Article

A QoS Prediction Approach Based on Truncated Nuclear Norm Low-Rank Tensor Completion

Hong Xia 1,2,3,*, Qingyi Dong 1, Jiahao Zheng 1, Yanping Chen 1,2,3, Cong Gao 1,2,3 and Zhongmin Wang 1,2,3

1 School of Computer Science and Technology, Xi’an University of Posts and Telecommunications, Xi’an 710121, China
2 Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi’an 710121, China
3 Xi’an Key Laboratory of Big Data and Intelligent Computing, Xi’an 710121, China
* Correspondence: xiahong@xupt.edu.cn

Abstract: With the rise of mobile edge computing (MEC), mobile services with the same or similar functions are gradually increasing. Usually, Quality of Service (QoS) has become an indicator to measure high-quality services. In the real MEC service invocation environment, due to time and network instability factors, users’ QoS data feedback results are limited. Therefore, effectively predicting the QoS value to provide users with high-quality services has become a key issue. In this paper, we propose a truncated nuclear norm Low-rank Tensor Completion method for the QoS data prediction. This method represents complex multivariate QoS data by constructing tensors. Furthermore, the truncated nuclear norm is introduced in the QoS data tensor completion to mine the correlation between QoS data and improve the prediction accuracy. At the same time, the general rate parameter is introduced to control the truncation degree of tensor mode. Finally, the prediction approximate tensor is obtained by the Alternating Direction Multiplier Method iterative optimization algorithm. Numerical experiments are conducted based on the public QoS dataset WS-Dream. The results indicate that our QoS prediction method has better prediction accuracy than other methods under different missing density QoS data.

Keywords: QoS prediction; tensor completion; truncated nuclear norm; collaborative filtering

1. Introduction

With the rapid growth of the Internet of Things (IoT) has placed new demands on cloud computing facilities: low latency, high storage, and high bandwidth [1,2]. These demands become more important as more and more services provided by mobile devices are moved to the cloud for storage and applications. However, present cloud computing [3] is not capable of satisfying these new demands. Mobile edge computing (MEC) [4] is a supplement to cloud computing, and edge servers also serve as service carriers to mobile users. The MEC mainly offers mobile services for mobile users who are close to edge computing servers. It solves the shortcomings of insufficient storage capacity, high latency, and low bandwidth for traditional applications to serve users.

With the fast development of MEC, an increasing number of MEC services with similar functionality are appearing at the edge of the network [5]. The main challenge now is to select the most satisfying and suitable service for the user from a very large number of mobile services. This selection of the most appropriate service from a large number of candidate services is called information overload. Therefore, recommending high-quality services for users has become an urgent need. Service recommendations now take into account two main factors: functional and non-functional. The functional contains detailed functional information of a service. The non-functionality generally refers to Quality of Service (QoS) [6]. With the increase of MEC services with similar functions, the non-functional attributes QoS becomes the key foundation for MEC service recommendation and MEC service composition [7]. MEC service QoS contains many attributes such as failure probability, response time, availability, throughput, price, popularity, and more. However,
in a real MEC service invocation environment, due to time and network instability, users’ QoS data feedback results are limited, which greatly reduces the user satisfaction and denies the benefits of MEC. So far, service recommendation technology based on QoS prediction has been widely used in Web services, effectively solving the problem of information overload [8]. However, studies involving applications in the field of MEC are few. To combat the issue, QoS prediction for MEC services must be conducted [9].

A real taxi service invocation scenario is shown in Figure 1. The left part is the user, the middle is the cloud server and its services, and the right is the major service providers. The table at the bottom shows the QoS generated by user George invoking the service. The server provides corresponding services according to user requests. As an example for George, the user in Figure 1, suppose that George, a mobile user, wants to invoke a car rental service provided by a service provider uploaded to a cloud server. Thus, the edge server near George provides three candidate services Server 1, Server 2, and Server 3 to George’s mobile device phone. For George, who wants to choose a rental car service, it is difficult to choose between the three rental car services. QoS is an important metric to differentiate the quality of service between functionally equivalent MEC services. In a real invocation scenario, due to a large number of candidate services, in the past, users usually invoked a limited number of MEC services and assigned the corresponding QoS values (car rental services are selected based on QoS an attribute response time). The edge server records that previously, George called services Server 1, Server 2, and Server 3 at T1, but called Server 1 and Server 2 at T2, resulting in the QoS value of Server 3 being unavailable at T2. Usually, not every user invokes every service provided by the server at every moment. In the case of sparse QoS data, it is difficult to perform relevant operations on MEC mobile services (service recommendation and service composition) [10]. This paper focuses on how to predict missing QoS data values, because QoS record prediction of missing values is a critical step in operations related to mobile services.

Figure 1. The MEC service invocation scenario.

In recent years, the problem of predicting missing values in service QoS has received extensive attention and is considered a key factor in service selection. Recommending services to users based on QoS data directly depends on the prediction accuracy of the QoS data. At this stage, many QoS-based service recommendation methods have been
Collaborative Filtering (CF) is a fairly mature technology that is widely used in various recommendation scenarios and has achieved good results. In service recommendation applications, CF has been proven to be an effective solution to the problem of information overload in service recommendations. Typical CF methods fall into two categories: neighborhood-based and model-based [11]. The neighborhood-based CF method [12,13] involves simply predicting QoS data based on similar users or services. However, in the real MEC platform invocation service environment, QoS data are often highly sparse, making it challenging to make predictions based on similar users and services. The matrix decomposition method in the model-based CF method [14–16] is to obtain two low-dimensional target matrices by learning latent factors. This approach only considers second-order data structures. Thus, considering the time factor that heavily affects QoS data, tensors need to be introduced to model existing QoS records. Although some studies have applied tensor decomposition to QoS prediction [17–19], these methods still have shortcomings in that they only consider the global structure of existing QoS records and do not consider the local relationships of QoS, so the accuracy of QoS prediction is not ideal. Therefore, for the low-rank characteristic of QoS data, we propose a model-based CF method to solve the shortcomings of existing methods. The main contributions of this paper are summarized as follows:

