Multi-dimensional Sequential Contrastive Learning for QoS Prediction

Yuyu Yin, Qianhui Di, Yuanqing Zhang, Tingting Liang(✉), Youhuizi Li, and Yu Li

School of Computer Science, Hangzhou Dianzi University, Hangzhou, China
{yinyuyu,dqh_2021,yuanqingzhang,liangtt,huizi,liyuconp}@hdu.edu.cn

Abstract. Quality of service (QoS) is the main factor in service selection and recommendation, and it is influenced by dynamic factors, such as network condition and user location, and static factors represented by the invocation sequence at a fixed time slice. In order to jointly consider these two factors, this work proposes a multi-dimensional sequential contrastive learning framework named MDSCL, which applies contrastive learning method to learn the sequence representations of both user and time dimensionalities. An overlap crop augmentation strategy is proposed to obtain positive examples for user sequences and time sequences, respectively. Besides, MDSCL includes an integrated feature extractor that combines WaveNet and BiLSTM to facilitate the long short-term feature capturing. Extensive experiments on WSDREAM have been conducted to verify the effectiveness of our approach.

Keywords: QoS prediction · Multi-dimensional contrastive learning · Multi-task training

1 Introduction

The continuous advancement of Service-Oriented Architecture (SOA) motivates the rapid growth of the number of Web services or Web APIs, which drives the production of software applications with more complex functions and accelerates the development of interoperable interaction through the Internet. The proliferation of Web services makes it difficult for consumers to effectively discover the high-quality services that meet their requirements. Therefore, how to provide the satisfactory service recommendation for consumers is becoming an imperative task.

For those Web services with similar functionalities, the traditional semantic matching based service selection methods are no longer operative. In this case, Quality of Service (QoS), which describes the non-functional characteristics of Web service, such as response time, reliability, throughput, becomes the main factor in service selection and recommendation. QoS prediction is the essential preparation for service recommendation which aims at providing the

H. Gao et al. (Eds.): CollaborateCom 2023, LNICST 562, pp. 514–531, 2024.
https://doi.org/10.1007/978-3-031-54528-3_28

appropriate services for consumers to satisfy both functional and non-functional requirements.

There exist many approaches of predicting unknown service QoS values by modeling the historical invocation information. Generally, these approaches can be categorized into three groups, which are memory-based, model-based, and context-aware methods. The memory-based methods [1,2] usually compute the similarity between users or Web services and predict the missing QoS values depending on the historical values of similar users or services. The model-based approaches [3,4] learn the potential characteristics of consumers and Web services with the machine learning algorithm for prediction. The context-based approaches [5,6] incorporate the contextual information, *e.g.,* geographical information, time, trustworthiness to facilitate QoS prediction, among which the temporal-aware methods have attracted great attention. Considering that a consumer might receive different QoS experiences when invoking the same service at different points of time, a deep learning based approach called DeepTSQP is proposed to learn the temporal feature representations by modeling the user-service invocation changes across different temporal slices [7]. Besides, a dynamic graph neural collaborative learning approach is utilized to model user-service historical temporal interactions and extract latent features of users and services at each time slice [8].

Although the aforementioned approaches improve the performance of service QoS prediction, most of them focus on the modeling of user invocation sequence while neglect the importance of sequential information in time dimensionality. Generally, many previous work models the user invocation sequence to predict the QoS values on all the services in the next time slice considering that the invocation conditions (*e.g., network condition, user location*) might be different across different time slices. However, besides the dynamic factor, the static factor represented by the invocation sequence at a fixed time slice should also be considered. The underlying intuition is that service state is consistent and user-service interactive features would be more prominent.

To compensate for this shortcoming, this work proposes a Multi-Dimensional Sequential Contrastive Learning framework named MDSCL for QoS prediction. Specifically, MDSCL performs contrastive learning on service invocation sequences from both user and time dimensionalities to learn the discriminative representations of user behaviours and temporal conditions. An overlap crop strategy is proposed for the sequence augmentation which generates two positive examples for the user sequence and temporal sequence, respectively. Furthermore, in order to obtain the representative sequential features, we integrate the WaveNet [9] and BiLSTM [10] to capture both long and short information for contrastive learning.

The main contributions of this work can be summarized as follows:

– We propose a multi-dimentional sequential contrastive learning framework named MDSCL for QoS prediction which considers invocation sequences of both user and time dimensionalities.

– We equip MDSCL with a overlap crop augmentation strategy for positive sequence generation and an integrated feature extractor for long short-term feature capturing.
– We conduct comprehensive experiments on a real-world dataset and compare the proposed MDSCL with the well-known existing methods. Experimental results demonstrate the effectiveness of our MDSCL.

2 Related Work

Among various research topics related to dynamic service QoS prediction, the prediction and estimation of actual QoS values of dynamic QoS attributes have been widely studied, and the methods are mainly classified into factor analysis-based methods, time-series-based methods, and hybrid-based methods that combine the above two methods.

Factor analysis is a variable simplification technique that starts from analyzing the correlation of multiple original variables, mainly by studying the correlation matrix of multiple variables and identifying a limited number of potential variables that govern the correlation to achieve the purpose of explaining complex problems with a few variables. In the field of QoS prediction, tensor decomposition methods can be used [11], which can extract user-specific, service-specific, and time-specific potential features from a three-dimensional matrix based on user-service-time, and then perform QoS prediction based on the potential features. On the basis of the basic matrix decomposition and tensor decomposition, it can be improved. For example, the biased non-negative latent factor decomposition model models the linear bias (LB) of users, services, and time points and uses LB vectors of the same dimensionality to form a first-order tensor of LB [12]. It applies additive GD and LB for each latent factor and controls the learning rate to offset the negative terms with the initial state of the corresponding parameters.

