitable for inductive settings.
- **Memory Inefficient**. The system assigns a unique embedding for new users, services, and context, which often occupies extensive storage resources. We find that there is an opportunity to optimize.

To meet our requirement of being inductive and memory efficient, we propose our compositional embeddings. The basic idea is to decouple ID-based embeddings. ID-based embeddings aggregate information from both QoS values and user/service context into a single vector. We decouple into personalized part and the common part. The personalized part is the information contained in the QoS subtree, while the common part is constructed by the sum of context embeddings (e.g., AS, Country, Service Provider). When a new user joins, as long as the user reports its context, then the common part can be computed. With QoS values reported by the users, the personalized part can be calculated by GNNs which will be introduced later. In this way, we successfully get rid of the transductive problem in previous methods to enable inductive prediction even for new users/services.

2) Message Passing: In this part, we describe the message-passing scheme in GNN to extract features on two subgraphs. Message-passing layers are utilized for embedding propagation, which capture the collaborative information in the subgraphs. Inspired by [9] and [10], each message passing layer consists of two stages: message construction stage and message aggregation stage.

Message Construction. On the QoS subtree, the message received by the target user u is formulated as:

$$\mathbf{m}_{u \leftarrow s}^{(L)} = \mathbf{W}_{us}^{(L)}(\mathbf{e}_s^{(L-1)} || q_{us}), \tag{1}$$

where $\mathbf{m}_{u \leftarrow s}^{(L)}$ denotes a message embedding which contains the information distilled from the invoked service embedding $\mathbf{e}_s^{(L-1)}$ and the corresponding QoS value q_{us} for an existing invoked record (u, s). L is the propagation layer. $\mathbf{e}_s^{(0)}$ denotes the initial embedding of the service s. $||$ is a concatenation operation. $\mathbf{W}_{us}^{(L)} \in \mathbb{R}^{d' \times d}$ is a trainable weight matrix which distills information for propagation layer L. d' denotes the transformation size. Analogously, we can obtain the message $\mathbf{m}_{s \leftarrow u}^{(L)}$ for the target service s in the L propagation layer from s's QoS subtree as:

$$\mathbf{m}_{s \leftarrow u}^{(L)} = \mathbf{W}_{su}^{(L)}(\mathbf{e}_u^{(L-1)} || q_{us}) \tag{2}$$

On the network environment subgraph, we adopt the same message construction operation to obtain the context message representation of the target user u and service s.

Message Aggregation In this stage, we aggregate the messages constructed on the target u's subtree. We define the aggregation function as:

$$\mathbf{e}_u^{(L)} = \sigma(\frac{1}{|\mathcal{N}_u|} \sum_{s \in \mathcal{N}_u} \mathbf{m}_{u \leftarrow s}^{(L)}), \tag{3}$$

where $\mathbf{e}_u^{(L)}$ denotes the representation embedding of the target user u obtained after L message propagation layer. $\mathcal{N}_u$ is the $(L-1)$-hop services that are invoked by the user u. σ denotes an activation function, such as LeakyReLU [11]. Analogously, we aggregate the messages of the target service s to achieve the representation $\mathbf{e}_s^{(L)}$ as:

$$\mathbf{e}_s^{(L)} = \sigma(\frac{1}{|\mathcal{N}_s|} \sum_{u \in \mathcal{N}_s} \mathbf{m}_{s \leftarrow u}^{(L)}) \tag{4}$$

For the network environment subgraph, we aggregate the messages by mean pooling on the user side and the service side to achieve the embedding of every layer. The context representation of the L propagation layer $\mathbf{p}_u^{(L)}$ and $\mathbf{p}_s^{(L)}$ can then be obtained, respectively. Noted that the initialized embedding $\mathbf{p}_u^{(0)}$ and $\mathbf{p}_s^{(0)}$ are exactly the same with $\mathbf{e}_u^{(0)}$, $\mathbf{e}_s^{(0)}$ when constructing messages for the network environment subgraph.

