

A Two-Stream Light Graph Convolution Network-based Latent Factor Model for Accurate Cloud Service QoS Estimation

Abstract—Historical Quality-of-Service (QoS) data regarding past user-service invocations are vital to understand the user behaviors and cloud service conditions. A matrix factorization (MF)-based collaborative filtering (CF) model has proven to be highly effective in performing representation learning to such QoS data. However, its performance is hindered by its linear interaction and implicit encoding of collaborative QoS signal. To address this critical issue, this paper presents a Two-stream Light Graph Convolution Network-based latent factor (TLGCN) model with the three-fold ideas: 1) constructing a multilayered and fully-connected network to represent services' nonlinear latent features; 2) integrating the user-service interactions, i.e., the bipartite graph structure into the representation learning process with a light graph convolution network for illustrating the high-order connectivity information in QoS data; and 3) incorporating the data density-oriented modeling mechanism into the input and output of TLGCN for high computational efficiency. Experimental results on two real QoS datasets demonstrate that the proposed TLGCN model significantly outperforms its state-of-the-art peers in both estimation accuracy for missing QoS data and computational efficiency.

Keywords—*Quality-of-Service, Representation Learning, Data Science, Cloud Service, Missing Data Estimation, Graph Neural Network, Non-Euclidean Data.*

I. INTRODUCTION

In industrial software applications based on service-oriented architectures [1-3], cloud services are taken as the fundamental components to achieve easy exchange of data among them over the World Wide Web. With this era of cloud computing, more and more service providers serve their customers by deploying cloud services, i.e., the number of online cloud services are explosively increasing [4-6]. And owing to the highly similar functionality of available candidate cloud services, it becomes a vital challenge for users to select appropriate ones and build a reliable system [7, 8].

Most nonfunctional characteristics of cloud services can be reflected by Quality-of-Service (QoS) data at both server and user sides [9-13], which is critical to service selection. Server-side QoS data, e.g., popularity and price, can be directly

provided by service providers. On the other hand, user-side data, e.g., response time and throughput, vary greatly among different users depending on many factors [13-15], e.g., invoking environment. Warming-up tests is a straightforward approach to retrieve user-side QoS data by actually invoking each cloud service [1-3]. However, owing to the large number of candidate cloud services and charged commercial service invocations in most cases, to perform such warming-up tests is very time-consuming and expensive thereby impractical.

Motivated by the success of collaborative filtering (CF) in e-commerce, researchers employ this technique to implement QoS estimation aiming at efficiently and accurately estimating missing QoS data based on historical ones [16-24]. A CF-based QoS estimator works on a given user-service QoS matrix, where the information of users, cloud services and their corresponding invocations are recorded in its rows, columns and entries, respectively. In real-world scenarios, the target QoS matrix is highly sparse owing to the fact that a user typically experiences just a very limited number of candidate cloud services [1-3, 16]. Hence, how to perform efficient and accurate representation learning based on such a highly sparse user-service matrix is the major problem of CF-based QoS estimation.

Among various CF-based QoS estimators, a matrix factorization (MF)-based latent factor analysis (LFA) model has proven to be effective in performing representation learning to QoS data [16, 20-28]. It works by mapping both users and services into a low-rank latent feature space to extract their latent features based on the known entries in a target user-service QoS matrix, and then perform inner products of corresponding latent features of users and cloud services to obtain the missing entries. *Zhu et al.* [20] incorporate a similarity-maintaining privacy preservation strategy into their location-aware low-rank matrix factorization model to achieve reliable and accurate QoS estimation. *Luo et al.* [25] adopt the principle of alternating direction method and ensemble mechanism to build an effective matrix factorization-based QoS estimator. *Yang et al.* [27] consider the correlation between users and services, i.e., incorporate their neighborhood information into the factorization process for accurate QoS estimation. *Ryu et al.* [23] propose a matrix factorization-based

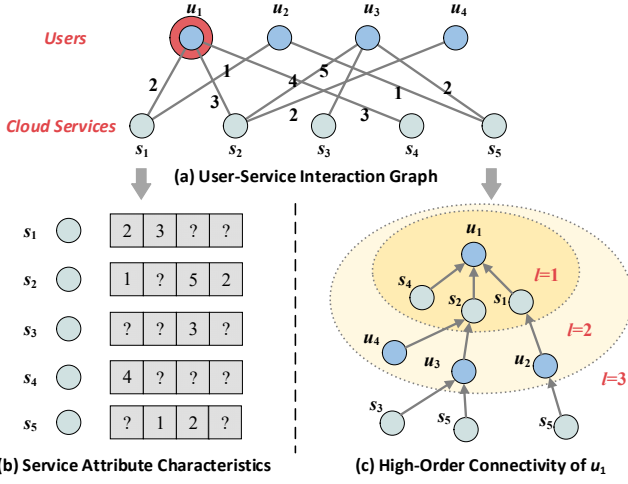


Fig. 1. An illustrative example of the attribute characteristics (e.g., response time and throughput) and high-order connectivity information in a user-service interaction graph.

preference propagation approach to use the fused invocation and neighborhood similarity information for addressing the cold start problem in QoS estimation. Wu *et al.* [28] propose a density peaks-based clustering method to detect the neighborhood and noises of QoS data for achieving a data-characteristic-aware QoS estimator.

Despite their effectiveness, existing matrix factorization-based LFA methods possess inherent deficiencies. As depicted in Fig. 1, the user-service interaction graph contains the attribute characteristics of users/services and high-order connectivity information. MF-based LFA models adopt inner product to combine the multiplication of the latent feature vectors of users and services, whose linear nature limits to fully capture the implicit information in them. Apart from this, they cannot well handle the non-Euclidean structure in QoS data, i.e., suffer the unavailability of leveraging the high-order connectivity information, resulting in extremely sparse collaborative QoS signals and inevitable accuracy loss.

Considering the great potential of graph convolutional networks (GCNs) to address the non-Euclidean data in other fields [28-32] and the unique attribute characteristics of user-service QoS data, this paper presents a **Two-stream Light Graph Convolution Network**-based latent factor analysis (TLGCN) model to perform highly accurate and efficient representation learning for QoS data with the following three-fold fundamental ideas:

- (1) Constructing a cloud service-oriented multilayered and fully-connected network to learn high level of services' nonlinear latent features;
- (2) Integrating the user-service interactions, i.e., the bipartite graph structure into the representation learning process with a light graph convolution network for illustrating high-order connectivity information;
- (3) Incorporating the data density-oriented modeling principle into the input and output of TLGCN for high computational efficiency.

To summarize, this paper achieves the following main contributions:

- (1) Proposing a TLGCN model with high representation learning ability for cloud service QoS data in both estimation accuracy and computational efficiency;
 - (2) Conducting extensive empirical studies on two commonly-adopted real QoS datasets to evaluate the TLGCN model.
- To the best of our knowledge, the proposed TLGCN model significantly outperforms its state-of-the-art peers in both estimation accuracy for missing cloud service QoS data and computational efficiency.

The remainder of this paper is organized as below. Section II presents the preliminaries. Section III describes the TLGCN model. Section IV gives the experimental results. In the end, Section V concludes this paper.

II. PRELIMINARIES

Table I summarizes the adopted notations of this paper. Note that a QoS matrix describing the relationship among user set and service set is the fundamental input for an LFA-based QoS estimator, which is defined as [9-13, 16-24]:

Definition 1. A QoS matrix. Given a user set U and a service set S , $Q^{|S| \times |U|}$ is a QoS matrix where each element $q_{s,u}$ describes a QoS record of invocation by $u \in U$ on $s \in S$.

Since a user usually invokes only a very limited number of candidate cloud services, Q is incomplete. Let Λ and O denote the known entry set and unknown one, an LFA-based QoS estimator is defined as [16-24]:

Definition 2. An LFA-based QoS estimator. Given Q , an LFA-based QoS estimator builds Q 's rank- F approximation \hat{Q} only based on Λ , generating an estimate $\hat{q}_{s,u}$ for specified $s \in S$ and $u \in U$ such that $\sum_{q_{s,u} \in \Lambda} (q_{s,u} - \hat{q}_{s,u})^2$ is minimized.

