Quality of Service Prediction via Large Language Models

Huiying Liu
School of Computer Science and Technology
Anhui University
Hefei, Anhui, China.
liuhuiying.ahu@hotmail.com

Zekun Zhang
Department of Computer Science
Stony Brook University
Stony Brook, New York, USA.
zekzhang@cs.stonybrook.edu

Lei Sang
School of Computer Science and Technology
Anhui University
Hefei, Anhui, China.
sanglei@ahu.edu.cn

Qilin Wu
School of Information Engineering
Chaohu University
Hefei, Anhui, China.
qlw@chu.edu.cn

Yiwen Zhang
School of Computer Science and Technology
Anhui University
Hefei, Anhui, China.
zhangyiwen@ahu.edu.cn

***Abstract*—Large language models (LLMs) have been the center of many research areas in recent years, and have a wide range of application scenarios from text understanding and content creation, to data analysis and human-computer interaction. After being trained on trillions of text tokens, LLMs exhibit excellent generalization abilities, and can work on various tasks with minimum alteration or finetuning. Such capability is potentially useful for the task of service recommendation, where the sparsity of training data has hindered the performance of many existing algorithms significantly. In this paper, we explore the possibility of using LLMs to predict the quality of service (QoS) value directly for a pair of web user and service. We propose the Large Language Model for Service Recommendation (LLMSRec), a framework that treats each web user and service as a descriptive natural language sentence, and learns to understand the QoS-related features of them from limited historical invocation data. On the WSDream dataset, LLMSRec can make accurate predictions under high sparsity levels, and outperforms QoS prediction baseline methods consistently.**

***Index Terms*—Service Recommendation, Quality of Service Prediction, Large Language Model.**

I. Introduction

The number of web services has been growing exponentially, causing the proliferation of services offering identical or highly similar functionalities [1], [2]. Driven by the widespread adoption of Service-Oriented Architecture and cloud computing technologies [3], this has led to an increasingly complex service ecosystem. While such abundance provides users with a wide range of options, it also complicates the task of recommending the most suitable service to specific users. This is further exacerbated by the dynamic nature of web services, where factors such as network conditions, geographical location, and user-specific contexts significantly influence the Quality of Service (QoS), which is the critical metric for evaluating and differentiating among these services. QoS encompasses non-functional attributes [4] such as response time and throughput, which are pivotal in determining the performance and suitability of a service in meeting user needs.

Despite the critical role of QoS in service recommendation, accurately predicting QoS values remains a significant challenge due to the data sparsity issue which is inherent to the task. The recommendation application needs to make accurate QoS prediction based on historical user-service interaction matrix, which is often highly sparse in real-world scenarios, as users typically invoke only a small subset of available services. Collecting more data points is impractical since it costs enormous time and resources to directly make invocation for users [5]. Collaborative filtering [6] is arguably the most widely used technique for QoS prediction. The data sparsity issue can significantly hamper its effectiveness in making accurate predictions. While many methods have been developed to address this challenge, they still fall short of achieving optimal performance in service-oriented applications.

Recently, large language models (LLMs) [7], [8] have shown various capabilities in different natural language processing tasks. LLMs are typically trained on vast amounts of data, and can achieve impressive performance even under the low-data constraint at testing time [9]–[13]. Indeed, LLMs are increasingly utilized in the broader field of recommendation systems, where their application has led to notable improvements in recommendation quality and user experience [14], [15]. This indicates that pretrained LLMs can be utilized to address the data-sparsity issue of QoS prediction. We claim that the excellent natural language processing ability of LLMs can be harnessed by converting the different attributes associated with web users and services into textual representations. In this paper, we propose the **L**arge **L**anguage **M**odel for **S**ervice **Rec**ommendation (LLMSRec) approach. LLMSRec treats each user and each service as a descriptive natural

This work is supported by the National Natural Science Foundation of China (No.62272001) and the Opening Foundation of State Key Laboratory of Cognitive Intelligence, iFLYTEK (COGOS-2023HE01).

language sentence, and uses LLMs to make QoS prediction. Specifically, for each historical invocation, we combine the descriptive sentences of the corresponding user and service, incorporating special QoS-specific tokens, forming the input for the LLM. By incorporating attributes of both the user and service, the LLM learns to understand their QoS-related features. We claim that LLMs are less prone to fail on the data sparsity issue for their great few-shot learning capabilities. We summarize the main contributions of this paper as follows:

- We propose the LLMSRec framework, which leverages the capabilities of LLMs to predict quality of service values for service recommendation. It eliminates the need for structured data input, and provides a more flexible and adaptable solution. Additionally, LLMSRec directly identifies patterns and relationships between users and services, bypassing traditional feature extraction algorithms and similarity-based computations.
- Our framework treats each web user and service as a natural language sentence, and enriches them with specially designed QoS-related tokens. This leverages LLM's strong text processing capabilities and enables the LLM to utilize its inherent ability to understand and process related information. Our approach improves prediction accuracy, reduces computational complexity, and enhances the overall efficiency of the model.
- LLMSRec achieves remarkable accuracy for throughput and response time prediction on the WSDream dataset under different density levels. This highlights the robustness and generalization ability of our method in handling real-world service recommendation tasks.

II. Related Work

In this paper we propose to use LLM for QoS prediction for the service recommendation task. In this section, we briefly introduce the state of the art in both QoS prediction and LLM research.

A. *QoS Prediction for Service Recommendation*

Quality of Service (QoS) prediction plays a pivotal role in service recommendation systems, especially when it comes to delivering high-quality and personalized recommendations. Collaborative Filtering (CF) [6] is one of the most widely used traditional methods for QoS prediction. Approaches based on CF can be further categorized into memory-based methods and model-based methods.