With L layers messages propagation, we obtain multiple representations for the target user u and service s, namely $\{\mathbf{e}_u^{(0)}, \mathbf{e}_u^{(1)}, ..., \mathbf{e}_u^{(L)}\}$ and $\{\mathbf{e}_s^{(0)}, \mathbf{e}_s^{(1)}, ..., \mathbf{e}_s^{(L)}\}$, respectively. Since the representations are generated from the respective QoS subtree, rich personalized semantics are contained in these representation embeddings. Thus, we concatenate the embeddings generated from the QoS subtree for prediction:

$$\mathbf{e} = \mathbf{e}_u^{(0)} || \mathbf{e}_u^{(1)} ... || \mathbf{e}_u^{(L)} || \mathbf{e}_s^{(0)} || \mathbf{e}_s^{(1)} ... || \mathbf{e}_s^{(L)}, \tag{5}$$

where || denotes the concatenation operation. Besides, we have $\{\mathbf{p}_u^{(0)}, \mathbf{p}_u^{(1)}, ..., \mathbf{p}_u^{(L)}\}$ and $\{\mathbf{p}_s^{(0)}, \mathbf{p}_s^{(1)}, ..., \mathbf{p}_s^{(L)}\}$ as context readouts via mean pooling distilled from the coarse-grain network environment subgraph, reflecting the common characteristics of the respective context environment they are within. For each propagation layer, we concatenate both user side and service side representation as the graph readout:

$$\mathbf{p}^{(L)} = \mathbf{p}_u^{(L)} || \mathbf{p}_s^{(L)}, \tag{6}$$

where $\mathbf{p}^{(L)}$ is the graph readout in the L-th propagation layer.

D. Link Prediction

In this stage, we first apply a multi-layer perceptron (MLP) neural network to reduce the embeddings from the QoS subtree:

$$\mathbf{e}^* = f(\mathbf{e}|\mathbf{\Theta}_f), \tag{7}$$

where $\mathbf{e}^*$ is a compressed intermediate embedding, and $\mathbf{\Theta}_f$ denotes the parameters of the MLP function $f(\cdot)$. For the network environment subgraph, since the context neighbors may contain noise nodes that are dissimilar with the target node, we design a shortcut for directly transmitting all the graph readout to the prediction layer instead of going through the multi-layer perceptron. So far, the prediction layer is capable of achieving a link prediction $\hat{r}_{us}$ by:

$$\mathbf{x} = \mathbf{e}^* || \mathbf{p}^0 || \mathbf{p}^1 || ... || \mathbf{p}^L, \tag{8}$$

$$\hat{r}_{us} = g(\mathbf{x}|\mathbf{\Theta}_g), \tag{9}$$

where || is a concatenation operation, x is the input embedding of the prediction layer, and $\mathbf{\Theta}_g$ denotes the parameters of a fully connected layer $g(\cdot)$.

IV. Experiments

In this section, we conduct experiments to show our model performance. Different experiments are designed to answer the following research questions:

RQ1 Does ISPA-GNN outperform baselines under high data sparsity?

RQ2 Is ISPA-GNN still effective when new users/services join?

RQ3 How does ISPA-GNN perform on memory efficiency?

RQ4 How do the propagation layers affect our model?

A. Experimental Settings

We conduct all our experiments on the WS-DREAM [12] dataset, a large-scale real-world QoS dataset that collected 1,974,675 web service invocation QoS records of response-time (RT) and throughput (TP) between 339 users and 5825 services from distributed locations. All experiments are carried out on a GPU server with 6 cores Intel Xeon E5 CPU, NVIDIA RTX2080Ti, and 32G RAM under Ubuntu OS. We implement our model based on PyTorch v1.7.0 and DGL v0.6, which are widely adopted deep learning framework and graph learning library, respectively. The experimental dataset is divided into two groups, namely the training matrix and the test matrix. The elements in the training matrix are randomly selected, and the remaining forms the test matrix. In order to accurately simulate the highly sparse QoS scenario, we vary the QoS record matrix density (MD) between 0.50% and 2.00%, with a step length of 0.50%, to form the training set. In terms of the loss function for our model training, we minimize the mean absolute error (MAE) between the predictions and the labels:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} I(i,j) \left| r_{ij} - \hat{r}_{ij} \right|, \tag{10}$$

where $I(i,j)$ is a mask function indicating the observed edges in graph G.

For the gradient descent algorithm, we apply Adam [13] as the optimizer. For other training detailed settings of our model, we set the messages propagation layers as 2, and fix the embedding dimension in every massage passing and aggregation layer as 32. The hidden unit of the MLP and the prediction layer are set to {128, 128} and {128}. For the training process, we set the batch size to 128. The initial learning rate is 0.002.

