计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2023,32(11):286−293 [doi: 10.15888/j.cnki.csa.009273]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

# 基于服务负载的时序 QoS 预测①

张红霞[1], 武梦德[1], 王登岳[1], 董　琰[2], 高增海[3]

[1](中国石油大学 (华东) 青岛软件学院、计算机科学与技术学院, 青岛 266580)
[2](中石化胜利油田分公司 信息化管理中心, 东营 257001)
[3](中石化胜利石油管理局有限公司 信息化技术服务中心, 东营 257001)
通信作者: 张红霞, E-mail: Zhanghx@upc.edu.cn

**摘　要**: 网络技术的发展和多接入边缘计算的兴起使得计算和网络资源的部署逐渐靠近终端. 随着服务数量的增多, 为了向用户更好地推荐服务, 如何在复杂、动态的边缘计算环境中实时、准确地预测服务质量 (quality of service, QoS) 成为一项挑战. 本文提出一种基于服务负载实时预测 QoS 的深度神经模型 (QPSL), 它可以为边缘计算中的 QoS 预测提供缺少的负载状况感知和周期感知. 首先, 对服务的负载状况进行特征表示, 并通过时序分解模块获取时序特征. 其次, 将 CNN 和 BiLSTM 结合, 学习潜在的时序关系, 生成不同时刻的状态向量. 然后, 基于 Attention 机制为历史时刻的状态向量分配权重, 从而构造未来时刻的状态向量. 最后, 将上下文嵌入向量与状态向量送入感知层完成实时 QoS 预测. 基于真实的融合数据集进行了大量的实验, 结果表明 QPSL 在响应时间和吞吐量任务上的 MAE 分别平均提升了 10.28% 和 10.87%, 优于现有的时间感知 QoS 预测方法.

**关键词**: 边缘计算; 多接入; QoS 预测; 时间感知; 实时预测; 预测模型; 深度学习

引用格式: 张红霞,武梦德,王登岳,董琰,高增海.基于服务负载的时序 QoS 预测.计算机系统应用,2023,32(11):286–293. http://www.c-s-a.org.cn/1003-3254/9273.html

## Time-series QoS Prediction Based on Service Load

ZHANG Hong-Xia[1], WU Meng-De[1], WANG Deng-Yue[1], DONG Yan[2], GAO Zeng-Hai[3]

[1](Qingdao Institute of Software & College of Computer Science and Technology, China University of Petroleum, Qingdao 266580, China)
[2](Information Management Center, Sinopec Shengli Oilfield Branch, Dongying 257001, China)
[3](Information Technology Service Center, Sinopec Shengli Petroleum Management Bureau Co. Ltd., Dongying 257001, China)

**Abstract**: The advance in network technology and the rise of multi-access edge computing have led to the deployment of computation and network resources closer to the end users. As the service numbers increase, it is a challenge to predict the quality of service (QoS) in real-time and accurately in the complex and dynamic edge computing environment to better recommend services to users. In this study, a deep neural model for real-time QoS prediction based on service load (QPSL) is proposed, which can provide missing load condition awareness and cycle awareness for QoS prediction in edge computing. Firstly, the service load condition is characterized, and the features of the time-series are obtained by the time-series decomposition module. Secondly, CNN and BiLSTM are combined to learn the potential time-series relationships and generate the state vectors at different time intervals. Then, the state vectors at future time intervals are constructed by assigning weights to the historical state vectors based on the Attention mechanism. Finally, contextual embedding vectors and state vectors are fed into the perception layer to complete the real-time QoS prediction. Extensive experiments are conducted based on a real fusion dataset, and the results show that QPSL improves MAE by 10.28% and 10.87% on average for response time and throughput tasks respectively, outperforming existing time-aware QoS prediction methods.

