Complex & Intelligent Systems (2025) 11:191
https://doi.org/10.1007/s40747-025-01824-w

ORIGINAL ARTICLE

Trust-aware privacy-preserving QoS prediction with graph neural collaborative filtering for internet of things services

Weiwei Wang[1] · **Wenping Ma**[1] · **Kun Yan**[1]

Received: 13 August 2024 / Accepted: 8 February 2025 / Published online: 28 February 2025

Abstract
The booming development of the Internet of Things (IoT) has led to an explosion of web services, making it more inconvenient for users to choose satisfactory services among numerous options. Therefore, ensuring quality of service (QoS) in a service-oriented IoT environment is crucial, highlighting QoS prediction as a prominent research focus. However, issues related to information credibility, user data privacy, and prediction accuracy in QoS prediction for IoT services have become significant challenges in current research. To tackle these issues, we propose TPP-GNCF, a trust-aware privacy-preserving QoS prediction framework that integrates graph neural networks with collaborative filtering methods. In TPP-GNCF, we filter out untrustworthy QoS values provided by users for certain services to select credible QoS values. Then, a message-passing graph neural network (MP-GNN) is utilized to effectively capture information transmission and relationships in the graph structure, while differential privacy is used to protect user node information. In addition, we use a similarity calculation method based on weight function in collaborative filtering to mine implicit embedded features that graph neural networks cannot directly utilize. Finally, the final missing QoS values are achieved by fusing graph neural predicted QoS and feature collaborative filtering predicted QoS. We conducted extensive experiments on the well-known WS-DREAM dataset. The results demonstrate that the TPP-GNCF framework not only surpasses existing schemes in performance but also effectively addresses issues of information credibility and user privacy.

Keywords QoS prediction · Gaussian distribution · GNN · Privacy protection · IoT services

Introduction

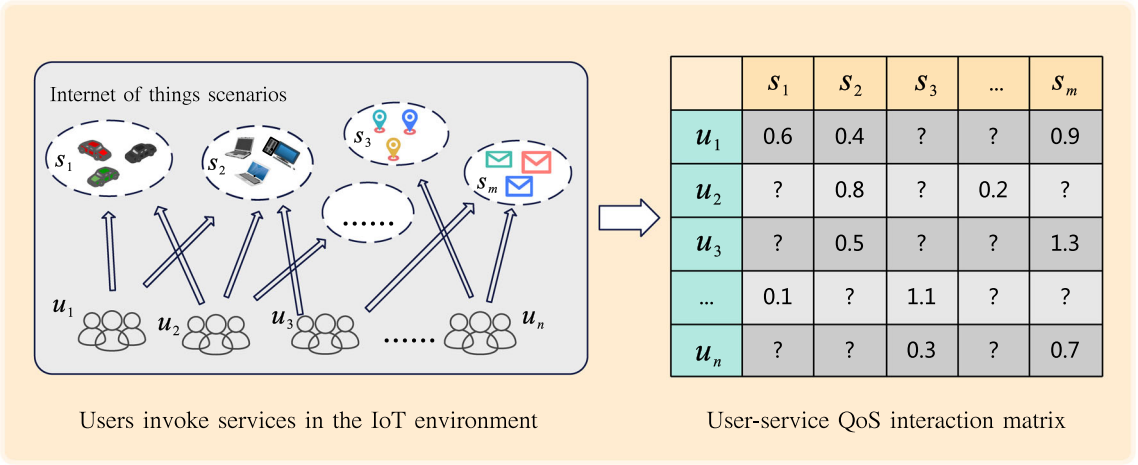
With the rapid expansion of IoT, network complexity and user needs have shown unprecedented growth, ensuring the quality of network services has become a crucial task [1–3]. Recent academic studies have emerged revealing the critical role of QoS prediction in network management. This trend reflects not only the importance attached by industry and academia to QoS prediction but also highlights its indispensability in meeting increasingly complex network requirements. QoS prediction becomes critical in this context, as it is an essential component of network services. The objective of QoS prediction is to accurately forecast web service performance indicators, such as delay, bandwidth, and reliability, to meet user requirements for service quality [4, 5]. Hence, as the popularization of artificial intelligence in daily life [6], people's demand for connectivity and intelligence in the IoT continues to increase [7–9], and the role of QoS prediction becomes increasingly prominent.

QoS typically refers to the non-functional attributes of a network or service, encompassing factors like response time, availability, reliability, throughput, and other pertinent characteristics [4, 5, 10, 11]. These non-functional attributes are critical to user experience and service quality, and QoS provides a way to quantify and evaluate these attributes. In network communications, QoS can indicate performance characteristics such as data transmission stability, latency, and bandwidth, thereby affecting users' perception and experience with network services [3]. In the service field, QoS broadly covers the reliability, security, response time, and other aspects of the service, and is used to measure whether the service can meet user expectations and requirements [10, 11]. Thus, QoS is not only the focus of service providers and

✉ Weiwei Wang
wweiwei@stu.xidian.edu.cn

✉ Wenping Ma
wp_ma@mail.xidian.edu.cn

[1] School of Telecommunications Engineering, Xidian University, Xi'an 710071, China



	s_1	s_2	s_3	...	s_m
u_1	0.6	0.4	?	?	0.9
u_2	?	0.8	?	0.2	?
u_3	?	0.5	?	?	1.3
...	0.1	?	1.1	?	?
u_n	?	?	0.3	?	0.7

Fig. 1 An illustration of QoS missing value prediction in IoT. Taking n users $\{u_1, u_2, u_3, \ldots, u_n\}$ and m services $\{s_1, s_2, s_3, \ldots, s_m\}$ as an example, it shows how users invoke web services, which can be modeled as a user-service QoS interaction matrix

network operators but also one of the key criteria for users to choose services and evaluate service quality. With the rapid growth of services in IoT scenarios, it is often impossible for ordinary users to essentially judge the quality of a service. Therefore, ensuring users can discover services that precisely match their requirements highlights the paramount importance of accurately predicting QoS values in the realm of web services.

Service-oriented QoS prediction is a complex data analysis task, which aims to predict the QoS value of each service invoked by the target user based on limited historical data. However, data sparsity and cold start problems make it difficult to obtain satisfactory recommendation results in this process. To accomplish the prediction task of missing values, the recommendation algorithm can be used to fill the missing values in the QoS prediction. Specifically, the QoS prediction problem can be converted into a recommendation task, and the recommendation system model can be trained using the QoS data that is already available. The trained model is then utilized to forecast missing QoS values. Based on the user's historical behavior and other relevant information, the recommendation system has the ability to anticipate the user's QoS preference for a certain service, thereby filling in the missing values. As illustrated in Fig. 1, an example of the QoS missing value prediction problem in the IoT environment is presented, which depicts how to transform relevant historical data into a user-service interaction matrix to model the QoS value prediction task in the process of user invoking a service. To achieve this prediction task, researchers initially adopted recommendation techniques such as collaborative filtering, which fills missing values based on user or item similarity [12–14]. By analyzing user behavior patterns and preferences, recommendation systems can identify users or services similar to target users or target services, thereby predicting missing QoS values. Collaborative filtering technology is extensively employed to predict unknown QoS values. This method primarily uses the calculation of user or service similar neighborhoods for prediction. However, this method is highly susceptible to the sparse nature of the user-service invocation matrix, which notably influences the precision of QoS predictions. To alleviate this special problem, researchers have used clustering algorithms [15, 16], matrix factorization (MF) [17–20], and other techniques [21–24] to improve QoS prediction. Recent studies have also employed deep learning [25, 26] and graph neural networks [27, 28] to enhance QoS prediction accuracy. However, QoS prediction in IoT services still faces three major challenges.

- **Data credibility issues**. Given various research solutions, it is widely acknowledged that the quality of data used for QoS prediction is crucial. Outliers stemming from untrustworthy QoS values provided by users for certain services can greatly affect the prediction performance.
- **User privacy security.** The given QoS values may reveal users' sensitive information, so it is imperative to design methods that prioritize protecting user privacy.
- **Poor prediction performance**. Efficient and accurate prediction algorithms are the key to QoS prediction for IoT services. How to strike a balance between complexity and performance is a hot topic.

To tackle the aforementioned issues, we design a novel fusion QoS prediction framework with trust-aware privacy protection named TPP-GNCF. First, we use the 3-σ rule of the Gaussian distribution to filter out possible outliers to select credible QoS values. Then, a MP-GNN is utilized to effectively capture the information transmission and relationships in the graph structure and differential privacy is used to protect the information of user nodes. In addition, we

use a similarity calculation method based on weight function in collaborative filtering to mine implicit embedded features that graph neural networks cannot directly utilize. Finally, the missing QoS values are obtained by fusing the QoS predicted by MP-GNN and feature collaborative filtering. In summary, the main innovations of our article are as follows:

- We filter out untrustworthy QoS values provided by users for certain services in the user-service matrix that may affect the model's understanding and prediction of user behavior. It will then utilize GNN to process the features of users and services, along with the interaction information between them, and apply differential privacy to protect the privacy of user nodes, thereby enabling the fusion of multi-source information for effective QoS prediction while protecting user privacy.
- We employ a weight function-based similarity calculation method in collaborative filtering to perform trust-aware collaborative prediction, aiming to deeply mine the implicit embedded features that graph neural networks cannot directly utilize.
- We propose a novel fusion QoS prediction framework, TPP-GNCF, a trust-aware and privacy-preserving scheme that determines the final missing QoS values by integrating trust-aware graph neural network predicted QoS and trust-aware collaborative filtering predicted QoS.
- Extensive comparative and ablation experiments are conducted on well-known QoS datasets to assess the superiority of TPP-GNCF. The results illustrate that TPP-GNCF outperforms existing solutions in performance.

The remainder of this work is structured as follows. We present some existing work in Sect."Literature overview". We introduced the relevant basic knowledge involved in this article, which includes differential privacy and message-passing graph neural networks in Sect. "Preliminaries". Section "Proposed method" introduces the relevant details of our scheme. Section "Experiment and evaluation" we primarily conducted numerous experiments and provided detailed discussion and analysis. Finally, we present a summary of the paper's conclusions and and future research ideas in Section "Conclusion".

Literature overview

QoS prediction is increasingly vital as a core component of IoT services. The growing network complexity and rising user demands further emphasize the significance of QoS prediction. Numerous studies on QoS prediction have emerged, underscoring its growing relevance in both industry and academia. This section reviews traditional QoS prediction approaches and privacy-preserving QoS prediction schemes.

QoS prediction

Research on QoS prediction in Web services has attracted widespread attention, and related research and development have increased significantly. Many earlier solutions were based on collaborative filtering methods for web service recommendations. Zheng et al. [29] integrated the traditional collaborative filtering methods based on user and service [30, 31] to design a hybrid QoS prediction scheme. This scheme is equipped with a confidence weight factor to form a new linear combination for the final QoS prediction. Additionally, collaborative filtering methods using MF for QoS prediction are also popular. Zheng et al. [19] first integrated similar users into MF for QoS prediction, proposing a neighborhood-integrated MF scheme. Xu et al. [32] presented a scheme for QoS value prediction utilizing probabilistic MF, which combines users' QoS values and geographical location information. Chang et al. [33] presented a graph-based MF scheme based on PMF, primarily extracting context information to construct a graph for QoS prediction. The contextual information mentioned by Chang et al. is extremely important in the IoT QoS prediction service. For instance, Gao et al. [34] designed a context-aware prediction scheme that uses fuzzy clustering algorithms to process contextual information for the IoT environment, and also presented a QoS prediction scheme that can extract local and global features. The proliferation of deep learning has led to the extensive utilization of neural network models in tasks concerning QoS prediction. Zou et al. [10] presented a new QoS prediction scheme by integrating the user neighborhoods selected by the collaborative method into the MF via a deep neural network, which overcomes the shortcomings of traditional schemes that cannot capture hidden features of users and services. Zhang et al. [35] proposed an LDCF model by considering the location correlation between users and services, combining MLP and similarity adaptive corrector to learn this correlation.