B. Baseline Methods

We compare our model with existing classic and state-of-the-art methods to demonstrate the advantages of ours. PMF [4] and NMF [5] are widely used model-based methods, while UPCC [14], IPCC [15] and UIPCC [3] are memory-based methods. LN_LFM [16] conducts MF incorporated with location information. NIMF [6] and EMF [7] are hybrid methods. For neural network based methods, we reproduce CSMF and DNM according to [17] and [8]. For fair comparison, we adopt model codes from open source library released by WSDream[1]for the aforementioned baseline methods.

C. Evaluation Metrics

In this paper, we focus on the accuracy of the prediction and employ the following classic predictive accuracy metrics to evaluate our model performance in comparison with other existing methods.

- **MAE** (Mean Absolute Error) measures the average absolute deviation between a predicted rating and the real rating. MAE is calculated as:

$$MAE = \frac{1}{N} \sum_{us} |r_{us} - \hat{r}_{us}|, \tag{11}$$

  where r_{us} is the real QoS value of service s observed by user u, and $\hat{r}_{us}$ is the predicted value, and N is the number of testing samples.

- **RMSE** (Root Mean Squared Error) is used to illustrate the degree of dispersion of the sample. RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{us} |r_{us} - \hat{r}_{us}|^2} \tag{12}$$

For both metrics, smaller value indicates a better performance.

[1] https://github.com/wsdream

TABLE II
Performance comparison of QoS response-time prediction using different matrix densities

Density	**MD=0.50%**		**MD=1.00%**		**MD=1.50%**		**MD=2.00%**	
Methods	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
UPCC	0.8903	1.8396	0.8664	1.8281	0.8042	1.7156	0.7389	1.5496
IPCC	0.8079	1.9147	0.7719	1.7709	0.7587	1.7106	0.7602	1.6954
UIPCC	0.8255	1.6987	0.8071	1.7000	0.7645	1.6357	0.7203	1.5229
PMF	0.8923	2.1224	0.8947	2.1346	0.8447	2.0642	0.7728	1.9509
NMF	0.8920	2.1218	0.8577	2.0740	0.7783	1.9453	0.7099	1.8270
LN_LFM	0.8491	**1.6509**	0.7754	**1.5749**	0.7276	**1.5219**	0.6878	**1.4787**
NIMF	0.8982	2.1376	0.7003	1.8177	0.7582	1.9400	0.7003	1.8177
EMF	0.8923	2.1224	0.8828	2.1221	0.7719	1.9877	0.6988	1.8612
CSMF	0.7935	1.9414	0.6636	1.7307	0.5951	1.5587	0.5835	1.5205
DNM	**0.7377**	1.9720	**0.6233**	1.7009	**0.5631**	1.5682	**0.5323**	1.5013
ISPA-GNN-1	0.5979	1.5781	0.4998	**1.4208**	0.4547	1.3511	0.4292	1.3110
ISPA-GNN-2	**0.5757**	**1.5725**	**0.4882**	1.4251	**0.4448**	**1.3466**	**0.4245**	**1.3016**
Gains	21.96%	4.75%	21.67%	9.78%	21.01%	11.52%	20.25%	11.98%

[1] The best values for both baselines and ISPA-GNN are marked in bold. The gains are calculated based on them.

TABLE III
Performance comparison of QoS throughput prediction using different matrix densities