1. In order to better express complex multivariate QoS data, we add the dimension of time based on the second-order “User-Service” to form a third-order tensor “User-Service-Time” to represent the QoS data. The QoS data tensor with the addition of time information can well express the complex ternary relationships between data;
2. In order to greatly exploit the QoS data correlation between MEC services, we propose the TLTC (Truncated nuclear norm Low-rank Tensor Completion) method to predict the QoS data. As the constructed QoS third-order tensor has low-rank characteristics, the TCTL method approximates the rank of a tensor by the truncated nuclear norm. Meanwhile, a general truncation rate parameter is introduced to control the degree of truncation of the tensor model in order to better analyze the potential characteristics of the QoS data tensor. Finally, the Alternating Direction Method Multiplies (ADMM) method is used for iterative optimization;
3. In order to prove the effectiveness of the QoS data prediction model based on the TLTC method proposed in this paper, we conducted an experimental evaluation on the public dataset WS-Dream. The metrics use Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to evaluate the prediction accuracy. Experimental results show that our QoS data prediction model outperforms other prediction methods.

The organizational framework of this paper is divided into the following sections. Section 2 introduces the related work of QoS data prediction. Section 3 introduces the relevant preparatory knowledge of the tensors involved in this article. Section 4 introduces the QoS data prediction framework and TLTC method proposed in this paper. Section 5 evaluates the proposed method through extensive comparative experiments. Section 6 summarizes the work of the full text.

2. Related Work

In this section, related work on QoS data prediction is briefly reviewed. This section also introduces the concept of Low-rank tensor completion and its application in time-series data prediction.

In recent years, QoS data prediction has received a lot of attention, and many QoS data prediction methods have been developed. Accurate prediction of missing QoS values mostly employs Collaborative Filtering methods. A typical neighborhood-based method is the CF method proposed by UPCC [20], which is based on user neighborhood. It measures the correlation between users through a Pearson’s correlation coefficient and predicts missing values using historical QoS data of similar users. This method only considers the relationship between users and does not associate services or time. Moreover, by analyzing the QoS dataset, it is concluded that the number of users is small compared to the number...
of services, so the overall prediction effect cannot meet the users’ needs. The work in [21] proposes an improved CF algorithm for adjacent user sites. It overcomes the disadvantage of low recall. In order to better capture the correlation between services. The authors of [22] and [23] propose the IPCC and UIPCC methods, respectively. Similar to the UPCC method, the IPCC method uses a parameter to measure the correlation between services. The UIPCC method is a simple combination of the two methods to integrate the prediction results. Furthermore, the QoS value is closely related to the temporal context to introduce time into the QoS prediction. This is also a kind of neighborhood-based method. The authors of [24] proposed a neighborhood-based CF method combined with time series for prediction. This method uses a feedback mechanism to compensate for the deficiencies of the predictive model. The work in [25] proposed a similar user-based CF method to capture temporal features in similarity computation. However, the high sparsity of QoS records in a real MEC environment makes it difficult to select similar users and services at a specific time.

The model-based method is different from the nearest neighbor-based CF method, which requires the use of methods such as machine learning and data mining, and deep mining of potential relationships between data, for example, building a predictive model to predict missing values based on users’ historical preference data for items [26]. Among them, the Probabilistic Matrix Factorization (PMF) [14] method in the model based CF has been successfully applied to the QoS prediction of Web services. The PMF models QoS data as a “User-Service” matrix, where each item in the matrix is a user-generated QoS value for a set of services, which use the Gaussian assumption to decompose the user service quality matrix to predict missing QoS values. The adaptive matrix factorization method is proposed in [27]. This method trains the matrix decomposition model through data transformation and adaptive weighting and predicts the QoS value of user demand services. Although matrix factorization-based QoS prediction methods are generally better than traditional neighborhood-based QoS prediction methods, there are still some shortcomings that do not consider the impact of time factors on QoS data. The non-negative tensor decomposition model is proposed in [18]. A data tensor “user-service-time” model is established using tensor CANDECOMP/PARAFAC (CP) decomposition, and non-negative constraints are added to the decomposition process. The authors of [19] proposed a service prediction method based on tensor decomposition that uses the mean value of the objective function for regularization. The decomposition method based on the tensor decomposition model considers the role of time information. However, in the process of tensor decomposition, the use of tensor decomposition results in a large amount of calculation and data loss. Furthermore, the non-convex relaxation of ranks is intractable, which means it is difficult to recover the CP rank tensor. Therefore, there is still room for improvement in prediction accuracy.

In addition, substantial research has been conducted in data prediction for Low-Rank Tensor Completion (LRTC) methods. Since the “rank” in tensor decomposition is an NP-Hard problem that is difficult to solve, some studies have solved it by introducing the nuclear norm (NN) as the minimized “rank” function. Many variants have been developed on this basis in recent years, but the NN cannot well approximate the “rank” function to achieve the optimal solution [28]. Therefore, the authors of [29] proposed to use the Truncated Nuclear Norm (TNN) to approximate the “rank” for non-convex optimization for tensor completion, and some studies also proved the advantages of using non-convex functions to approximate the “rank” of the tensor [30]. At the same time, the works in [31,32] uses Low-rank tensors to complete the “rank” approximation for data prediction of traffic flow data with temporal and low-rank characteristics and achieved good results. For example, in [33], the TNN approximate tensor “rank” method is used for prediction. Truncated nuclear norm regularization is leveraged to mine structural attributes of tensors and correlations between adjacent elements.

Overall, the fundamental challenge of missing data prediction is to describe correlations and dependencies in the data. Time series data are unique because of their strong similarity and periodicity, and time-series data have a certain temporal correlation and low-rank properties [34]. To better capture QoS correlations and take advantage of the
global structure across time, data-missing values are predicted by building the data into tensors. In general, the original second-order “User-Service” matrix is transformed into a third-order “User-Service-Time” tensor by introducing “time” as the time dimension. Inspired by the tensor completion in the prediction of time-series data, we propose a TLTC QoS data prediction method for the low-rank characteristics of QoS data. Low-rank tensor completion methods have demonstrated better predictive accuracy than other methods [35].

3. Preliminaries

This section introduces the concept of tensors and some operations on tensors. The evolution of the basic models of LRMC and LRTC is then detailed.