To solve the legacy problems of using neural networks to predict QoS values in the past, Zou et al. [25] designed an adaptive QoS prediction framework that integrates location-aware neural prediction methods and domain-based collaborative prediction methods to generate an adaptive QoS prediction scheme. Li et al. [36] introduced a flexible framework of topology-aware neural models that can achieve accurate QoS predictions by effectively utilizing context, eliminating the complex interaction process between systems. Zhang et al. [26] proposed the user-service graph for the first time by using the deep connections between users and services. They further constructed a user-service feature vector set to build a prediction scheme based on dual-stream deep learning, utilizing user and service data to the fullest

to update its feature vector to get more precise prediction outcomes. To maintain the accuracy and stability of service recommendations in the presence of untrustworthy users, Wu et al. [27] designed a framework that integrates reputation into a graph neural network for QoS prediction. This approach aims to achieve a robust and accurate prediction result. Liu et al. [28] also used graph neural networks for QoS prediction. They conducted a more detailed exploration to illustrate the advantages of GNN for QoS prediction tasks. However, a shortcoming is that implicit features are not fully considered when extracting user and service features.

Existing solutions have made significant progress in QoS prediction, enhancing the stability and performance of network services while providing a robust foundation for user experience and the growing network demands in the IoT environment. However, most advanced solutions overlook the critical issue of user privacy protection.

Privacy-preserving QoS prediction

The continuous advancements in science and technology, coupled with the popularization of privacy policies, have led to a steady increase in people's awareness of privacy protection. This trend has attracted academic attention to the privacy issues in QoS prediction. Zhu et al. [37] introduced a privacy-preserving QoS prediction scheme based on random perturbations, aiming to simply and effectively improve the privacy of the prediction method. On this basis, the authors designed P-UIPCC and P-PMF two classic QoS prediction methods with privacy protection. Liu et al. [38] introduced a QoS prediction method based on differential privacy by employing the Laplace mechanism to introduce noise, the framework ensures the protection of sensitive information. In particular, they designed two types of methods, namely directly and simply applying differential privacy to user data (DPS) and aggregated data differential privacy (DPA). DPA first aggregates user data and then applies differential privacy to enhance the practicality of QoS data. Then, Liu et al. [39] proposed a novel QoS prediction scheme, namely shared collaborative web service QoS prediction and designed a novel differential privacy method for it to achieve data sharing. This framework can not only achieve shared collaboration QoS prediction but can also prevent the leakage of relevant private information. Zhang et al. [40] studied the privacy protection issue of QoS prediction in mobile edge computing scenarios. They proposed a novel QoS prediction method that protects privacy by using Laplacian noise in mobile edge scenarios, which effectively solves the issue of privacy leakage of user data in edge environments. However, differential privacy-based methods for privacy protection strike a trade-off dilemma between maintaining privacy and ensuring accuracy, because enhancing privacy in multiple calculations involves adding more noise, which inevitably reduces accuracy.

We explored the delicate balance between privacy preservation and prediction accuracy by analyzing previous QoS prediction schemes. As elucidated in [28], various graph structures can result in significant differences in accuracy. Different from previous prediction solutions based on graph neural networks, this paper mainly uses a fusion method of MP-GNN combined with collaborative filtering to perform QoS prediction tasks. Compared with traditional graph neural network structures, MP-GNN can better capture the information transfer process in graph data, thereby improving the accuracy of predictions.

Preliminaries

Differential privacy

Differential privacy [41, 42] is defined by a randomization mechanism M that maps a dataset $\mathcal{D}$ to a range of values $\mathcal{R}$. If for any subset $\mathcal{T}$ in the range $\mathcal{R}$ and any two adjacent databases D_i and D_i' with $D_i, D_i' \in \mathcal{D}$, the following inequalities are satisfied:

$$\Pr[M(D_i) \in \mathcal{T}] \leq e^{\varepsilon} \Pr[M(D_i') \in \mathcal{T}] + \delta. \tag{1}$$

Then, we state that the randomization mechanism M satisfies (ε, δ)-differential privacy. The Gaussian mechanism provides (ε, δ)-differential privacy with a slack term δ, ensuring that the probability of leaking user privacy for any possible output does not exceed δ. For instance, setting δ to 10^{-3} implies that only a probability of 10^{-3} can be tolerated to violate strict differential privacy. Hence, the smaller δ is, the higher the probability that the mechanism satisfies differential privacy. $\varepsilon > 0$ represents the privacy budget, indicating the degree of privacy leakage. This parameter is crucial for privacy protection and is inversely related to noise levels. When the ε value is small, it means that the risk of privacy leakage is relatively low. For standard continuous variables, the Gaussian mechanism can be employed to ensure (ε, δ)-differential privacy. To achieve this, we need to ensure that the noise distribution $\eta \sim \mathcal{N}(0, \sigma^2)$ satisfies (ε, δ)-differential privacy. We select the constant a such that $a \geq \sqrt{2\ln(1.25/\delta)}$, where δ falls within the interval $(0, 1)$. Therefore, the noise scale $\sigma \geq a\Delta f/\varepsilon = \sqrt{2\ln(1.25/\delta)}\Delta f/\varepsilon$, where Δf represents the sensitivity of the function f. This sensitivity can be obtained by $\max_{D_i, D_i'} ||f(D_i) - f(D_i')||$, where f is a real-valued function. In this Gaussian mechanism, N represents the noise sample value added to the dataset. These noise samples are obtained by sampling from a Gaussian distribution with mean 0 and variance $\frac{2\ln(1.25/\delta)}{\varepsilon^2}$.

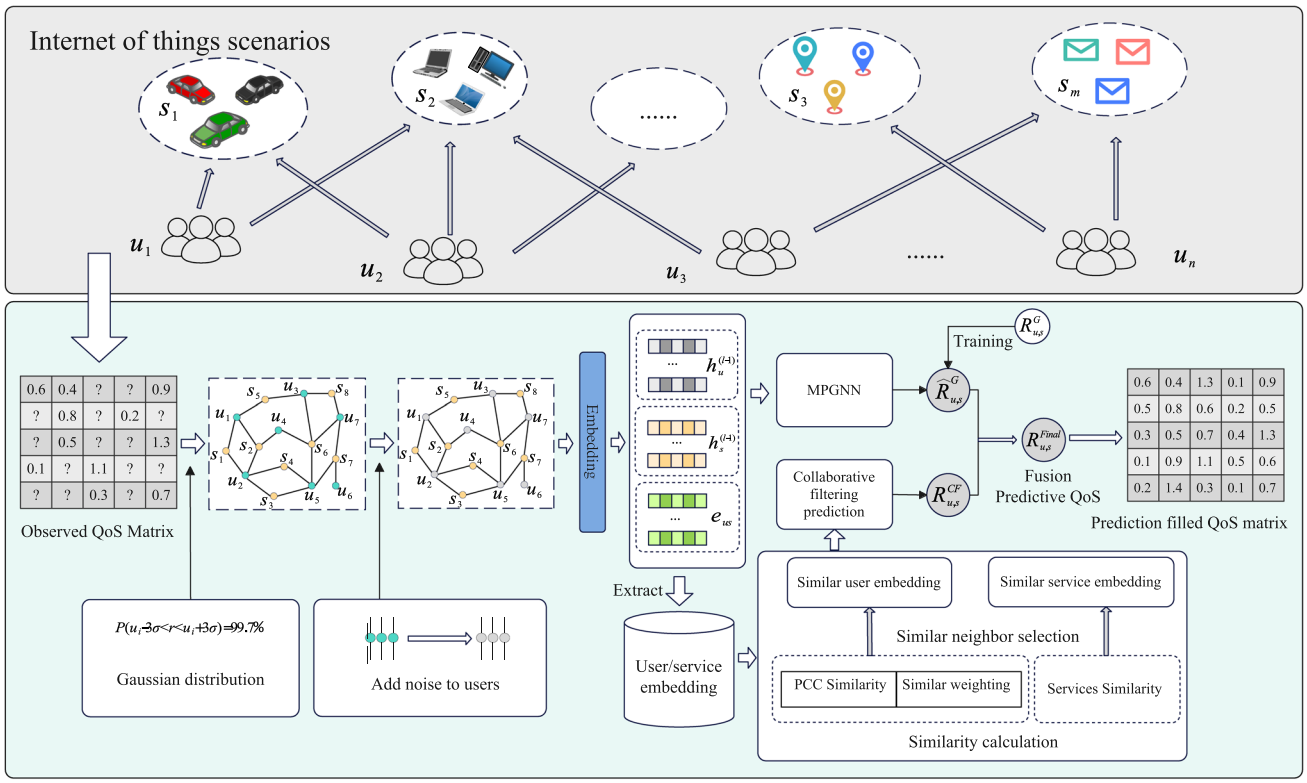


Fig. 2 Overall framework preview of TPP-GNCF. TPP-GNCF first adopts the 3-σ principle to select credible QoS values, constructs a trust-aware graph, and adds noise to user data nodes to improve privacy. Next, a message passing graph neural network is used to predict QoS on the trust-aware graph, and the model is trained through a loss function until the optimal solution is obtained. Then, collaborative filtering is performed for QoS prediction based on the similarity calculation method of the weight function to further mine implicit embedded features. Finally, the QoS prediction of the trust-aware graph neural network is fused with the QoS prediction results of the trust-aware collaborative filtering to obtain the final QoS prediction value

These noise samples can be added to the original data for privacy protection.

Message passing graph neural network

The MP-GNN [43] is a neural network model designed for processing graph data. It learns the relationships between nodes and the structural characteristics of the graph by passing messages between nodes. Let $G = (V, E)$ be a graph, where V represents the set of nodes and E represents the set of edges. Each node $v \in V$ is associated with a feature representation x_v. The layer index of the GNN is denoted as $l = 0, 1, \ldots, L-1$. Through multiple rounds of message passing and node updates, the GNN can learn the global characteristics of the graph at the node level, enabling it to achieve good performance on various tasks. The following presents the general form of a MP-GNN, which mainly comprises two parts: the message passing and the readout.

Message passing stage: During the process of message passing, each node combines its own characteristics with those of its neighboring nodes to generate and deliver messages. This process can be iteratively repeated multiple times to allow nodes to aggregate more information. Let $N(v)$ denote the set of neighbors of node v. The formula for message passing is typically expressed as follows:

$$m_{vu}^{(l+1)} = M\left(x_v^{(l)}, x_u^{(l)}\right), \forall (u, v) \in V, \tag{2}$$

where $m_{vu}^{(l+1)}$ is the message sent by node v to node u, M is the message passing function, and t is the number of layers of GNN. After receiving messages from neighbor nodes, each node needs to aggregate these messages to update its representation.

$$m_v^{(l+1)} = \mathrm{AGG}\left(\{m_{vu}^{(l+1)} : u \in N(v)\}\right), \forall u \in V, \tag{3}$$

where AGG is an aggregation function, which summarizes the messages of neighbor nodes into an aggregate message $m_v^{(t)}$. Upon receiving aggregated messages, nodes need to use these messages to update their feature representations. Typically, the update process combines the characteristics of the node itself with the aggregated messages, undergoing

some nonlinear transformation.

$$x_v^{(l+1)} = U\left(x_v^{(l)}, m_v^{(l+1)}\right), \tag{4}$$

where $x_v^{(l+1)}$ is the feature representation of node v at layer $l+1$, and U is the node update function.