TABLE I. SYMBOL APPOINTMENT

Symbol	Description
U, S	Concerned user and service sets.
Q	An $ S \times U $ QoS matrix between U and S .
Λ, O	Known and unknown entry sets of Q .
F	Latent feature space dimension.
\hat{Q}	Q 's rank- F approximation.
$q_{s,u}, \hat{q}_{s,u}$	Single entries in Q and \hat{Q} .
K	Number of hidden layers of fully-connected network.
H^k	$ S \times D$ services' nonlinear latent feature matrix in the k -th layer.
$h_{s,f}^k$	Single entries in H^k .
W^k, B^k	Weight matrix and bias vector of k -th hidden layer.
$w_{s,f}^k, b_f^k$	Single variables in W^k and B^k .
A	$(U + S) \times (U + S)$ adjacency matrix of the user-service graph.
D	A 's degree matrix.
\tilde{A}	A 's symmetrically normalized matrix.
L	Number of propagation layers of light graph convolution.
E^l	$(U + S) \times D$ combined feature matrix of l -th convolution layer.
e_u^l, e_s^l	Latent features of u and s in the l -th graph convolution layer.
M, N	Resultant latent feature matrices of users and services.
$m_{s,f}, n_{u,f}$	Single entries in M and N .
C	Bias vector of the output layer.
c_u	Single variables in C .
$ \cdot $	Cardinality of the involved set.
$ak(\cdot)$	k -th layer activation function.
ε	Loss function.
$N(s), N(u)$	Neighborhood sets of s and u .
$\Lambda(s), \Lambda(u)$	Subsets of Λ related to each s and u in Q .
Ψ	Testing set from Λ .
α_l	Weight of l -th graph convolution layer features in E .
ω	Hyper parameter controlling the importance of E and H^k .

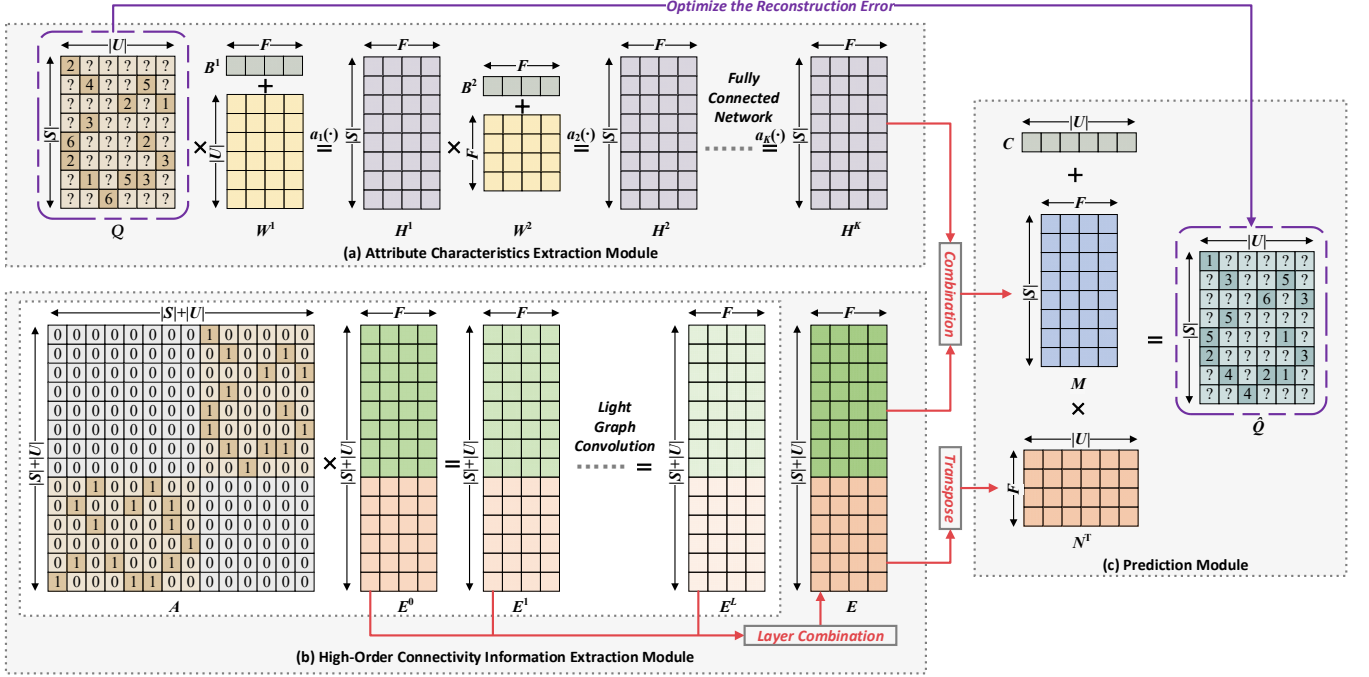


Fig. 2. The structure and flowchart of the proposed TLGCN model.

III. A TLGCN MODEL

In this section, we introduce the proposed TLGCN model, which can be divided into User-oriented TLGCN (U-TLGCN) and Service-oriented TLGCN (S-TLGCN). Note that they enjoy the same structure and the only difference between them is the former takes each user's records regarding all invoked cloud services as the input data, while the latter takes each service's invocation records as the input data. Hence, in this paper, we mainly present the Service-oriented TLGCN model for better readability and simplification, whose structure and flowchart are illustrated in Fig. 2.

As shown in Fig. 2, TLGCN consists of the following three main modules:

- The attribute characteristics extraction module receives the input QoS matrix and adopts a fully-connected network to acquire the nonlinear service representations;
- The high-order connectivity information extraction module iteratively performs light graph convolution to learn smooth representation for users and services on their interaction graph;
- The prediction module performs estimation, e.g., inner product, to obtain the unknown QoS data based on the resultant combined latent features of users and services.

TLGCN's detailed descriptions are as below.

A. Attribute Characteristics Extraction Module

We argue that by feeding the known invocation records in a user-service QoS matrix, the target LFA model can learn more accurate nonlinear representations for services. Hence, we design the attribute characteristics extraction module and incorporate the data density-oriented modeling mechanism into it to accommodate the sparsity of QoS data. Fig. 2(a) illustrates

the whole architecture of it and Fig. 3 depicts the propagation process of a specified service s , whose details are as below.

Input Layer. The input layer receives service-oriented QoS data, i.e., a $|S| \times |U|$ QoS matrix Q regarded as the attribute characteristics of services. As illustrated in Figs. 2(a) and 3, the known data are marked with real numbers denoting the user-service invocation records (e.g., throughput values) and the remaining unknown ones are marked with question marks. Since the real-world QoS data are mostly highly sparse, the prior methods filling the unknown parts with artificial values, e.g., zero values, are time-consuming and can cause accuracy loss. Hence, following the data density-oriented principle, TLGCN only activates the input layer nodes based on the known user-service interactions to achieve high efficiency.

Hidden Layer. The attribute characteristics extraction module achieves multiple fully-connected hidden layers to learn high level of nonlinear latent features. As shown in Fig. 2(a), H^k denotes the k -th layer latent feature matrix, W^k and B^k denote the corresponding weight matrix and bias vector in k -th layer. Note that the k -th layer node count F^k is set uniformly for simplicity, i.e., $F^1 = F^2 = \dots = F^K = F$. For the first layer, we present the mapping formula as:

$$h_{s,f}^1 = a_1 \left(\sum_{u \in \Lambda(s)} q_{s,u} w_{u,f}^1 + b_f^1 \right), \quad (1)$$

where $q_{s,u}$, $h_{s,f}^1$, $w_{u,f}^1$ and b_f^1 are the single entries in Q , H^1 , W^1 and B^1 , $\Lambda(s)$ represents the known entry subset related to s , and $a_1(\cdot)$ indicates the first-layer activation function, which can be hyperbolic tangent (tanh), sigmoid or Rectified Linear Unit (ReLU), among others. Note that since $|\Lambda(s)| \ll |U|$, (2) reduces the computational and storage cost greatly.

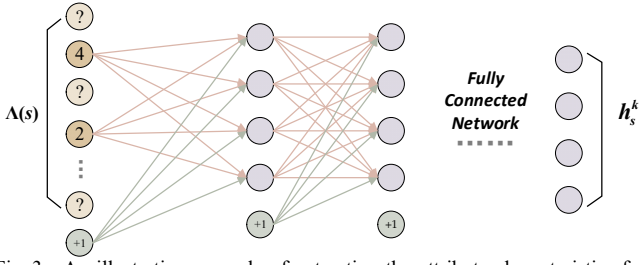


Fig. 3. An illustrative example of extracting the attribute characteristics for service s . In the first layer, all input nodes regarding unknown QoS data are not activated to obtain high computational efficiency.