3.1. Expression of Tensors

Tensors are higher-order generalizations of arrays, and in recent years, they have emerged as powerful analytical tools for data representation. Now, the use of tensors to express and process high-dimensional data has received extensive attention. It can better express the complex relationships of multivariate data. This section only introduces the background knowledge of tensors used in the paper. Generally, a vector is a first-order tensor, which is represented by bold and lowercase letters \( \mathbf{x}, \mathbf{y} \). A matrix is a second-order tensor, which is represented by \( \mathbf{X}, \mathbf{Y} \), such as matrix \( \mathbf{X} \in \mathbb{R}^{I_1 \times I_2} \). Scalars denote \( x, y \) with lowercase letters. The tensors can be represented by bold Euler letters \( \mathbf{X}, \mathbf{Y} \), then an \( N \)-order tensor can be represented as \( \mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times \cdots \times I_N} \).

Tensor unfold: The tensor unfolds along the \( n \)-th called tensor matricization. Tensor matricization is the unfolding of tensors into a matrix format with a predefined pattern ordering. Figure 2 shows unfolding of the tensor, matrices \( \mathbf{X}^{(1)} \in \mathbb{R}^{I_1 \times I_2 I_3}, \mathbf{X}^{(2)} \in \mathbb{R}^{I_2 I_3 I_1} \) and \( \mathbf{X}^{(3)} \in \mathbb{R}^{I_3 I_1 I_2} \) are tensors respectively, 1, 2, 3 modules are unfolded. At the same time, \( \text{fold}(\cdot) \) is used to convert a matrix into a \( k \)-th order high-order tensor \( \text{fold}_k(\mathbf{X}^{(k)}) = \mathbf{X} \).

Frobenius norm: Assuming a matrix \( \mathbf{X} \) is given, its Frobenius norm is defined as \( \| \mathbf{X} \|_F = \sqrt{\sum_{ij} x_{ij}^2} \), which is the square root of the sum of all elements. A tensor is an extension of a matrix, given a tensor \( \mathbf{X} \), then the Frobenius norm is defined as \( \| \mathbf{X} \|_F = \sqrt{\sum_{i_1,i_2,\ldots,i_N} x_{i_1,i_2,\ldots,i_N}^2} \).

Tensor inner product: Given two tensors, when the order of the two tensors is the same, the inner product is a number. Thus follows Equation (1):

\[
L = \sum_{i_1, i_2, \ldots, i_N} x_{i_1,i_2,\ldots,i_N} y_{i_1,i_2,\ldots,i_N}
\]
3.2. Low-Rank Matrix Completion

First, we introduce the concept of Low-rank matrix completion (LRMC). For a partially observable matrix \( Y \in \mathbb{R}^{I_1 \times I_2} \), the LRMC model is presented in Equation (2):

\[
\min_X \text{rank}(X) \\
\text{s.t.} X_{\Omega} = Y_{\Omega}
\]  

where \( X_{\Omega} \in \mathbb{R}^{I_1 \times I_2} \) is the recovered matrix that we hope to find, and \( \Omega \) is the index set of observed elements, where \( \text{rank}(X) \) is the algebraic rank of the matrix \( X \). The matrix rank minimization problem in the formula is NP-Hard, as the function is a non-convex function. The most common method is to use the NN to approximate the rank of the matrix, which can be transformed into a convex optimization problem. Equation (2) can be transformed into Equation (3):

\[
\min_X : \|X\|_* \\
\text{s.t.} X_{\Omega} = Y_{\Omega}
\]

where \( \|X\|_* = \sum_{i=1}^{\min(m,n)} \delta_i \) is the nuclear norm, \( \delta_i \) is i-th singular value of the matrix.

3.3. Low-Rank Tensor Completion

Tensors are generalized from higher-order matrices. The Low-rank tensor completion (LRTC) is a general tensor completion method. The prior condition is to assume that the elements of this tensor are of low rank in the partial observations. This is the same as the LRMC algorithm. Given a third-order tensor \( Y \in \mathbb{R}^{U \times S \times T} \), the model of LRTC is:

\[
\min_{\mathcal{X}} \text{rank}(\mathcal{X}) \\
\text{s.t.} \mathcal{P}_{\Omega}(\mathcal{X}) = \mathcal{P}_{\Omega}(Y)
\]

\( \mathcal{X} \in \mathbb{R}^{U \times S \times T} \) is the tensor to be recovered, and \( \Omega \) is the index set of observed entries. For tensors, the minimization of tensor rank is an NP-Hard problem that cannot be computed, just like the minimization of matrix rank. To solve this problem, the same solution as matrix completion uses the NN approximation to minimize. Therefore, the above Equation (4) can be transformed into Equation (5):

\[
\min_{\mathcal{X}} : \|\mathcal{X}\|_* \\
\text{s.t.} \mathcal{P}_{\Omega}(\mathcal{X}) = \mathcal{P}_{\Omega}(Y)
\]

\( \mathcal{P}(\mathcal{X}) \) represents the weight parameter.

4. Prediction Framework And Method

4.1. Prediction Framework

In the real MEC mobile service application environment, the QoS data and the value declared by the provider will be slightly different. Usually, due to time and network constraints, only limited QoS data are available to each user calling the service. It is difficult to provide users with high-quality services and accurate recommendations using these missing QoS data. Therefore, we propose a TLTC-based QoS prediction framework for the low-rank characteristics of QoS data to predict the lost values of QoS data. Figure 3 shows the QoS data prediction framework model proposed in this paper. First, the user obtains sparse QoS data from the server. After data processing, a tensor data model is constructed. After the training and optimization of the TLTC model, the complete recovered QoS data are obtained. Then, the predicted QoS values are sorted, and the service that meets the user’s needs is recommended to the user.
When a user invokes a service, it takes time to accumulate. The QoS values of the services invoked by users are recorded to form a series of QoS datasets. The QoS attributes data set is represented as \(<\text{UserID}, \text{ServiceID}, \text{TimeID}, \text{Value}\>\). As shown in Figure 3, the collected data are sparse. As shown in Figure 4, we construct the QoS data into a third-order QoS data tensor. Specifically, it is assumed that the user whose User ID is 0 invokes the service whose Service ID is 0 when the Time ID is 0 and 2. The user whose User ID is 0 invokes the service whose Service ID is 2 when the Time ID is 0, 1, and 2. Therefore, the QoS values obtained by the same service invoked by the user in multiple time intervals can be stored in the data tensor model as time series. The obtained QoS data are constructed as a third-order tensor \(Y \in \mathbb{R}^{U \times S \times T}\), and the missing values are filled with 0. The specific construction of the tensor algorithm is as in Algorithm 1. The key task of our paper is to predict the unknown QoS value in the above third-order tensor.