Readout stage: In MP-GNN, the readout stage produces the final node representation h_G, where L is the last layer of the neural network. Typically, one can linearly transform the final representation of a node and then apply an activation function to produce the final output.

$$h_G = R\left(\{x_v^{(L)} : v \in V\}\right). \tag{5}$$

Proposed method

Overview of the TPP-GNCF

In this article, the proposed TPP-GNCF scheme mainly contains four key stages: (a) Trust graph construction and embedding processing; (b) Trust-Aware Graph Neural QoS Prediction; (c) Trust-aware collaborative filtering QoS prediction; and (d) Fusion QoS prediction. It is worth noting that stages (b) and (c) of TPP-GNCF can independently predict unknown QoS values, but in this work, we design a fusion model that combines these two schemes to predict the final QoS value. The overall process framework of TPP-GNCF is demonstrated in Fig. 2, and each stage will be described in detail below.

Trust graph construction and embedding processing

Trusted QoS value selection

To ensure the quality and stability of the graph structure, we need to pre-process the user data to remove outliers that have an adverse effect on the training and prediction of the model before constructing the user-service interaction graph. These outliers often represent untrustworthy values provided by users, which are meaningless for predicting effective QoS values.

In IoT service scenarios, QoS data typically follows a Gaussian distribution [1, 44], which makes the 3-σ rule an ideal choice for removing outliers. Its unique statistical properties can effectively retain reliable data while accurately filtering outliers, significantly enhancing prediction accuracy. Therefore, the 3-σ rule derived from the Gaussian distribution $N(\mu, \sigma^2)$ is used to identify trustworthy value for QoS prediction. Based on this, it can be determined whether the QoS value is within the range of $(\mu - 3\sigma, \mu + 3\sigma)$. If so, the user's QoS is considered trustworthy; otherwise, it may be abnormal or untrustworthy. Referring to [16, 44], we assume that the user's QoS value follows a Gaussian distribution. We compute the standard deviation σ and mean μ from the user's QoS historical data and use the 3-σ rule to determine if a user's QoS is trustworthy. Based on the Gaussian distribution, it is observed that approximately 99.7% of the QoS values r_{us} fall within the interval $(\mu - 3\sigma, \mu + 3\sigma)$. Hence, the probability P of the service s_j as observed by the user u_i can be calculated as follows:

$$P(\mu_s - 3\sigma_s < r_{us} < \mu_s + 3\sigma_s) = 99.7\%, \tag{6}$$

where σ_s and μ_s are the standard deviation and mean of the QoS values provided by user u for service s respectively. Figure 3 shows a schematic diagram of the application of the 3-σ rule to filter the data from the QoS datasets. The outliers stem from the untrustworthy QoS values provided by users for certain services.

Graph construction and embedding processing

We construct the user-service interaction graph after selecting trustworthy QoS values, where nodes represent users and services, and their interactive relationships constitute edges. We denote the user node set as $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ and the service node set as $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$, where n and m represent the total numbers of users and services, respectively. The set $\mathcal{E}$ represents the edges of user-service interactions. $G = (\mathcal{U} \cup \mathcal{S}, \mathcal{E})$ represents the user-service interaction graph, where $\mathcal{U} \cup \mathcal{S}$ is the union of the user and service node sets.

To enhance user privacy, noise can be injected into trusted user nodes subsequent to constructing the conventional graph. The purpose of adding noise is to blur the real identities of user nodes, making it difficult to infer the true identity of users. For each user node u_i, a random noise ϵ_i is introduced to obfuscate its true identity, resulting in a new node identifier $\tilde{u}_i$, i.e., $\tilde{u}_i = u_i + \eta_i$. Thus, we denote the set of user nodes after noise addition as $\widetilde{\mathcal{U}} = \{\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_m\}$. The corresponding user-service interaction graph after adding noise is denoted by $G' = (\widetilde{\mathcal{U}} \cup \mathcal{S}, \mathcal{E})$, where $\widetilde{\mathcal{U}}$ represents the set of user nodes after adding noise. Through the above construction, a user-service interaction graph can be constructed. This graph serves as a foundation for analyzing and modeling the interactions between users and services to achieve QoS prediction tasks while protecting the user's privacy information.

Next, as illustrated in Fig. 2, we convert the user and service nodes into binary representations through one-hot encoding after constructing the user-service interaction graph. Suppose d represents the embedding dimension. Then, the one-hot encoding of users and services can be converted into feature embeddings denoted as $h_u \in \mathbb{R}^d$ for users and $h_s \in \mathbb{R}^d$ for services, respectively, as elaborated below:

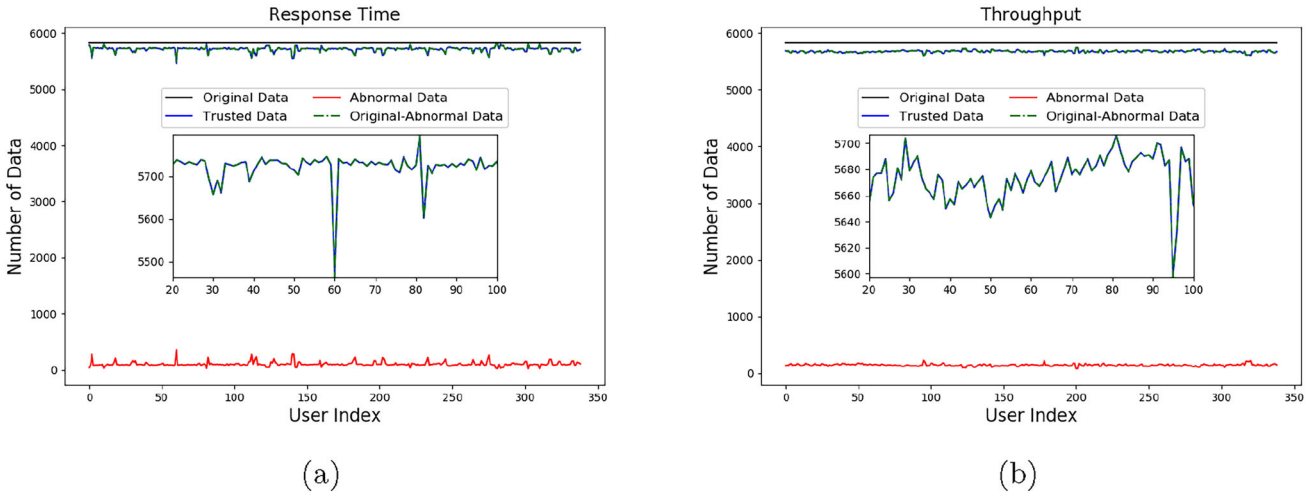


Fig. 3 Schematic diagram of outlier filtering and the number of data in each part under response time and throughput

One-hot encoding. We can convert user and service nodes into binary representations using one-hot encoding. Assume that the number of users and services are n and m, respectively. The one-hot encoding of users and services will result in vectors of length n and m respectively, where one element in each vector is 1, representing the user or the index of the service, while the other elements are 0.

Feature embedding. Once the one-hot encoding is completed, the next step is to map these encodings to the feature embedding space. For a user node u with a one-hot encoding o_u of dimension n, its feature embedding h_u is calculated as $h_u = W_u \cdot o_u$, where W_u is a $d \times n$ weight matrix. Similarly, for a service node s with a one-hot encoding o_s of dimension m, its feature embedding h_s is calculated as $h_s = W_s \cdot o_s$, where W_s is a $d \times m$ weight matrix. Finally, we represent the edge feature embedding as the Hadamard product of the user and service feature embeddings, $e_{us} = h_u \odot h_s$, where $\odot$ denotes the Hadamard product.

Therefore, through one-hot encoding and feature embedding mapping, we can represent user nodes and service nodes as d-dimensional feature embedding vectors for subsequent training of the message-passing graph neural network model and recommendation tasks. The pseudocode of the algorithm for this part of the scheme is shown in Algorithm 1.

Algorithm 1 Trust graph construction and embedding processing.

Require: User set $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$; service set $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$; interaction set $\mathcal{E}$ with trusted QoS value; embedding dimension d; ϵ: privacy budget; δ: slack term
Ensure: Feature embeddings $h_u \in \mathbb{R}^d$ for users and $h_s \in \mathbb{R}^d$ for services, and the edge feature embedding $e_{us} \in \mathbb{R}^d$
Step 1: Construct the Initial Interaction Graph.
Construct an initial user-service interaction graph G using $\mathcal{U}$, $\mathcal{S}$, and trusted QoS values.
Step 2: Inject Noise for user nodes.
for each user node $u_i \in \mathcal{U}$ **do**
  Generate a new user node identifier $\tilde{u}_i$.
end for
Define the new user node set $\widetilde{\mathcal{U}}$ and update the interaction graph to G'
Step 3: One-Hot Encoding.
for each user $\tilde{u}_i \in \widetilde{\mathcal{U}}$, $s_j \in \mathcal{S}$ **do**
  $o_u \leftarrow$ one-hot vector of length n.
  $o_s \leftarrow$ one-hot vector of length m.
end for
Step 4: Feature Embedding.
Initialize weight matrices $W_u \in \mathbb{R}^{d \times n}$ and $W_s \in \mathbb{R}^{d \times m}$.
for $\tilde{u}_i \in \widetilde{\mathcal{U}}$, $s_j \in \mathcal{S}$ **do**
  Compute the feature embedding $h_{\tilde{u}_i} = W_u \cdot o_u$ and $h_{s_j} = W_v \cdot o_s$, respectively.
end for
Calculate the edge feature embedding $e_{us} = h_u \odot h_s$, where $\odot$ denotes the Hadamard product.
Return h_u, h_s, and e_{us}

Trust-aware graph neural QoS prediction

In this section, we employ the MP-GNN to process the graph $G(V, E)$ after its construction, thereby fusing multi-source information for effective QoS prediction while achieving user privacy protection. Figure 4 illustrates a trust-aware graph neural QoS prediction framework comprising four functional modules: message generation, aggregation, update, and readout function. The level index of the graph neural network (GNN) is denoted as $l = 0, 1, \ldots, L-1$. Each node updates its feature representation through message passing on the graph, utilizing information from its neighboring nodes to update its own representation. Specifically, each node aggregates and updates its hidden features through a specific function. In each layer of the graph neural network, a node's representation is updated through the representations of its