Memory-based methods in QoS prediction focus on utilizing historical user-item interactions to estimate the preferences of users for unseen services. These methods operate by measuring the similarity between users or items based on their past behaviors or characteristics. Zheng *et al.* [16] propose a collaborative filtering-based approach for predicting the QoS values of Web services and making service recommendations. This method integrates both user and service similarities, utilizing past usage data through a user-collaborative mechanism to gather QoS information. Hu *et al.* [17] propose a time-aware collaborative filtering approach for web service recommendation. Their method integrates time information into both similarity measurement and QoS prediction, considering its impact on web service performance.

On the other hand, model-based approaches for QoS prediction focus on capturing the underlying patterns in user-service interactions by learning predictive models from historical data. These methods often involve techniques such as matrix factorization [18]–[20] and deep neural networks [21]–[23]. Xu *et al.* [24] introduce a method that incorporates users' geographical information into QoS prediction for Web service recommendations. It employs probabilistic matrix factorization to predict QoS values and integrates geographical-based user neighbor feature vectors into the model's learning process. Recently, deep neural network based methods [25]–[27] have gained significant attention in the field of QoS prediction due to their ability to automatically extract complex features and model non-linear relationships within large datasets. Zou *et al.* [28] introduce the neighborhood-based collaborative residual learning framework, which combines a location-aware deep residual network and collaborative relationships to improve QoS prediction accuracy by extracting latent features from users and services.

Our proposed LLMSRec method can also be categorized as a model-based method using deep neural networks. Compared to other methods, it does not train the network from scratch on QoS data. Instead, it re-purposes a pretrained LLM network to the QoS prediction task. And unlike many other model-based methods, which design specific network architectures to work with highly structured input data such as matrix or graph, LLMSRec uses unstructured textual input, which gives it more flexibility.

B. *Large Language Model for Recommendation*

Recently, LLMs have gained significant attention for their excellent performance in various nature language processing tasks such as machine translation, text summarization, sentiment analysis, and dialogue systems [7], [8], [29]. LLMs have also been found to have the capability for making recommendations [14], [15], and are starting to play a more important role in recommendation systems. For instance, RLMRec [30] uses LLMs for user/item profiling and semantic alignment. Similarly, AlphaRec [31] shows that item representations derived from LLMs can outperform traditional collaborative filtering models, highlighting LLMs' potential to improve recommendation accuracy by capturing implicit user preferences.

Despite the success of LLMs in various recommendation tasks, their application in QoS prediction for service recommendation remains largely under-explored. Our proposed LLMSRec framework integrates the strengths of LLMs into QoS prediction for web service recommendation. As QoS prediction is not a nature language processing task, it is necessary to refactor QoS data into a suitable prompt format that LLMs can interpret. In LLMSRec, we generate descriptive sentences from several useful attributes of web users and services, which are then used to construct the prompt and fed to the LLMs for predicting QoS values.

III. Methods

In this section, we layout the pipeline of the proposed LLM-SRec framework, including constructing descriptive sentences for web services and users from their corresponding attributes, inserting special QoS-related tokens to construct the prompts, and training details. The overall architecture is illustrated in Figure I.

A. Descriptive Sentence Construction

Each web user and service in the WSDream dataset has some corresponding attributes, such as their location, network provider, IP address, and autonomous system. Those attributes are useful for QoS prediction. For instance, a web service and a web user that are physically close to each other should have lower latency thus potentially higher throughput and lower response time. However, we argue that not all attributes are needed. For example, the value of IP addresses does not necessarily represent any locational or networking information, and general-purpose LLMs are known to perform poorly on numerical inputs [33], [34]. As a result, we utilize the ID, country of location (Country), and the autonomous system (AS) for web users, and the ID, country of location (Country), URL address (URL), network provider (NP), and autonomous system (AS) for web services.

From those attributes, we construct descriptive sentences to represent the web services and users. Since the LLMs are trained on corpus that mostly consists of natural language sentences. To match this, we also use sentences that are natural. Specifically, we use the template "`web user number _ID_, located in _Country_, in autonomous system _AS_.`" for web users, and "`web service number _ID_, at url _URL_, hosted by _NP_, in autonomous system _AS_.`" for services. If a certain attribute is missing in the dataset, we will not include the corresponding sub-sentence in the description. Pairs of descriptive sentences from a web user and a web service are used to make the input prompt for our LLM-based QoS prediction model.

B. Special Tokens and Input Prompt

Trained LLMs are expected to extract useful information from the descriptive sentences. However, they are not trained under the context of QoS prediction. To better guide the model, we use special tokens to wrap the descriptive sentences to get the input prompt. Specifically, we insert the start-of-service token `<sos>` and end-of-service token `<eos>` around the descriptive sentence of the service, and start-of-user token `<sou>` and end-of-user token `<eou>` around the descriptive sentence of the user. Then the two wrapped sentences are concatenated to get the input prompt.

Since the `<eou>` token is the last token in the prompt, we also give it the purpose of QoS prediction, thus renaming it to `<eou_qos>`. In other words, the corresponding feature vector of the last token is used for generating the predicted QoS value. An example of construction of the descriptive sentences and the prompt is shown in Figure I.

C. Model Architecture

We apply a pretrained LLM and apply task-specific modifications for QoS prediction. The model architecture is illustrated in Figure I. Please note that our proposed method can work with any pretrained LLM with different architectures. Most of our experiments are based on the Qwen2.5-0.5B model [32], of which the architecture is shown in the figure. Qwen2.5-0.5B is the smallest model in the state-of-the-art Qwen2.5 family [32], but we argue that it is still capable enough for the QoS prediction task. We conduct additional experiments with larger LLMs in §IV-F. To adopt the LLM to the task, the special tokens introduced in §III-B are added to the tokenizer, and the corresponding embedding vectors are added and randomly initialized. Then we replace the head of the LLM, which maps feature vectors to the probability distribution on a vocabulary, with a linear layer that maps the feature vectors to a scalar QoS value.