The essence of the factor analysis algorithm is to find potential factors in the data and apply the potential factors to explain the phenomena shown by the original variables. However, in practice, potential factors are difficult to extract accurately, and the potential characteristics of users are dynamically changing, as shown by the fact that the QoS values of users in a specific time interval will be influenced not only by the QoS values of similar users, but also by the QoS values in previous time intervals. Therefore, QoS prediction of dynamic services can be performed by time series. The method is divided into a statistical based method and a neural network based method. Statistical based time series methods include by ARIMA model [13–15]. The neural network based time series method mainly includes the prediction by using recurrent neural network RNN [3,16].

The hybrid approach is to choose two or more of the most suitable methods among the single methods of factor analysis and time series mentioned above. However, it is not the case that the more methods there are, the better the final prediction performance. Rather, its key factors should be considered, for example, if the prediction accuracy is considered, i.e., the method with the lowest

prediction error. A typical approach is to combine matrix decomposition with other methods. For example, it can be combined with network embedding, where a reputation-aware network embedding is used to learn the hidden representation of the user [17]. Then a user-based matrix decomposition method is used to predict the unknown QoS values. Also, it can be combined with convolutional neural networks [18] to extract QoS features through a joint deep network of potential factor embedding methods, matrix decomposition, and convolutional neural networks [19]. In addition, it can also be combined with graph networks [20] to first deep mine potential relationships based on the QoS matrix [21], and then combine multi-source information from user-aware context and service-aware context with graph networks to adaptively combine the QoS values by cutting irrelevant edges into several subgraphs and establishing a Gaussian mixture model (GMM) of QoS values as a fusion method based on the local information of the subgraphs and the global information of the integral graph. Local and global information to complete the final QoS prediction. Meanwhile, it can also be combined with deep neural network DNN [22,23], with LSTM [24–26], etc. Although all of the above methods take the time factor into account in QoS prediction, they tend to extract dynamic features on user sequences at different time slices and thus utilize it for QoS prediction to improve the final prediction accuracy. However, these methods do not consider the sequence information on fixed time slices, which is also important and reflects the user interaction information in the same external state.

3 Proposed Model

In this section, we start with stating the problem formulation of prediction task, and then describe the overall framework of MDSCL and the details of different components.

3.1 Problem Formulation

Suppose the QoS prediction scenario includes N users $\mathcal{U} = \{u_1, u_2, \cdots, u_N\}$ and M services $\mathcal{S} = \{s_1, s_2, \cdots, s_M\}$. If user $u_i \in \mathcal{U}$ invokes service $s_j \in \mathcal{S}$ at time slice t, the QoS value would be recorded. These QoS values generated by the N users invoking M services at multiple time slices forms a three-dimensional QoS matrix $Q \in \mathbb{R}^{N \times M \times T}$, where T denotes the number of time slices. Fix the user dimensionality and concatenate the invocation sequences in different time slices to form a one-dimensional user invocation sequence $Seq^u = [Seq_1^u, Seq_2^u, \cdots, Seq_t^u]$, where $Seq_t^u = [(u, s_1, t, q_{u,s_1,t}), (u, s_2, t, q_{u,s_2,t}), \cdots, (u, s_M, t, q_{u,s_M,t})]$ is the invocation sequence of user u at time slice t. $q_{u,s,t}$ is an element of matrix Q which denotes the QoS value of user u on service s at time slice t. The values of services that users do not invoke at each time slice are filled in using the padding technique. In an analogical manner, the sequence of time dimensionality can be constructed as $Seq^t = [Seq_1^t, Seq_2^t, \cdots, Seq_m^t]$, where

$Seq_m^t = [(u_1, s_m, t, q_{u_1,s_m,t}), (u_2, s_m, t, q_{u_2,s_m,t}), \cdots, (u_N, s_m, t, q_{u_N,s_m,t})]$ is the invocation sequence of service s_m at time slice t. In this work, given the historical invocation sequence of both user and time dimensionalities, Seq^u and Seq^t, the goal is to precisely predict QoS values for the invocation on the $t+1$ time slice.

3.2 MDSCLFramework

This paper proposes a multi-dimensional sequential contrastive learning framework MDSCL, aiming at better extracting the user dynamic and time static features from service invocation data for accurate QoS prediction. Figure 1 shows the overview of the proposed framework. MDSCL consists of three phases: sequential data filtering, multi-dimensional sequential contrastive learning, and joint training. Firstly, the sequences of two different dimensionalites are transformed from the historical invocation data and inputted into the Kalman filter layer to get the smoothed QoS sequences. In the second step, the sequences are inputted into a multi-dimensional sequential augmentation module including the overlap crop to generate two positive sequence representations and an integrated feature extractor to obtain the representative sequential embeddings. Finally, we combine the contrastive learning losses of different dimensionalities and one prediction loss for model training.

3.3 Sequential Data Filtering

As shown in Fig. 1, the sequences of two different dimensionalities are generated from the historical invocation data. Specifically, the invocation sequence of user and time dimensionalities are respectively denoted by Seq^u and Seq^t as mentioned in Sect. 3.1. Since some interfering factors are present in the QoS sequence, the Kalman algorithm, as a state-space model-based filtering algorithm, could be chosen to filter the noise in the QoS data.

The original sequences Seq^u and Seq^t could be transformed into $KSeq^u = KalmanSmoother(Seq^u)$ and $KSeq^t = KalmanSmoother(Seq^t)$ through the Kalman smoother. The Kalman smoothing algorithm includes forward filtering algorithm and reverse filtering algorithm [27]. Specifically, the Kalman filter has the following state and measurement definitions:

$$\mathbf{x}_{k+1} = \mathbf{F}_{k+1,k}\mathbf{x}_k + \mathbf{W}_k, \mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{V}_k, \tag{1}$$

where $\mathbf{x}_k$ and $\mathbf{y}_k$ are the state and measurement at moment k, $\mathbf{F}_{k+1,k}$ and $\mathbf{H}_k$ are the transition and measurement matrix, $\mathbf{W}_k$ and $\mathbf{V}_k$ are the process and measurement matrix which are assumed to be white, zero mean Gaussian, and the covariance matrix of $\mathbf{V}_k$ is $\mathbf{R}_k$. In the forward filtering algorithm, the one-step prediction of the state, and the state estimationare respectively as follows:

$$\hat{\mathbf{x}}_{k/k-1} = \Phi\hat{\mathbf{x}}_{k-1}, \hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k/k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k/k-1}), \tag{2}$$