**Key words**: edge computing; multi-access; quality of service (QoS) prediction; time-aware; real-time prediction; prediction model; deep learning

① 基金项目: 山东省自然科学基金 (ZR2020MF006, ZR2022LZH015)
收稿时间: 2023-04-07; 修改时间: 2023-05-06; 采用时间: 2023-05-17; csa 在线出版时间: 2023-07-21
CNKI 网络首发时间: 2023-07-21

随着网络技术的不断进步, 智能终端数量日益增多. 与此同时, 服务数量显著增长, 边缘计算[1] 环境下的服务推荐成为研究热点, 而服务推荐的关键是 QoS 预测问题. 在复杂、动态的边缘计算场景中, 用户随时间移动, 不同时刻接入的服务器不同, 导致边缘服务器的负载持续变化, 从而影响服务的调用. 如: 当某边缘服务器附近的用户增多时, 服务器的负载显著增加, 用户通过该服务器调用服务时 QoS 显著降低. 此外, 用户在不同时刻调用同一服务的 QoS 值也存在明显差异. 以往的 QoS 预测[2–4] 利用张量分解学习用户、服务和时间的潜在特征完成预测, 但预测准确性低. 而当前深度学习在推荐系统等领域取得优异成果[5], 特别是使用深度神经模型预测用户点击率, 极大提升了预测效果. 这主要得益于深度神经网络通过多层表示结构有效地提取特征, 从而放大输入中与任务相关的重要因素、限制不相关的因素. 此外, 激活函数的使用增强了深度模型的非线性建模能力, 实现对任意复杂函数的近似, 建立一个高准确度的预测模型. QoS 预测和点击率预测极为类似, 因此深度神经模型也适用于 QoS 预测. 近年来也确实涌现了大量基于深度学习实现 QoS 预测的研究[6–8].

当前的深度 QoS 预测方法主要研究使用各种上下文缓解数据稀疏性, 从而提升预测表现. 然而, 边缘计算环境的复杂性、动态性使得当前的方法, 特别是时间感知类的方法仍然存在一些限制.

1) 考虑到边缘计算场景中服务负载的波动, 过高的服务负载会影响用户的 QoS 体验. 但现有的 QoS 预测模型没有刻画不同时刻服务的负载状况, 缺乏对负载状况的感知能力.

2) 边缘环境内的用户在每天的相似时间段内趋向于在同一位置范围调用相同的服务, 即存在某种周期. 但现有的方法没有对周期性的感知能力.

为了解决以上挑战, 本文提出了一种深度神经模型 (QPSL) 用于实时、准确的 QoS 预测, 其框架如图 1 所示. 在一个服务调用中, QPSL 不光考虑了用户和服务的影响, 还考虑了随时序变化的服务负载的影响. 具体地, 该模型首先对用户和服务在某一时刻的上下文独热向量进行嵌入操作, 从而得到相应的嵌入向量. 其次, 对同一时刻服务的负载状况进行新的特征表示, 通过时序分解模块挖掘特征表示中的时序信息, 包括趋势项、季节项. 卷积层基于趋势项、季节项叠加得到的特征进行局部交互, 从而过滤出较重要的因素, 获得对周期性的感知能力. BiLSTM 接收局部交互的结果, 学习任一时刻的关键因素与状态向量的关系以及整个时间序列上的时序关系, 增强周期性感知能力. Attention 层区分不同时刻的状态向量的重要程度, 获取不同时刻的周期信息对未来时刻的影响. 最后, 感知层基于嵌入向量和 Attention 层的输出实现多个属性的实时预测.

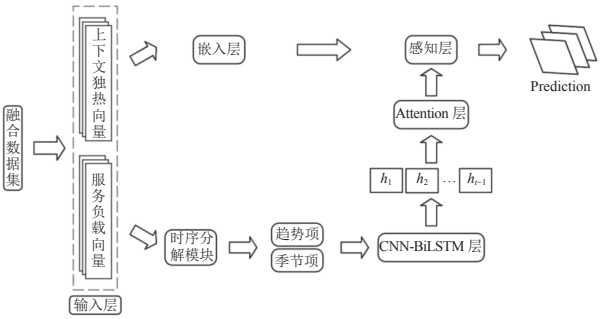


图 1 QPSL 的整体框架

本文的主要贡献如下.