Without loss of generality, the input prompt is first converted to a sequence of N tokens $\{t_1, t_2, \cdots, t_N\}$ by the tokenizer. Please note that t_N is the `<eou_qos>` token. Then it is mapped to N feature vectors $\mathbf{X}^1 = \{\mathbf{x}_1^1, \mathbf{x}_2^1, \cdots, \mathbf{x}_N^1\}$ by the embedding layer. The sequence of vectors is processed by a series of L transformer decoder blocks [29]. In each block, the feature vectors are first mapped to the query, key, and value vectors by corresponding projection layers q-proj, k-proj, and v-proj

$$\mathbf{Q}^l = \mathbf{W}_q^l \mathbf{X}^l + \mathbf{b}_q^l, \tag{1}$$
$$\mathbf{K}^l = \mathbf{W}_k^l \mathbf{X}^l + \mathbf{b}_k^l, \tag{2}$$
$$\mathbf{V}^l = \mathbf{W}_v^l \mathbf{X}^l + \mathbf{b}_v^l, \tag{3}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the corresponding weight and bias of the projection layers in the l-th block. Positional embeddings $\mathbf{E}$ are added to the query and key vectors, and all vectors are processed by a multi-head attention module (MHA)

$$\mathbf{Z}^l = MHA(\mathbf{Q}^l + \mathbf{E}, \mathbf{K}^l + \mathbf{E}, \mathbf{V}^l), \tag{4}$$

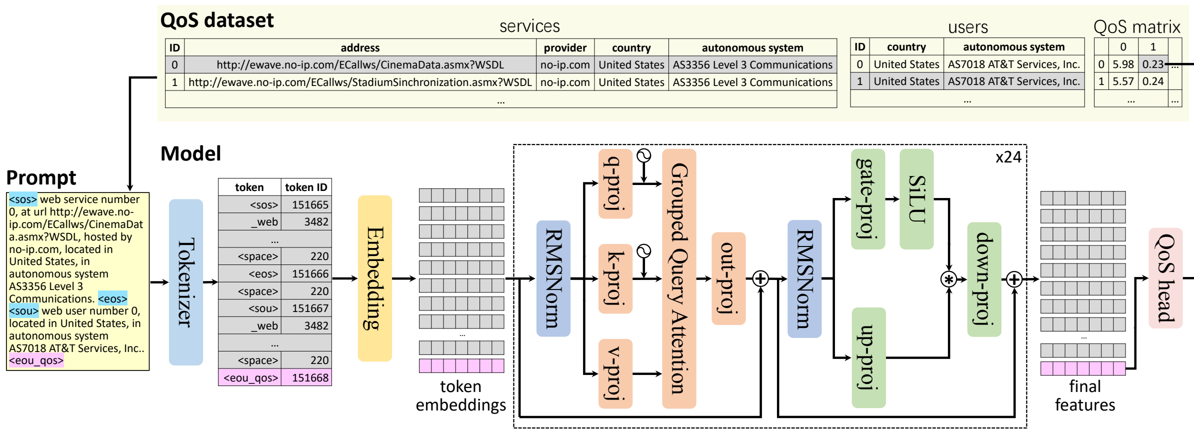
and followed by an output project layer o-proj

$$\mathbf{O}^l = \mathbf{W}_o^l \mathbf{Z}^l + \mathbf{b}_o^l. \tag{5}$$

Please note that for Qwen2.5-0.5B specifically, grouped query attention (GQA), which is a variant of ordinary multi-head attention, is used for more efficient computation. After the projection, the feature vectors are mapped to the output feature vectors by a feed-forward network (FFN), which is essentially a multi-layer perceptron network

$$\mathbf{X}^{l+1} = FFN^l(\mathbf{O}^l), \tag{6}$$

and $\mathbf{X}^{l+1}$ is the input of the $(l+1)$-th block. Normalization layers and residual connections are also added to each transformer decoder block, and their exact implementation differs by different LLM architectures.


QoS dataset

services

ID	address	provider	country	autonomous system
0	http://ewave.no-ip.com/ECallws/CinemaData.asmx?WSDL	no-ip.com	United States	AS3356 Level 3 Communications
1	http://ewave.no-ip.com/ECallws/StadiumSinchronization.asmx?WSDL	no-ip.com	United States	AS3356 Level 3 Communications
...				

users

ID	country	autonomous system
0	United States	AS7018 AT&T Services, Inc.
1	United States	AS7018 AT&T Services, Inc.
...		

QoS matrix

	0	1	
0	5.98	0.23	...
1	5.57	0.24	
	...		...

Prompt

<sos> web service number 0, at url http://ewave.no-ip.com/ECallws/CinemaData.asmx?WSDL, hosted by no-ip.com, located in United States, in autonomous system AS3356 Level 3 Communications. <eos> <sou> web user number 0, located in United States, in autonomous system AS7018 AT&T Services, Inc.. <eou_qos>

token	token ID
<sos>	151665
_web	3482
...	
<space>	220
<eos>	151666
<space>	220
<sou>	151667
_web	3482
...	
<space>	220
<eou_qos>	151668

FIGURE I: Architecture and training pipeline of the LLMSRec model. Please note that although the proposed method can theoretically work with any pretrained LLM, we illustrate the architecture with Qwn2.5-0.5B [32], which is used in most of the experiments. For a pair of web service and web user, descriptive sentences are constructed using their corresponding attributes, from which the input prompt is composed by inserting special tokens for QoS prediction. The prompt is processed by the LLM, and the predicted QoS value is calculated by the QoS head.