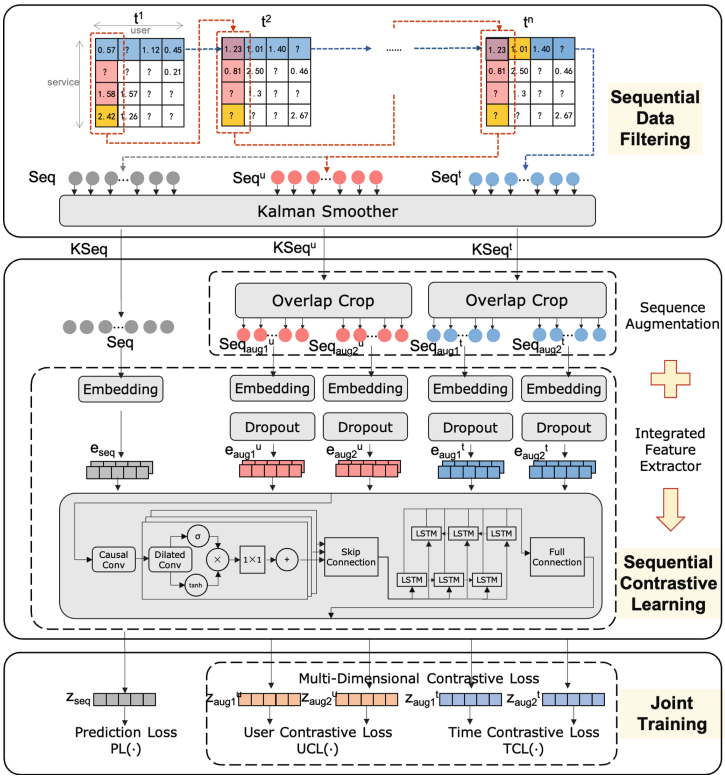


Fig. 1. The overall framework of MDSCL. The red and blue dots represent the sequences of user and time dimensionalities derived from the original invocation records. The gray dots represent the invocation sequence of user u on the first t time slices being used to predict the QoS values on the $t+1$ time slice, which are actually the same as the red dots.

where $\hat{\mathbf{x}}_{k/k-1}$ is the *a priori estimate* of $\mathbf{x}_k$, $\hat{\mathbf{x}}_{k-1}$ is the *a posteriori estimate* of $\mathbf{x}_{k-1}$, Φ is the $k-1$ moment to k moment one-step transfer matrix. The mean square error of the one-step prediction $\mathbf{P}_{k/k-1}$ and estimation $\mathbf{P}_k$ can be calculated, and the Kalman gain $\mathbf{K}_k$ could be obtain. Continuously iterating the above calculation process, the state vector estimation $\{\hat{\mathbf{x}}_k\}_{(k=1,2,\cdots,t)}$ can be finally obtained. In the inverse filtering algorithm, the smoothing equation is:

$$\hat{\mathbf{x}}_{k/t} = \hat{\mathbf{x}}_k + \mathbf{A}_k(\hat{\mathbf{x}}_{k+1/t} - \hat{\mathbf{x}}_{k+1/k}), \tag{3}$$

where $\mathbf{A}_k = \mathbf{P}_k \Phi^T \mathbf{P}^{-1}_{k+1/k}$, and the smoothed mean square error $\mathbf{P}_{k/t}$ can be calculated. Suppose $\hat{\mathbf{x}}_{t/t} = \hat{\mathbf{x}}_t$, the above process is integrated on the basis of forward filtering, and the smoothed state vector estimate $\{\hat{\mathbf{x}}_{k/t}\}_{(k=t-1,t-2,\cdots,0)}$,i.e., the output $KSeq^t$ is obtained recursively from the state vector estimate $\{\hat{\mathbf{x}}_{k/t}\}$.

3.4 Multi-dimensional Sequential Contrastive Learning

In order to learn the discriminative sequence representations containing both dynamic and static characteristics, this paper proposes a multi-dimensional contrastive learning which is the core component of MDSCL framework. The multi-dimensional contrasitve learning includes separate contrastive learning in both user and time dimensionalities. The sequence of user dimentionality is the invocation sequence of a specific user on all services across multiple time slices, which contains the user dynamic features. And the sequence of time dimensionality is the invocation sequence of all users on multiple services at a specific time slice, from which the service static features can be captured.

Specifically, the multi-dimensional contrastive learning module consists of three parts: sequence augmentation, integrated feature extraction, and contrastive loss function.

3.4.1 Sequence Augmentation

In this part, the sequence augmentation module generates two invocation sequences for the smoothed user sequence $KSeq^u$ and time sequence $KSeq^t$, respectively. The newly generated sequences are denoted by Seq^u_{aug1}, Seq^u_{aug2} and Seq^t_{aug1}, Seq^t_{aug2}. Since the sequences of user and time dimensionalities are processed with the same augmentation and feature extraction operations, we represent them with $KSeq$ uniformly in the next sections for simplicity. Specifically, each smoothed sequence $KSeq$ is fed into two successive layers, overlap crop and embedding layer with dropout, to obtain two representations for the augmented invocation sequences which contain the similar sequential characteristics. These augmented representations are used as positive pairs for the following multi-dimensional sequential contrastive learning.

The previous sequence augmentation methods [28,29] usually use the ordinary crop method, where some sub-objects of the original object are randomly selected as the augmented object. However, the ordinary crop method is only applicable to the domain where the original object is simple, such as computer vision. In the QoS prediction task, the QoS sequences have their unique time distributions, it is prone to destroy the time distribution of original sequence with the traditional crop method adopted, resulting in a low correlation between the time distributions of the two augmented sequences after cropping. Such augmented sequences are less referable and less effective for the subsequent contrastive learning.