1) 本文通过对服务负载状况进行特征表示, 刻画边缘系统中负载的时序变化, 完善对负载状况的感知能力. 而时序分解模块以及 CNN-BiLSTM-Attention 层的提出实现并提升了模型的周期感知能力.

2) 本文提出了 QPSL 深度神经模型, 用于充分挖掘和学习服务负载下的周期性信息, 从而利用服务负载这一辅助信息实时预测 QoS.

3) 在真实的数据集上进行了大量的实验, 结果表明 QPSL 使响应时间和吞吐量任务上的 MAE 分别平均提升了 10.28% 和 10.87%, 说明了 QPSL 的有效性.

## 1 相关工作

近年来, 边缘计算的相关研究备受关注, 因为它弥补了云计算的缺陷. 随着边缘计算的发展, 边缘计算环境下的 QoS 预测逐渐吸引到相关人员.

由于边缘环境中用户频繁移动, 边缘环境中的 QoS 预测常常会关注用户移动性. Wang 等[9] 认为用户的移动性会使 QoS 预测值偏离真实值, 提出一种新的协同过滤方法, 该方法基于边缘服务器的相似度和距离选择 Top-K 个相似的邻居, 实现 QoS 预测. Zhang 等[10] 尝试基于长短期记忆网络预测用户轨迹, 确定候选边缘服务器, 通过基于位置的协同过滤完成预测. 张鹏程等[11] 归纳了移动边缘环境下的一些问题, 并针对这些

问题提出了一种移动边缘环境下隐私保护的 Web 服务 QoS 预测方法. 用户移动性造成的冷启动使得基于协同过滤的方法在边缘计算环境中使用时存在部分缺陷. Yin 等[12] 提出了一种混合方法. 首先, 基于改进的自编码器实现对稀疏输入的处理, 并缓解冷启动问题. 之后, 提出一种基于欧氏距离的相似度计算方法, 解决模型过拟合的问题.

边缘环境中观测到的 QoS 值还有着实时性强的特点, 即网络状况和服务负载频繁波动, 使得用户观测到的 QoS 经常随时间变化. 时间感知的 QoS 预测方法由此进入研究者们的视线. 此类方法通常考虑用户、服务、时间这 3 个维度, 因此主要基于张量分解实现. Zhang 等[2] 最早提出一种使用张量分解预测 QoS 的方法 WS-Pred, 该方法利用 QoS 平均值正则化目标函数. Chen 等[3] 将 QoS 数据扩展到时间维度, 并将时间正则化与张量分解结合, 挖掘相邻时间间隔的时间序列特性. Yan 等[4] 通过截断奇异值分解 (SVD) 提取 QoS 矩阵的压缩矩阵, 用于扩展 ARIMA 模型, 从而同时预测多个 QoS 值. Wang 等[13] 认为移动客户端的 QoS 体验与多维度 (时间、空间) 相关, 需要挖掘多维 QoS 数据中的结构关系, 从而提出一种新的张量分解方法. Ngaffo 等[14] 在当前时间间隔上使用矩阵分解技术实现 QoS 预测, 而针对连续时间序列, 基于 ARIMA 模型提出一种预测方法对未来某一时间间隔的 QoS 进行预测. 张雅倩[15] 针对以往的服务质量预测方法存在的问题以及移动边缘环境的特点, 将时间和时空信息先后融合进矩阵分解中.

深度学习在其他领域的成功应用使得研究者们开始探索深度 QoS 预测. 熊伟等[16] 提出一种通用的时空感知的 QoS 预测方法, 通过深度学习准确建模时间和空间信息. Xiong 等[6] 提出了一种基于矩阵分解的个性化 LSTM 预测模型, 分别学习用户侧和服务侧的潜在表示随时间的变化, 从而预测未来时刻的 QoS 值. 传统的服务质量预测方法很少利用上下文数据或者忽略请求时间信息, 因此这些方法无法很好地捕获依赖因素并准确预测 QoS 值. 为了解决这个问题, Li 等[7] 提出一种深度神经网络来感知上下文数据和时间序列信息, 用于在包含显式和隐式因素的动态场景下预测 QoS. Zou 等[8] 提出一种具有门控递归单元 (GRU) 的深度神经网络, 学习和挖掘用户和服务之间的时间特征. 陈慢慢[17] 研究了基于对比学习方法实现对未来时刻 QoS 预测的任务和基于循环神经网络实现对任意时刻 QoS 预测的任务.