After the last transformer decoder block, the feature vector that corresponds to the `<eou_qos>` token is mapped to a scalar QoS prediction value $\hat{r}$ by the QoS head

$$\hat{r} = \mathbf{W}_{qos}\mathbf{x}_N^{L+1} + \mathbf{b}_{qos}. \tag{7}$$

Since in each block, the information from different input tokens is aggregated by the attention layers, the feature vector of `<eou_qos>` has the global contextual information of the descriptive sentences of both the web service and user, thus can be used for accurate QoS prediction.

D. *Model Training*

The model is trained by minimizing the L_1 loss between the predicted QoS value $\hat{r}$ and the ground truth QoS value r

$$\mathcal{L} = |\hat{r} - r|. \tag{8}$$

In the experiments, relatively small-scale LLMs such as Qwen2.5-0.5B are used. However, training all their parameters still requires a substantial amount of memory, considering all the intermediate features and optimizer states. Thus we apply low-rank adapters (LoRA) [35] to the model to reduce memory footprint at training time. LoRA is applied to most of the projection layers in each transformer decoder block, including the output projection layer after the attention module, and the projection layers in the feed-forward network. The query, key, and value projection layers, the token embedding layer, and the QoS head are still trained with all parameters.

IV. Experiments

In this section, we present the details of the WSDream dataset and on how the LLMSRec model is trained. The performance of LLMSRec is compared to various baseline methods. The effect of LoRA training and the size of LLMs on LLMSRec is also tested and discussed.

TABLE I: Statistics of the WSDream dataset. We show the number of web users and services, the number of unique values of attributes used by LLMSRec, and the number of interactions in the throughput and response time subsets.

Dataset Attribute	Unique Values
User	339
Service	5,825
Country (User)	31
Autonomous System (User)	137
Country (Service)	74
Autonomous System (Service)	993
URL Address (Service)	5,825
Network Provider (Service)	2,699
Throughtput	1,831,253
Response Time	1,873,838

A. *Dataset*

To evaluate the performance of LLMSRec and compare it with other methods, we use WSDream, a ubiquitous QoS prediction dataset collected by Zheng *et al.* [36]. The dataset captures web interactions among 339 users and 5,825 services. Each user and service has an ID, as well as several additional attributes. Several of such attributes are utilized by LLMSRec in the form of descriptive sentences as described in § III-A. The number of users and services, and the number of unique values of the attributes are shown in Table I.

WSDream consists of subsets for two different QoS metrics: throughput and response time. The throughput subset contains 1,831,253 records, and the response time subset has 1,873,838 records. To better simulate the data sparsity challenge faced by real-world service recommendation applications, we shuffle each subset and split it into training and testing sets based on different density values, with the training set containing only 5%, 10%, 15%, and 20% of the QoS values. This allows

TABLE II: The QoS prediction performance in terms of Mean Average Error (MAE) and Root Mean Squared Error (RMSE) of baseline methods and the proposed LLMSRec. The performance on different densities and the throughput and response time subsets are shown. For each combination of subset and density, the best baseline method is underlined, and we show the percentage improvement from the best baseline achieved by LLMSRec as gain. LLMSRec outperforms all baselines consistently.

(a) Throughput.

Methods	Density=5%		Density=10%		Density=15%		Density=20%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UIPCC	26.757	60.799	22.370	54.456	20.219	50.704	18.928	48.295
RegionKNN	25.632	67.868	24.838	67.551	24.584	67.314	24.036	66.176
LACF	23.169	58.967	19.626	53.105	17.795	49.766	16.667	47.625
PMF	19.082	57.883	15.994	48.071	14.670	44.013	13.924	41.714
LRMF	19.109	58.072	15.949	48.272	14.597	44.068	13.921	41.788
PSO-USRec	23.332	60.155	19.740	54.254	17.839	50.209	16.787	47.395
LMF-PP	18.301	51.777	15.913	46.142	14.745	42.993	14.103	41.408
DCALF	18.624	51.412	15.343	45.901	14.066	42.624	13.549	41.219
LDIF	17.200	48.140	15.810	45.500	15.400	40.840	15.100	39.650
LLMSRec	**12.461**	**44.499**	**10.513**	**38.924**	**9.779**	**37.038**	**9.230**	**35.100**
Gain	27.55%	7.56%	31.48%	14.45%	30.48%	9.31%	31.87%	11.48%

(b) Response time.

Methods	Density=5%		Density=10%		Density=15%		Density=20%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UIPCC	0.625	1.388	0.582	1.330	0.501	1.250	0.450	1.197
RegionKNN	0.588	1.543	0.548	1.513	0.526	1.513	0.516	1.521
LACF	0.637	1.444	0.566	1.342	0.516	1.276	0.483	1.230
PMF	0.569	1.537	0.487	1.316	0.452	1.221	0.431	1.169
LRMF	0.555	1.495	0.485	1.296	0.454	1.209	0.435	1.163
PSO-USRec	0.565	1.358	0.506	1.274	0.471	1.222	0.444	1.181
LMF-PP	0.529	1.341	0.473	1.242	0.447	1.210	0.426	1.161
DCALF	0.513	1.373	0.454	1.245	0.435	1.200	0.425	1.176
LDIF	0.403	1.305	0.402	1.303	0.303	1.301	0.301	1.300
LLMSRec	**0.359**	**1.291**	**0.316**	**1.196**	**0.296**	**1.170**	**0.283**	**1.144**
Gain	10.84%	1.09%	21.32%	3.67%	2.44%	2.55%	6.05%	1.52%

better performance assessment of the proposed method under different levels of data availability.