To alleviate this problem, this work designs an overlap crop method for the QoS sequence augmentation, as shown in Fig. 2. Firstly, a segment of QoS subsequence $[q_{start}, \cdots, q_{end}]$ is arbitrarily selected as the overlap region in the smoothed invocation sequence $KSeq = [q_0, \cdots, q_t]$. The length of the subsequence is decided by a preset parameter α, *i.e.,* $|[q_{start}, \cdots, q_{end}]| = \alpha \cdot |KSeq|$. Here we only use the symbols of QoS values to denote the invocation record for simplicity. q_{start} and q_{end} can be the same as q_1 and q_t. Next, one of the augmented sequences $Seq_{aug1} = [q_i, \cdots, q_{end}]$ is generated, where $i = \beta \cdot start$. The other augmented sequence $Seq_{aug2} = [q_{start}, \cdots, q_j]$ can also be determined,

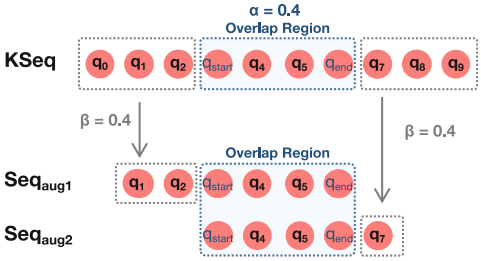


Fig. 2. A description of the overlap method.

where $j = end + \beta \cdot |t - end|$. Fig. 2 shows an example of overlap crop. Once the overlap region is decided, two augmented sequences are derived by expanding the region in different directions with the parameter β.

With the proposed overlap crop method, part of the time distribution $[q_{start}, \cdots, q_{end}]$ in the original sequence $[q_0, ..., q_t]$ is preserved, making the two augmented sequences contain common temporal characteristics and facilitating the subsequent multi-dimensional contrastive learning.

3.4.2 Integrated Feature Extractor

3.4.2.1 Embedding-Dropout Layer

The augmented sequences Seq_{aug1}, Seq_{aug2} generated by the overlap crop are fed into the subsequent embedding layer with dropout to capture the sequential features of the augmented samples from two dimensionalities.

$$\mathbf{E}_{aug1} = Dropout(F(Seq_{aug1})), \tag{4}$$

$$\mathbf{E}_{aug2} = Dropout(F(Seq_{aug2})), \tag{5}$$

where $F(\cdot)$ denotes the embedding layer and $Dropout(\cdot)$ is the dropout operation. $\mathbf{E}_{aug1} = [\mathbf{e}_{aug1,1}, \cdots, \mathbf{e}_{aug1,l}] \in \mathbb{R}^{l \times d}$ is the embeddings of the augmented sequence Seq_{aug1}, where l denotes the length of the sequence and d is the embedding size. Here the dropout is introduced to add some noise by randomly setting input units to 0 with a frequency of rate to the augmented representations, which is able to improve the robustness of the model.

3.4.2.2 Long Short-Term Feature Extractor

Considering that the sequences augmented by the proposed overlap crop are usually not too short, it requires a feature extractor that has the ability of capturing both long-term and short-term sequential information. WaveNet [9] is originally proposed for speech recognition and generation and achieves the impressive performance on sequential feature learning. Compared with the basic convolutional neural network (CNN), the casual and dilated convolutions of WaveNet effectively expands the receptive field, which increases its ability to handle long

sequences. Since WaveNet processes the sequence only in one direction and might loses the sequential features of another direction, we propose to integrate it with Bidirectional LSTM (BiLSTM) model [10]. LSTM is a type of recurrent neural network (RNN) which maintains memory of the input internally, making it well suited for solving long continuous data like time series. The performance of BiLSTM is better than LSTM in the series prediction problem [30] as it considers more contextual information, and we utilize the BiLSTM model as long short-term feature extractor in this work.

Since the representations of the augmented sequences, namely $\mathbf{E}_{aug1}$, $\mathbf{E}_{aug2}$ are going to pass through the same layers, in this section, we denote the representation with $\mathbf{E}_{aug}$ uniformly. The augmented sequence embedding $\mathbf{E}_{aug}$ is firstly fed into a casual convolutional layer, which ensures to keep the ordering of the input sequence. Subsequently, this filtered tensor is fed into a stacked dilated convolutional layer. Each layer performs the same operation, which is formulated as follows:

$$\mathbf{v}_k = tanh(\mathbf{W}_{f,k} * \mathbf{u}_k) \otimes \sigma(\mathbf{W}_{g,k} * \mathbf{u}_k), \tag{6}$$

where $\mathbf{W}_{f,k}$ and $\mathbf{W}_{g,k}$ are the learnable dilated convolution filters, f and g denote 'filter' and 'gate', k denotes the layer index, $\mathbf{u}_k$ denotes the input of the k-th layer, and the input for the first layer is the output of causal convolution layer. $*$ denotes the convolution operator, $\otimes$ is the element-wise multiplication operator, $\sigma(\cdot)$ is the sigmoid activation function.

After the gated activation unit, $\mathbf{v}_k$ is processed with a 1×1 convolution layer and residual operation, and the output is the input of the next dilated convolutional layer, while the sikp connection is used to obtain the output of WaveNet: :

$$\mathbf{u}_{wave} = \mathbf{u}_0 + \sum_{k=1}^{d} \mathbf{u}_k = \mathbf{u}_0 + \sum_{k=0}^{d-1} (\mathbf{W} * \mathbf{v}_k + \mathbf{u}_k) \tag{7}$$