现有的方法在 QoS 预测准确性上已经取得优异的成果, 但它们没有考虑边缘计算中服务负载波动的影响, 缺乏负载状况的感知能力. 同时, 也没有充分考虑边缘计算中的周期性, 缺乏周期感知能力.

## 2 QPSL 方法

### 2.1 输入层

我们以元组的形式定义用户-服务调用, 具体如下:

$$USI=(UserID,ServiceID,ServerLongitude,\\ ServerLatitude,ServiceLoad) \tag{1}$$

其中, *UserID* 表示服务器中用户的 ID, *ServiceID* 表示部署在服务器上的服务的 ID, *ServerLongitude* 表示用户接入的服务器所在的经度, *ServerLatitude* 表示用户接入的服务器所在的纬度, *ServiceLoad* 表示服务器中服务的负载状况.

在 *USI* 中, 除了 *ServiceLoad* 这一字段, 剩余的每个字段 (上下文) 通常都有多个不同的值 (特征), 为了保留语义信息以及将其输入到神经网络中, 需要通过独热编码将每个字段分别转换为一个高维稀疏的二元向量即独热向量. 如: 某一时刻某一服务器中存在 5 个用户, 用户 $U1$ 调用了某个服务, 则 *USI* 中的 *UserID* 为[01000]. 在获取每个上下文对应的独热向量后, 将它们拼接在一起, 从而得到输入层中的上下文独热向量.

对于 *ServiceLoad* 这一字段, 它被定义为一个向量 *sl*, 大小等于某一时刻服务器中用户的总数, 值是根据某一用户在某一时刻是否调用某一服务确定的. 以服务器 *SN*1 为例进行说明, $T1$ 时刻 *SN*1 中存在 8 个用户, 服务 $S1$ 被用户 $U1$, $U3$, $U4$, $U7$ 调用, 则 $sl$ = [01011001]. 该向量可以反映服务负载的状况, 向量越稠密, 服务负载越高. 同时, 该向量包含用户与服务之间的时序关系.

### 2.2 嵌入层

由于上下文独热向量在语义表达上效果较差, 我们使用嵌入层将它映射到低维空间, 从而准确表征独热向量中每个特征的语义信息. 经过嵌入操作后, 每个上下文映射得到一组实值向量 $\{e_i^d\}$, 其中, $i$ 表示特征在独热向量中的下标, $d$ 表示低维空间的具体维度. 更进一步地, 只保留每个上下文中标记为 1 的特征对应的嵌入向量, 最终, 一个上下文被表示为一个嵌入向量 $e^d$. 通过这种方式, 可以得到一组嵌入向量 $e_c=[e_{UID}^d, e_{SID}^d,$

$e^d_{SLD}, e^d_{SLA}]$, 它们分别对应式 (1) 中的各个上下文.

### 2.3 时序分解模块

在边缘计算环境中, 任一用户的移动行为在一段长连续时间内具有周期性, 而多名用户的周期性移动构成了服务负载的周期性. 因此, 使用时序分解模块分解时序数据、获取服务负载的周期性信息是必不可少的. 在时间序列分解领域[18], 某一时刻的数据可以视为趋势项、季节项、剩余项等相加或者相乘的结果, 即加法模型或乘法模型.