![Figure 3. QoS prediction model based on TLTC.](image)

![Figure 4. Construction of "User-Service-Time" Tensor Model.](image)
Algorithm 1 presents the construction of the QoS data tensor model. The input is the collected sparse QoS data quadruple \(< UserID, ServiceID, TimeID, Value >\). The output is the QoS data tensor “User-Service-Time” \(Y \in \mathbb{R}^{U \times S \times T}\) with time information added.

**Algorithm 1:** Tensor Construction “User-Service-Time”.

1: Load all QoS value quadruplets (user, service, time, value).
2: Use the set of (user, service, 1, value) construct a (user, service) matrix \(N^{(1)}\) that takes all \(I\) users as the rows and all \(J\) services as the columns at the time interval 1.
3: Set the elements of the matrix \(N^{(1)}\) value with QoS value.
4: Reconstruct all the matrices \(N^{(1)}, N^{(2)}, \cdots, N^{(K)}\) for \(K\) time periods.
5: Return \(Y \in \mathbb{R}^{U \times S \times T}\)

4.2. Method

The TLTC method is derived from the matrix truncated nuclear norm as a basis. Let us first introduce the definition of the TNN of a matrix. Suppose a matrix \(Z \in \mathbb{R}^{I_1 \times I_2}\) and a non-negative integer \(r < \min(I_1, I_2)\) are given, \(r\) is the rank of the matrix. The TNN of the matrix \(Z\) can be defined as the sum of the smallest singular values of \(\min(I_1, I_2) - r\). In this definition, large singular values 1 to \(r\) in the matrix are not taken into account in truncation, as in Equation (6).

\[
\|Z\|_{r,\ast} = \sum_{i=r+1}^{\min(I_1,I_2)} \delta_i(Z) \tag{6}
\]

where \(\delta_i(Z)\) represents the \(i\)-th singular value of the matrix. The collation to follow is: \(\delta_1 \geq \delta_2 \geq \delta_3 \geq \cdots \geq \delta_{\min\{m,n\}}\).

The definition of TNN for matrices cannot be directly applied to high-dimensional tensors. Therefore, we refer to [28,33] for the definition of tensor truncation nuclear norm. As the basis of LRTC for TNN minimization, the truncation of each tensor mode will be automatically assigned if the rate parameter can be set appropriately. For the tensor \(Z\), the TNN definition model of the tensor follows Equation (7):

\[
\|Z\|_{\gamma,\ast} = \sum_{k=1}^{d} \alpha_k \|Z^{(k)}\|_{r_k,\ast} \tag{7}
\]

where the truncation for each tensor mode is: \(r_k = \lceil \gamma \cdot \min\{n_k, \prod_{h \neq k} n_h\} \rceil, k \in \{1,2,3,\cdots,d\}\).

Where \(d\) represents the mode of tensor unfold. The parameter \(\gamma\) is a general rate parameter that controls the degree of truncation of the tensor \(d\) modulo, where \(r_k\) should be satisfied with \(1 \leq r_k \leq \min\{n_k, \prod_{h \neq k} n_h\}\), and \(\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_d\) represents the weight parameters for the tensor \(Z\) when it is expanded into a matrix to calculate TNN.

According to the definition of tensor TNN, the model of TLTC for QoS data tensor \(Y \in \mathbb{R}^{U \times S \times T}\) can be into Equation (8):

\[
\min_{\mathcal{Z}} \sum_{k=1}^{d} \alpha_k \|Z^{(k)}\|_{r_k,\ast}
\]

\[s.t. \mathcal{P}_\Omega(\mathcal{Z}) = \mathcal{P}_\Omega(\mathcal{Y})\]

Prevent the dependency of its variables from being guaranteed when the tensor is unfolded. Add an extra set of constraints to this. Introduce auxiliary tensor \(G\). Under certain conditions \(Z_k = G, k = 1,2,3\). For this, the TLTC model is transformed following Equation (9):
The ADMM framework optimally solves each variable similarly. Derive the augmented observation information at each iteration.

\[ \min_{\mathcal{G}, \mathcal{Z}, \mathcal{T}} \frac{3}{k=1} \left( a_k \| \mathcal{Z}(k) \|_{F}^2 + \frac{\rho_k}{2} \| \mathcal{Z}(k) - \mathcal{G}(k) \|_F^2 \right) \]

(9)

For the optimization algorithm, the ADMM algorithm is used for iterative optimization. The ADMM framework optimally solves each variable similarly. Derive the augmented Lagrangian function of formula of Equation (10).

\[ \mathcal{L}(\mathcal{G}, \{ \mathcal{Z}_k, \mathcal{T}_k \}) = \frac{3}{k=1} \left( a_k \| \mathcal{Z}(k) \|_{F}^2 + \frac{\rho_k}{2} \| \mathcal{Z}(k) - \mathcal{G}(k) \|_F^2 \right) \]

(10)

Therefore, ADMM optimization iteration has converted the tensor prediction problem into the following three solving problems:

\[ \mathcal{Z}^{l+1}_{k} = \arg \min_{\mathcal{Z}} \mathcal{L}(\mathcal{G}, \{ \mathcal{Z}^{l+1}_{k}, \mathcal{T}^{l}_{k} \}) \]

\[ \mathcal{G}^{l+1}_{k} = \arg \min_{\mathcal{G}} \mathcal{L}(\mathcal{G}, \{ \mathcal{Z}^{l+1}_{k}, \mathcal{T}^{l}_{k} \}) \]

\[ \mathcal{T}^{l+1}_{k} = \mathcal{T}^{l}_{k} + \rho_k(\mathcal{Z}^{l+1}_{k} - \mathcal{G}^{l+1}_{k}) \]