The sequential feature representations filtered by WaveNet are used as the input to the BiLSTM module for further extraction of long short-term features, and the calculation process is as follows:

$$\mathbf{h}_t = LSTM(\mathbf{g}_t, \mathbf{h}_{t-1}), \mathbf{h}_t = LSTM(\mathbf{g}_t, \mathbf{h}_{t-1}), \mathbf{h}_t = \mathbf{W}_t^f \mathbf{h}_t + \mathbf{W}_t^b \mathbf{h}_t + \mathbf{b}_t, \tag{8}$$

where $\mathbf{g}_t$ denotes the input of the BiLSTM network at time slice t, the input of the first moment of the network is the output of WaveNet $\mathbf{u}_{wave}$. $\mathbf{h}_t$ denotes the forward output of BiLSTM at time slice t and $\mathbf{h}_t$ is the reverse output of BiLSTM at time slice t. $\mathbf{W}_t^f$ and $\mathbf{W}_t^b$ are the weight matrices, and $\mathbf{b}_t$ is the bias. We choose the last hidden vector $\mathbf{h}_t$ as the final output of the BiLSTM and perform a full connection layer $FC(\cdot)$ of this output as the representation of one augmented sequence $\mathbf{z}_{aug}$:

$$\mathbf{z}_{aug} = FC(\mathbf{h}_t) \tag{9}$$

3.4.3 Multi-dimensional Contrastive Loss

The multi-dimensional contrastive loss function is designed to minimize the difference between the augmented sequence pair and maximize the distance between the negative sequence pair. Here we use the inner product of sequence representations to measure the difference between sample pairs.

Given a training batch with size $BN = BN^u + BN^t$, where BN^u is the number of sequences of user dimensionality and BN^t is the number of sequences of time dimensionality, we can get $2BN^u$ positive sequences of user dimensionality and $2BN^t$ those of time dimensionality. For each augmented sequence pair of user dimensionality (Seq_{aug1}, Seq_{aug2}), the rest $2(BN^u - 1)$ is used as the negative samples. The contrastive loss function of user dimensionality can be defined as:

$$\mathcal{L}_{cl}^{u} = -\log \frac{\exp((\mathbf{z}_{aug1}^{u} \cdot \mathbf{z}_{aug2}^{u})/\tau)}{\exp((\mathbf{z}_{aug1}^{u} \cdot \mathbf{z}_{aug2}^{u})/\tau) + \sum \exp((\mathbf{z}_{aug1}^{u} \cdot \mathbf{z}^{u-})/\tau)}, \tag{10}$$

where τ is a temperature hyper-parameter [31]. In the same manner, the contrastive loss function of time dimensionality can be defined as follows:

$$\mathcal{L}_{cl}^{t} = -\log \frac{\exp((\mathbf{z}_{aug1}^{t} \cdot \mathbf{z}_{aug2}^{t})/\tau)}{\exp((\mathbf{z}_{aug1}^{t} \cdot \mathbf{z}_{aug2}^{t})/\tau) + \sum \exp((\mathbf{z}_{aug1}^{t} \cdot \mathbf{z}^{t-})/\tau)}. \tag{11}$$

3.5 Joint Training

The task of the proposed multi-dimensional contrastive learning aims at learning the discriminative representations for service invocation sequences of both user and time dimensionalites by closing the augmented positive sequence pairs and separating the negative ones. The aforementioned two contrastive losses $\mathcal{L}_{cl}^{u}$ and $\mathcal{L}_{cl}^{t}$ well serve this purpose.

For the QoS prediction task, as mentioned in the problem formulation, the goal is to precisely predict the QoS values on the next time slice $t+1$. The original sequence $Seq^u = [Seq_1^u, Seq_2^u, \cdots, Seq_t^u]$ is fittered by the Kalman smoother and input into the integrated feature extractor to generate the sequential representations in the same manner with that of the augmented sequences. The target service s_{target} to be predicted at time $t+1$ is fed into the embedding layer to obtain the representation $\mathbf{e}_{target}$. With the learned representation of the original sequence $\mathbf{z}_t$ and $\mathbf{e}_{target}$, the final QoS value can be predicted by $\mathbf{z}_t \cdot \mathbf{e}_{target}$. To effectively learn the QoS prediction task, we adopt the squared loss as objective function:

$$L_{pred} = \frac{1}{|\mathcal{D}_{t+1}|} \sum_{i \in \mathcal{D}_{t+1}}^{N} (\hat{y}_i - y_i)^2, \tag{12}$$

where $\mathcal{D}_{t+1}$ denotes the set of observed QoS values of all users on all services at the $t+1$ time slice. y_i is the true QoS value, and $\hat{y}_i$ is the predicted QoS value.

To jointly learn the QoS prediction task and the additional multi-dimensional contrastive learning task, we combine the predictive loss and the constrastive

learning loss of two dimensionalites to obtain the final objective function as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{pred} + \lambda(\mathcal{L}_{cl}^{u} + \mathcal{L}_{cl}^{t}), \tag{13}$$

where λ denotes the fixed coefficient for sequential contrastive loss function of the user and time dimensionalies.

4 Experiments

4.1 Datasets

We conduct all experiments on a publicly available dataset WSDREAM[1], which was collected from real-world Web services and suitable for large scale experiments. The dataset is a time-aware dataset consisting of 27,392,643 response time records obtained from 142 users from different regions sending requests to 4500 real-world Web services over 64 consecutive time slices. Since each time period is 16 h, according to a time interval of every 15 min, there are a total of 64 time slices per user. We select response time as a typical QoS values for our study.

4.2 Expermimential Setup

4.2.1 Implementation Details

In reality, since a user may invoke only a small number of services, the corresponding QoS values contain a large number of missing values, leading the user-service QoS matrix with high sparsity. In order to simulate the actual service invocation scenario, we remove some QoS values randomly on the basis of the original data to generate eight datasets of different sparsity levels, namely 1%−5%, 10%, 15%, and 20%. When dividing the dataset, 60% of the QoS values are used as the training set, and the remaining 40% are used for model testing. In addition, in order to eliminate the coincidence of experimental results, each experiment is conducted 5 times for each data density and the average result is used as the final result.