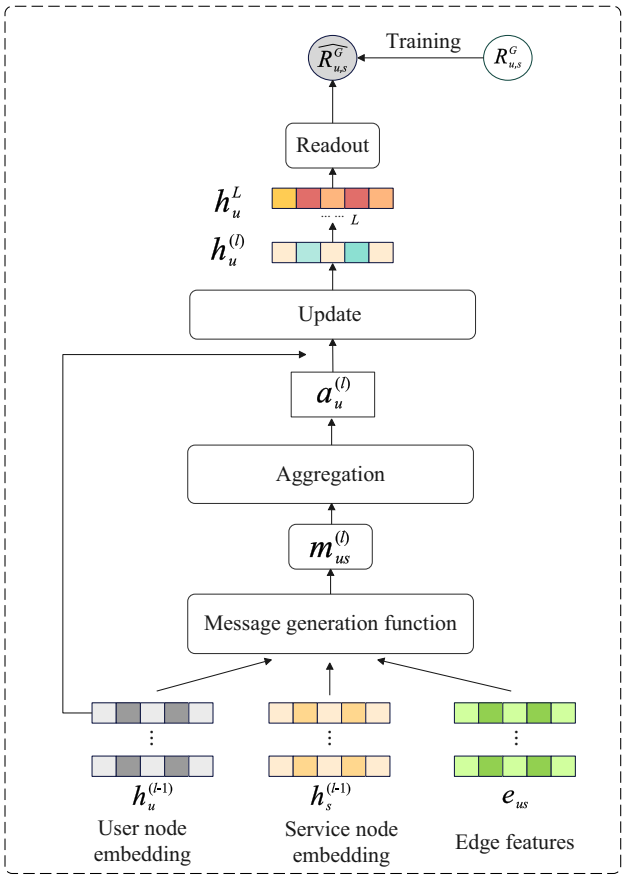


Fig. 4 The flowchart of trust-aware graph neural QoS prediction

neighboring nodes. Hence, the following is a description of the node feature at layer l:

In a typical MP-GNN framework, M is commonly implemented using a feedforward neural network. This function integrates the messages from node u, its neighboring node s, and the edge e_{us} to generate a new node representation $m_{us}^{(l)}$. Specifically, M can be a multilayer perceptron (MLP), with inputs being the features $h_u^{(l-1)}$ and $h_s^{(l-1)}$ of nodes u and s, respectively, along with the information from edge e_{us}. The output is the updated node feature $m_{us}^{(l)}$.

$$m_{us}^{(l)} = M(W^{(l)}\Phi(h_u^{(l-1)}; h_s^{(l-1)}; e_{us}) + b^{(l)}), \tag{7}$$

where Φ represents the concatenation of the feature of node u, its neighbor node v, and the feature of edge e_{us}. $W^{(l)}$ denotes the weight matrix, while $b^{(l)}$ represents the bias vector.

The AGG function aggregates the information from all neighbor nodes s of node u. In this context, we use averaging as the aggregation method. Specifically, for node u, its feature $a_u^{(l)}$ is averaged using the features $m_{us}^{(l)}$ of its neighboring nodes s. It is defined as follows:

$$a_u^{(l)} = \frac{1}{|N_u|}\sum_{s\in N_u} m_{us}^{(l)}, \tag{8}$$

where N_u denote the set of neighbor nodes of node u, $|N_u|$ represent the number of neighbor nodes.

The update function updates the feature $a_u^{(l)}$ of node u. It takes $h_u^{(l-1)}$ and $a_u^{(l)}$ as inputs and outputs the updated node feature $h_u^{(l)}$.

$$h_u^{(l)} = \mathrm{U}(\Phi(h_u^{(l-1)}; a_u^{(l)})), \tag{9}$$

where Φ means concatenating $h_u^{(l-1)}$ and $a_u^{(l)}$. Here, we also use an MLP to implement the update function. By combining the aforementioned functions, node features can be updated in each layer of the GNN to facilitate the message passing process.

In the MP-GNN, after computing the L-layer GNN node feature propagation, the node features are sequentially updated and stored in the hidden layers, ultimately forming the hidden feature representation of the last layer. Then, through a readout process, the final node features $h_u^{(L)}$ are transformed into global graph features, which are then output. Typically, one can linearly transform the final feature of a node and apply an activation function to generate the final output. The process can be represented as follows:

$$\hat{R}_{u,s}^{G} = \sigma(Wh_u^{(L)} + b), \tag{10}$$

where $\hat{R}_{u,s}^{G}$ is the predicted output of node u, σ denotes the activation function, W is the weight matrix, and b represents the bias term.

Graph neural QoS prediction and model training

To effectively utilize GNN for QoS prediction, the above model must be trained reliably. The loss function for the graph structure is defined as follows:

$$\mathcal{L} = \frac{1}{|V|}\sum_{(u,s)\in V}\left|R_{u,s}^{G} - \hat{R}_{u,s}^{G}\right|, \tag{11}$$

where $|V|$ is the total number of nodes in the graph, (u, s) represents each node, $R_{u,s}^{G}$ represents the true value, $\hat{R}_{u,s}^{G}$ represents the predicted value. During training, we adopt adaptive moment estimation as the optimizer.

Given the user-service invocation records, the update function's value can be transformed through a fully connected layer with the sigmoid activation function to predict the missing QoS value $R_{u,s}^{G}$ using the trained TPP-GNCF, as described in Eq. (10). In short, we opt for ReLU as the activation function for the MLP during the entire GNN message

propagation process and use the sigmoid activation function in the prediction stage.

Trust-aware collaborative filtering QoS prediction

In the above section, we designed a graph neural network that can predict by utilizing the embedded features of users and services. However, these embedded features have not been fully exploited for QoS prediction. In particular, some implicit relationships between embedded features cannot be directly captured by GNN. To improve the accuracy of QoS prediction, we introduce a weight function-based similarity computation to perform trust-aware collaborative prediction.

Given the extracted feature embedding vectors, we calculate the similarity between the embedding vectors representing the user and the service, respectively. Suppose there are two user embedding vectors h_{u_i} and h_{u_j}. We employ the Pearson correlation coefficient (PCC) [45] to calculate the similarity between them:

$$\mathrm{SIM}(h_{u_i}, h_{u_j}) = \frac{\sum_{k=1}^{d}(h_{u_i}^{(k)} - \bar{h}_{u_i})(h_{u_j}^{(k)} - \bar{h}_{u_j})}{\sqrt{\sum_{k=1}^{d}(h_{u_i}^{(k)} - \bar{h}_{u_i})^2}\sqrt{\sum_{k=1}^{d}(h_{u_j}^{(k)} - \bar{h}_{u_j})^2}}, \tag{12}$$

where d represents the dimension of the embedding vector, $h_{u_i}^{(k)}$ is the embedding value of user u_i in the k-th dimension, $\bar{h}_{u_i}$ is the average of the embedding in all dimensions of user u_i. Although PCC can provide good accuracy, to further achieve higher accuracy, we propose an improved PCC method. This improved similarity is calculated as follows:

$$\mathrm{ISIM}(u_i, u_j) = \mathrm{SIM}(h_{u_i}, h_{u_j}) \times \mathrm{H}(h_{u_i}, h_{u_j}), \tag{13}$$

where $\mathrm{SIM}(h_{u_i}, h_{u_j})$ is Eq.(12), and $\mathrm{H}(h_{u_i}, h_{u_j})$ is a weight function. In particular, $\mathrm{H}(h_{u_i}, h_{u_j})$ is defined as follows:

$$\mathrm{H}(h_{u_i}, h_{u_j}) = T(U)^{w(h_{u_i}, h_{u_j})}, \tag{14}$$

where $T(U) = \frac{1}{ln(2+|U|)}$, U represents the set of trusted users invoking the service, and $|U|$ represents the number of users providing trusted QoS values. $w(h_{u_i}, h_{u_j})$ can be expressed as follows:

$$w(h_{u_i}, h_{u_j}) = \frac{1}{\sqrt{\sum_{k=1}^{d}(h_{u_i}^{(k)} - h_{u_j}^{(k)})^2}}. \tag{15}$$

Similar to Eq. (12), the similarity between service embedding h_{s_i} and h_{s_j} is computed as follows:

$$\mathrm{SIM}(h_{s_i}, h_{s_j}) = \frac{\sum_{k=1}^{d}(h_{s_i}^{(k)} - \bar{h}_{s_i})(h_{s_j}^{(k)} - \bar{h}_{s_j})}{\sqrt{\sum_{k=1}^{d}(h_{s_i}^{(k)} - \bar{h}_{s_i})^2}\sqrt{\sum_{k=1}^{d}(h_{s_j}^{(k)} - \bar{h}_{s_j})^2}}, \tag{16}$$

where $h_{s_i}^{(k)}$ is the embedding value of service s_i in the k-th dimension, $\bar{h}_{s_i}$ is the average of the embedding values across all dimensions of service s_i.

Next, we can determine a set of the most similar user neighbor sets $N(u_i)$ and service neighbor sets $N(s_j)$, respectively, based on the similarity between the obtained user and service embedding vectors for QoS prediction. The prediction formula is as follows:

$$\begin{cases} R_{u,s}^{U} = \bar{r_u} + \frac{\sum_{u_a \in N(u)} \mathrm{ISIM}(h_u, h_{u_a})(r_{a,s} - \bar{r_a})}{\sum_{u_a \in N(u)} \mathrm{ISIM}(h_u, h_{u_a})} \\ R_{u,s}^{S} = \frac{\sum_{s_b \in N(s)} \mathrm{SIM}(h_s, h_{s_b}) \cdot r_{u,b}}{\sum_{s_b \in N(s)} \mathrm{SIM}(h_s, h_{s_b})}, \end{cases} \tag{17}$$

where $R_{u,s}^{U}$ and $R_{u,s}^{S}$ represent the QoS prediction values based on similar neighbor sets $N(u_i)$ and $N(s_j)$ respectively. $\bar{r_u}$ is the average QoS across all services for user u. $r_{a,s}$ signifies the actual QoS provided by neighbor user u_a for service s. $\bar{r_a}$ reflects the average QoS across all services offered by neighbor user u_a. r_{ub} is the predicted QoS value of user u to neighbor service s_b.

Previous studies have shown that either $R_{u,s}^{U}$ or $R_{u,s}^{S}$ fails to achieve good prediction performance. Therefore, a combination of the two methods is still used for prediction in this work. The final formula of trust-aware collaborative prediction is as follows:

$$R_{u,s}^{CF} = \omega R_{u,s}^{U} + (1 - \omega) R_{u,s}^{S}. \tag{18}$$

In Eq. (18), ω is a weight parameter that represents the relative importance of $R_{u,s}^{U}$ and $R_{u,s}^{S}$ in the final prediction. We set $\omega = 0.5 (0 \leq \omega \leq 1)$ to indicate that the weights of the two parts are equal. Equation (18) describes the final QoS prediction result by collaborative filtering and uses the weight parameter ω to adjust the contributions of the two prediction results to obtain the QoS prediction value.

Fusion Qos prediction

As mentioned earlier in this section, both methods introduced above can perform QoS prediction, but neither achieves very high prediction accuracy. Therefore, to enhance prediction accuracy for missing values, TPP-GNCF integrates the above two prediction methods using the parameter ζ to create a linear fusion prediction model to achieve improved prediction performance. The final QoS fusion prediction model can be expressed as follows:

$$R_{u,s}^{Final} = \zeta \hat{R}_{u,s}^{G} + (1 - \zeta) R_{u,s}^{CF}, \tag{19}$$

where $R_{u,s}^{Final}$ represents the ultimate predicted QoS value of the target user, $\hat{R}_{u,s}^{G}$ is the QoS prediction value obtained through the GNN, $R_{u,s}^{CF}$ is the QoS prediction value obtained

by collaborative filtering. Parameter ζ is defined as the relative importance of our proposed TPP-GNCF framework to $\hat{R}^G_{u,s}$ and $R^{CF}_{u,s}$. Here $\zeta \in [0, 1]$.