本文中将任一时刻的服务负载向量 ($sl$) 视为各种因素相加的结果, 因此使用加法模型构成时序分解模块. 由于不同的服务的负载不一定相同, 因此$sl$的长度会不一致. 为了能够有效地分解出影响因素, 首先需要统一$sl$的长度, 即使用 0 填充. 其次, 使用滑动平均法提取出趋势项, 具体来说, 基于每个时刻的$sl$分别计算平均值, 得到每个时刻的趋势项$sl_t \in R^{1\times m}$, 其中, $m$是所有时刻中服务负载向量的最大长度. 然后, 在每个时刻的输入序列中减去趋势项, 即可得到季节项$sl_s \in R^{1\times m}$和剩余项$sl_r \in R^{1\times m}$.

$$\begin{cases} sl_t = AvgPool(Padding(sl)) \\ sl_s + sl_r = sl - sl_t \end{cases} \tag{2}$$

在得到趋势项$sl_t$后, 通过去噪操作剥离出剩余项$sl_r$, 从而得到季节项$sl_s$. 趋势项和季节项分别反映了长期趋势性和周期波动性. 最终, 拼接得到的各个分项, 从而得到$ts = [sl_t, sl_s] \in R^{1\times 2m}$. 时序分解模块的使用使得模型初步具备周期感知能力, 可以有效感知边缘系统的负载状况.

### 2.4 CNN-BiLSTM-Attention 层

对于时序特征向量$ts$, 它是由趋势项和季节项拼接得到, 我们认为这种方式不能有效获取$sl$中潜在的周期信息. 此外, 周期信息在一个时间序列内是逐渐变化的, 并且是双向影响的. 最后, 预测时刻的周期信息是由之前多个时刻的周期信息共同决定, 但每个时刻的影响是不同的. 针对上述问题, 我们提出了 CNN-BiLSTM-Attention 层逐个解决.

如图 2 所示, 首先对$ts$使用卷积层进行局部特征交互, 从而获取信息更加充分、有用的周期性特征. 具体地, 为了执行卷积操作, 使$ts$变形为$f^0 \in R^{1\times 1\times 2m}$, 以$f^0$作为第 1 个卷积神经网络的输入特征, 通过一个权重矩阵$w_0$对输入特征进行一维时间维度上的卷积, 从而得到新的特征$f^1$. 为了获取重要的周期性特征, 减少不必要的参数, 本文中对局部交互的结果使用了最大池化层, 即在$f^1$上进行最大池化从而获取$f^1_p$. 以此类推, 在执行多次卷积神经网络与最大池化后得到最终的特征$f^l_p$. 式 (3) 描述了这一过程:

$$\begin{cases} f^1 = ReLU(f^0 \otimes w_0 + b_0) \\ f^1_p = maxpool(f^1) \\ f^2 = ReLU(f^1 \otimes w_1 + b_1) \\ f^2_p = maxpool(f^2) \\ \quad\vdots \\ f^l = ReLU(f^{l-1} \otimes w_{l-1} + b_{l-1}) \\ f^l_p = maxpool(f^l) \end{cases} \tag{3}$$

其中, $l$是卷积层的总数, $w_i \in R^{1\times w^i\times c^i}$, 1、$w^i$、$c^i$分别是第$i$个卷积神经网络中卷积核的高度、宽度、数量, $b_i$表示偏置, $\otimes$表示卷积操作. 在整个卷积层执行完后, 需要改变$f^1_p$的形状, 从而得到$x \in R^{1\times n}$. 通过卷积层的学习可以初步获取不同时刻的周期信息、实现对周期的初步感知. 为了确定这些周期信息随时间的变化模式、依赖关系, 我们将卷积层提取的序列特征数据输入到 BiLSTM 层进行隐含状态学习.

BiLSTM 是一种独特的 RNN 结构, 它由两层 LSTM 网络组成, 可以同时学习前向和后向的序列特征之间的依赖, 充分感知时间序列数据之间的上下关系. LSTM 单元通过遗忘门、更新门、输出门等结构完成单元内的指定任务. 遗忘门、更新门、输出门都是以当前的输入信息和前一个隐藏状态信息作为输入. 遗忘门输出一个 [0, 1] 范围的值, 输出 0 代表完全丢弃, 输出 1 代表完全保留, 从而确定应当丢弃或保留的信息. 在进行遗忘之后, 下一阶段需要更新单元状态中的信息, 这一操作主要由更新门完成. 更新门确定具体的更新内容以及比例, 结合遗忘门的输出以及前一个单元状态, 获取当前的单元状态. 输出门用于产生一个最终的输出 (即当前隐藏状态信息). 卷积层处理后的$t$时刻的输入$x_t$在 LSTM 单元的处理流程如下:

$$\begin{cases} f_t = Sigmoid(w_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t = Sigmoid(w_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{c}_t = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \\ c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \\ o_t = Sigmoid(w_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t = o_t \odot \tanh(c_t) \end{cases} \tag{4}$$