Density	**MD=0.50%**		**MD=1.00%**		**MD=1.50%**		**MD=2.00%**	
Methods	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
UPCC	52.5699	104.1792	49.2746	101.4672	40.2990	87.1788	34.0954	73.5738
IPCC	**31.1793**	77.5450	30.0654	**74.0104**	29.9783	72.7395	31.0753	73.6052
UIPCC	34.6277	**75.0232**	33.6681	74.4862	31.7632	72.4633	30.8620	70.2904
PMF	41.5016	105.3395	34.1022	93.1312	29.5836	84.3541	26.5696	77.6570
NMF	41.5094	105.3577	34.1011	93.1223	29.5782	84.2814	26.5750	77.5530
LN_LFM	40.2361	84.8351	34.1250	75.7049	31.0520	71.1949	28.3710	66.6500
NIMF	39.7875	102.8491	31.9814	88.1882	28.7872	79.7137	27.3017	74.7455
EMF	44.1035	110.8194	35.7752	97.9137	31.1697	87.1053	28.9771	80.0971
CSMF	36.6621	99.2980	**28.9750**	80.9790	**24.6179**	**70.0257**	**22.1117**	**62.5988**
DNM	34.8189	98.7324	28.2889	82.7692	26.1544	79.2907	22.8549	73.2483
ISPA-GNN-1	30.5991	79.4281	27.3405	72.4542	24.6495	67.4802	21.8470	61.5726
ISPA-GNN-2	**29.7816**	**78.3509**	**25.8139**	**70.0355**	**23.4371**	**65.4268**	**21.1362**	**59.9950**
Gains	4.48%	-4.44%	8.75%	5.37%	4.80%	6.57%	4.41%	4.16%

[1] The best values for both baselines and ISPA-GNN are marked in bold. The gains are calculated based on them.

D. Comparison with Baseline Methods (RQ1)

Table II and Table III shows the performance comparison results. Specifically, ISPA-GNN-1 denotes the ISPA-GNN model without activating the network environment subgraph processing and only applies GNN on two QoS subtrees. ISPA-GNN-2 is the intact design of our model. We test our proposed model for 5 times in every density and take the average of the five as the final results. According to the results, we have the following observations:

- For the baseline models, neural-based models like DNM are generally superior to those MF-based and PCC-based methods. This confirms that the neural network models have powerful non-linearity approximation ability in handling the QoS prediction tasks. However, when it comes to the extreme sparse scenario, the preponderance of the DNM becomes smaller as the lack of graph-based information may constrain the neural network to exploit its powerful feature extraction ability perfectly. Hence, abundant graph-based information is crucial for model inference, whereby sufficient auxiliary information is guaranteed for further processing.
- Compared to those pure neighborhood-based CF methods, the performance of CSMF, DNM, and LN_LFM verify that with more context introduced into the model, the prediction accuracy further improves. This demonstrates that contextual information has positive effects on QoS prediction.
- Compared with the baseline methods, the graph neural network (GNN) based model (ISPA-GNN-1) consistently performs flawlessly on all sparse matrix densities. Such improvement by a large margin is attributed to the ability of the GNN that captures the high-order connectivity on the subgraph pattern. Our pure GNN-based model ISPA-GNN-1 achieves a satisfying performance in the extreme sparse scenario by explicitly injecting the collaborative signal into model inference. It proves that the user-service

TABLE IV
PERFORMANCE COMPARISON OF QOS RESPONSE-TIME PREDICTION *w.r.t* THE COLD START ISSUE

Holdout		**0%**	**5%**	**10%**	**15%**	**20%**	**25%**
ISPA-GNN-1	**MAE**	0.4998	0.5037	0.5077	0.5184	0.5212	0.5359
	RMSE	1.4208	1.4457	1.4428	1.5030	1.4729	1.4909
ISPA-GNN-2	**MAE**	0.4882	0.4959	0.5002	0.5079	0.5183	0.5219
	RMSE	1.4251	1.4432	1.4404	1.4844	1.4738	1.4752
Degradation-MAE		0%	1.59%	2.46%	4.03%	6.17%	6.91%
Degradation-RMSE		0%	1.28%	1.08%	4.16%	3.42%	3.52%

interaction subgraph contains rich semantics. Leveraging the collaborative signal is crucial to refining a better representation of a user or a service in the QoS prediction problem while facing a highly sparse situation.

- ISPA-GNN-2 gains the best performance among other baseline methods in both QoS attributes datasets. Specifically, ISPA-GNN-2 improves over the strongest baselines *w.r.t* the MAE by over 20% in different matrix densities on response time. When the matrix density is decreased sequentially, the improvement of the ISPA-GNN-2 model becomes more significant compared to the ISPA-GNN-1 that only learns from the two QoS subtrees. The network environment subgraph provides the GNN with adequate neighbor nodes that contribute to refining the common context feature. This also demonstrates the contribution of the coarse-grain network environment in capturing the feature, which leads the model to alleviate the cold start failure caused by extreme data sparsity.