Privacy analysis

To demonstrate that adding Gaussian noise to user nodes satisfies differential privacy, it is essential to show that the process satisfies the formal definition of differential privacy. Specifically, considering adjacent databases D and D', the noise adding mechanism $\mathcal{M}$ must satisfy Eq. (1). To ensure user privacy, Gaussian noise is added to user identities during the construction of the user-service interaction graph, $\tilde{u}_i = u_i + \eta_i$, where $\eta_i \sim \mathcal{N}(0, \sigma^2)$. According to the properties of the Gaussian mechanism, for any two adjacent datasets D and D', and any output $\mathcal{T}$, the following holds:

$$\frac{\Pr[\mathcal{M}(D) \in \mathcal{T}]}{\Pr[\mathcal{M}(D') \in \mathcal{T}]} \leq \exp\left(\frac{\|f(D) - f(D')\|_2^2}{2\sigma^2}\right). \tag{20}$$

Since user nodes u_i and u_i' produce different outputs only when a single user changes, the sensitivity $\Delta f = 1$. When $\|f(D) - f(D')\|_2 \leq \Delta f$, and if $\sigma \geq \frac{\Delta f\sqrt{2\log(1.25/\delta)}}{\epsilon}$, then $\exp\left(\frac{\Delta f^2}{2\sigma^2}\right) \leq \exp(\epsilon)$. Therefore,

$$\Pr[\mathcal{M}(D) \in \mathcal{T}] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in \mathcal{T}]. \tag{21}$$

Furthermore, since the noise η_i is applied independently to each user node u_i, and the distribution of each noise is symmetric, for any single user node change between adjacent databases D and D', its impact on the overall output is limited. According to the properties of the Gaussian mechanism, the change in the output distribution satisfies the differential privacy condition. Therefore, Eq. (1) holds. Based on the above analysis, the construction process of the entire trust-aware graph satisfies (ϵ, δ)-differential privacy, making it difficult for attackers to infer the true identity of the user from the obfuscated node identifiers, thereby effectively protecting user privacy.

Complexity analysis

The algorithm pseudocode of TPP-GNCF is displayed in Algorithm 2. The complexity of TPP-GNCF mainly comes from two parts: Trust-aware graph neural QoS prediction and Trust-aware collaborative filtering QoS prediction.

For the trust-aware graph neural QoS prediction method, if d is the embedding dimension, the implementation complexity of the message passing function M is $O(d^2e)$, where e is the number of edges. The aggregation function aggregates information from neighbor nodes. For each node u, the information of all its neighbor nodes v needs to be aggregated. Assume that each node has N_k neighbors, the number of edges is e, and the number of nodes is $n + m$. Each user node has $|N(u)|$ neighbors, and the aggregation complexity of each node is $O(|N(u)|\, d)$, and the complexity of the total aggregation function is the sum of the complexity of all nodes $\sum_{u\in V} O(|N(u)|\, d)$, since $\sum_{u\in V} O(|N(u)d|) = 2e$, the overall complexity is $O(2ed)$. Therefore, the complexity of the aggregation function is $O(de)$. The update function updates the previous layer representation of the node and the aggregated representation, with a complexity of $O((n+m)d^2)$. The readout function linearly transforms the final representation of each node and applies an activation function, with a complexity of $O((n+m)d)$. Since the graph neural network has L layers, the total complexity of the entire graph neural network is $O(L(m+n+e)d^2)$.

For the trust-aware collaborative filtering QoS prediction scheme, we set the total number of users to n, the total number of services to m, and the embedding dimension size to d. For the similarity calculation of each pair of users and each pair of service embedding vectors, the total complexity is $O(n^2d)$ and $O(m^2d)$, respectively. Additionally, the complexity of the calculation of the weight function $H(h_{u_i}, h_{u_j})$ is $O(d)$. Hence, the complexity of calculating similarity is still $O(n^2d)$ and $O(m^2d)$, respectively. For the calculated user and service embedding vector similarities, the complexity to select the most similar neighbor set is

Algorithm 2 TPP-GNCF for QoS prediction

Require: M: user-service invocation matrix; d: embedding dimension; ζ: weight coefficient; α: percentage of untrustworthy value; $K_u(K_s)$: maximum number of similar user (service) embedding
Ensure: QoS prediction value of TPP-GNCF.;
1: **Stage(I): Trust graph construction and embedding processing.**
2: Select a trusted QoS value using 3-σ of the Gaussian distribution from Eq.(6);
3: Execute Algorithm 1 to obtain h_u and h_s;
4: **Stage(II): Trust-Aware Graph Neural QoS Prediction.**
5: **for** $l = 0 \cdots L$ **do**
6: **for** $u, s \in V$ **do**
7: Calculate $m^{(l)}_{us}$, $a^{(l)}_u$, and $h^{(l)}_u$ according to Eqs.(7)-(9), respectively.
8: **end for**
9: **end for**
10: **for** all $u \in V$ **do**
11: Calculate $R^G_{u,s}$ from the Eq.(10).
12: **end for**
13: **Stage(III): Trust-aware collaborative filtering QoS prediction.**
14: Calculate the similarity of user embedding and service embedding according to Eqs.(13) and (16), respectively.
15: Identify a set of similar neighbors $N(u_i)$ and $N(s_j)$ based on Eqs.(13) and (16), respectively.
16: Calculate $R^U_{u,s}$ and $R^S_{u,s}$ based on $N(u_i)$ and $N(s_j)$, respectively.
17: Calculate $R^{CF}_{u,s}$ based on $R^U_{u,s}$ and $R^S_{u,s}$ from Eq.(18).
18: **Stage(IV): Fusion QoS prediction.**
19: Calculate the ultimate predicted QoS value $R^{Final}_{u,s}$ based on $R^G_{u,s}$ and $R^{CF}_{u,s}$ utilizing Eq.(19).

$O(nk \log n)$ and $O(mk \log m)$, respectively, where k is the number of selected neighbors. The complexity of the prediction part based on user embedding and service embedding is $O(nk+mk)$. The final complexity of collaborative prediction is $O(nm)$. Since $O(n^2d + m^2d)$ is the largest complexity term, the time complexity of this part of the algorithm can be simplified to $O(n^2d + m^2d)$. The above analysis reveals that the complexity of TPP-GNCF can be expressed as $O(L(m + n + e)d^2 + n^2d + m^2d)$.

Experiment and Evaluation

We undertake a series of experiments to fully verify the effectiveness of the TPP-GNCF proposed in this section. To accomplish this goal, we will solve three problems in the next experiments.

- **RQ1:** How do different parameters impact the performance of TPP-GNCF?
- **RQ2:** How does TPP-GNCF perform compared to existing schemes?
- **RQ3:** How do the several variants of TPP-GNCF perform?

Experiment environment and dataset

The experiments described in this work were carried out on a single computing system using the PyCharm development environment, with Python 3.6 as the major programming language. The hardware specifications include an Intel(R) Core(TM): i5-11400F, 2.60 GHz CPU with 32.0 GB RAM, and NVIDIA GeForce RTX 3080Ti.

To validate the effectiveness of TPP-GNCF's QoS prediction, a sequence of experiments was executed utilizing the WS-DREAM dataset [29]. Currently, the WS-DREAM dataset is extensively employed to assess the effectiveness of various QoS prediction methods, predominantly centered around two categories of user-service QoS invocations: response time (RT) and throughput (TP), which encompasses a total of 1,974,675 historical QoS invocation records gathered from 339 users and 5,825 web services. In this paper, we take into account the sparsity of the user-service matrix in the real-world environment. We set the QoS dataset to four distinct low densities (5%, 10%, 15%, and 20%) for training the model on RT and TP. Subsequently, to assess prediction performance, the other data samples under each density were utilized as the test set.

Evaluation metrics

To assess the disparity between the true and predicted values in the test set, we utilize three widely accepted prediction accuracy metrics to compare the predictive performance of TPP-GNCF with other methods: mean absolute error (MAE), root mean square error (RMSE), and normalized mean absolute error (NMAE). These metrics quantify the accuracy of the predicted QoS values, offering complementary insights into prediction performance. The MAE measures the average absolute error between the predicted and actual values, providing an intuitive evaluation of the overall prediction error. MAE is defined as

$$\text{MAE} = \frac{\sum_{u,s} \left| R_{u,s} - \bar{R}_{u,s} \right|}{N}, \tag{22}$$

where $R_{u,s}$ and $\bar{R}_{u,s}$ indicate the true and predicted QoS values of user u invoking service s, respectively, and N is the total number of predicted QoS values. RMSE is a metric that squares the errors, making it particularly effective at highlighting significant deviations and sensitive to outliers. Especially when the error distribution is uneven, RMSE can better reflect the overall performance of the model. It is calculated as

$$\text{RMSE} = \sqrt{\frac{\sum_{u,s} (R_{u,s} - \bar{R}_{u,s})^2}{N}}. \tag{23}$$

In addition, to further evaluate the model's performance, we introduce the NMAE when comparing it with existing solutions, which can better reflect the relative accuracy of the model prediction results by eliminating scale dependence, making the results interpretable across different datasets and prediction models, thereby providing a more fair and accurate performance comparison. NMAE is given by

$$\text{NMAE} = \frac{\text{MAE}}{R_{\max} - R_{\min}}, \tag{24}$$

where $R_{\max}$ and $R_{\min}$ denote the maximum and minimum observed QoS values in the dataset, respectively.

Compared methods

We conducted comparative experiments using ten distinct approaches detailed below to assess the performance of TPP-GNCF.

- **UPCC** [30]: This method uses the collaborative filtering method to find similar neighbors of users through PCC, and then predicts empty QoS values through historical information.
- **IPCC** [31]: This is a collaborative filtering prediction method that mainly relies on PCC to compute service similarity and estimate missing QoS values with the help of historical invocation matrix information.

- **UIPCC** [29]: This scheme mainly linearly integrates UPCC and IPCC to forecast empty QoS values based on weight parameters. Originally, it was also named WSRec.
- **NIMF** [19]: This method is the first to incorporate similar users into MF for service QoS prediction, which is mainly a neighborhood integration MF that utilizes PCC and identifies user neighborhoods based on historical information.
- **NDMF** [10]: This method is a QoS prediction framework that integrates neighbor-integrated deep matrix factorization, which extracts latent user and service characteristics through nonlinear interaction functions to improve traditional prediction schemes.
- **NRCF** [46]: This method presents a standard recovery CF scheme to address the issues of personalized web services. It introduces a novel similarity metric to enhance the accuracy of identifying similar users and web services.
- **PMF** [32]: This method primarily employs probabilistic matrix factorization for predicting QoS values.
- **PSO-USRec** [47]: This is a QoS prediction scheme that uses swarm intelligence search, which remodels the difficult problem of QoS prediction as a optimization problem to improve the prediction performance.
- **FHC-DQP** [48]: This is a unique privacy-preserving collaborative federated QoS prediction framework that mainly uses federated learning to protect user-service invocation information and improves distributed QoS prediction performance through multi-stage federated collaborative learning and user clustering.
- **CNCF** [34]: This method combines neural collaborative filtering and clustering algorithms to design a QoS prediction framework that considers contextual information for IoT services.

Sensitivity analysis of the parameters (RQ1)

1) **Impact of** ζ. In our fusion model, ζ reflects the relative importance of our proposed TPP-GNCF framework to $R_{u,s}^{G}$ and $R_{u,s}^{CF}$. In Eq. (19), we impose the constraint $\zeta \in [0, 1]$. Therefore, to explore how ζ affects the prediction performance of our scheme in this section, we conducted experiments on the parameter ζ in the range 0 to 1, and the outcomes are illustrated in Fig. 5.

Figure 5 presents the performance comparison between RT and TP across various parameter ζ values. As illustrated in Fig. 5, for the RT property, the MAE increases as the value of ζ decreases, whereas for the TP property, the MAE increases as the value of ζ increases. However, when ζ is set to 0.8 and 0.5, TPP-GNCF achieves smaller MAE and RMSE values on the RT property, respectively. When ζ is set to 0.2, TPP-GNCF achieves lower values on the TP property. This trend indicates that in TPP-GNCF, both $R_{u,s}^{G}$ and $R_{u,s}^{CF}$ play an effective role. Therefore, considering the sensitivity of parameter ζ, our subsequent experiments set $\zeta = 0.8$ on RT and $\zeta = 0.2$ on TP.