For the $(2\sim K)$ -th hidden layers, the computing process is based on a fully-connected network, which is given as:

$$h_{s,f}^k = a_k \left(\sum_{d=1}^F h_{s,d}^{k-1} w_{d,f}^k + b_f^k \right), \quad (2)$$

where $h_{s,f}^k$, $h_{s,d}^{k-1}$, $w_{d,f}^k$ and b_f^k are the single entries in H^k , H^{k-1} , W^k and B^k respectively, and $a_k(\cdot)$ is the activation function. Different activation functions are tested in our implementation, and we find that uniformly adopting sigmoid for all layers plays the best effect.

B. High-Order Connectivity Information Extraction Module

As discussed in [28-32], to explicitly exploit the high-order connectivity information from user-service interaction graph can acquire stronger collaborative QoS signal thereby augmenting node representations. In recent years, graph convolution networks become popular owing to their high accuracy and scalability in capturing high-order connectivity information, which work by iteratively performing message passing to aggregate multi-hop neighborhood information. When addressing non-Euclidean QoS data, its complete message passing layer with self-connection is defined as:

$$E^{l+1} = \sigma(\hat{D}^{-0.5} \hat{A} \hat{D}^{-0.5} E^l W^l), \quad (3)$$

where $\hat{A} = A + I$ and $\hat{D} = D + I$. A is a $(|U|+|S|) \times (|U|+|S|)$ adjacency matrix directly constructed from the bipartite user-service interaction graph as:

$$A = \begin{pmatrix} 0 & Q_{adj} \\ Q_{adj}^T & 0 \end{pmatrix}. \quad (4)$$

where Q_{adj} is Q 's adjacency matrix. D is A 's diagonal degree matrix, in which each entry D_{ii} denotes the number of invocation records regarding each user or service, i.e., the nonzero entries in A 's i -th row vector. And I is the identity matrix used to integrate the self-connections on nodes. E^l and W^l denote the latent feature matrix and feature transformation weight matrix of the l -th layer. $\sigma(\cdot)$ is a nonlinear activation function such as ReLU.

Despite GCN's wide success in various graph learning fields, several recent studies [29-32] argue that by appropriately simplifying GCN, the performance on CF tasks can be further boosted. Inspired by this, we design the high-order connectivity information extraction module as depicted in Fig. 2(b) and adopt the following message passing strategy in each layer:

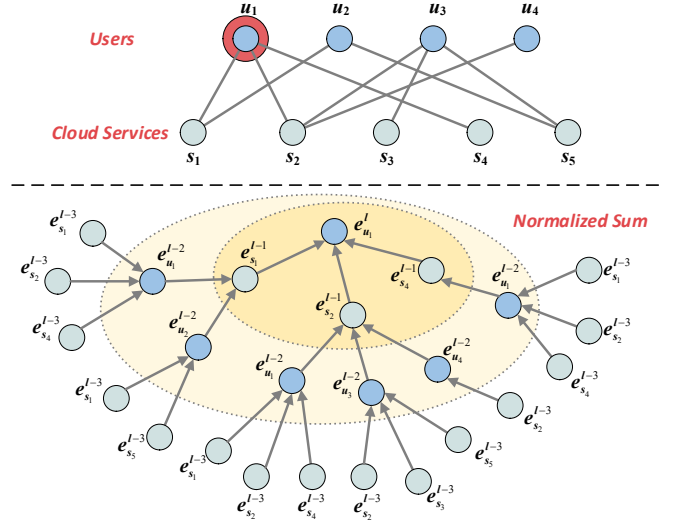


Fig. 4. An illustrative example of extracting the high-order connectivity information for user u_1 . In light graph convolution, the feature transformation, self-connection, and nonlinear activation are all removed, only the normalized sum of neighborhood latent features are aggregated towards next layer, whose simplified structure reduces training difficulty thereby achieving higher estimation accuracy and efficiency compared with the complete one.

$$E^{l+1} = D^{-0.5} A D^{-0.5} E^l. \quad (5)$$

Note that compared with (3), (5) removes: 1) the feature transformation, i.e., W^l ; 2) the nonlinear activation function, i.e., $\sigma(\cdot)$; and 3) the self-connection, i.e., I . In this way, for each specified user or service, the $(l+1)$ -th layer latent feature can be obtained by only aggregating the normalized sum of its neighborhood features. Specifically, in (5), the light graph convolution operations, i.e., the propagation rules for user u and service s are defined as:

$$\begin{aligned} e_u^{l+1} &= \sum_{s \in N(u)} \frac{1}{\sqrt{|N(u)|} \sqrt{|N(s)|}} e_s^l, \\ e_s^{l+1} &= \sum_{u \in N(s)} \frac{1}{\sqrt{|N(s)|} \sqrt{|N(u)|}} e_u^l. \end{aligned} \quad (6)$$

where $\frac{1}{\sqrt{|N(u)|} \sqrt{|N(s)|}}$ is the symmetric normalization form,

in which $N(u)$ and $N(s)$ denote the directly-connected neighbor node sets of u and s . It can avoid the feature scale increasing with multiple graph convolution operations and assign different importance to each neighbor for higher accuracy and diversity. Fig. 4 illustrates the detailed propagation process of user u_1 by performing (6). After propagating L layers, we can obtain the final representations by adopting the layer combination, i.e., the weighted sum of the latent features propagated at each layer as:

$$\begin{aligned} E &= \alpha_0 E^0 + \alpha_1 E^1 + \alpha_2 E^2 + \dots + \alpha_L E^L \\ &= \alpha_0 E^0 + \alpha_1 \tilde{A} E^0 + \alpha_2 \tilde{A}^2 E^0 + \dots + \alpha_L \tilde{A}^L E^0, \end{aligned} \quad (7)$$

where $\tilde{A} = D^{-0.5} A D^{-0.5}$ is A 's symmetrically normalized matrix; $\alpha_i \geq 0$ denotes the importance of l -th layer features to constitute

the final representations, we set it uniformly as $1/(L+1)$ here. Note that (7) plays a similar effect to self-connection, and it can alleviate the over-smoothing problem. In the whole propagation process, E^0 is the only parameter matrix to optimize, which is easy to train thereby gaining high efficiency and accuracy.

C. Prediction Module

For a QoS graph, the attribute characteristics and multi-hop neighborhood information greatly enrich each node's representation. Hence, by performing attribute characteristics extraction and light graph convolution, we address the data sparsity problem to some extent and obtain complementary latent features. After that, in the prediction module, the services' final latent features are combined as:

$$M = \omega E_{1-|S|} + (1 - \omega) H^K, \quad (8)$$

where M is the final service feature matrix, and ω denotes the importance of $E_{1-|S|}$ and H^K in constituting the final latent features, which is treated as a hyper parameter to be tuned carefully. And we extract N from E as the resultant user latent feature matrix, as shown in Fig. 2(c). Based on M and N , TLGCN estimates the unknown entries in Q as:

$$\hat{q}_{s,u} = \sum_{f=1}^F m_{s,f} n_{u,f} + c_u, \quad (9)$$

where $\hat{q}_{s,u} \in Q$ is the estimation for each $q_{s,u}$, and $c_u \in C$ is the bias for u to enlarge the solution space. And then as discussed in Section II, we utilize the commonly-adopted Euclidean distance-based objective function to optimize the variables as:

$$\varepsilon = \frac{1}{2} \sum_{q_{s,u} \in \Lambda} (q_{s,u} - \hat{q}_{s,u})^2, \quad (10)$$

where Λ denotes the known entry set. And like the input layer in the attribute characteristics extraction module, (10) follows the data density-oriented modeling principle to optimize the variables only on the known entries in Q . Note that the combination and estimation strategies can be further extended to boost the performance of TLGCN with different methods, e.g., an attention-based neural network.

D. CUDA-Parallelized Sparse Matrix Computation

To address the high sparsity of QoS data, TLGCN adopts the data-density oriented principle in its input and output parts, i.e., only based on the known entry set without any data filling, for avoiding time and storage consumption. However, existing mainstream deep learning frameworks are immature in sparse computing and they cannot absolutely support TLGCN's numerous sparse matrix operations in GPU. Hence, following the previous studies [33, 34], an efficient CUDA-parallelized sparse matrix computation module is specially implemented for GPU acceleration of TLGCN.