B. Implementation Details

The experiments are carried out with `PyTorch` [37], and the implementation and pretrained weights of the LLMs are taken from `HuggingFace` [38]. Since QoS prediction does not require the model to have question-answering or complex task-solving capabilities, the base pretrained model is used. We use Qwen2.5-0.5B for in the experiments, and compare it with other LLMs in § IV-F. The model weights are converted to `float32` precision before training. For all LoRA layers, we use a rank of 64 and α of 64. In all experiments, the model is trained for 20 epochs on the training set with batch size of 64 and a constant learning rate of 10^{-4}, using the AdamW [39] optimizer.

To evaluate the performance of the model, Mean Average Error (MAE) and Root Mean Squared Error (RMSE) metrics on the testing set are measured by

$$\text{MAE} = \frac{\sum_{j=1}^{M} |r_j - \hat{r}_j|}{M}, \tag{9}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{j=1}^{M} (r_j - \hat{r}_j)^2}{M}}, \tag{10}$$

where $(r_j, \hat{r}_j)$ is a pair of ground truth and predicted QoS values, and M is the number of samples in the testing set. The MAE and RMSE of the final model are used for evaluation.

C. Baseline Models

We selected several baseline methods for QoS prediction for comparison with the proposed LLMSRec. These methods range from classical collaborative filtering to advanced deep neural network model-based methods. For a fair and consistent evaluation, the same densities and evaluation metrics as those used by LLMSRec are used across all baselines. The baselines are briefly introduced below.

UIPCC [16] is a hybrid collaborative filtering method designed to predict missing QoS values by combining user and service similarity information. It dynamically integrates two similarity computation strategies through a balanced weighting mechanism, enhancing prediction accuracy and robustness.

RegionKNN [40] is a scalable hybrid collaborative filtering algorithm. It introduces a region-based model to capture the unique characteristics of users and services. RegionKNN makes QoS predictions by integrating geographic information with an enhanced memory-based collaborative filtering.

LACF [41] also integrates the geographic locations of users and services. It focuses on identifying similar users and services within the same or neighboring geographic regions.

By incorporating location information into a similarity measurement process, the accuracy of QoS value prediction is enhanced.

PMF [42] is a collaborative filtering model that integrates probabilistic models into matrix factorization to enhance QoS prediction accuracy. It leverages Bayesian inference to estimate latent features of users and services, enabling robust predictions under high sparsity.

LRMF [43] is a QoS prediction method that integrates user reputation and location information to improve accuracy. By addressing the issue of unreliable QoS values from untrustworthy users, LRMF enhances traditional matrix factorization with reputation-aware mechanisms.

PSO-USRec [44] models QoS prediction as a global search optimization problem in the QoS distribution space, leveraging an improved particle swarm optimization technique. By diversifying initial solutions and smoothing outlier particles, it avoids local optimization and achieves more accurate predictions.

LMF-PP [45] integrates geographical information, invocation records, and neighborhood similarity for QoS prediction. It employs preference propagation to fuse these components, effectively addressing the data sparsity issue.

DCALF [46] is a QoS prediction model that extracts dense latent factors from sparse interaction records to identify user and service neighborhoods while reducing noise. It incorporates density peaks-based clustering to simultaneously detect neighborhoods and noise, enabling precise representation and prediction of QoS values.

LDIF [47] integrates location similarity and feature interactions through a scanning interaction structure. It efficiently captures low and high-order feature interactions while using an early-stop mechanism to minimize unnecessary computations, offering a lightweight yet effective solution.

D. Comparison with Baselines

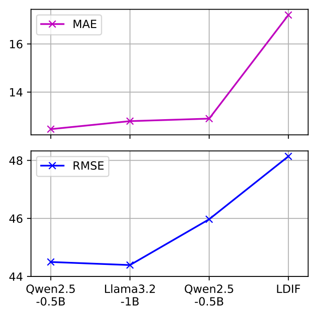
We run experiments of the proposed LLMSRec trained from Qwen2.5-0.5B on different density levels from 5% to 20% on the throughput and response time subsets of WSDream, and evaluate the MAE and RMSE metrics. The results are compared with the baseline methods in Table II. LDIF achieves the highest accuracy overall, being the best baseline method on most of the subsets and density levels. The proposed LLMSRec achieves higher accuracy than all baseline methods consistently. On the throughput subset the advantage is especially high, in some cases with more than 30% of improvement over LDIF. This indicates that the pretrained LLM can learn to make accurate QoS predictions after being trained on a very limited amount of data, effectively addressing the data sparsity issue. The performance gain on response time subset is more subtle, probably due to the fact that response time values are smaller, so it requires more fine-grained prediction and is more difficult to improve.

The gain of LLMSRec over the baseline methods is more significant on MAE compared to RMSE. This is likely caused by the fact that LLMSRec is trained with L_1 loss, which corresponds to minimizing the MAE instead of RMSE. Nevertheless, it still outperforms the baselines with the RMSE metric. Similar to the baseline methods, LLMSRec can make more accurate QoS prediction when trained on higher density, indicating the LLM can capture and memorize more correlation information among the web users and services when exposed to more samples.