2) **Impact of** d. In the embedding processing stage, the parameter d involving the embedding of user and service nodes represents the dimension size of the embedding. Embedding dimensions play a crucial role in model training as they directly impact the quality of the embedding vectors. Therefore, we experimented with different embedding dimensions, including 2, 4, 8, 16, 32, 64, and 128, and varied the matrix density from 5% to 20% for further exploration. To simplify the experiment, we only verified and interpreted the results on RT.

As illustrated in Fig. 6, both MAE and RMSE exhibit an upward trend when the embedding dimensions are 2, 4, and 8. Starting from an embedding dimension of 16, MAE exhibits a downward trend, reaching its minimum value at an embedding dimension of 64. However, when the embedding dimension is 128, the MAE increases again. Apart from a few outliers, it is evident that MAE demonstrates superior performance with an increasing embedding dimension. This indicates that as the embedding dimension increases, the model demonstrates effective feature extraction capabilities. However, these outliers may be due to the small embedding dimension making the embedding vector too uncomplicated to fit the outliers well. Taking everything into consideration, in the subsequent experiments, we established the embedding dimension as 64.

3) **Impact of** α. The parameter α represents the percentage of untrustworthy values. To explore the influence of α on the accuracy of predictions, we selected untrustworthy value ratios of 10%, 15%, 20%, and 25%, and conducted experiments for RT and TP. Figure 7 illustrates the RT and TP prediction results under various matrix densities.

As illustrated in Fig. 7, the values of MAE and RMSE gradually become larger as the percentage of untrustworthy values increases, indicating that the more untrustworthy values, the worse the prediction results. The results illustrate that eliminating untrustworthy values significantly benefits prediction accuracy.

4) **Impact of** K_u **and** K_s. The parameters K_u and K_s represent the maximum number of similar user embeddings and service embeddings, which are utilized to generate neighbor sets for h_{u_i} and h_{s_j} to predict empty QoS values. To investigate the impact of K_u and K_s on model performance, we conducted experiments under conditions with density matrix values of 5%, 10%, 15%, and 20%. We adjusted K_u from 10 to 30 with a step size of 5 and K_s from 100 to 500 with a step size of 100. The corresponding results are presented in Figs. 8 and 9.

Figure 8 illustrates that for both RT and TP, the MAE and RMSE values gradually decrease as K_u increases. This trend occurs because the increase in K_u leads to a gradual rise

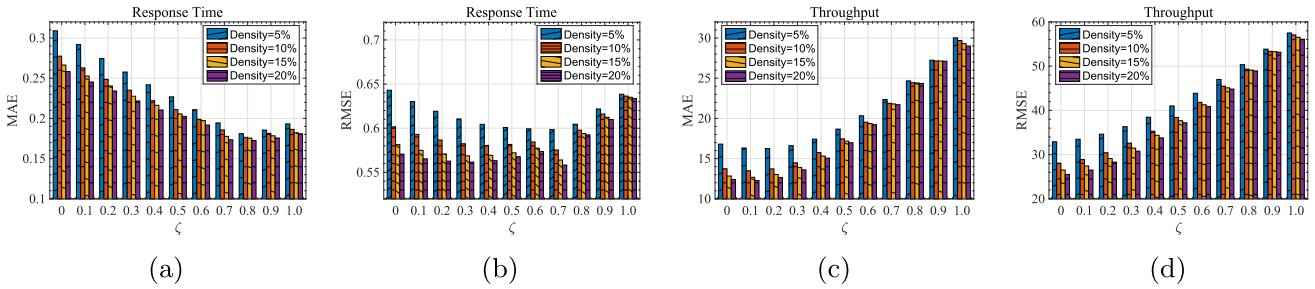


Fig. 5 Parameter sensitivity analysis of ζ of TPP-GNCF under RT and TP

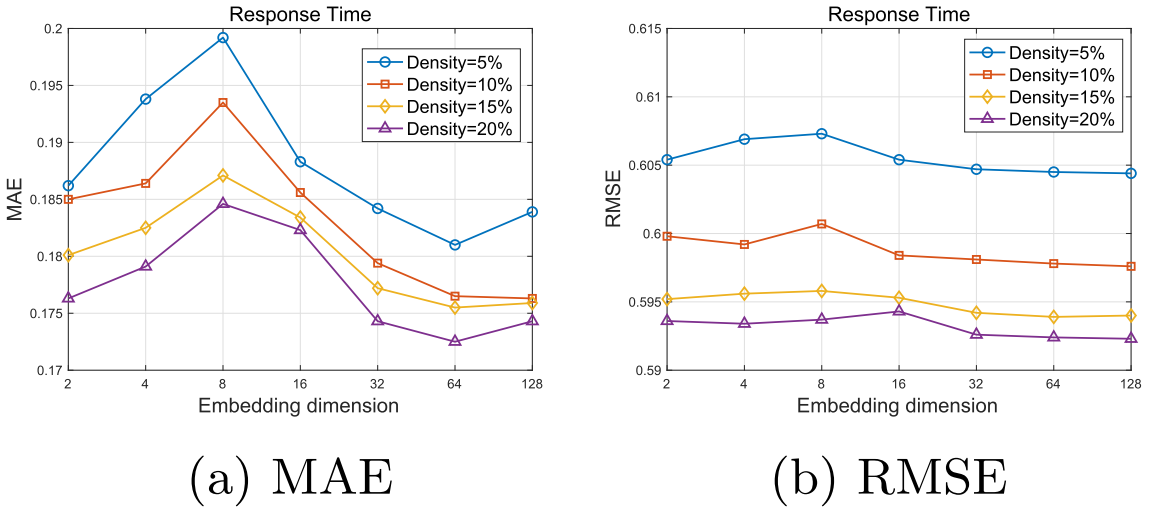


Fig. 6 The impact of embedding dimension d

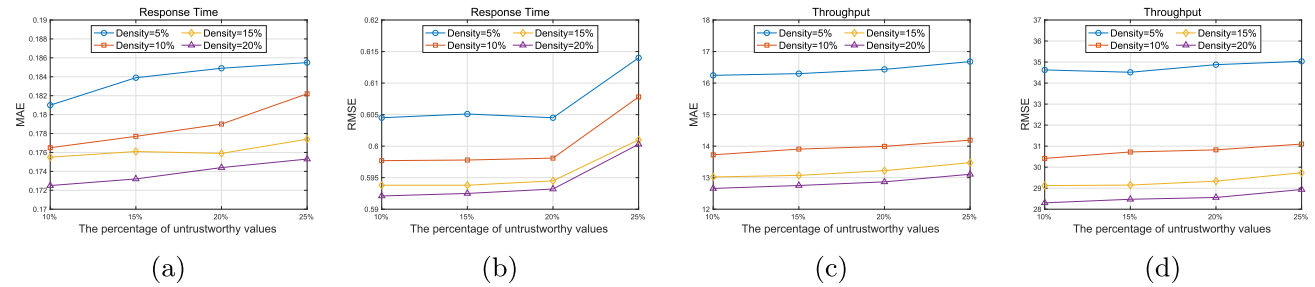


Fig. 7 The impact of the percentage of untrustworthy values

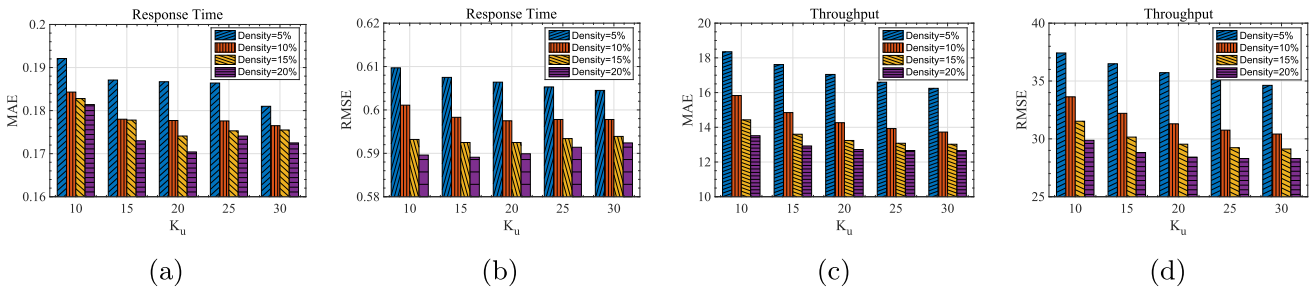


Fig. 8 Impact of K_u

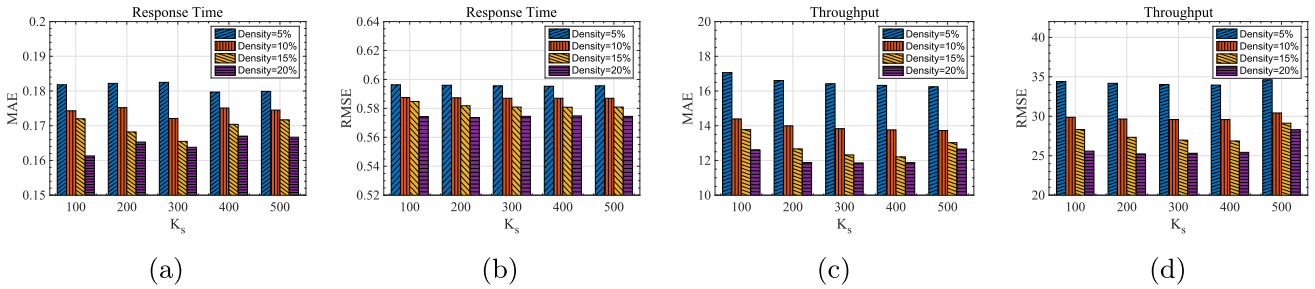


Fig. 9 Impact of K_s

in the number of learnable similar user embedding vectors, thereby offering more potential features and consequently contributing to improved prediction results. In Fig. 8a, the change in MAE is nearly negligible when the K_u value ranges between 15 and 20. When K_u=25, the MAE value is even slightly higher than that when K_u is 20. As shown in Fig. 9, with the increase of K_s, it is observed that the changes of MAE and RMSE at RT and TP are very small, especially the value of RMSE. Since the direct correlation between the model's prediction time and the values of K_u and K_s, we established K_u=30 and K_s=500 for the experimental.

To assess the feasibility of TPP-GNCF, we conducted two parts of experiments, comprising comparisons with existing methods and ablation experiments. Specific experiments and analyses are discussed further below.

Performance comparison and analysis (RQ2)

During the comparison experiments, the QoS invocation matrices of RT and TP were separated into four distinct densities, which are 5%, 10%, 15%, and 20%, respectively. All comparative experiments were performed on the training sets of RT and TP, and the MAE, RMSE, and NMAE were utilized to assess the performance of the prediction on the test set. The parameter settings for TPP-GNCF are presented in Table 1. To ensure fairness in comparing prediction results, we executed TPP-GNCF multiple times and calculated the average value for comparison.