As illustrated in Fig. 5, two key schedulers, i.e., SM and warp, are adopted in CUDA programming to achieve two-level sparse matrix computation parallelism. Compressed Sparse Row (CSR) format is utilized to re-represent a sparse user-service QoS matrix, which is constituted by three arrays: a) ptr array storing the row pointers, which denote the position of each row's first entry; b) idx array storing the column

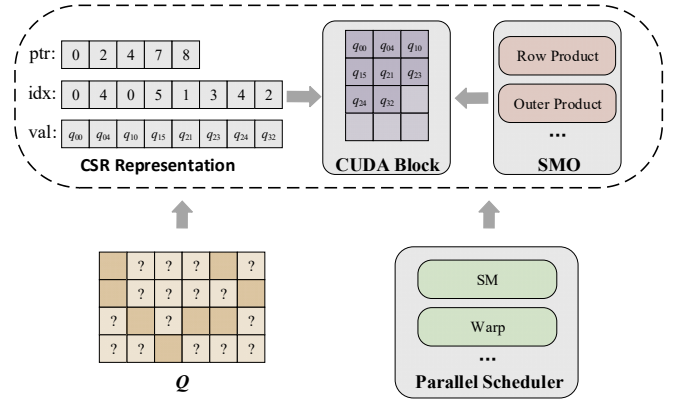


Fig. 5. CUDA-parallelized sparse matrix computation module.

indexes; and c) val array storing the values of the known data, i.e., the known invocation records in QoS data. Note that given a target QoS matrix Q , the size of ptr array is $|S|$, i.e., the number of rows or services of Q , while the size of idx and val arrays is $|\Lambda|$, i.e., the number of Q 's known entries. Based on such CSR format, we achieve efficient GPU computing by designing some sparse matrix operators (SMOs) [33, 34], e.g., the outer product one and the row product one.

Generally, the size of block significantly influences the performance of CUDA-parallelized computation. Since a warp consists of 32 threads, it is better to set the size of block as a multiple of 32. And owing to that GPU's various architectures has different numbers of SM, we follow the previous studies [33, 34] to set the block size to 32×32 in this paper.

IV. EXPERIMENTS AND RESULTS

We perform extensive experiments on two real-world QoS datasets to evaluate the proposed TLGCN model, and we aim at answering the following three research questions (RQs):

- **RQ1.** Is the multilayered design of attribute characteristics extraction and light graph convolution helpful for improving the estimation accuracy of a TLGCN model?
- **RQ2.** How do different hyper parameter settings, e.g., the latent feature combination weight ω and latent feature dimension F , affect the performance of a TLGCN model?
- **RQ3.** How does the proposed TLGCN model perform compared with state-of-the-art QoS estimators?

A. General Settings

Datasets. Two real cloud service QoS data¹ collected by the WS-Dream system are applied in our experiments, which are the largest publicly-available QoS datasets and widely adopted in prior studies for QoS estimation [15-28]. On both datasets, we carefully design multiple cases to validate each model's performance under different low densities, whose details are shown in Table II. Note that the column Train:Validation:Test denotes the ratio of training data, validation data and testing data, e.g., 1%:4%:95% indicates that 1% of Λ are chosen randomly as the training data, 4% of Λ are chosen randomly as the validation data, and the remaining 95% are chosen as the testing data.

¹https://wsdream.github.io/dataset/wsdream_dataset1.html

TABLE II. PROPERTIES OF TESTING CASES

Dataset	A	U	S	No.	Train:Validation:Test
Response Time (D1)	1,873,838	339	5,825	D1.1	1%:4%:95%
				D1.2	2%:8%:90%
				D1.3	3%:12%:85%
				D1.4	4%:16%:80%
Throughput (D2)	1,831,253	339	5,825	D2.1	1%:4%:95%
				D2.2	2%:8%:90%
				D2.3	3%:12%:85%
				D2.4	4%:16%:80%

Evaluation Metrics. For each tested model, this study mainly concerns its estimation accuracy for missing cloud service QoS data. Hence, we use two commonly-adopted evaluation metrics [16-24], i.e., the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE), to measure this. Generally, the RMSE and MAE are formulated as:

$$RMSE = \sqrt{\frac{\sum_{q_{s,u} \in \Psi} (q_{s,u} - \hat{q}_{s,u})^2}{|\Psi|}},$$

$$MAE = \left(\frac{\sum_{q_{s,u} \in \Psi} |q_{s,u} - \hat{q}_{s,u}|_{abs}}{|\Psi|} \right),$$

where Ψ denotes the testing dataset, $|\Psi|$ denotes the number of entries of it.

Note that to evaluate the computational efficiency of an involved model, the GPU training time on each testing case is carefully recorded. We implement all experiments in Python 3.7, except that the compressed sparse matrix parallel program is written with CUDA C and compiled with CUDA 10.1. All empirical tests are uniformly deployed on a server with a 2.4-GHz Intel Xeon 4214R CPU, four NVIDIA RTX 3090 GPUs, and 128-GB RAM.

Comparison models. Eight state-of-the-art QoS estimators are involved in our comparison, whose details are as below.

M1. Mult-VAE [35]. A generative model which builds a variational AutoEncoder and introduces a multinomial distribution into data to perform parameter estimation.

M2. NeuMF [17]. A widely adopted CF baseline which combines multilayered perception and inner product into matrix factorization to achieve nonlinear estimation.

M3. MetaMF [24]. A federated meta matrix factorization model for QoS data, which builds a federated learning framework to generate private embeddings of users.

M4. LR-GCCF [32]. A linear GCN-based model which removes nonlinearities to enhance the performance and empirically explains the layer concatenation operation.

M5. LightGCN [29]. A light GCN-based QoS estimator, which removes feature transformation and nonlinear activation to obtain efficient and accurate representation.

M6. DGCN-HN [36]. A deep GCN-based model, which uses hybrid normalization for flexible aggregation and adopts residual connection to address over-smoothing problem.

M7. HMLET [28]. A hybrid method of linear and non-linear CF, which is a GCN-based model and proposes a gating module to select each node's best propagation method.

M8. TLGCN. The model given in Section III.

Training Settings. The following settings are employed for each tested model.

- We initialize all the trained variables randomly with Xavier method, and optimize all involved models with Adam;
- For M1-3, we adopt the model architectures suggested in their original papers, and for M4-8, i.e., all GCN-based models, the latent feature dimension is fixed to 200;
- To achieve fair comparison, we carefully tune each model's hyper parameters to achieve the best performance;
- On each testing case, we repeat the splitting and training process for ten times and record the final average results;
- Each model's training process terminates if: a) the training epoch reaches a preset threshold, i.e., 1000; or b) its estimation accuracy keeps decreasing for 30 epochs.

TABLE III. THE RMSE AND MAE OF TLGCN WITH DIFFERENT NUMBER OF ATTRIBUTE CHARACTERISTICS EXTRACTION ON ALL TESTING CASES.

No.	1 Layer (K=1)		2 Layers (K=2)		3 Layers (K=3)		4 Layers (K=4)	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
D1.1	1.7595	0.6898	1.7240	0.6963	1.7129	0.6865	1.7321	0.6902
D1.2	1.5480	0.6471	1.5280	0.6091	1.5230	0.5989	1.5292	0.6035
D1.3	1.4847	0.6290	1.4512	0.5765	1.4455	0.5600	1.4505	0.5611
D1.4	1.4424	0.6078	1.4035	0.5455	1.3967	0.5336	1.4042	0.5307
D2.1	0.9014	0.3378	0.8728	0.3327	0.8660	0.3280	0.8643	0.3297
D2.2	0.7907	0.3033	0.7779	0.3011	0.7715	0.2944	0.7718	0.2981
D2.3	0.7074	0.2688	0.6539	0.2445	0.6894	0.2720	0.6880	0.2695
D2.4	0.6677	0.2473	0.6047	0.2248	0.6535	0.2594	0.6541	0.2587

TABLE IV. THE RMSE AND MAE OF TLGCN WITH DIFFERENT NUMBER OF LIGHT GRAPH CONVOLUTION ON ALL TESTING CASES.