(11)

The relationship order of these three variables is derived as: \( \mathcal{Z}^{l+1}_1 \rightarrow \ldots \rightarrow \mathcal{Z}^{l+1}_3 \rightarrow \mathcal{G}^{l+1}_1 \rightarrow \mathcal{T}^{l+1}_1 \rightarrow \ldots \rightarrow \mathcal{T}^{l+1}_3 \). Let Equation (11) \( \rho_k = \rho \), the variable \( \mathcal{T}^{l+1}_{k} \) is converted to: \( \mathcal{T}^{l+1} := \mathcal{T}^{l} + \rho(\mathcal{Z}^{l+1} - \mathcal{G}^{l+1}) \). Where \( \mathcal{T}, \mathcal{Z}, \mathcal{G} \) is a fourth-order tensor. Please refer to the [33] if there are too many explanations.

In the following, we provide the derivation formulae for the variables \( \mathcal{Z}^{l+1}_{k} \) and \( \mathcal{G}^{l+1}_{k} \).

\[ \mathcal{Z}^{l+1}_{k} = \arg \min_{\mathcal{Z}} a_k \| \mathcal{Z}(k) \|_{F}^2 + \frac{\rho_k}{2} \| \mathcal{Z}(k) - \mathcal{G}(k) \|_F^2 + \left\langle \mathcal{Z}(k), \mathcal{T}(k) \right\rangle \]

\[ \mathcal{G}^{l+1}_{k} = \arg \min_{\mathcal{G}} a_k \| \mathcal{Z}(k) \|_{F}^2 + \frac{\rho_k}{2} \| \mathcal{Z}(k) - \mathcal{G}(k) \|_F^2 + \left\langle \mathcal{G}(k), \mathcal{T}(k) \right\rangle \]

(12)

According to the formula in Equation (12), we can iteratively obtain \( \mathcal{Z}^{l+1}_{k} \). However, the final \( G(\mathcal{Z}(k)) \) is a non-convex function, and the optimal solution cannot be achieved by using the usual solution. We refer to [33] to derive Equation (13).

\[ \mathcal{Z}^{l+1}_{k} := f o l d_k(Udiag(\sigma(\mathcal{Z}(k)))V^T) \]

(13)

The \( Udiag(\sigma(\mathcal{Z}(k)))V^T \) denotes the singular value decomposition of the tensor unfold matrix, where \( U \) and \( V \) are a matrix.

\[ \mathcal{G}^{l+1}_{k} := \arg \min_{\mathcal{G}} \sum_{k=1}^{3} \left( \frac{\rho_k}{2} \| \mathcal{Z}^{l+1}_{k}(k) - \mathcal{G}(k) \|_F^2 - \left\langle \mathcal{G}(k), \mathcal{T}(k) \right\rangle \right) \]

\[ \mathcal{G}^{l+1}_{k} := \frac{1}{\sum_{k=1}^{3} \rho_k} \sum_{k=1}^{3} \left( \rho_k \mathcal{Z}^{l+1}_{k} + \mathcal{T}(k) \right) \]

(14)

where \( \mathcal{P}_\Omega(\mathcal{G}^{l+1}) = \mathcal{P}_\Omega(\mathcal{Y}) \) is a fixed constraint, in order to ensure the transformation of observation information at each iteration.

We described the TLTC model and its ADMM optimization algorithm in detail above, as shown in Algorithm 2.
Algorithm 2: TLTC Optimization Imputation Algorithm.

1: Initialize $\mathcal{P}_Ω(G_0^k) = \mathcal{P}_Ω(Z)$.
2: Set parameter $\rho_k, a_k, \gamma, l = 0, \mathcal{T}^0 = 0$.
3: While not converged do
4: $\rho = \min\{1.05 \times \rho, \rho_{\text{max}}\}$
5: For $k = 1, 2, 3$ do
6: Update $Z_{l+1}^k \leftarrow \text{Equation (13)}$
7: EndFor
8: Update $\mathcal{T}^{l+1}$
9: Update $G^{l+1} \leftarrow \text{Equation (14)}$
10: $l = l + 1$
11: EndWhile
12: Return $\hat{Y} \in \mathbb{R}^{U \times S \times T}$

5. Experiment Description

In this section, we present experiments on the QoS data of MEC to verify the prediction accuracy of the TLTC method. The QoS data set uses Throughput attributes and Response Time attributes. We use the evaluation metrics MAE and RMSE. Experiments were compared with seven methods in the case of different data density missing rates. At the same time, we experimentally analyze the influence of integer truncation rate parameters and tensor density on the experimental results. As the experimental operations are randomly removed from the data, this affects the stability of the results. In order to avoid the contingency of the experiment, each experiment was run ten times and the average value was taken as the result.

5.1. Database And Baseline Models

We use the well-known large-scale data WS-Dream for experiments. We preprocessed the original dataset and constructed a third-order tensor data structure, “User-Service-Time”. The dataset records the Throughput and Response Time data when 142 users invoke 4532 services in 64 intervals (15 min intervals). Both datasets contain 30,287,611 records. The response time indicates the time duration between the service user sending the request and receiving the corresponding response, whereas throughput is indicated as the average rate of successful message size delivery per second over the communication channel. The detailed statistical information about the QoS attributes data is shown in Table 1. This paper mainly studies two representative attributes of Throughput and Response Time in QoS data. However, it can be used to directly predict any other QoS attribute without modification.

| Statistics       | Throughput      | Response Time |
|------------------|-----------------|---------------|
| Scale of QoS values | 0–1000 kbps    | 0–20 s        |
| Mean of QoS values  | 9,609 kbps     | 3.165 s       |
| Standard Deviation | 50.11 s         | 6.12 s        |
| Num. of Users     | 142             | 142           |
| Num. of Services  | 4532            | 4532          |
| Num. of Time Intervals | 64           | 64            |
| Interval of Time Slots | 15 min       | 15 min        |
| Num. of Records   | 30,287,611      | 30,287,611    |

In order to better approximate the real calling environment in this paper, we need to delete the QoS dataset randomly to ensure the sparsity of its QoS data. The detailed description is shown in Table 2. In this paper, the density of the training dataset is set between 10% and 30% in increments of 5%. For fairness of the evaluation, we assign the same initial assumptions to the model, and under the same training and testing datasets,
MAE and RSME indicators are used to compare the error between the predicted value and the original value.