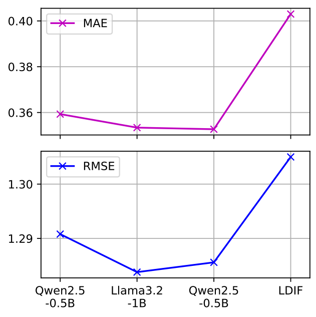
E. Effects of LoRA Training

In the experiments we use LoRA to reduce the memory usage during training. However, since the number of trainable parameters is significantly reduced, there is a risk of lower representativeness leading to underfitting. So we train another set of models using full parameters of Qwen2.5-0.5B, and compare the results with LoRA in Table III. We also show the GPU memory usage of both cases. It is shown that LoRA training actually gets better accuracy than full parameter training. The effect of LoRA of model performance is found to be inconsistent on different datasets and tasks [48], [49]. Since WSDream is smaller compared to datasets used for training modern LLMs, using less number of parameters for training might reduce the risk of overfitting thus achieving higher accuracy. And LoRA training uses significantly less memory than full parameter training. Please note that the reduction in memory usage will be more pronounced for larger LLMs.

F. Comparison of Different LLMs



(a) Throughput.



(b) Response time.

FIGURE II: Comparison of the performance of different LLMs. The best baseline method (LDIF [47]) is also shown for perspective. All LLMs achieve significantly higher accuracy than the best baseline method.

Qwen2.5-0.5B, which is the smallest model in the Qwen2.5 family [32], is used in the main experiments. We further tested

TABLE III: Comparison of LoRA and full parameter training of Qwen2.5-0.5B with density=5%. LoRA gets higher accuracy while using less memory.

Training	Throughput		Response Time		Memory (GiB)
	MAE	RMSE	MAE	RMSE	
LoRA	12.461	44.499	0.359	1.291	10.14
Full Parameter	14.258	48.183	0.367	1.326	13.02

two larger models Llama3.2-1B [50] and Qwen2.5-1.5B. The same training hyper-parameters and LoRA settings are used for fair comparison. The performance is compared in Figure II, and we show the best baseline method (LDIF [47]) to put the performance in perspective. It is shown that all LLMs can achieve significantly lower error than LDIF, regardless of their size and architecture. This indicates that the proposed method is general and applicable to different LLMs, rather than only working with a specific model.

The relative performance among different LLMs is inconsistent. For instance, Qwen2.5-1.5B gets lower MAE and RMSE than Qwen2.5-0.5B on response time subset, but higher MAE and RMSE than Qwen2.5-0.5B on throughput subset. However, the differences among different LLMs are much smaller compared to the difference between Qwen2.5-0.5B and the best baseline method, showing the robustness and generalization ability of the proposed method.

V. Perplexity Analysis

Modern LLMs are commonly trained on very large text corpus with many trillions of tokens [32], [50]. Although the methodology of collecting such corpus and its domain composition are discussed with some level of details in the technical reports, the exact sources are rarely disclosed by the developers. This poses the risk of training on the testing set. For a language task such as text completion or question answering, if its testing set is publicly available, it can be possible that the testing set is already included in the training data of the model. This will cause the model to overfit on the task and inflate the performance. For our experiments, although WSDream is public, it is not a language dataset by nature, and the QoS prediction is not regarded as a textual task. So it is unlikely that the descriptive sentences used by our experiments are present in the training set of the LLMs we used. Yet the possibility cannot be fully ruled out.

Perplexity can be used to measure the possibility of an LLM being trained on a dataset. An auto-regressive language model calculates the probability distribution of the next token t_N given context $t_1, t_2, \ldots, t_{N-1}$

$$p(t_N|t_1, t_2, \ldots, t_{N-1}; \boldsymbol{\theta}), \tag{11}$$

where $\boldsymbol{\theta}$ is the model parameter. For a sentence in a dataset in the form of a sequence of tokens $t_1, t_2, \ldots, t_N$, the perplexity (PPL) can be defined as

$$PPL = \exp[-\frac{1}{N}\sum_{i=2}^{N} \log p(t_i|t_1, t_2, \ldots, t_{i-1}; \boldsymbol{\theta})]. \tag{12}$$

If the model is well trained on the sentence, it can predict the correct next token with high confidence, resulting in lower perplexity. On the contrast, if the PPL value is high, the sentence is not likely to be in the model's training set.

We compare the perplexity on Qwen2.5-0.5B of the descriptive sentences we constructed from WSDream with several public textual datasets. WikiText-2 [51] is a dataset that consists of texts of good-quality Wikipedia articles. We treat each paragraph in the training split as an input sentence. CMRC-2018 [52] is a question answering dataset for the reading comprehension task, and we use each article in the training split as an input sentence. Both datasets have been publicly available for years, so there is a good chance that they are included in the training set of Qwen2.5-0.5B. For both WikiText-2 and CMRC-2018, we also test their randomized versions, where for each sentence, the sub-sentence (text pieces separated by commas, colons, semicolon, periods, exclamation marks, or question marks) are randomly shuffled. The randomized sentences are incoherent and are very unlikely in the training set.

The distribution of the perplexity values of the sentences in each dataset is compared in Figure IV. Both WikiText-2 and CMRC-2018 have perplexity with the average at around 30. This is still relatively high due to the smaller size of Qwen2.5-0.5B. After shuffling the sub-sentences, the perplexity increases significantly for both datasets to the average of above 50. The descriptive sentences of WSDream, however, have perplexity even higher than the randomized versions of WikiText-2 and CMRC-2018, with the average around 70. This shows that the model cannot predict the next token for the descriptive sentences with high accuracy and certainty, implying that the model has not seen similar text during training. So the model is not trained on testing set, and the performance obtained in our experiments is not inflated.