No.	1 Layer (L=1)		2 Layers (L=2)		3 Layers (L=3)		4 Layers (L=4)		5 Layers (L=5)		6 Layers (L=6)		7 Layers (L=7)		8 Layers (L=8)	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
D1.1	1.7477	0.6939	1.7330	0.6875	1.7129	0.6865	1.7057	0.6836	1.7104	0.6780	1.7099	0.6794	1.6976	0.6760	1.6986	0.6755
D1.2	1.5344	0.6079	1.5250	0.6045	1.5230	0.5989	1.5197	0.5984	1.5131	0.5984	1.5119	0.5959	1.5076	0.5959	1.5063	0.5972
D1.3	1.4533	0.5667	1.4490	0.5674	1.4455	0.5601	1.4454	0.5564	1.4441	0.5603	1.4435	0.5567	1.4434	0.5583	1.4427	0.5657
D1.4	1.4007	0.5407	1.3965	0.5373	1.3967	0.5337	1.3962	0.5312	1.4019	0.5345	1.3995	0.5307	1.3990	0.5324	1.3962	0.5328
D2.1	0.8724	0.3312	0.8667	0.3284	0.8660	0.3280	0.8613	0.3267	0.8564	0.3258	0.8543	0.3257	0.8528	0.3261	0.8508	0.3260
D2.2	0.7807	0.2995	0.7781	0.2984	0.7715	0.2944	0.7668	0.2931	0.7598	0.2916	0.7558	0.2901	0.7504	0.2890	0.7524	0.2895
D2.3	0.7043	0.2793	0.6962	0.2753	0.6894	0.2720	0.6826	0.2701	0.6748	0.2684	0.6718	0.2675	0.6695	0.2670	0.6665	0.2664
D2.4	0.6620	0.2618	0.6565	0.2603	0.6535	0.2594	0.6509	0.2585	0.6483	0.2570	0.6460	0.2564	0.6448	0.2572	0.6430	0.2572

B. Effect of Multilayered Structure (RQ1)

To investigate whether TLGCN can benefit from the multilayered structure of attribute characteristics extraction and light graph convolution, we vary the depth of two modules, i.e., we search K in $\{1, 2, 3, 4\}$ and L in $\{1, 2, 3, 4, 5, 6, 7, 8\}$. The experimental results are shown in Tables III and IV. Jointly analyzing them, we have the following findings:

a) **Increasing the depth of the attribute characteristics extraction module significantly enhances TLGCN's estimation accuracy for missing QoS data.** For instance, as recorded in Table III, by fixing other hyper parameters on D1.3, as K varies from 1 to 3, TLGCN's RMSE decreases from 1.4847 to 1.4455, which achieves the estimation error gap (i.e., $(Error_{high} - Error_{low})/Error_{high}$) at 2.65%. As far the MAE, it decreases from 0.6290 to 0.5600 with the estimation error gap of 10.97%. Similar situations can be found on other testing cases. In most cases, TLGCN

can achieve the best performance as $K=2$ or 3, and stacking more layers leads to its overfitting.

b) **Capturing the high-order connectivity information from user-service interaction graph substantially enhances the collaborative QoS signal.** As summarized in Table IV, on all eight testing cases, the estimation errors decrease greatly with the increase of L . For instance, on D2.1, as $L=1$ while others being fixed, TLGCN achieves the RMSE and MAE at 0.8724 and 0.3312. However, as L increases to 8, the values of them both decrease to 0.8508 and 0.3260, i.e., the estimation error gap in RMSE and MAE are at 2.48% and 1.57%. Note that on most cases, TLGCN achieves the highest accuracy as $L=8$, whose great improvements can be attributed to the abundance of high-order connectivity information and layer combination addressing the over-smoothing problem. Hence, it is vital to explicitly exploit the collaborative signal from user-service interactions.

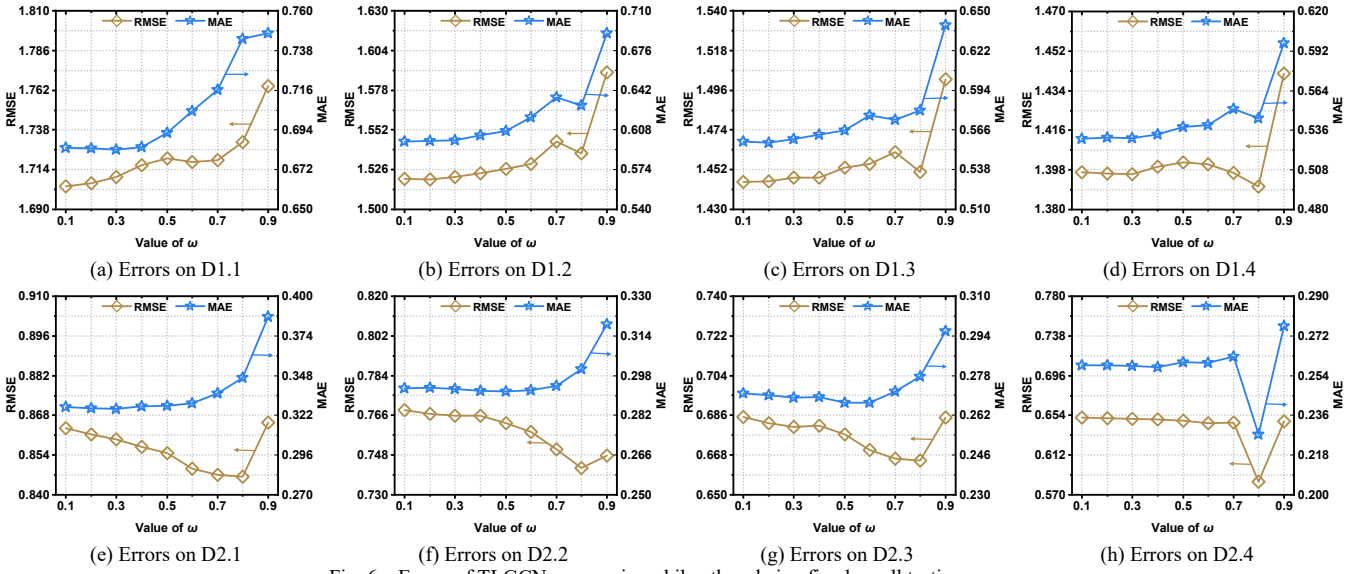


Fig. 6. Errors of TLGCN as ω varies while others being fixed on all testing cases.

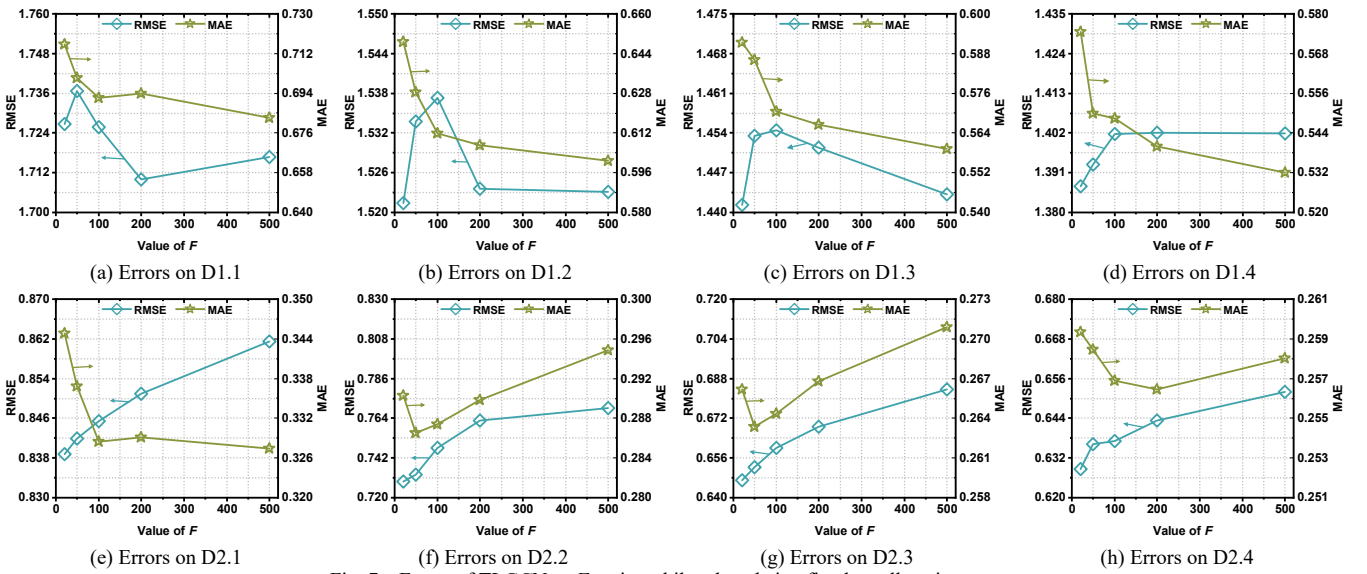


Fig. 7. Errors of TLGCN as F varies while others being fixed on all testing cases.

TABLE V. THE RMSE, WIN/LOSS COUNTS AND FRIEDMAN TEST RESULTS OF TLGCN ON ALL TESTING CASES.