Table 2. QoS data preprocessing.

| No. | Train/Test | Training Data  | Testing Data    |
|-----|------------|----------------|-----------------|
| 1   | 10%:90%    | 3,028,761      | 27,258,850      |
| 2   | 15%:85%    | 4,543,142      | 25,744,469      |
| 3   | 20%:80%    | 6,057,522      | 24,230,089      |
| 4   | 25%:75%    | 7,571,903      | 22,715,708      |
| 5   | 30%:70%    | 9,086,283      | 21,201,328      |

In order to verify the effectiveness of the algorithm, we choose the most classic seven QoS prediction algorithms to compare the prediction accuracy on the same dataset. The following baseline models are compared: UPCC [20], IPCC [22], UIPCC [23], PMF [24], NTF [18], WSPred [19], CLUS [36].

5.2. Evaluation Metrics

In order to accurately evaluate the prediction criteria of QoS, the MAE and RMSE are used as the criteria to compare with other baselines. The MAE formula and RMSE formula are Equation (15) [37]:

\[
MAE = \frac{1}{N} \sum_{i,j,k} | y_{ijk} - \hat{y}_{ijk} | ,
\]

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i,j,k} (y_{ijk} - \hat{y}_{ijk})^2} .
\]

where \( y_{ijk} \) represents the real QoS value of the test data set, true QoS value of user \( i \) calling service \( j \) at interval \( k \). \( \hat{y}_{ijk} \) represents the predicted QoS value of the training data set after model training. The smaller the MAE and RMSE values, the better the prediction effect.

5.3. Parameter Settings

The weight parameter \( \alpha_k \), the truncation rate parameter \( \gamma \) and the learning rate \( \rho \) in ADMM, these three parameters will directly affect the prediction accuracy of the model. The setting of the truncation rate parameter \( \gamma \) can be obtained by cross-validation. The low-rank tensor is feasible for processing time-series data with the same parameter selection as in [31], using cross-validation for selection. We also performed a local analysis while performing cross-validation to select parameters. If cross-validation is used, selecting the appropriate weight parameter \( \alpha_k \) requires extensive computation. We refer to the parameter setting of the HaLRTC model in [28] and set the parameter to \( \alpha_1 = \alpha_2 = \alpha_3 = 1/3 \). The \( \rho \) parameter is learned in ADMM, which determines the convergence of the entire model. We set \( \rho = 1 \times 10^{-5} \) and \( \rho = 1 \times 10^{-4} \) in the two QoS attribute datasets of Throughput and Response Time, respectively. The maximum number of iterations is set to 20 and 50 to achieve convergence.

5.4. Analysis of Results

In the comparison experiments, we set the initial conditions of all algorithmic models to be the same and use the same evaluation criteria and datasets. The Neighborhood-based CF methods such as UIPCC, IPCC and IPCC set their neighborhood users to 10 and the number of neighborhood services to 50. The latent factor matrix dimension is set to 20 in tensor factorization or matrix factorization. Some methods are unable to construct tensors for testing. We use the same special treatment as in [18] to compress the tensor into a “user-service” matrix. We calculated MAE and RMSE for these methods and compared them with our TLTC method.
The setting of the truncation rate parameter is selected by cross-validation. However, the prior knowledge of QoS data is relatively small. In order to study the effect of the truncation rate parameter on the performance of the algorithm, the global optimum is guaranteed. This paper refers to the parameter settings of time-series data processing in [26,31]. As shown in Figure 5, we will study the effect of the truncation rate parameter on the prediction accuracy. The tensor density is adjusted from 10% to 30% with a step of 5%. We observe the MAE and RMSE values of Throughput and Response Time for QoS data at different tensor densities.

Figure 5 presents the experimental results of the MAE and RMSE of the Throughput. Figure 5c,d presents the experimental results of the MAE and RMSE of the Response Time. First, for the QoS dataset of Throughput attributes, the truncation rate parameters 0.01, 0.1, 0.15, 0.2, 0.25 were used for testing. The number of iterations was chosen to be 20. The experiment shows that when the integer truncation value of the tensor is 0.01, MAE decreases more stably than other values when the density of the tensor is between 10% and 30%. It shows that the difference value is handled well. Moreover, the obtained MAE value is relatively low, indicating that the prediction effect is better when the truncation rate parameter value is 0.01. For the value of RMSE, when the truncation rate parameter is 0.01, although the value is not ideal, the convergence rate is faster than other values. In overall evaluation, the prediction effect is the best when the value of the truncation rate parameter is 0.01. Although the MAE value of the Throughput is relatively low when the value of the truncation rate parameter is 0.1, its RMSE value is relatively high. We should expect MAE values to be much smaller than RMSE values.

For the QoS dataset with Response time attributes, as shown in Figure 5c,d, the parameters 0.01, 0.1, 0.15, 0.2, 0.25 were used for testing, and the number of iterations was 50 converged. It can be observed from the experimental results that, as for the Throughput QoS dataset, when the truncation rate parameter is set to 0.01, the values of MAE and RMSE are the lowest. Therefore, the prediction effect is the most ideal.

Experimental results show that under different tensor densities, the prediction accuracy of our TLTC method has smaller RMSE and MAE values when the truncation rate parameter is 0.01. According to prior experience, the truncation rate parameter is locally verified to achieve the desired prediction accuracy and save a lot of computational costs. With the increase of tensor density, when the truncation rate parameter is 0.01, the prediction accuracy of the TLTC method gradually increases. This means that providing more QoS data can lead to higher prediction accuracy.