E. Extrapolation Performance (RQ2)

In this section, we further investigate the extrapolation performance of our model when facing the newly joined users/services. We consider a scenario where a well-trained QoS model is deployed while new users and services are constantly joining in. To this end, we randomly hold out {5%, 10%, 15%, 20%, 25%} of users and services from the dataset and fix the matrix density as 1% to form the training matrix from the remaining dataset. The rest of the records from the original dataset are all divided into the testset. Other settings of the model remain the same. This experiment is performed on the response-time dataset.

Table IV summarizes the experimental results. With the increased percentage of holdout elements in the training set, ISPA-GNN incurs a performance degradation both in MAE and RMSE metrics. However, our proposed model still outperforms all other baseline methods training on the original set when we jointly compare the result with Table II. This result declares that a comprehensive additional quality degradation (6.91% in MAE and 3.52% in RMSE) leads to an extrapolation (up to 25% unseen participants) ability of our model. Comparing with ISPA-GNN-1, ISPA-GNN-2 achieves a better performance under every holdout rate in MAE metrics. It verifies the effectiveness of the coarse-grain network environment subgraph to reveal the common preference and thus ensure the model performance.

TABLE V
EMBEDDING QUANTITY BETWEEN MF-BASED MODELS AND ISPA-GNN

Methods	**Users**	**Services**	**Users+Services**
MF-based	339	5825	6164
ISPA-GNN	139	2737	2876
Reduction	59.00%	53.01%	53.34%

F. Memory Efficiency of ISPA-GNN(RQ3)

We gain insight into the memory efficiency of our model compared with other transductive methods. They assign embeddings for each UserID, ServiceID, and ContextID. We use compositional embeddings instead in ISPA-GNN without using ID. We conduct an embedding quantity comparison between transductive models and our ISPA-GNN. The result shows in Table V. Our model uses [*Country, AS*] as users context and [*Country, AS, Provider*] as services context and takes only 139 and 2737 embeddings to initialize the users and services representations, compared to the 339 and 5825 in other models. We gain 53.34% memory efficiency improvement by embedding sharing technique than the traditional ID initialization does. Besides, such a contextual embedding technique can easily generalize to unseen nodes because the context environment has already existed in the previous training in most cases. Noted that, for a very-large-scale recommender system, the gap between our scheme and others will enlarge.

G. Impact of Propagation Layers (RQ4)

To investigate how the layers in GNNs affect the performance of ISPA-GNN, we vary the parameter of the message passing layers from 1 to 5 to conduct the experiments. In particular, we run our model 5 times on the RT and TP datasets under the MD of 1.00%. Other parameters keep the same as the default settings. Figure 3 and Figure 4 plotted the results.

The result shows that a small number of convolution layers (two layers in RT and one layer in TP) usually leads to the best performance of the model. We realize that the key reason may lie in two folds: 1) High orders users and services may introduce noises to the node representation and training process, leading the model to achieve a suboptimal performance. 2) The prior work [18] demonstrates that the over-smoothing problem will weaken the embedding representation and therefore hinder the performance of the model. In the QoS system of WSDream, ISPA-GNN achieves peak performance with two layers in the RT dataset and one layer in the TP dataset. However, when the convolution layer of our model is

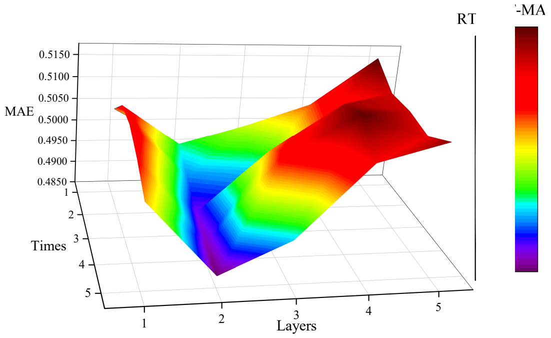

Fig. 3. Various convolution layers *w.r.t* the performance of ISPA-GNN in the QoS response-time dataset.

1, the message from the high-order will not be received during the training process. Hence we set the default convolution layers of our model as 2 in the default experiment settings.

V. Related Work

Accurate QoS values are critical for various QoS-driven approaches to cloud services. Collaborative Filtering (CF), which is a mature technique in recommender systems for predicting unknown ratings [1], has been applied to facilitate QoS prediction tasks. Existing CF-based approaches can be categorized into memory-based, model-based, and deep learning-based approaches.