其中, $w_*$表示权重矩阵, $b_*$表示偏置, $\odot$表示元素级的乘

法, $h_{t-1}$表示前一时刻的输出, $f$、$i$、$c$、$o$、$h$分别是遗忘门、更新门、单元激活向量、输出门、隐藏状态. 简单起见, 上述流程可以表示为$h_t = lstm(x_t, \theta)$. BiLSTM 网络由正向 LSTM 层和反向 LSTM 层组成, 则输出结果为:

$$\begin{cases} \overrightarrow{h_t} = \overrightarrow{lstm(x_t, \theta)} \\ \overleftarrow{h_t} = \overleftarrow{lstm(x_t, \theta)} \\ h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \end{cases} \tag{5}$$

其中, $\overrightarrow{h_t}$为前向 LSTM 网络的计算结果, $\overleftarrow{h_t}$为后向 LSTM 网络的计算结果, 拼接得到的$h_t$为 BiLSTM 网络的最终输出. BiLSTM 网络能够很好地获取时间序列的全局特征信息, 使得不同时刻的 BiLSTM 的输出充分隐含前向和后向的数据间依赖关系, 可以增强周期感知能力. 为了充分利用所有时刻的状态向量, 引入注意力机制来控制不同时刻状态向量的重要程度, 从而实现对预测时刻的周期信息的构造.

注意力机制首先应用于图像处理领域[19], 它会根据特征的重要程度分配相应的权重, 使得神经网络具备专注某些特征的能力, 同时解决 BiLSTM 面临的信息丢失问题. 该部分的计算过程如下所示:

$$\begin{cases} \alpha_i = Softmax(s(h_i, Q)) = \dfrac{\exp(h_i^{\mathrm{T}} \cdot Q)}{\sum_{j=1}^{t-1} \exp(h_j^{\mathrm{T}} \cdot Q)} \\ e_{ts} = \sum_{i=1}^{t-1} \alpha_i h_i \end{cases} \tag{6}$$

其中, $Q$是一个查询向量, $\alpha_i$是$h_i$与$Q$之间的相关程度, $e_{ts}$是加权汇总后的最终结果.