To further investigate the behavior of the pretrained model, we feed Qwen2.5-0.5B with partial descriptive sentences of WSDream and let it complete the sentence. We use greedy decoding to avoid any randomness. Some completed sentences are compared with the corresponding original sentences in Figure III. The model can write coherent sentences based on partial inputs in the context of web services and users. But it does not possess the specific information about each service or user, only general information about networks and countries. Thus we can safely conclude that the information of the web services and users of WSDream is not included in the training set of the model, and is learned along with the QoS prediction training.

VI. Conclusion

In this paper, we have proposed the Large Language Model for Service Recommendation (LLMSRec) framework for QoS prediction. It utilizes the great generalization ability of pretrained LLMs, and can learn to understand the QoS-related features of them under the data sparsity constrain. LLMSRec extracts useful information for QoS prediction from each web user and service by using their corresponding unstructured descriptive natural language sentences as input. It consistently outperforms the baseline models on the WSDream dataset in different settings. Our work demonstrates the versatility of LLMs in service recommendation applications, and we hope it can inspire more exploration in this area.

original sentence	completed sentence
web service number 3637, at url http://www.smithsfoodanddrug.com/_vti_bin/UserProfileService.asmx?wsdl, hosted by smithsfoodanddrug.com, located in Germany, in autonomous system AS32577 The Kroger Co..	web service number 3637, at url http://www.smithsfoodanddrug.com/_vti_bin/UserProfileService.asmx?wsdl, hosted by smithsfoodanddrug.com, is not available. The error message indicates that the service…
web service number 5573, at url http://www.url.org/Service.asmx?WSDL, hosted by url.org, located in Italy, in autonomous system AS47750 Oneitalia S.p.a..	web service number 5573, at url http://www.url.org/Service.asmx?WSDL, is a SOAP web service. The service is designed to handle requests from clients and return responses in XML format. The service…
web user number 86, located in Germany, in autonomous system AS8365 Man-da.de GmbH.	web user number 86, located in Germany, is a member of the European Union and is a member of the European…
web user number 139, located in Austria, in autonomous system AS1110 University of Innsbruck.	web user number 139, located in Austria, in the 19th century, was the first to use the term "sociology" in a scientific…

FIGURE III: The completion results of Qwen2.5-0.5B on partial descriptive sentences of WSDream. The partial, original remaining, and completed parts are indicates by different colors. It is evident that the model does not possess the specific information of the users and services.

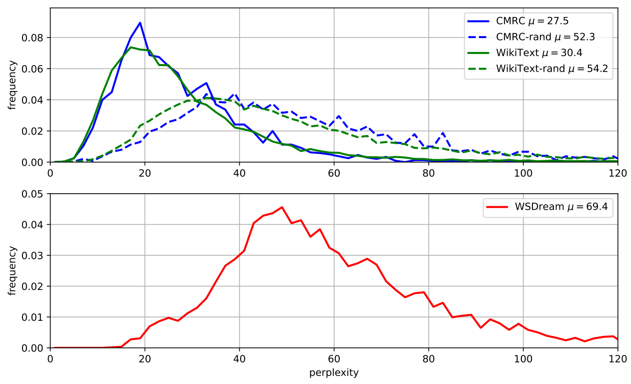


FIGURE IV: Perplexity (PPL) of Qwen2.5-0.5B model on several datasets. It is clear that the PPL value of public textual datasets (CMRC and WikiText) are relatively low, while the PPL value on the descriptive sentences of WSDream is significantly higher, even higher than randomized CMRC and WikiText. This indicates that the high performance of the proposed model is unlikely to be inflated.

REFERENCES

[1] S. Li, H. Luo, and G. Zhao, "bi-HPTM: An effective semantic matchmaking model for web service discovery," in *IEEE International Conference on Web Services*, 2020.

[2] Y. Zhang, L. Wu, Q. He, F. Chen, S. Deng, and Y. Yang, "Diversified quality centric service recommendation," in *IEEE International Conference on Web Services (ICWS)*, 2019.

[3] Y. Zhang, X. Ai, Q. He, X. Zhang, W. Dou, F. Chen, L. Chen, and Y. Yang, "Personalized quality centric service recommendation," in *Service-Oriented Computing*, 2017.

[4] M. Silic, G. Delac, and S. Srbljic, "Prediction of atomic web services reliability based on k-means clustering," in *Joint Meeting on Foundations of Software Engineering*, 2013.

[5] Z. Wu, D. Ding, Y. Xiu, Y. Zhao, and J. Hong, "Robust qos prediction based on reputation integrated graph convolution network," *IEEE Transactions on Services Computing*, vol. 17, no. 3, pp. 1154–1167, 2024.

[6] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *International Conference on World Wide Web*, 2001.

[7] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J. Nie, and J. rong Wen, "A survey of large language models," *ArXiv*, 2023.

[8] S. Minaee, T. Mikolov, N. Nikzad, M. A. Chenaghlu, R. Socher, X. Amatriain, and J. Gao, "Large language models: A survey," *ArXiv*, 2024.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[10] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *ArXiv*, 2019.

[11] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, and et al., "Language models are few-shot learners," *OpenAI*, 2020.

[12] M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, and et al., "Phi-3 technical report: A highly capable language model locally on your phone," *ArXiv*, 2024.

[13] Meta AI, "Introducing meta Llama 3: The most capable openly available llm to date," 2024. [Online]. Available: https://ai.meta.com/blog/meta-llama-3/

[14] L. Wu, Z. Zheng, Z. Qiu, H. Wang, H. Gu, T. Shen, C. Qin, C. Zhu, H. Zhu, Q. Liu, H. Xiong, and E. Chen, "A survey on large language models for recommendation," *World Wide Web*, vol. 27, no. 5, p. 60, 2024.

[15] X. Ren and C. Huang, "Easyrec: Simple yet effective language models for recommendation," *ArXiv*, 2024.