Tables 2 and 3 exhibit the experimental results of MAE and RMSE for different schemes on RT and TP, respectively. It is well-known that in QoS prediction, lower values of MAE and RMSE imply enhanced predictive performance. Therefore, it is evident that TPP-GNCF outperforms other schemes significantly in both RT and TP. It is worth noting that, similar to the literature [48, 49], we directly use the results shown in PSO-USRec, FHC-DQP, NDMF, and CNCF. As illustrated in Table 2, UPCC and IPCC show poorly compared to other methods because these two schemes directly predict missing QoS values of users and services, respectively, without additional processing. UIPCC significantly improves QoS prediction performance by combining collaborative filtering methods for user and service neighborhoods. PMF is a basic method using matrix factorization, which is improved over the previous methods in QoS prediction. PSO-USRec uses a swarm intelligence search method for prediction, showing slightly improved performance over previous solutions. NRCF addresses the issue of personalized web service by introducing a novel similarity calculation and collaborative filtering method, thereby improving QoS prediction. NIMF utilizes a traditional neighborhood-incorporated matrix factorization method, showing slight improvement over PMF by incorporating neighborhoods in model training. FHC-DQP significantly outperforms previous methods, primarily improving prediction performance through federated learning and user clustering. NDMF improves the traditional matrix factorization QoS prediction scheme by extracting latent user and service features through nonlinear interaction functions. For IoT services, CNCF combines neural collaborative filtering and clustering algorithms to develop a QoS prediction scheme that considers contextual information, which greatly improves the performance. At the same time, combined with Table 3, we can see that the prediction performance of the PMF scheme on RT is relatively poor, but the prediction performance on TP is better. In addition, NMAE is introduced for comparison with existing solutions, as shown in Table 4, as it better reflects the relative accuracy of model prediction results. Regarding the NMAE metric, TPP-GNCF achieves at least a 48% improvement on RT and a 0.8% improvement on TP compared to other advanced solutions. It is evident that the proposed TPP-GNCF outperforms all comparative solutions in RT and TP. This improvement is attributed to the fact that TPP-GNCF introduces graph convolutional networks and trust-aware weighting mechanisms, which utilize the 3-σ rule to filter out outliers to select reliable QoS data, significantly improving data quality and effectively alleviating the challenges posed by sparse data. Furthermore, trust-aware collaborative prediction based on weight function enables the model to capture the trust relationship between users and services, and deeply mine latent features, thereby mitigating the poor performance of traditional methods on low-density datasets. This enables

Table 1 Parameter setting for performance comparison experiment

Parameter	Value	Meaning
ζ	0.8(RT), 0.2(TP)	Weight coefficient
K_u	30	The maximum number of similar user embedding
K_s	500	The maximum number of similar service embedding
d	64	Embedding dimension
α	0.1	The percentage of untrustworthy values
ϵ_1, ϵ_2	0.6, 0.9	Privacy budget
δ	0.0001	Slack term

Table 2 Comparison of QoS prediction performance among different methods in terms of Response Time

Method	Density=5%		Density=10%		Density=15%		Density=20%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	0.6353	1.3805	0.5528	1.3109	0.5131	1.2590	0.4856	1.2196
IPCC	0.6337	1.3979	0.5918	1.3430	0.5102	1.2588	0.4558	1.2050
UIPCC	0.6243	1.3896	0.5813	1.3293	0.5010	1.2510	0.4492	1.1966
PMF	0.5698	1.5355	0.4868	1.3179	0.4528	1.2206	0.4328	1.1712
NRCF	0.5536	1.4378	0.4897	1.3525	0.4544	1.2931	0.4289	1.2430
NIMF	0.5588	1.4826	0.4916	1.3109	0.4569	1.2299	0.4357	1.1844
PSO-USRec	0.5655	1.3580	0.5064	1.2742	0.4706	1.2221	0.4440	1.1813
FHC-DQP	0.5100	1.4000	0.4340	1.3160	0.3950	1.2360	0.3380	1.2050
NDMF	0.4880	1.3495	0.4304	1.2349	0.3845	1.1569	0.3665	1.1294
CNCF	**0.3690**	**0.6720**	**0.3380**	**0.6210**	**0.3290**	**0.6060**	**0.3140**	**0.5920**
TPP-GNCF-ϵ_1	0.3101	0.6457	0.3022	0.6380	0.2984	0.6363	0.2978	0.6357
TPP-GNCF-ϵ_2	0.2787	0.6237	0.2785	0.6176	0.2744	0.6157	0.2696	0.6151
TPP-GNCF	**0.1776**	**0.5946**	**0.1737**	**0.5839**	**0.1668**	**0.5759**	**0.1633**	**0.5690**
Gains	51.87%	11.52%	48.61%	5.97%	49.30%	4.97%	47.99%	3.89%

The best values of the baseline methods and the TPP-GNCF model are highlighted in bold

Table 3 Comparison of QoS prediction performance among different methods in terms of Throughput

Method	Density=5%		Density=10%		Density=15%		Density=20%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	28.5879	62.5354	26.4949	59.9933	23.5350	56.4801	21.5375	53.6545
IPCC	27.0193	62.3098	26.2085	58.7562	23.7279	56.1661	22.9322	54.1675
UIPCC	26.5895	62.2850	25.3856	58.5963	23.0015	56.0079	21.0526	53.0957
PMF	19.0454	57.4000	16.0610	48.0965	14.9004	44.1907	14.1076	41.9384
NRCF	23.5318	60.1241	18.9637	53.0881	16.1386	48.1330	14.4772	44.6851
NIMF	23.4704	60.6047	18.2285	50.4683	16.0876	46.8689	14.5792	42.6737
PSO-USRec	23.3323	60.1550	19.7397	54.2544	17.8389	50.2085	16.7872	47.3949
FHC-DQP	17.2660	51.5200	14.3470	46.6050	13.4950	44.9160	12.6380	42.1970
NDMF	**16.3818**	50.9612	**13.9317**	43.9095	**12.5043**	42.5319	**11.7204**	39.9431
CNCF	21.0270	**38.6710**	18.1890	**33.5740**	17.3730	**32.3760**	16.8260	**31.4640**
TPP-GNCF-ϵ_1	17.2536	38.8207	16.1393	36.6113	15.7245	35.7555	15.5176	35.3650
TPP-GNCF-ϵ_2	16.8593	36.7455	15.0744	33.8160	14.3979	32.5898	14.0558	31.9353
TPP-GNCF	**16.2451**	**34.1611**	**13.7812**	**29.7082**	**12.1670**	**26.8150**	**11.8372**	**25.4090**
Gains	0.83%	11.66%	1.08%	11.51%	2.70%	17.18%	−0.99%	19.24%

The best values of the baseline methods and the TPP-GNCF model are highlighted in bold

Table 4 Comparison of NMAE values for QoS prediction across different methods in terms of Response Time (RT) and Throughput (TP)

Method	Density=5%		Density=10%		Density=15%		Density=20%	
	RT	TP	RT	TP	RT	TP	RT	TP
UPCC	0.03177	0.02858	0.02764	0.02650	0.02566	0.02355	0.02428	0.02155
IPCC	0.03169	0.02702	0.02959	0.02621	0.02551	0.02373	0.02279	0.02293
UIPCC	0.03122	0.02659	0.02907	0.02539	0.02505	0.02300	0.02246	0.02105
PMF	0.02849	0.01905	0.02434	0.01606	0.02264	0.01490	0.02164	0.01411
NRCF	0.02769	0.02351	0.02450	0.01898	0.02272	0.01612	0.02143	0.01449
NIMF	0.02794	0.02347	0.02458	0.01823	0.02285	0.01609	0.02179	0.01458
PSO-USRec	0.02828	0.02333	0.02532	0.01974	0.02353	0.01784	0.02220	0.01679
FHC-DQP	0.02550	0.01727	0.02170	0.01435	0.01975	0.01350	0.01690	0.01264
NDMF	0.02440	**0.01638**	0.02152	**0.01393**	0.01923	**0.01250**	0.01833	**0.01172**
CNCF	**0.01845**	0.02103	**0.01690**	0.01819	**0.01645**	0.01737	**0.01570**	0.01683
TPP-GNCF-ϵ_1	0.01551	0.01725	0.01511	0.01614	0.01492	0.01572	0.01489	0.01552
TPP-GNCF-ϵ_2	0.01394	0.01686	0.01393	0.01507	0.01372	0.01440	0.01348	0.01406
TPP-GNCF	**0.00888**	**0.01625**	**0.00869**	**0.01378**	**0.00834**	**0.01217**	**0.00817**	**0.01184**
Gains	51.9%	0.8%	48.6%	1.1%	49.3%	2.7%	48.0%	−1.0%

The best values of the baseline methods and the TPP-GNCF model are highlighted in bold

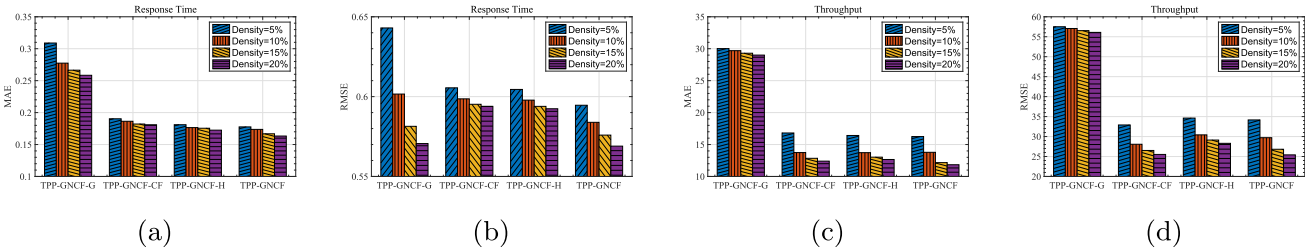


Fig. 10 Ablation experiment

TPP-GNCF to handle sparse data more robustly and achieve better prediction performance.

To analyze the QoS prediction performance of TPP-GNCF in terms of privacy protection, we selected privacy budgets of $\epsilon_1 = 0.6$ and $\epsilon_2 = 0.9$, along with a slack term $\delta = 0.0001$ for the comparative experiments. Obviously, after introducing the privacy budget, the prediction performance of TPP-GNCF declined compared to the original. Specifically, compared to TPP-GNCF, TPP-GNCF-ϵ_1 and TPP-GNCF-ϵ_2 significantly reduce the MAE by about 39% and 45.16%, and the RMSE by 7.5% and 10.49% on the RT, respectively. Similarly, on the TP dataset, TPP-GNCF-ϵ_1 and TPP-GNCF-ϵ_2 showed significant reductions in MAE of about 23.72% and 15.78%, and RMSE of 28.15% and 20.43%, respectively. Moreover, TPP-GNCF-ϵ_1 also outperforms the other competing schemes by improving the MAE and RMSE by 15.96% and 3.91%, respectively, across the four matrix densities in the RT dataset. TPP-GNCF also slightly improves the MAE and RMSE compared to other competing schemes by 2.7% and 19.24% on the TP, respectively. However, it is worth noting that across the four matrix densities of the RT dataset, although the values of both MAE and RMSE gradually decrease, the differences in the decreases between each matrix density are relatively small. The above trend is also reflected in the NMAE metric. TPP-GNCF-ϵ_1 and TPP-GNCF-ϵ_2 both show decreases in performance on RT and TP datasets compared to TPP-GNCF.

Ablation experiments and analysis (RQ3)

In this section, we conduct ablation experiments to evaluate the benefits of the fusion model by considering three variants of TPP-GNCF: TPP-GNCF-CF, TPP-GNCF-G, and TPP-GNCF-H. TPP-GNCF-CF represents that the scheme does not consider the impact of collaborative filtering; TPP-GNCF-G indicates that the scheme does not consider the impact of GNN; TPP-GNCF-H indicates that the scheme does not utilize the improved PCC.