For memory-based CF methods, xPCC (eg. UPCC [14] and IPCC [15]) methods applied Pearson Correlation Coefficient (PCC) to calculate the similarity between users/services and then obtain the predicting QoS values. UIPCC [3] attempted to combine both UPCC and IPCC approaches to gain better QoS prediction. As the QoS values might vary from invocation contexts, many context-aware methods (i.e., [19]–[21]) were proposed by using the contextual information for performance improvement. However, the main weaknesses of these memory-based CF methods are that they are prone to low performance when the data is sparse.

In terms of model-based CF methods, matrix factorization (MF) is the most widely-used approach undertaking the QoS prediction tasks [2]. Concretely, Zheng et al. [4] proposed a probabilistic matrix factorization (PMF) to decompose the user-service matrix for personalized QoS prediction. Analogous to PMF, several typical model-based approaches such as NMF [5], LN_LFM [16] and AMF [22], are all MF-driven. Approaches exploiting additional side information (i.e., location) were proposed to improve the accuracy of the prediction [23]. Wu et al. [17] made full use of implicit and explicit contextual factors in the QoS data and proposed a general context-sensitive matrix factorization approach (CSMF). The major drawback of MF is its transductive characteristic. Some studies combine memory-based and model-based CF approaches to achieve hybrid methods (eg. NIMF [6] and EMF [7]). Such hybrid methods can generally obtain better prediction results. However, they also inherit drawbacks from both memory-based and model-based methods.

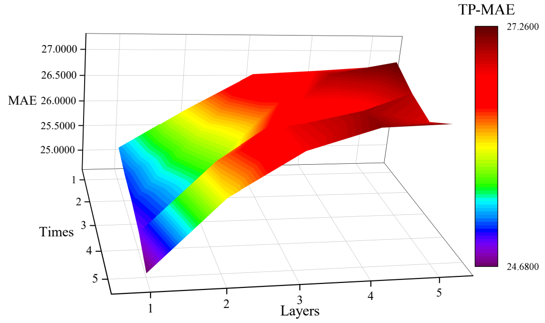

Fig. 4. Various convolution layers *w.r.t* the performance of ISPA-GNN in the QoS throughput dataset.

Another research line is neural network-based (NN-based) attempts on QoS prediction tasks. Wu et al. [8] designed a Deep Neural Model (DNM), capturing rich contextual features for multiple attributes QoS prediction. Xiong et al. [24] proposed a personalized LSTM (P-LSTM), which can capture the dynamic latent representations of multiple users and services. Furthermore, Zhou et al. [25] suggested a spatio-temporal context-aware collaborative multilayered neural network model. The outstanding performance that the aforementioned NN-based methods have achieved demonstrates that incorporating neural networks with QoS prediction tasks is a promising direction.

Inspired by the prosperity of graph-based neural network, recent research has made some attempts in the recommendation domain. Wang et al. [9] proposed a graph-based recommendation framework named Neural Graph Collaborative Filtering (NGCF), which explicitly encodes the collaborative signal and the connectivity information to the node embedding for collaborative filtering. He et al. [10] further optimized the design of NGCF and proposed LightGCN. Liu et al. [18] designed an Interest-aware Message-Passing GCN (IMP-GCN) model to avoid propagating negative information from higher-order neighbors. Such solutions performs decently for recommender systems and the GNN-based models are yelling the powerful performance for the CF-related tasks.

VI. Conclusion

We study the inductive QoS prediction problem under extreme data sparsity. Previous transductive models struggle to handle new users/services appropriately, and the highly sparse records may degrade the model performance. To address these problems, we proposed ISPA-GNN with two novel designs. First, instead of learning latent representation for matrix completion, we directly learn local QoS patterns around the target user and service via neighborhood sampling. We further improve the model performance via context-guided neighborhood subgraph sampling. We exploit GNNs to learn user/service embeddings for collaborative filtering with two subgraphs extracted. Second, we use compositional embeddings rather than assigning unique embedding for each user/service to enable inductive inference and reduce memory usage. We conduct extensive experiments to prove the effectiveness of our model and show its nice properties for real-world cloud service recommendation. In future work, we intend to incorporate GNN with time series, which will infer the services' QoS fluctuation.