Experimental results show the effectiveness of our TLTC method. We select seven classical prediction methods to compare with the QoS data prediction methods of our TLTC model. In the same training and testing sets, the evaluation metrics MAE and RMSE of the prediction accuracy of these methods are obtained in this paper through extensive experiments as shown in Figure 6. Figure 6a,b shows the results of MAE and RMSE values for the response time. Figure 6c,d presents the MAE and RMSE results of Throughput. As can be seen from Figure 6, the neighborhood-based CF methods UPCC, IPCC, UIPCC and matrix factorization PMF methods have lower prediction accuracy than the WSPred, NTF, and TLTC methods using tensor models. Other methods only utilize the second-order static relationship of the “User-Service” model and do not consider the time information in the “User-Service-Time” model. For example, the TLTC framework proposed in this paper significantly improves the prediction accuracy of QoS data, and obtains smaller MAE and RMSE values for both Response Time and Throughput under different QoS data missing densities. For methods that also use the tensor model WSPred and NTF, they also add time information. However, when using tensor decomposition for predicting QoS data, it does not pay much attention to the data loss due to decomposition during prediction, and the prediction accuracy is much worse than expected. We adopt the method of low-rank tensor completion, using the automatic unfold tensor mode with TNN, it can better focus on strong temporal correlation between QoS data. Therefore, the prediction accuracy of our proposed TLTC method is higher than that of other methods. In addition, because the range of Throughput is 0–1000 kbps, and the range of response time is only 0–20 s,
the MAE and RMSE for Throughput are much larger than the MAE and RMSE values for Response Time.

![Graphs showing MAE and RMSE values for Throughput and Response Time](image)

**Figure 5.** The effect of the truncation rate parameter.

The experiments show that the prediction accuracy is positively correlated with the tensor density of QoS data. It can be seen from Figure 6 that the density of the training tensor is changed from 10% to 30%, and the step size increases by 5%, the prediction accuracy of method also gradually changes. The prediction accuracy of the TLTC method increases with the increase of training density. This is because the greater the density of QoS data which provides more useful information, the better the prediction effect of the method.
6. Conclusions

In this paper, to better express the global structure of QoS data in the real MEC environment, we add time information to the current QoS data to construct a “User-Service-Time” third-order tensor. To this end, for the low-rank properties of the QoS data tensor, we propose a TLTC method to predict the missing values of QoS data. The TLTC method avoids the disadvantages of data loss and ignoring local data caused by traditional tensor decomposition. The method can better capture the temporal correlation between different users and different services by regularizing the TNN. Considering the hidden correlation of QoS data, a general rate parameter is introduced to control the truncation degree of all tensor patterns. Finally, the low-rank estimate of the target tensor is obtained to predict the missing values of the QoS data using ADMM iterative optimization of the approximation tensor. In this paper, experimental studies on the public large-scale QoS dataset WS-Dream indicate that our method outperforms other methods with higher prediction accuracy.

Author Contributions: All authors participated in some part of the work for this article. Investigation, H.X.; methodology, H.X., Q.D.; software, Q.D., H.X.; supervision, Y.C.; writing—original draft preparation, H.X. and Q.D.; writing—review and editing, J.Z., Z.W. and C.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by Science and Technology Project in Shaanxi Province of China (Program No. 2019ZDLGY07-08), Natural Science Basic Research Program of Shaanxi Province, China (Grant No. 2020JM-582), Scientific Research Program Funded by Shaanxi Provincial Education Department (No. 21JP115), Natural Science Basic Research Program of Shaanxi (Program No. 2021JQ-719), Xi’an University of Posts and Telecommunications Key Innovation Fund Project of Science and
Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Acknowledgments: The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ngu, A.H.; Gutierrez, M.; Metsis, V.; Nepal, S.; Sheng, Q.Z. IoT middleware: A survey on issues and enabling technologies. *IEEE Internet Things J.* 2017, 4, 1–20. [CrossRef]

2. Tran, T.; Hosseini, M.; Pompili, D. Mobile edge computing: Recent efforts and five key research directions. *IEEE Commun. Mag.* 2017, 12, 29–33. [CrossRef]

3. Ren, T.F.; Zhou, M.T.; Yu, C.B.; Shen, F.; Li, J.; He, W. An Integrated Edge and Cloud Computing Platform for Multi-Industrial Applications. In Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, Beijing, China, 15 July 2021; pp. 212–215.

4. Li, H.; Shou, G.; Hu, Y.; Guo, Z. Mobile Edge Computing: Progress and Challenges. In Proceedings of the 2016 4th IEEE International Conference on Mobile Cloud Computing, Services and Engineering (Mobile Cloud), Oxford, UK, 29 March 2016; pp. 83–84.

5. Liu, Z.Z.; Sheng, Q.Z.; Xu, X.; Chu, D.; Zhang, W.E. Context-Aware and Adaptive QoS Prediction for Mobile Edge Computing Services. *IEEE Trans. Serv. Comput.* 2022, 15, 400–413.

6. Huo, Y.; Qiu, P.; Zhai, J.; Fan, D.; Peng, H. Multi-objective service composition model based on cost-effective optimization. *Appl. Intell.* 2018, 48, 651–669.

7. Yin, Y.; Xu, H.; Liang, T.; Chen, M.; Gao, H.; Longo, A. Leveraging Data Augmentation for Service QoS Prediction in Cyber-physical Systems. *ACM Trans. Internet Technol.* 2021, 21, 1–25. [CrossRef]

8. Wang, S.; Huang, L.; Hsu, C.H.; Yang, F. Collaboration reputation for trustworthy Web service selection in social networks. *J. Comput. Syst. Sci.* 2016, 82, 130–143. [CrossRef]

9. Wang, S.; Zhao, Y.; Huang, L.; Xu, J.; Hsu, C.H. QoS prediction for service recommendations in mobile edge computing. *J. Parallel Distrib. Comput.* 2019, 127, 134–144. [CrossRef]

10. Li, S.; Wen, J.; Wang, X. From reputation perspective: A hybrid matrix factorization for qos prediction in location-aware mobile service recommendation system. *Mob. Inf. Syst.* 2019, 8950508. [CrossRef]

11. Chen, Z.; Shen, L.; Li, F.; You, D.; Mapetu, J.P.B. Web service QoS prediction: When collaborative filtering meets data fluctuating in big-range. *World Wide Web* 2020, 23, 1715–1740. [CrossRef]