The MAE and RMSE results for RT and TP of TPP-GNCF and its three variants at various matrix densities are illustrated in Fig. 10. For RT, we found that the performance of TPP-GNCF-G was the worst compared to the other three variants, following the trend: TPP-GNCF-G > TPP-GNCF-CF > TPP-GNCF-H > TPP-GNCF. In other words,

TPP-GNCF exhibited the best prediction performance. For TP, TPP-GNCF-CF showed the worst performance, with the trend: TPP-GNCF-CF > TPP-GNCF-G > TPP-GNCF-H > TPP-GNCF. However, the prediction performance of all three variants is inferior to that of the fusion model, TPP-GNCF. This also illustrates the absolute advantage of the improved fusion model presented in this paper. In summary, TPP-GNCF demonstrates the best prediction performance for both RT and TP, proving that the improved fusion model significantly enhances accuracy and reliability across various matrix densities.

Conclusion

To explore the balance issue between user privacy, data credibility, and QoS prediction performance in IoT services, in this work, we design a trust-aware privacy-preserving QoS prediction framework based on the fusion model, named TPP-GNCF. First, we filter out untrustworthy QoS values that may affect the model's understanding and prediction of user behavior, which helps improve QoS prediction accuracy. Then, we utilize message-passing graph neural networks to effectively capture information passing and relationships in the graph structure and differential privacy is employed to protect the information of user nodes. Finally, the missing QoS value prediction is achieved by fusing a graph neural network to predict QoS values and collaborative filtering to predict QoS. Experimental results demonstrate that TPP-GNCF is superior to the comparative solutions, achieving improvements of at least 47.99% and 3.89% in MAE and RMSE on RT, and at least 0.83% and 11.51% on TP, respectively. Additionally, for the NMAE metric, our scheme also demonstrates at least a 48% and 0.8% improvement on RT and TP, respectively.

In future work, we intend to further expand the framework to improve its applicability to larger-scale datasets and a wider variety of IoT services. Moreover, we will explore methods for integrating additional contextual information, including user preferences and dynamic changes in service, into the model to improve the precision of QoS prediction. Simultaneously, we will also explore more efficient privacy protection technologies to further reduce computational complexity and resource consumption while ensuring user privacy.

Author Contributions WW: conceptualization, methodology, software, writing—original draft, writing—review & editing. WM: funding acquisition, supervision. KY: software, validation.

Funding This work was supported by the Key Industry Innovation Chain Project of Shaanxi Provincial Science and Technology Department, China, under Grant 2022ZDLGY03-08.

Data availability The authors confirm that the data supporting the findings in this article are available at the public links https://wsdream.github.io/#dataset.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

References

1. Liu J, Chen Y (2019) A personalized clustering-based and reliable trust-aware qos prediction approach for cloud service recommendation in cloud manufacturing. Knowl-Based Syst 174:43–56
2. Li L, Li S, Zhao S (2014) Qos-aware scheduling of services-oriented internet of things. IEEE Trans Ind Inform 10(2):1497–1505
3. Shu Z, Taleb T (2020) A novel qos framework for network slicing in 5g and beyond networks based on sdn and nfv. IEEE Netw 34(3):256–263
4. Zheng Z, Li X, Tang M, Xie F, Lyu MR (2020) Web service qos prediction via collaborative filtering: a survey. IEEE Trans Serv Comput 15(4):2455–2472
5. Ghafouri SH, Hashemi SM, Hung PC (2020) A survey on web service qos prediction methods. IEEE Trans Serv Comput 15(4):2439–2454
6. Aggarwal A (2024) Artificial Intelligence in Radiation Oncology, pp. 135–150 . https://doi.org/10.1201/9781003450153-11
7. Kumar A, Sato Y, Oishi T, Ono S, Ikeuchi K (2014) Improving gps position accuracy by identification of reflected gps signals using range data for modeling of urban structures. Seisan Kenkyu 66(2):101–107
8. Kumar A, Banno A, Ono S, Oishi T, Ikeuchi K (2013) Global coordinate adjustment of the 3d survey models under unstable gps condition. Seisan Kenkyu 65(2):91–95
9. KUMAR A (2020)Atmospheric delay correction of rinex gps data . TEST Engineering & Management 83:24877–24882
10. Zou G, Chen J, He Q, Li K-C, Zhang B, Gan Y (2020) Ndmf: Neighborhood-integrated deep matrix factorization for service qos prediction. IEEE Trans Netw Service Manag 17(4):2717–2730
11. Shi L-L, Liu L, Jiang L, Zhu R, Panneerselvam J (2022) Qos prediction for smart service management and recommendation based on the location of mobile users. Neurocomputing 471:12–20
12. Wu X, Cheng B, Chen J (2015) Collaborative filtering service recommendation based on a novel similarity computation method. IEEE Trans Serv Comput 10(3):352–365

13. Ajaegbu C (2021) An optimized item-based collaborative filtering algorithm. J Ambient Intell Hum Comput 12(12):10629–10636
14. Cui Z, Xu X, Fei X, Cai X, Cao Y, Zhang W, Chen J (2020) Personalized recommendation system based on collaborative filtering for iot scenarios. IEEE Trans Serv Comput 13(4):685–695
15. Hussain W, Merigó JM, Raza MR, Gao H (2022) A new qos prediction model using hybrid iowa-anfis with fuzzy c-means, subtractive clustering and grid partitioning. Inform Sci 584:280–300
16. Su K, Xiao B, Liu B, Zhang H, Zhang Z (2017) Tap: A personalized trust-aware qos prediction approach for web service recommendation. Knowl-Based Syst 115:55–65
17. Zhu J, He P, Zheng Z, Lyu MR (2017) Online qos prediction for runtime service adaptation via adaptive matrix factorization. IEEE Trans Parallel Distributed Syst 28(10):2911–2924
18. Wu H, Yue K, Li B, Zhang B, Hsu C-H (2018) Collaborative qos prediction with context-sensitive matrix factorization. Future Generation Comput Syst 82:669–678
19. Zheng Z, Ma H, Lyu MR, King I (2012) Collaborative web service qos prediction via neighborhood integrated matrix factorization. IEEE Trans Serv Comput 6(3):289–299
20. Lo W, Yin J, Li Y, Wu Z (2015) Efficient web service qos prediction using local neighborhood matrix factorization. Eng Appl Artificial Intell 38:14–23
21. Ding L, Liu J, Kang G, Xiao Y, Cao B (2023) Joint qos prediction for web services based on deep fusion of features. IEEE Transactions on Network and Service Management
22. Lv S, Yi F, He P, Zeng C (2021) Qos prediction of web services based on a two-level heterogeneous graph attention network. IEEE Access 10:1871–1880
23. Tang M, Tang W, Xie F (2023) Accurately predicting quality of services in iot via using self-attention representation and deep factorization machines. IEEE Transactions on Intelligent Transportation Systems
24. Zou G, Li T, Jiang M, Hu S, Cao C, Zhang B, Gan Y, Chen Y (2022) Deeptsqp: Temporal-aware service qos prediction via deep neural network and feature integration. Knowl-Based Syst 241:108062
25. Zou G, Wu S, Hu S, Cao C, Gan Y, Zhang B, Chen Y (2022) Ncrl: Neighborhood-based collaborative residual learning for adaptive qos prediction. IEEE Transactions on Services Computing
26. Zhang P, Huang W, Chen Y, Zhou M (2023) Predicting quality of services based on a two-stream deep learning model with user and service graphs. IEEE Transactions on Services Computing
27. Wu Z, Ding D, Xiu Y, Zhao Y, Hong J (2023) Robust qos prediction based on reputation integrated graph convolution network. IEEE Transactions on Services Computing
28. Liu M, Xu H, Sheng Q.Z, Wang Z (2023) Qosgnn: Boosting qos prediction performance with graph neural networks. IEEE Transactions on Services Computing
29. Zheng Z, Ma H, Lyu MR, King I (2010) Qos-aware web service recommendation by collaborative filtering. IEEE Trans Serv Comput 4(2):140–152
30. Shao L, Zhang J, Wei Y, Zhao J, Xie B, Mei H (2007) Personalized qos prediction forweb services via collaborative filtering. In: Ieee International Conference on Web Services (icws 2007), pp. 439–446 . IEEE
31. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295
32. Mnih A, Salakhutdinov R.R (2007) Probabilistic matrix factorization. Advances in neural information processing systems **20**
33. Chang Z, Ding D, Xia Y (2021) A graph-based qos prediction approach for web service recommendation. Applied Intelligence, 1–15
34. Gao H, Xu Y, Yin Y, Zhang W, Li R, Wang X (2019) Context-aware qos prediction with neural collaborative filtering for internet-of-things services. IEEE Internet Things J 7(5):4532–4542
35. Zhang Y, Yin C, Wu Q, He Q, Zhu H (2019) Location-aware deep collaborative filtering for service recommendation. IEEE Trans Syst Man Cybernet 51(6):3796–3807
36. Li J, Wu H, Chen J, He Q, Hsu C-H (2021) Topology-aware neural model for highly accurate qos prediction. IEEE Trans Parallel Distributed Syst 33(7):1538–1552
37. Zhu J, He P, Zheng Z, Lyu M.R (2015) A privacy-preserving qos prediction framework for web service recommendation. In: 2015 IEEE International Conference on Web Services, pp. 241–248 . IEEE
38. Liu A, Shen X, Li Z, Liu G, Xu J, Zhao L, Zheng K, Shang S (2019) Differential private collaborative web services qos prediction. World Wide Web 22:2697–2720
39. Liu A, Shen X, Xie H, Li Z, Liu G, Xu J, Zhao L, Wang FL (2020) Privacy-preserving shared collaborative web services qos prediction. J Intell Inform Syst 54:205–224
40. Zhang P, Jin H, Dong H, Song W, Bouguettaya A (2020) Privacy-preserving qos forecasting in mobile edge environments. IEEE Trans Serv Comput 15(2):1103–1117
41. Dwork C, Roth A, *et al.* (2014) The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science **9**(3–4), 211–407
42. Xie B, Hu C, Huang H, Yu J, Xia H (2023) Dci-pfgl: Decentralized cross-institutional personalized federated graph learning for iot service recommendation. IEEE Internet of Things Journal
43. Gilmer J, Schoenholz S.S, Riley P.F, Vinyals O, Dahl G.E (2017) Neural message passing for quantum chemistry. In: International Conference on Machine Learning, pp. 1263–1272 . PMLR
44. Liu J, Chen Y (2019) Hap: a hybrid qos prediction approach in cloud manufacturing combining local collaborative filtering and global case-based reasoning. IEEE Trans Serv Comput 14(6):1796–1808
45. Yin J, Lo W, Deng S, Li Y, Wu Z, Xiong N (2014) Colbar: A collaborative location-based regularization framework for qos prediction. Inform Sci 265:68–84
46. Sun H, Zheng Z, Chen J, Lyu MR (2012) Personalized web service recommendation via normal recovery collaborative filtering. IEEE Trans Serv Computi 6(4):573–579
47. Chen J, Mao C, Song WW (2023) Qos prediction for web services in cloud environments based on swarm intelligence search. Knowl-Based Syst 259:110081
48. Zou G, Lin S, Hu S, Duan S, Gan Y, Zhang B, Chen Y (2023) Fhc-dqp: Federated hierarchical clustering for distributed qos prediction. IEEE Transactions on Services Computing
49. Zhang P, Ren J, Huang W, Chen Y, Zhao Q, Zhu H (2023) A deep-learning model for service qos prediction based on feature mapping and inference. IEEE Transactions on Services Computing