No.	M1	M2	M3	M4	M5	M6	M7	M8
D1.1	1.9396±7.7E-4•	1.9718±4.0E-3•	1.9229±2.4E-2•	1.8621±9.2E-5•	1.8147±1.4E-3•	2.0220±2.6E-3•	1.8685±1.2E-2•	1.6990±5.0E-3
D1.2	1.9141±5.0E-4•	1.7398±7.0E-3•	1.8414±3.1E-2•	1.8443±8.6E-4•	1.6361±8.5E-4•	1.8406±1.6E-3•	1.6499±3.8E-3•	1.5210±4.9E-3
D1.3	1.8973±9.1E-4•	1.6148±1.6E-3•	1.7820±2.6E-2•	1.7536±8.2E-2•	1.5848±2.8E-4•	1.7278±1.9E-3•	1.5771±3.0E-3•	1.4503±1.7E-3
D1.4	1.8839±3.3E-4•	1.5318±4.2E-3•	1.7446±1.4E-2•	1.5449±1.2E-2•	1.5514±2.0E-4•	1.6525±1.3E-3•	1.5169±3.2E-3•	1.4016±6.3E-4
D2.1	1.1404±1.1E-3•	1.0480±1.2E-2•	1.0827±8.4E-3•	1.0971±3.8E-4•	0.9775±2.8E-3•	1.1756±1.2E-3•	1.0578±1.3E-2•	0.8501±9.1E-4
D2.2	1.1266±6.5E-4•	0.7710±4.8E-3•	0.9995±3.3E-2•	1.0310±1.3E-2•	0.7769±1.4E-3•	1.0756±1.4E-3•	0.8061±1.8E-2•	0.7422±2.3E-3
D2.3	1.1171±1.3E-3•	0.6727±6.0E-3•	0.9334±1.2E-2•	0.8805±1.5E-2•	0.6674±6.5E-4•	0.9548±2.9E-3•	0.6825±5.3E-3•	0.6125±2.2E-3
D2.4	1.0992±7.6E-4•	0.6696±5.6E-2•	0.8760±7.7E-3•	0.7467±7.8E-3•	0.5982±1.4E-3•	0.8566±9.3E-4•	0.6411±5.2E-3•	0.5718±2.4E-3
Win/Loss	8/0	8/0	8/0	8/0	8/0	8/0	8/0	—
F-Rank*	7.63	3.75	6.00	5.25	2.63	6.50	3.25	1.00

*A lower F-rank value denotes a higher estimation accuracy for missing QoS data; and • indicates that M8's RMSE is lower than its peers.

TABLE VI. THE MAE, WIN/LOSS COUNTS AND FRIEDMAN TEST RESULTS OF TLGCN ON ALL TESTING CASES.

No.	M1	M2	M3	M4	M5	M6	M7	M8
D1.1	0.9399±1.4E-3•	0.7936±2.6E-3•	0.8768±1.8E-2•	0.8899±8.0E-3•	0.7771±8.5E-4•	0.7590±5.2E-4•	0.7571±1.1E-3•	0.6872±1.4E-3
D1.2	0.9108±5.6E-3•	0.7025±5.3E-3•	0.8814±1.3E-2•	0.8782±8.0E-3•	0.7162±3.1E-3•	0.7452±6.6E-4•	0.7043±1.6E-3•	0.6041±2.8E-3
D1.3	0.8925±2.4E-3•	0.6413±1.2E-3•	0.8635±1.3E-2•	0.8521±1.4E-2•	0.6845±4.3E-4•	0.7223±4.3E-4•	0.6651±2.2E-3•	0.5632±2.0E-3
D1.4	0.8562±1.9E-3•	0.6100±2.3E-3•	0.8264±1.3E-2•	0.7498±1.1E-2•	0.6542±6.1E-4•	0.7019±5.9E-4•	0.6288±3.3E-3•	0.5380±1.7E-3
D2.1	0.5136±5.1E-4•	0.3934±5.0E-3•	0.5472±1.5E-2•	0.5549±9.2E-3•	0.3875±8.0E-4•	0.4380±1.1E-3•	0.3983±2.8E-3•	0.3278±9.8E-4
D2.2	0.4838±7.1E-4•	0.3035±2.7E-3•	0.5267±2.3E-2•	0.5354±6.8E-3•	0.3147±1.7E-3•	0.3942±3.1E-4•	0.3238±6.9E-3•	0.2864±1.1E-3
D2.3	0.4767±4.8E-4•	0.2302±6.4E-4	0.4812±2.0E-2•	0.4889±1.3E-2•	0.2647±9.0E-4•	0.3660±6.2E-4•	0.2840±2.9E-3•	0.2303±1.0E-3
D2.4	0.4775±1.4E-3•	0.2160±7.4E-4•	0.4283±1.7E-2•	0.3889±6.6E-3•	0.2352±8.0E-4•	0.3351±8.4E-4•	0.2674±2.6E-3•	0.2121±8.4E-4
Win/Loss	8/0	7/1	8/0	8/0	8/0	8/0	8/0	—
F-Rank*	7.25	2.38	6.88	6.88	3.38	4.75	3.38	1.13

*A lower F-rank value denotes a higher estimation accuracy for missing QoS data; and • indicates that M8's RMSE is lower than its peers.

TABLE VII. THE TIME COST TO CONVERGE IN RMSE (SEC.), WIN/LOSS COUNTS AND FRIEDMAN TEST RESULTS OF TLGCN ON ALL TESTING CASES.

No.	M1	M2	M3	M4	M5	M6	M7	M8
D1.1	20±0.89•	2586±116.63•	324±46.59•	267±28.61•	207±7.62•	263±6.16•	990±215.41•	5±1.17
D1.2	20±1.04•	1623±307.52•	206±47.65•	187±11.42•	274±14.13•	370±18.87•	1377±157.37•	6±2.03
D1.3	19±1.69•	2872±433.92•	172±33.05•	687±384.08•	261±12.60•	541±10.72•	1755±108.29•	7±2.26
D1.4	20±2.50	2391±482.72•	128±29.60•	1028±42.94•	232±10.53•	645±23.18•	1915±113.27•	21±7.80
D2.1	48±47.72•	3178±456.85•	306±9.68•	314±30.50•	252±17.90•	229±5.14•	1208±251.00•	7±0.46
D2.2	42±1.94•	2627±450.25•	243±43.49•	821±63.68•	355±17.80•	346±14.48•	1196±96.74•	5±0.26
D2.3	45±2.72•	2691±233.52•	175±21.71•	796±40.22•	452±28.96•	549±15.20•	2281±88.89•	38±4.57
D2.4	45±3.65•	1841±337.03•	127±13.27•	771±33.35•	495±17.40•	740±16.64•	2421±220.69•	30±1.46
Win/Loss	7/1	8/0	8/0	8/0	8/0	8/0	8/0	—
F-Rank*	1.88	7.88	3.75	5.50	4.13	4.63	7.13	1.13

*A lower F-rank value denotes a higher efficiency when addressing a QoS matrix; and • denotes that M8's converging time cost is less than its peers.

TABLE VIII. THE TIME COST TO CONVERGE IN MAE (SEC.), WIN/LOSS COUNTS AND FRIEDMAN TEST RESULTS OF TLGCN ON ALL TESTING CASES.

No.	M1	M2	M3	M4	M5	M6	M7	M8
D1.1	25±2.02•	2552±61.90•	80±18.40•	236±44.67•	247±12.38•	157±13.53•	1205±229.80•	17±3.82
D1.2	33±9.81•	1544±318.33•	82±69.83•	170±13.89•	418±33.15•	444±14.30•	1791±167.32•	13±4.98
D1.3	40±2.70•	2851±426.92•	137±93.57•	300±269.10•	410±12.41•	604±12.57•	2037±105.61•	30±9.28
D1.4	41±1.98•	2470±376.01•	153±34.35•	850±30.21•	438±13.84•	693±19.62•	2112±123.96•	25±12.28
D2.1	55±54.89•	3102±368.68•	98±32.95•	259±81.15•	309±24.14•	180±2.30•	991±182.41•	15±5.12
D2.2	52±2.73•	2632±457.16•	192±102.37•	276±220.43•	445±20.52•	329±8.12•	974±89.25•	8±0.67
D2.3	50±2.87•	2655±228.16•	189±34.68•	655±44.81•	442±24.25•	534±7.79•	1856±65.92•	33±2.78
D2.4	45±2.36•	1971±320.04•	155±14.60•	642±32.04•	449±26.37•	712±3.48•	2044±152.14•	28±2.32
Win/Loss	8/0	8/0	8/0	8/0	8/0	8/0	8/0	—
F-Rank*	2.00	7.75	3.00	4.88	5.00	5.13	7.25	1.00

*A lower F-rank value denotes a higher efficiency when addressing a QoS matrix; and • denotes that M8's converging time cost is less than its peers.