For all the baseline methods, we take the same modeling approach for sequencial input, i.e., the sequencial modeling approach in our MDSCL for the user dimension. In addition, we take the same window size and number of samples to control the sequence length and optimize the computational cost. We adjust the parameters of the baselines according to the guidelines of the original paper, so that the model parameters are different for different baselines, but the parameters are the same for each baseline at different sparse. For our MDSCL, we adjust the parameter settings to achieve optimal predictions. We utilize the Adam optimizer with a batch size of 256 and set the learning rate to 0.001. The temperature for the multi-dimensional contrastive learning is set to 0.04 and the weights of contrastive losses are set to 0.1. We use an early stopping technique to train the model.

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

4.2.2 Metrics

We use Mean Absolute Error (MAE) [2] and Root Mean Square Error (RMSE) [2] as the metrics for the evaluation, which are defined as:

$$MAE = \frac{\sum_{i=1}^{n} |real_i - predicted_i|}{n}, RMSE = \sqrt{\frac{\sum_{i=1}^{n} (real_i - predicted_i)^2}{n}}, \tag{14}$$

where n is the number of predicted QoS values i.e. the response time, and $real_i$ and $predicted_i$ are the true and predicted values.

4.2.3 Baselines

We compare the proposed MDSCL with the following sequential prediction models and QoS prediction methods:

- TASR [32] is based on a time-aware service prediction method that combines an enhanced time-aware similarity-based collaborative filtering approach with an ARIMA model.
- CTF [33] is a QoS prediction method based on matrix decomposition, which reduces outliers and increases the robustness of outliers through Corsi loss, while taking the time factor into account in the matrix decomposition.
- RNN [34] is a recurrent neural network for processing serial data.
- LSTM [35] is long short-term memory network that improves the problem of gradient disappearance and gradient explosion in RNN and has long-term dependence.
- Trsfm [36] is based on a self-attentive mechanism deep learning model.
- RNN + Trsfm incorporates the Transfomer mechanism into RNN temporal prediction.

4.3 Performance Comparison

To verify the effectiveness of the proposed MDSCL in QoS prediction task, we compare it with the baseline methods. We use the same training, testing datasets and the same form of data input to run all these methods. Table 1 summarizes the performance of MDSCL and the baseline methods at different sparsity in terms of MAE and RMSE. The best results are shown in bold. Generally, our MDSCL achieves the best prediction accuracy for all sparsity. It shows the adaptability of the realistic QoS environment, which learns sequential features from both user and time dimensionalities and optimizes the prediction model and contrasive learning by the multi-task joint training.

Specifically, our MDSCL consistently outperforms the baselines at each sparsity, especially the extremely sparsity of 1%, with MAE of 0.626 and RMSE of 0.949. When the sparsity is between 1% and 5%, the performance of our method tends to be stable in terms of both metrics, benefiting from the robustness of our model for sparse data processing. When the sparsity exceeds 5%, our method decreases slightly, it is probably because the sequences of both dimensionalities would become complex and noisy when the training data gets denser, which

might increases the probability of randomly overlap cropping invalid sequence. Thus, the proposed MDSCL is more suitable for the situation where the training QoS data is sparse.

It is observed that our method performs better than the most efficient sequential model LSTM which obtains the highest prediction accuracy in all baselines. On the one hand, we improve the feature extraction by combining the WaveNet and BiLSTM. Compared to LSTM, the proposed integrated feature extractor has the stronger extraction capability in handling long sequences as they expand the receptive field and leverage the more contextual information. More importantly, we improve the feature optimization by replacing the traditional prediction task with a joint task of prediction and contrastive learning, so that the final learned features are more representative and thus achieve higher prediction accuracy. In addition, the other time series-based methods, such as RNN, exhibit lower prediction accuracy. A possible reason is that RNN is deficient for extracting long-term dependencies, while QoS data are usually long-term correlated, making it the worst performer with a MAE of 1.505 at 1% sparsity. It also leads to the effect of the combination of RNN and transformer with the highers values of MAE on 4%, 10%, 15%, and 20%, reaching 1.159, 1.233, 1.195, and 1.208, respectively. Traditional-based methods, such as CTF, which relies on matrix decomposition, consistently exhibit lower prediction accuracy on RMSE, especially at a low sparsity of 1% sparsity, with an RMSE of 2.951. However, in general, it can be seen that the time series-based prediction methods consistently outperform the traditional prediction methods.

4.4 Ablation Study

We conduct an ablation study for the proposed MDSCL to explore the performance of each module. Table 2 shows the predicted results of the variants which removes components for multi-dimensional sequential contrastive learning module and joint training one by one.

The results show that the composition of each part leads to the best performance. In particular, both MAE and RMSE are significantly higher after we use the traditional augmentation method, which randomly applies two of crop, mask, and random order to generate positive samples. It indicates that our proposed overlap crop could be used to generate effective positive samples for the multi-dimensional contrastive learning.

In the joint training module, it can be seen that both contrastive losses are effective in reducing MAE and RMSE. Particularly, the contribution of contrastive loss of user dimensionality is more significant than that of time dimensionality. We further validate that removing the contrastive loss function leads to the worse performance. Overall, both contrasive loss of user dimensionality and contrastive loss of time dimensionality are able to facilitate the learning of sequence representations and further improve the prediction performance.

Table 1. Performance comparison for different sparsity (Best results in bold numbers).