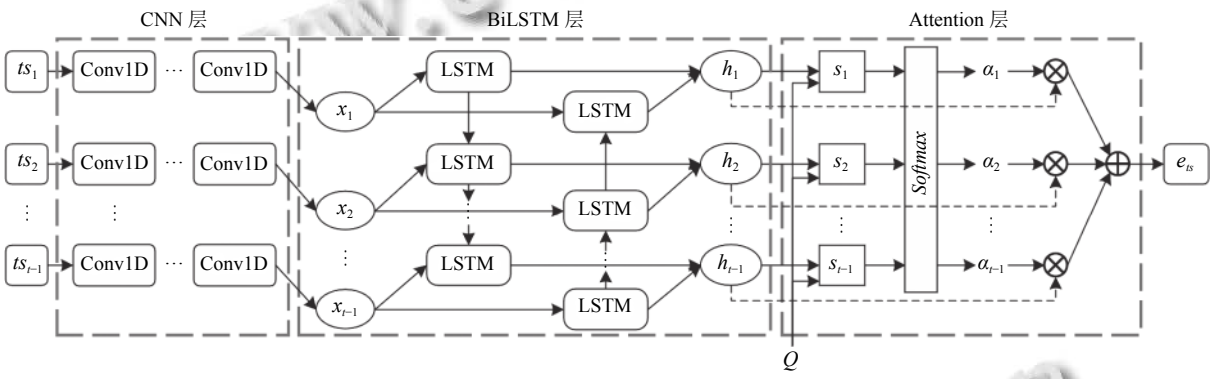


图 2 CNN-BiLSTM-Attention 层

### 2.5 感知层

为用户提供优质的服务, 总是需要考虑服务质量的多个属性. 如: 响应时间、吞吐量和可靠性等. 这些 QoS 属性受共同上下文的影响, 彼此高度相关. 但不同的上下文对服务质量属性有不同的影响: 吞吐量对服务器端的情况更敏感, 而响应时间对底层网络的条件更敏感. 为了感知不同的 QoS 属性, 每个预测任务使用独立的感知模块:

$$\begin{cases} e = [e_c, e_{ts}] \\ qos_{\text{task}} = PL_{\text{task}}(e) \end{cases} \tag{7}$$

其中, $qos_{\text{task}}$表示任意 QoS 属性的预测值, $PL_{\text{task}}$表示完成任意 QoS 属性预测任务所需要的感知模块. 这些感知模块是由多层感知机构成, 它可以为不同的预测任务提供相应的特征选择和加权功能.

通过 CNN-BiLSTM-Attention 层, QPSL 方法可以更好地基于$sl$分解得到的趋势项和季节项学习到服务本身的周期性信息, 捕获服务的周期性信息随时间的变化模式, 实现不同时刻的周期性信息对预测时刻的周期性信息的影响的计算, 从而在预测时刻获取较高准确度的服务的周期信息. 同时, 基于用户在预测时刻的上下文嵌入向量, QPSL 可以在高维空间中完成预测时刻用户调用服务的高精度建模, 从而达到利用服务负载实时预测 QoS 的目的.

## 3 实验

### 3.1 实验设置

我们融合两个真实数据集构造边缘环境数据集. 第 1 个数据集由上海电信提供[20], 可以访问网址 https://github.com/BuptMecMigration/Edge-Computing-Dataset 获取. 该数据集记录了用户调用服务的时间信息和接入的服务器的经纬度信息. 第 2 个数据集是 WS-Dream[21], 可在网址 https://github.com/wsdream/

wsdream.github.io 下载. 该数据集包括 142 个用户在 64 个不同时间片调用的 4 500 个 Web 服务的 QoS. 具体的融合方法借鉴了文献 [9] 的经验: 首先通过用户 ID 将两个数据集关联, 然后按照 64 个时间片展开, 形成时序边缘数据集.

为了保证公平性, 我们搭建统一的实验环境, 具体的配置为 Windows 10 操作系统、 英伟达 RTX 2060 显卡、 因特尔 i5-12400F、 16 GB 运行内存以及 PyTorch 深度学习框架.

### 3.2 性能比较

我们进行比较实验验证 QPSL 方法的优越性, 以 MAE 和 RMSE 作为评价指标. 此外, 取多次实验结果的平均值作为最终的预测结果. 在基线方法的选择上, 主要选择时间感知类的 QoS 预测方法, 包括 WSPred[2]、SERPRED[4]、PLMF[6]、QSPC[7]、DeepTSQP[8]、TASERM[14] 和 LMDC[22].