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

[17] Y. Hu, Q. Peng, X. Hu, and R. Yang, "Time aware and data sparsity tolerant web service recommendation based on improved collaborative filtering," *IEEE Transactions on Services Computing*, vol. 8, no. 5, pp. 782–794, 2015.

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

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

[20] Y. Zhang, K. Wang, Q. He, F. Chen, S. Deng, Z. Zheng, and Y. Yang, "Covering-based web service quality prediction via neighborhood-aware matrix factorization," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1333–1344, 2021.

[21] C. Wei, Y. Fan, and J. Zhang, "Time-aware service recommendation with social-powered graph hierarchical attention network," *IEEE Transactions on Services Computing*, vol. 16, no. 3, pp. 2229–2240, 2023.

[22] Y. Yin, Q. Di, J. Wan, and T. Liang, "Time-aware smart city services based on qos prediction: A contrastive learning approach," *IEEE Internet of Things Journal*, vol. 10, no. 21, pp. 18 745–18 753, 2023.

[23] Y. Zhang, C. Yin, Z. Lu, D. Yan, M. Qiu, and Q. Tang, "Recurrent tensor factorization for time-aware service recommendation," *Applied Soft Computing*, vol. 85, p. 105762, 2019.

[24] Y. Xu, J. Yin, W. Lo, and Z. Wu, "Personalized location-aware qos prediction for web services using probabilistic matrix factorization," in *Web Information Systems Engineering*, X. Lin, Y. Manolopoulos, D. Srivastava, and G. Huang, Eds., 2013.

[25] Z. Wang, C.-A. Sun, and M. Aiello, "Context-aware iot service recommendation: A deep collaborative filtering-based approach," in *IEEE International Conference on Web Services (ICWS)*, 2022.

[26] T. Lu, X. Zhang, Z. Wang, and M. Yan, "A feature distribution smoothing network based on gaussian distribution for qos prediction," in *IEEE International Conference on Web Services (ICWS)*, 2023.

[27] 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 *IEEE International Conference on Web Services (ICWS)*, 2021.

[28] G. Zou, S. Wu, S. Hu, C. Cao, Y. Gan, B. Zhang, and Y. Chen, "Ncrl: Neighborhood-based collaborative residual learning for adaptive qos prediction," *IEEE Transactions on Services Computing*, vol. 16, no. 3, pp. 2030–2043, 2023.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[30] X. Ren, W. Wei, L. Xia, L. Su, S. Cheng, J. Wang, D. Yin, and C. Huang, "Representation learning with large language models for recommendation," in *ACM Web Conference*, 2024, p. 3464–3475.

[31] L. Sheng, A. Zhang, Y. Zhang, Y. Chen, X. Wang, and T.-S. Chua, "Language representations can be what recommenders need: Findings and potentials," in *International Conference on Learning Representations*, 2025.

[32] Qwen, A. Yang, B. Yang, B. Zhang, and et al., "Qwen2.5 technical report," *ArXiv*, 2024.

[33] G. Feng, K. Yang, Y. Gu, X. Ai, S. Luo, J. Sun, D. He, Z. Li, and L. Wang, "How numerical precision affects mathematical reasoning capabilities of llms," *ArXiv*, 2024.

[34] I. Mirzadeh, K. Alizadeh, H. Shahrokhi, O. Tuzel, S. Bengio, and M. Farajtabar, "Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models," *ArXiv*, 2024.

[35] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

[36] Z. Zheng and M. R. Lyu, "Ws-dream: A distributed reliability assessment mechanism for web services," in *IEEE International Conference on Dependable Systems and Networks*, 2008.

[37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: an imperative style, high-performance deep learning library," in *International Conference on Neural Information Processing Systems*, 2019.

[38] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "Huggingface's transformers: State-of-the-art natural language processing," *ArXiv*, 2019.

[39] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2017.

[40] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *IEEE International Conference on Web Services*, 2010.

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

[42] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *International Conference on Neural Information Processing Systems*, 2007.

[43] S. Li, J. Wen, F. Luo, T. Cheng, and Q. Xiong, "A location and reputation aware matrix factorization approach for personalized quality of service prediction," in *IEEE International Conference on Web Services (ICWS)*, 2017.

[44] J. Chen, C. Mao, and W. W. Song, "Qos prediction for web services in cloud environments based on swarm intelligence search," *Knowledge-Based Systems*, vol. 259, p. 110081, 2023.

[45] D. Ryu, K. Lee, and J. Baik, "Location-based web service qos prediction via preference propagation to address cold start problem," *IEEE Transactions on Services Computing*, vol. 14, no. 3, pp. 736–746, 2021.

[46] 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 Transactions on Knowledge and Data Engineering*, vol. 34, no. 6, pp. 2525–2538, 2022.

[47] S. Zhu, J. Ding, and J. Yang, "Location-aware deep interaction forest for web service qos prediction," *Applied Sciences*, vol. 14, no. 4, 2024.

[48] S. Ghosh, C. K. R. Evuru, S. Kumar, R. S., D. Aneja, Z. Jin, R. Duraiswami, and D. Manocha, "A closer look at the limitations of instruction tuning," in *International Conference on Machine Learning*, 2024.

[49] Y. Hao, Y. Cao, and L. Mou, "Flora: Low-rank adapters are secretly gradient compressors," in *International Conference on Machine Learning*, 2024.

[50] Meta AI, "Llama 3.2: Revolutionizing edge ai and vision with open, customizable models," 2024. [Online]. Available: https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/

[51] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *ArXiv*, 2016.

[52] Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. Ma, S. Wang, and G. Hu, "A span-extraction dataset for Chinese machine reading comprehension," in *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.