TABLE IX. RESULTS OF THE WILCOXON SIGNED-RANKS TEST IN RMSE.

Comparison	R+*	R-	p-value**
M8 vs M1	36	0	0.0039
M8 vs M2	36	0	0.0039
M8 vs M3	36	0	0.0039
M8 vs M4	36	0	0.0039
M8 vs M5	36	0	0.0039
M8 vs M6	36	0	0.0039
M8 vs M7	36	0	0.0039

*A higher R+ value denotes M8 has a higher accuracy than its peers;

**The accepted hypotheses are highlighted with the significance level of 0.1.

TABLE X. RESULTS OF THE WILCOXON SIGNED-RANKS TEST IN MAE.

Comparison	R+*	R-	p-value**
M8 vs M1	36	0	0.0039
M8 vs M2	35	1	0.0078
M8 vs M3	36	0	0.0039
M8 vs M4	36	0	0.0039
M8 vs M5	36	0	0.0039
M8 vs M6	36	0	0.0039
M8 vs M7	36	0	0.0039

*A higher R+ value denotes M8 has a higher accuracy than its peers;

**The accepted hypotheses are highlighted with the significance level of 0.1.

TABLE XI. RESULTS OF WILCOXON SIGNED-RANKS TEST ON THE CONVERGING TIME COST IN RMSE.

Comparison	R^+ *	R^-	p -value**
M8 vs M1	35	1	0.0078
M8 vs M2	36	0	0.0039
M8 vs M3	36	0	0.0039
M8 vs M4	36	0	0.0039
M8 vs M5	36	0	0.0039
M8 vs M6	36	0	0.0039
M8 vs M7	36	0	0.0039

*A higher R^+ value denotes M8 has a higher efficiency than its peers;

**The accepted hypotheses are highlighted with the significance level of 0.1.

C. Analysis of Hyper Parameter Sensitivity (RQ2)

The performance of a deep learning-based model can be affected by many hyper parameters, e.g., the batch size, initial learning rate, and regularization coefficient. For TLGCN: the batch size is fixed at 512; the learning rate is searched amongst $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05\}$; and the coefficient of L_2 regularization is tuned in $\{10^{-5}, 10^{-4}, \dots, 10^1, 10^2\}$. Considering two unique hyper parameters of TLGCN, i.e., the latent feature combination weight ω and latent feature dimension F . Figs. 6 and 7 depict the influence of them while others being fixed. From them, we have the following findings:

a) **ω significantly affects TLGCN's RMSE/MAE.** For instance, as depicted in Fig. 6(b), on D1.2, as ω varies from 0.1 to 0.9 while others being fixed, the RMSE increases from 1.5200 to 1.5895 with the RMSE increment of 4.37%. Considering the MAE, it increases from 0.5983 to 0.6910 with the MAE increment of 13.42%. Similar situations are encountered in other cases. Note that ω is data-dependent, as shown in Fig. 6, on D1.1-1.3, TLGCN achieves the lower RMSE as $\omega=0.1$, while on D1.4-2.4, the lower RMSE is achieved as $\omega=0.8$. Hence, for different datasets, ω should be tuned carefully to obtain a satisfactory performance.

b) **F also greatly influences TLGCN's estimation accuracy, and it performs differently in RMSE and MAE.** For instance, as illustrated in Fig. 7(a), on D1.1, by fixing other hyper parameters, as $F=50$, TLGCN achieves the RMSE at 1.7366. However, as F is set to 200, the RMSE decreases to 1.7099 with the estimation error gap of 1.54%. Note that F 's effects vary greatly in RMSE and MAE, for instance, as shown in Fig. 7(e), on D2.1, as F increases from 20 to 500, the RMSE increases from 0.8388 to 0.8615 which achieves the RMSE increment at 2.63%. However, the MAE decreases from 0.3448 to 0.3274 with the estimation error gap of 5.05%. We attribute this phenomenon to that the testing cases are still not big enough, and we will test TLGCN's performance on larger-scale datasets to deeply understand it in our future work.

D. Comparison with State-of-the-Art Models (RQ3)

We perform detailed comparisons with seven state-of-the-art QoS estimators. Tables V-VIII present the RMSE, MAE, GPU time cost to converge in RMSE and MAE of M1-8 on all testing cases. To deeply understand the experimental results, in Table V-VIII, the second-to-last rows record the win/loss counts of M8 versus other models; the last rows summarize the Friedman statistical results of all tested models [37]. Note that as significance level=0.05, the hypothesis that all involved models are significantly different is accepted. And the results of Wilcoxon signed-ranks test are recorded in Table IX-XII

TABLE XII. RESULTS OF WILCOXON SIGNED-RANKS TEST ON THE CONVERGING TIME COST IN MAE.

Comparison	R^+ *	R^-	p -value**
M8 vs M1	36	0	0.0039
M8 vs M2	36	0	0.0039
M8 vs M3	36	0	0.0039
M8 vs M4	36	0	0.0039
M8 vs M5	36	0	0.0039
M8 vs M6	36	0	0.0039
M8 vs M7	36	0	0.0039

*A higher R^+ value denotes M8 has a higher efficiency than its peers;

**The accepted hypotheses are highlighted with the significance level of 0.1.

[37], which is helpful to check whether M8 possesses higher estimation accuracy for missing QoS data and computational efficiency than its peers. From these summarizations, we achieve the following findings:

- M8, i.e., TLGCN significantly outperforms its peers in estimation accuracy for missing QoS data.** Owing to its full modeling for the attribute characteristics and high-order connectivity information, on all testing cases, M8 achieves satisfactory performance gain. For instance, as recorded in Table V, on D2.1, M8 achieves the RMSE at 0.8501, which is about 25.46% lower than M1's 1.1404, 18.88% lower than M2's 1.0480, 21.48% lower than M3's 1.0827, 22.51% lower than M4's 1.0971, 13.03% lower than M5's 0.9775, 27.69% lower than M6's 1.1756, and 19.64% lower than M7's 1.0578. Similar situations in RMSE and MAE are also encountered on other testing cases. More persuasively, M8 also has the lowest F-rank value and supported Wilcoxon signed-ranks test results, e.g., as shown in Table V, in RMSE, M8's F-rank value is 1.00, which is lower than 7.63, 3.75, 6.00, 5.25, 2.63, 6.50, and 3.25 achieved by M1-7.
- M8, i.e., TLGCN has substantially higher computational efficiency than that of its peers when addressing a QoS matrix.** As recorded in Tables VII-VIII and XI-XII, since it adopts data density-oriented principle and light graph convolution, M8's time cost in RMSE/MAE is far less than its peers. For instance, as shown in Table VII, on D1.1, in RMSE, M8 takes 5 seconds to converge, which is 25.00% of 20 seconds by M1, 0.19% of 2586 seconds by M2, 1.54% of 324 seconds by M3, 1.87% of 267 seconds by M4, 2.42% of 207 seconds by M5, 1.90% of 263 seconds by M6, and 0.51% of 990 seconds by M7; MAE is similar.

E. Summary

We summarize that TLGCN achieves important virtues:

- High estimation accuracy for missing user-service QoS data;
- Highly competitive computational efficiency.

V. CONCLUSION

Aiming at performing efficient and accurate representation learning for QoS data, this paper proposes a TLGCN model. It constructs a multilayered fully-connected network to extract services' attribute characteristics; uses light graph convolution to learn high-order connectivity information from user-service interactions; and incorporates data density-oriented principle to achieve high computational efficiency. Experimental results on two real QoS data demonstrate that TLGCN significantly outperforms the state-of-the-arts. Following the recent researches [38], we plan to boost the strengths of its graph convolution on larger-scale QoS data in our future work.