12. Zhang, L.C.; Li, C.J.; Yu, Z.L. Dynamic Web service selection group decision-making based on heterogeneous QoS models. *J. China Univ. Posts Telecommun.* 2012, 19, 80–90. [CrossRef]

13. Sun, X.; Wang, S.; Xia, Y.; Zheng, W. Predictive-trend-aware composition of web services with time-varying Quality-Of-Service. *IEEE Access* 2020, 8, 1910–1921. [CrossRef]

14. Mnih, A.; Salakhutdinov, R. Probabilistic matrix factorization. In Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS ’07, Vancouver, BC, Canada, 3–6 December 2007; pp. 1257–1264. [CrossRef]

15. 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. Manag.* 2020, 17, 2717–2730. [CrossRef]

16. Zhu, X.; Jing, X.Y.; Wu, D.; He, Z.; Cao, J.; Yue, D.; Wang, L. Similarity-maintaining privacy preservation and location-aware low-rank matrix factorization for QoS prediction based web service recommendation. *IEEE Trans. Serv. Comput.* 2021, 14, 889–902. [CrossRef]

17. Wang, Q.; Chen, M.; Shang, M.; Luo, X. A momentum-incorporated latent factorization of tensors model for temporal-aware QoS missing data prediction. *Neurocomputing* 2019, 367, 299–307. [CrossRef]

18. Zhang, W.; Sun, H.; Liu, X.; Guo, X. Temporal QoE-aware web service recommendation via non-negative tensor factorization. In Proceedings of the WWW ’14: 23rd International World Wide Web Conference, Seoul, Korea, 7–11 April 2014; pp. 585–596.

19. Zhang, Y.; Zheng, Z.; Lyu, M.R. WS-Pred: A time-aware personalized QoS prediction framework for web services. In Proceedings of the 2011 IEEE International Symposium on Software Reliability Engineering, Washington, DC, USA, 29 November 2011; pp. 210–219. [CrossRef]

20. Shao, L.; Zhang, J.; Wei, Y.; Zhao, J.; Xie, B.; Mei, H. Personalized QoS prediction for web services via collaborative filtering. In Proceedings of the 2007 IEEE International Conference on Web Services, Salt Lake City, UT, USA, 9 July 2007; pp. 439–446. [CrossRef]
21. Zhang, K.; Wenwu, Z.; Li, H. Improved Collaborative Filtering Algorithm Based on Network Site Users. In Proceedings of the 2021 2nd International Conference on Big Data and Informatization Education, Hangzhou, China, 2–4 April 2021; pp. 212–215. [CrossRef]

22. Chung, K.Y.; Lee, D.; Kim, K.J. Categorization for grouping associative items mining in Item-based collaborative filtering. In Proceedings of the 2011 International Conference on Information Science and Applications, Jeju, Korea, 26 April 2011; pp. 1–6.

23. Zheng, Z.; Ma, H.; Lyu, M.R.; King, I. QoS-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. 2011, 4, 140–152.

24. Yan, H.; Peng, Q.; Hu, X.; Rong, Y. Web service recommendation based on time series forecasting and collaborative filtering. In Proceedings of the 2015 IEEE International Conference on Web Services, Washington, DC, USA, 27 June 2015; pp. 233–240.

25. 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. 2018, 107, 103–115.

26. Zheng, W.; Wang, Y.; Xia, Y.; Wu, Q.; Wu, L.; Guo, K.; Li, W.; Luo, X.; Zhu, Q. On dynamic performance estimation of fault-prone Infrastructure-as-a-Service clouds. Int. J. Distrib. Sens. Netw. 2017, 13, 1550147717718514.

27. Zhu, J.; He, P.; Zheng, Z.; Lyu, M.R. Online QoS prediction for runtime service adaptation via adaptive matrix factorization. IEEE Trans. Parallel Distrib. Syst. 2017, 28, 2911–2924. [CrossRef]

28. Liu, J.; Musialski, P.; Wonka, P.; Ye, J. Tensor completion for estimating missing values in visual data. IEEE Trans. Pattern Anal. Mach. Intell. 2013, 35, 208–220.

29. Song, Y.; Li, J.; Chen, X.; Zhang, D.; Tang, Q.; Yang, K. An efficient tensor completion method via truncated nuclear norm. J. Vis. Commun. Image Represent. 2020, 70, 9p. [CrossRef]

30. Yao, Q.; Kwok, J.T.; Wang, T.; Liu, T.Y. Large-scale Low-rank matrix learning with non-convex regularizers. IEEE Trans. Pattern Anal. Mach. Intell. 2019, 41, 2628–2643. [CrossRef]

31. Chen, X.; Chen, Y.; Saunier, N.; Sun, L. Scalable Low-rank tensor learning for spatiotemporal traffic data imputation. Transp. Res. Part Emerg. Technol. 2021, 129, 14p. [CrossRef]

32. Chen, X.; Sun, L. Low-Rank Autoregressive Tensor Completion for Multivariate Time Series Forecasting. arXiv 2020, arXiv:2006.10436. [CrossRef]

33. Chen, X.; Yang, J.; Sun, L. A Nonconvex Low-Rank Tensor Completion Model for Spatiotemporal Traffic Data Imputation. Transp. Res. Part Emerg. Technol. 2020, 117, 12p. [CrossRef]

34. Chen, X.; Lei, M.; Saunier, N.; Sun, L. Low-Rank Autoregressive Tensor Completion for Spatiotemporal Traffic Data Imputation. IEEE Trans. Intell. Transp. Syst. 2021, 23, 12301–12310. [CrossRef]

35. Su, Y.R.; Wu, X.H.; Liu, W.X. Low-rank Tensor Completion by Sum of Tensor Nuclear Norm Minimization. IEEE Access 2019, 7, 134943–134953. [CrossRef]

36. Silic, M.; Delac, G.; Srbljic, S. Prediction of atomic web services reliability for QoS-aware recommendation. IEEE Trans. Serv. Comput. 2015, 8, 425–438.

37. Chen, Y.; Zhang, Y.; Xia, H.; Gao, C.; Wang, Z.; Wang, F.; Li, G. A hybrid tensor factorization approach for QoS prediction in time-aware mobile edge computing. Appl. Intell. 2022, 52, 8056–8072.