Acknowledgement

This research was financially supported by Guangdong Province Special Fund for Science and Technology ("major special projects + task list") Project (No.STKJ2021201), 2020 Li Ka Shing Foundation Cross-Disciplinary Research Grant (No.2020LKSFG08D), Special Projects in Key Fields of Guangdong Universities (No.2022ZDZX1008, 2020ZDZX3073), Guangdong Basic and Applied Basic Research Foundation (No.2021A1515012527), and in part by Science and Technology Planning Project of Guangdong Province (2019B010116001).

References

[1] Y. Zhang, X. Zhang, P. Zhang, and J. Luo, "Credible and online qos prediction for services in unreliable cloud environment," in *2020 IEEE International Conference on Services Computing (SCC)*, 2020, pp. 272–279.

[2] Z. Zheng, L. Xiaoli, M. Tang, F. Xie, and M. R. Lyu, "Web service qos prediction via collaborative filtering: A survey," *IEEE Transactions on Services Computing*, pp. 1–1, 2020.

[3] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Qos-aware web service recommendation by collaborative filtering," *IEEE Transactions on Services Computing*, vol. 4, no. 2, pp. 140–152, 2011.

[4] Z. Zheng and M. R. Lyu, "Personalized reliability prediction of web services," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 22, no. 2, pp. 1–25, 2013.

[5] Y. Zhang, Z. Zheng, and M. R. Lyu, "Exploring latent features for memory-based qos prediction in cloud computing," in *2011 IEEE 30th International Symposium on Reliable Distributed Systems*, 2011, pp. 1–10.

[6] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative web service qos prediction via neighborhood integrated matrix factorization," *IEEE Transactions on Services Computing*, vol. 6, no. 3, pp. 289–299, 2013.

[7] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "An extended matrix factorization approach for qos prediction in service selection," in *2012 IEEE Ninth International Conference on Services Computing*, 2012, pp. 162–169.

[8] H. Wu, Z. Zhang, J. Luo, K. Yue, and C.-H. Hsu, "Multiple attributes qos prediction via deep neural model with contexts*," *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1084–1096, 2021.

[9] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019, pp. 165–174.

[10] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 639–648.

[11] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013, p. 3.

[12] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating qos of real-world web services," *IEEE Transactions on Services Computing*, vol. 7, no. 1, pp. 32–39, 2014.

[13] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.

[14] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized qos prediction forweb services via collaborative filtering," in *IEEE International Conference on Web Services (ICWS 2007)*, 2007, pp. 439–446.

[15] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175–186.

[16] D. Yu, Y. Liu, Y. Xu, and Y. Yin, "Personalized qos prediction for web services using latent factor models," in *2014 IEEE International Conference on Services Computing*, 2014, pp. 107–114.

[17] H. Wu, K. Yue, B. Li, B. Zhang, and C.-H. Hsu, "Collaborative qos prediction with context-sensitive matrix factorization," *Future Generation Computer Systems*, vol. 82, pp. 669–678, 2018.

[18] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing gcn for recommendation," in *Proceedings of the Web Conference 2021*, 2021, pp. 1296–1305.

[19] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "Qos prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 134–144, 2019.

[20] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation," in *2012 IEEE 19th International Conference on Web Services*, 2012, pp. 202–209.

[21] F. Chen, S. Yuan, and B. Mu, "User-qos-based web service clustering for qos prediction," in *2015 IEEE International Conference on Web Services*, 2015, pp. 583–590.

[22] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "Online qos prediction for runtime service adaptation via adaptive matrix factorization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 10, pp. 2911–2924, 2017.

[23] M. Tang, W. Liang, Y. Yang, and J. Xie, "A factorization machine-based qos prediction approach for mobile service selection," *IEEE Access*, vol. 7, pp. 32 961–32 970, 2019.

[24] R. Xiong, J. Wang, Z. Li, B. Li, and P. C. K. Hung, "Personalized lstm based matrix factorization for online qos prediction," in *2018 IEEE International Conference on Web Services (ICWS)*, 2018, pp. 34–41.

[25] Q. Zhou, H. Wu, K. Yue, and C.-H. Hsu, "Spatio-temporal context-aware collaborative qos prediction," *Future Generation Computer Systems*, vol. 100, pp. 46–57, 2019.