Sparsity		TASR	CTF	RNN	Trsfm	RNN + Trsfm	LSTM	MDSCL
1%	MAE	1.107	1.239	1.505	1.193	1.183	1.125	**0.626**
	RMSE	1.897	2.951	2.435	2.233	2.218	2.151	**0.949**
2%	MAE	1.058	1.218	1.314	0.881	1.149	0.882	**0.647**
	RMSE	1.695	2.764	2.256	1.868	2.165	1.848	**0.954**
3%	MAE	1.071	1.197	1.163	0.926	1.118	0.846	**0.646**
	RMSE	1.661	2.653	2.263	1.927	2.219	1.808	**0.956**
4%	MAE	1.077	1.087	1.118	0.935	1.159	0.842	**0.645**
	RMSE	1.655	2.674	2.171	1.921	2.224	1.784	**0.960**
5%	MAE	1.173	1.071	1.124	0.934	1.130	0.846	**0.652**
	RMSE	1.663	2.636	2.228	1.954	2.217	1.810	**0.993**
10%	MAE	1.059	1.045	1.171	0.980	1.233	0.833	**0.732**
	RMSE	1.639	2.593	2.253	1.997	2.360	1.811	**1.111**
15%	MAE	1.132	1.019	1.160	0.963	1.195	0.871	**0.767**
	RMSE	1.605	2.671	2.241	2.002	2.301	1.854	**1.181**
20%	MAE	1.015	1.021	1.166	0.933	1.208	0.842	**0.801**
	RMSE	1.590	2.592	2.252	1.975	2.308	1.800	**1.232**

4.5 Parameter Sensitivity

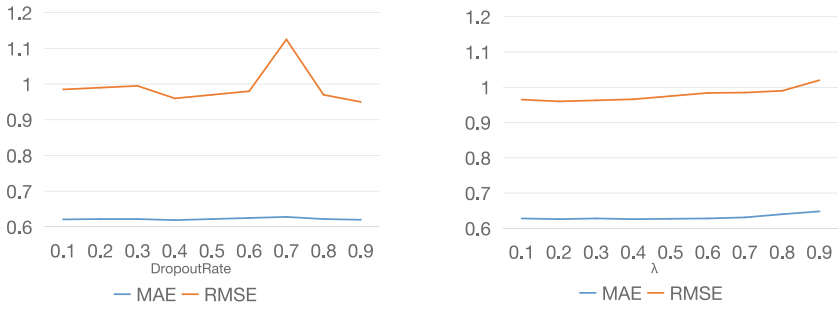
We conduct sensitivity analysis on two hyperparameters, namely *DropoutRate* and λ. *DropoutRate* denotes the hyperparameters in the dropout layer in the module of integrated feature extractor, which decides the percentage of neurons to be randomly deleted. λ is the coefficient of multi-dimensional contrastive loss in the joint training phase. To make the results more convincing, dataset with d of 1% is used to simulate the data in sparse environment.

For *DropoutRate*, Fig. 3(a) shows the results in terms of MAE and RMSE with different parameter values. It can be observed that different *DropoutRate* has essentially no effect on the results for MAE, while RMSE reaches a minimum at 0.4 and then tends to increase gradually, peaking at 0.7. It might be because the dropout of the embeddings of the augmented sequences prevents the model from overfitting and improves the robustness, and the MAE is basically steady. However, when the *DropoutRate* is too large and too much information is lost in the augmented sequences, it leads to a decrease in RMSE. Overall, we choose $DropoutRate = 0.4$ to get the best performance on both metrics.

Table 2. Ablation study of MDSCL on WSDREAM with 1%.

Method		MAE	RMSE
Multi-Dimensional Sequential Contrastive Learning	w/o Overlap Crop	0.661	1.032
	w/o Embedding Drop	0.656	1.030
	Traditional Augmentation	0.684	1.054
	w/o WaveNet	0.640	0.997
	w/o BiLSTM	0.642	1.001
Joint Learning	w/o User Contrastive Loss	0.646	1.011
	w/o Time Contrastive Loss	0.643	0.998
	w/o Contrastive Loss	0.647	1.005
MDSCL		**0.626**	**0.949**

For λ, Fig. 3(b) shows the performance with different λ. It can be seen that with the increase of λ, both MAE and RMSE show a slowly increasing trend. This is because the prediction task still dominates in QoS prediction, and the multi-dimensional contrastive learning task is only an auxiliary task to improve the prediction accuracy by learning the more representative sequence feature embeddings. If the weight of the multi-dimensional contrastive learning task is too high, it would lead to overfitting of the model and thus reduce the prediction performance. Therefore, we choose to set $\lambda = 0.1$ to obtain the optimal prediction results.



(a) Performance with different *DropoutRate*

(b) Performance with different λ

Fig. 3. Parameter sensitivity of *DropoutRate*, λ in MDSCL with d=1%.

5 Conclusion

In this paper, we propose a multi-dimentional sequential contrastive learning framework (MDSCL). Specially, a augmentation strategy of overlap crop is proposed in order to generate positive sequence pairs, and an integrated feature

extractor with an embedding layer with dropout, WaveNe module, and BiLSTM module is designed to capture long short-term sequential features. Finally a contrastive learning framework that considers both the user and time dimensionalities is combined with the original prediction objective to facilitate the task of QoS prediction. The experimental evaluation demonstrate the effectiveness of our MDSCL. In the future, we will further work on the representation of QoS and learn more fine-grained information to improve the accuracy of QoS prediction.

Acknowledgements. This work is supported in part by National Key R&D Program of China under grant 2022YFF0903300, National Natural Science Foundation of China under grant U20A20173 and 62002088, Natural Science Foundation of Zhejiang Province under grant LY22F020009.