REFERENCES

- [1] Y. Yang, Z. Li, J. Zhang, and Y. Chen, "Transfer learning for web services classification," in *Proc. of 2021 IEEE Int. Conf. on Web Services*, Chicago, USA, 2021, pp. 185-191.
- [2] D. Wu, Q. He, X. Luo, M. Shang, Y. He, and G. Wang, "A posterior-neighborhood-regularized latent factor model for highly accurate web service QoS prediction," *IEEE Trans. on Services Computing*, DOI: 10.1109/TSC.2019.2961895, 2019.
- [3] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A study on big knowledge and its engineering issues," *IEEE Trans. on Knowledge and Data Engineering*, vol. 31, no. 9, pp. 1630-1644, 2019.
- [4] S. Li, H. Luo, and G. Zhao, "Bi-HPTM: an effective semantic matchmaking model for web service discovery," in *Proc. of 2020 IEEE Int. Conf. on Web Services*, Beijing, China, 2020, pp. 433-440.
- [5] S. Badsha, X. Yi, L. Khalil, D. Liu, S. Nepal, E. Bertino, and K. Y. Lam, "Privacy preserving location-aware personalized web service recommendations," *IEEE Trans. on Services Computing*, vol. 14, no. 3, pp. 791-804, 2021.
- [6] Y. Xiao, G. Kang, J. Liu, B. Cao, and L. Ding, "WSGCN4SLP: weighted signed graph convolutional network for service link prediction," in *Proc. of 2021 IEEE Int. Conf. on Web Services*, Chicago, USA, 2021, pp. 135-144.
- [7] X. Luo, M. Zhou, S. Li, Y. Xia, Z. You, Q. Zhu, and H. Leung, "Incorporation of efficient second-order solvers into latent factor models for accurate prediction of missing QoS data," *IEEE Trans. on Cybernetics*, vol. 48, no. 4, pp. 1216-1228, 2018.
- [8] G. N. Rai, and G. R. Gangadharan, "Model checking based web service verification: a systematic literature review," *IEEE Trans. on Services Computing*, vol. 14, no. 3, pp. 747-764, 2021.
- [9] P. Zhang, H. Jin, H. Dong, W. Song, and L. Wang, "LA-LMRBF: online and long-term web service qos forecasting," *IEEE Trans. on Services Computing*, vol. 14, no. 6, pp. 1809-1823, 2021.
- [10] L. Ding, G. Kang, J. Liu, Y. Xiao, and B. Cao, "QoS prediction for web services via combining multi-component graph convolutional collaborative filtering and deep factorization machine," in *Proc. of 2021 IEEE Int. Conf. on Web Services*, Chicago, USA, 2021, pp. 551-559.
- [11] M. Karakus, E. Guler, and S. Uludag, "QoSChain: provisioning inter-as qos in software-defined networks with blockchain," *IEEE Trans. on Network and Service Management*, vol. 18, no. 2, pp. 1706-1717, 2021.
- [12] J. Liu, and Y. Chen, "HAP: a hybrid QoS prediction approach in cloud manufacturing combining local collaborative filtering and global case-based reasoning," *IEEE Trans. on Services Computing*, vol. 14, no. 6, pp. 1796-1808, 2021.
- [13] H. Wu, and X. Luo, "Instance-frequency-weighted regularized, nonnegative and adaptive latent factorization of tensors for dynamic QoS analysis," in *Proc. of 2021 IEEE Int. Conf. on Web Services*, Chicago, USA, 2021, pp. 560-568.
- [14] H. Sun, C. Peng, D. Yue, Y. L. Wang, and T. Zhang, "Resilient load frequency control of cyber-physical power systems under QoS-dependent event-triggered communication," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 4, pp. 2113-2122, 2020.
- [15] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE Trans. on Services Computing*, vol. 7, no. 1, pp. 32-39, 2014.
- [16] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A. C. Ammari, and A. Alabdulwahab, "Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 27, no. 3, pp. 524-537, 2016.
- [17] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. of the 26th Int. Conf. on World Wide Web*, Perth, Australia, 2017, pp. 173-182.
- [18] Z. Zheng, L. Xiaoli, M. Tang, F. Xie, and M. R. Lyu, "Web service QoS prediction via collaborative filtering: a survey," *IEEE Trans. on Services Computing*, DOI: 10.1109/TSC.2020.2995571, 2020.
- [19] D. Ryu, K. Lee, and J. Baik, "Location-based web service QoS prediction via preference propagation to address cold start problem," *IEEE Trans. on Services Computing*, vol. 14, no. 3, pp. 736-746, 2021.
- [20] X. Zhu, X. Jing, D. Wu, Z. He, J. Cao, D. Yue, and L. Wang, "Similarity-maintaining privacy preservation and location-aware low-rank matrix factorization for QoS prediction based web service recommendation," *IEEE Trans. on Services Computing*, vol. 14, no. 3, pp. 889-902, 2021.
- [21] X. Luo, H. Wu, Z. Wang, J. Wang, and D. Meng, "A novel approach to large-scale dynamically weighted directed network representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, DOI: 10.1109/TPAMI.2021.3132503, 2021.
- [22] Y. Lin, P. Ren, Z. Chen, Z. Ren, D. Yu, Jun Ma, M. Rijke, and X. Cheng, "Meta matrix factorization for federated rating predictions," in *Proc. of the 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Xi'an, China, 2020, pp. 981-990.
- [23] D. Ryu, K. Lee, and J. Baik, "Location-based web service QoS prediction via preference propagation to address cold start problem," *IEEE Trans. on Services Computing*, vol. 14, no. 3, pp. 736-746, 2021.
- [24] Y. Zhang, P. Zhang, Y. Luo, and J. Luo, "Efficient and privacy-preserving federated QoS prediction for cloud services," in *Proc. of 2020 IEEE Int. Conf. on Web Services*, Beijing, China, 2020, pp. 549-553.
- [25] X. Luo, M. Zhou, Z. Wang, Y. Xia, and Q. Zhu, "An effective scheme for QoS estimation via alternating direction method-based matrix factorization," *IEEE Trans. on Services Computing*, vol. 12, no. 4, pp. 503-518, 2019.
- [26] Y. Yang, Z. Zheng, X. Niu, M. Tang, Y. Lu, and X. Liao, "A location-based factorization machine model for web service qos prediction," *IEEE Trans. on Services Computing*, vol. 14, no. 5, pp. 1264-1277, 2021.
- [27] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and X. Wu, "A data-characteristic-aware latent factor model for web services QoS prediction," *IEEE Trans. on Knowledge and Data Engineering*, DOI: 10.1109/TKDE.2020.3014302, 2020.
- [28] T. Kong, T. Kim, J. Jeon, J. Choi, Y.-C. Lee, N. Park, and S.-W. Kim, "Linear, or non-linear, that is the question," in *Proc. of the 15th ACM Int. Conf. on Web Search and Data Mining*, Tempe, USA, 2022, pp. 517-525.
- [29] X. He, K. Deng, X. Wang, Y. li, Y. Zhang, and M. Wang, "Lightgcn: simplifying and powering graph convolution network for recommendation," in *Proc. of the 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Xi'an, China, 2020, pp. 639-648.
- [30] K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, and X. He, "UltraGCN: ultra simplification of graph convolutional networks for recommendation," in *Proc. of the 30th ACM Int. Conf. on Information and Knowledge Management*, Queensland, Australia, 2021, pp. 1253-1262.
- [31] F. Wu, A. H. S. Júnior, T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proc. of the 36th Int. Conf. on Machine Learning*, Long Beach, USA, 2019, pp. 6861-6871.
- [32] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "revisiting graph based collaborative filtering: a linear residual graph convolutional network approach," in *Proc. of the 34th AAAI Conference on Artificial Intelligence*, New York, USA, 2020.
- [33] J. Lee, S. Kang, Y. Yu, Y.-Y. Jo, S.-W. Kim, and Y. Park, "Optimization of GPU-based sparse matrix multiplication for large sparse networks," in *Proc. of the 36th IEEE Int. Conf. on Data Engineering*, Dallas, USA, 2020, pp. 925-936.
- [34] S. Pal, J. Beaumont, D.-H. Park, A. Amarnath, S. Feng, C. Chakrabarti, H.-S. Kim, D. Blaauw, T. N. Mudge, and R. G. Dreslinski, "Outerspace: an outer product based sparse matrix multiplication accelerator," in *Proc. of the 24th IEEE Int. Symposium on High Performance Computer Architecture*, Vienna, Austria, 2018, pp. 724-736.
- [35] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proc. of the 27th Int. Conf. on World Wide Web*, Lyon, France, 2018, pp. 689-698.
- [36] W. Guo, Y. Yang, Y. Hu, C. Wang, H. Guo, Y. Zhang, R. Tang, W. Zhang, and X. He, "Deep graph convolutional networks with hybrid normalization for accurate and diverse recommendation," in *Proc. of 3rd Workshop on Deep Learning Practice for High-Dimensional Sparse Data with KDD*, Virtual Event, China, 2021.
- [37] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1-30, 2006.
- [38] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: a survey," *IEEE Trans. on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 249-270, 2022.