References

1. Shao, L., Zhang, J., Wei, Y., Zhao, J., Xie, B., Mei, H.: Personalized qos prediction forweb services via collaborative filtering. In: IEEE International Conference on Web Services (ICWS 2007), pp. 439–446. IEEE (2007)
2. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Qos-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4**(2), 140–152 (2010)
3. White, G., Palade, A., Clarke, S.: Forecasting qos attributes using lstm networks. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2018)
4. Yin, Y., Chen, L., Xu, Y., Wan, J., Zhang, H., Mai, Z.: Qos prediction for service recommendation with deep feature learning in edge computing environment. Mobile Netw. Appli. **25**, 391–401 (2020)
5. Lo, W., Yin, J., Deng, S., Li, Y., Wu, Z.: Collaborative web service qos prediction with location-based regularization. In: 2012 IEEE 19th International Conference on Web Services, pp. 464–471. IEEE (2012)
6. Zhu, X., et al.: Similarity-maintaining privacy preservation and location-aware low-rank matrix factorization for qos prediction based web service recommendation. IEEE Trans. Serv. Comput. **14**(3), 889–902 (2018)
7. Zou, G., et al.: Deeptsqp: temporal-aware service qos prediction via deep neural network and feature integration. Knowl.-Based Syst. **241**, 108062 (2022)
8. Hu, S., Zou, G., Zhang, B., Wu, S., Lin, S., Gan, Y., Chen, Y.: Temporal-aware qos prediction via dynamic graph neural collaborative learning. In: Service-Oriented Computing: 20th International Conference, ICSOC 2022, Seville, Spain, November 29-December 2, 2022, Proceedings, pp. 125–133. Springer (2022). https://doi.org/10.1007/978-3-031-20984-0_8
9. Oord, A.v.d., Dieleman, S., et al.: Wavenet: a generative model for raw audio. arXiv preprint arXiv:1609.03499 (2016)
10. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11), 2673–2681 (1997)
11. Zhang, Y., Zheng, Z., Lyu, M.R.: Wspred: A time-aware personalized qos prediction framework for web services. In: 2011 IEEE 22nd International Symposium on Software Reliability Engineering, pp. 210–219. IEEE (2011)
12. Luo, X., Wu, H., Yuan, H., Zhou, M.: Temporal pattern-aware qos prediction via biased non-negative latent factorization of tensors. IEEE Trans. Cybern. **50**(5), 1798–1809 (2019)

13. Li, M., Hua, Z., Zhao, J., Zou, Y., Xie, B.: ARIMA model-based web services trustworthiness evaluation and prediction. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) ICSOC 2012. LNCS, vol. 7636, pp. 648–655. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34321-6_51
14. Xia, Y., Ding, J., Luo, X., Zhu, Q.: Dependability prediction of ws-bpel service compositions using petri net and time series models. In: 2013 IEEE Seventh International Symposium on Service-Oriented System Engineering, pp. 192–202. IEEE (2013)
15. Rahman, Z.U., Hussain, O.K., Hussain, F.K.: Time series qos forecasting for management of cloud services. In: 2014 Ninth International Conference on Broadband and Wireless Computing, Communication and Applications, pp. 183–190. IEEE (2014)
16. Chen, D., Gao, M., Liu, A., Chen, M., Zhang, Z., Feng, Y.: A recurrent neural network based approach for web service qos prediction. In: 2019 2nd International Conference on Artificial Intelligence and Big Data (ICAIBD), pp. 350–357. IEEE (2019)
17. Keshavarzi, A., Toroghi Haghighat, A., Bohlouli, M.: Online qos prediction in the cloud environments using hybrid time-series data mining approach. Iranian J. Sci. Technol. Trans. Electrical Eng. **45**, 461–478 (2021)
18. Xia, Y., Ding, D., Chang, Z., Li, F.: Joint deep networks based multi-source feature learning for qos prediction. IEEE Trans. Serv. Comput. **15**(4), 2314–2327 (2021)
19. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. arXiv preprint arXiv:1602.07360 (2016)
20. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
21. Chang, Z., Ding, D., Xia, Y.: A graph-based qos prediction approach for web service recommendation. Appli. Intell. 1–15 (2021)
22. Zou, G., Chen, J., He, Q., Li, K.C., Zhang, B., Gan, Y.: Ndmf: neighborhood-integrated deep matrix factorization for service qos prediction. IEEE Trans. Netw. Serv. Manage. **17**(4), 2717–2730 (2020)
23. Xu, J., Xiao, L., Li, Y., Huang, M., Zhuang, Z., Weng, T.H., Liang, W.: Nfmf: neural fusion matrix factorisation for qos prediction in service selection. Connect. Sci. **33**(3), 753–768 (2021)
24. Xiong, R., Wang, J., Li, Z., Li, B., Hung, P.C.: Personalized lstm based matrix factorization for online qos prediction. In: 2018 IEEE International Conference on Web Services (ICWS), pp. 34–41. IEEE (2018)
25. Chen, X., Li, B., Wang, J., Zhao, Y., Xiong, Y.: Integrating emd with multivariate lstm for time series qos prediction. In: 2020 IEEE International Conference on Web Services (ICWS), pp. 58–65. IEEE (2020)
26. Sahu, P., Raghavan, S., Chandrasekaran, K., Usha, D.: Time-aware online QoS Prediction Using LSTM and Non-negative Matrix Factorization. In: Sheth, A., Sinhal, A., Shrivastava, A., Pandey, A.K. (eds.) Intelligent Systems. AIS, pp. 369–376. Springer, Singapore (2021). https://doi.org/10.1007/978-981-16-2248-9_35
27. Georgiadis, S.D., Ranta-aho, P.O., Tarvainen, M.P., Karjalainen, P.A.: Single-trial dynamical estimation of event-related potentials: a kalman filter-based approach. IEEE Trans. Biomed. Eng. **52**(8), 1397–1406 (2005)
28. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: Simclr: a simple framework for contrastive learning of visual representations. In: International Conference on Learning Representations, vol. 2 (2020)

29. Xie, X., et al.: Contrastive learning for sequential recommendation. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE), pp. 1259–1273. IEEE (2022)
30. Siami-Namini, S., Tavakoli, N., Namin, A.S.: The performance of lstm and bilstm in forecasting time series. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 3285–3292. IEEE (2019)
31. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3733–3742 (2018)
32. Ding, S., Li, Y., Wu, D., Zhang, Y., Yang, S.: Time-aware cloud service recommendation using similarity-enhanced collaborative filtering and arima model. Decis. Support Syst. **107**, 103–115 (2018)
33. Ye, F., Lin, Z., Chen, C., Zheng, Z., Huang, H.: Outlier-resilient web service qos prediction. In: Proceedings of the Web Conference 2021, pp. 3099–3110 (2021)
34. Medsker, L.R., Jain, L.: Recurrent neural networks. Design Appli. **5**, 64–67 (2001)
35. Graves, A.: Long short-term memory. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-24797-2_4
36. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30 (2017)