从表 1 的实验结果可以看出, 所有方法的预测准确性都随着数据密度的增加而提高, 因为数据密度越大, 上下文信息越丰富, 模型可以学习到更多潜在关系. 在基于张量分解的方法中, SERPRED 和 TASERM 总是优于 WSPred, 因为它们使用的信息更加丰富, 应对数据稀疏性的效果更好. 此外, 随着数据密度增加, SERPRED 相对于 TASERM 的优越性逐渐消失, 这可以归结为数据量对 QoS 压缩矩阵的影响. 在基于深度学习的方法中, LMDC 在响应时间任务上的预测表现逐渐优于 PLMF, 因为 LMDC 方法需要基于大量数据计算相似度. QSPC 的预测表现始终优于 LMDC 和 PLMF, 因为它考虑的上下文因素更充分, 更好地缓解数据稀疏性. DeepTSQP 总是优于其他方法, 因为它同时使用了深度神经网络和协同过滤方法. 特别地, 与同样使用深度神经网络和协同过滤方法的 LMDC 相比, DeepTSQP 能够产生更高阶的特征, 从而得到更高的准确性. 基于深度学习的方法始终优于基于张量分解的方法, 这说明深度学习可以捕获时间序列数据中的变化并有效地学习非线性关系, 从而更准确地预测 QoS. 最后, 本文提出的 QPSL 方法在两个任务上始终优于基线方法, 并且使 MAE 分别提升 12.8% 和 12.92%, RMSE 分别提升 10.9% 和 10.12%. 我们认为这主要得益于时序分解模块和 CNN-BiLSTM-Attention 层的应用. 时序分解模块将 $sl$ 分解得到趋势项和季节项, 从而赋予 QPSL 周期性的感知能力. CNN-BiLSTM-Attention 层学习特征交互, 提升 QPSL 的周期感知能力.

表 1 预测性能比较

| QoS | Method | 5% | | 10% | | 15% | | 20% | |
|---|---|---|---|---|---|---|---|---|---|
| | | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE |
| Response time (RT) | WSPred | 0.7713 | 1.8126 | 0.7378 | 1.7769 | 0.6832 | 1.6925 | 0.6562 | 1.6284 |
| | SERPRED | 0.7420 | 1.7511 | 0.6982 | 1.7197 | 0.6701 | 1.6745 | 0.6554 | 1.6098 |
| | TASERM | 0.7563 | 1.7829 | 0.7198 | 1.7595 | 0.6783 | 1.6753 | 0.6411 | 1.6016 |
| | LMDC | 0.7412 | 1.7473 | 0.6896 | 1.7132 | 0.6515 | 1.6548 | 0.6351 | 1.5818 |
| | PLMF | 0.7298 | 1.7379 | 0.6873 | 1.6897 | 0.6639 | 1.6439 | 0.6523 | 1.5578 |
| | QSPC | 0.7164 | 1.6924 | 0.6704 | 1.6403 | 0.6457 | 1.6017 | 0.6207 | 1.5407 |
| | DeepTSQP | 0.6983 | 1.5823 | 0.6393 | 1.5086 | 0.5863 | 1.4483 | 0.5524 | 1.4106 |
| | QPSL | **0.6334** | **1.4256** | **0.5768** | **1.3467** | **0.5324** | **1.2824** | **0.4816** | **1.2572** |
| | Gains (%) | **9.3** | **9.90** | **9.80** | **10.70** | **9.20** | **11.50** | **12.8** | **10.90** |
| Throughput (TP) | WSPred | 15.6246 | 42.8502 | 14.1853 | 41.5393 | 13.3451 | 39.3652 | 12.1474 | 37.8323 |
| | TASERM | 15.0180 | 41.6257 | 13.5824 | 40.8813 | 12.5742 | 39.1149 | 11.6381 | 37.1049 |
| | SERPRED | 15.2163 | 42.1591 | 13.7140 | 41.0027 | 12.5905 | 39.0250 | 11.5692 | 36.7025 |
| | LMDC | 14.5123 | 41.5236 | 13.2057 | 40.0142 | 12.1240 | 38.7347 | 11.2554 | 36.2536 |
| | PLMF | 13.1532 | 38.5736 | 12.3621 | 37.2525 | 11.2358 | 36.9125 | 10.1535 | 34.9356 |
| | QSPC | 12.8679 | 37.7245 | 11.8640 | 36.2445 | 10.1514 | 34.3683 | 9.9080 | 32.8353 |
| | DeepTSQP | 12.2453 | 36.6142 | 11.2954 | 35.1634 | 10.6240 | 33.5153 | 9.3562 | 31.1254 |
| | QPSL | **11.1357** | **34.1390** | **10.1052** | **32.2671** | **9.2513** | **30.1245** | **8.3323** | **28.1356** |
| | Gains (%) | **9.06** | **6.76** | **10.54** | **7.10** | **12.92** | **10.12** | **10.94** | **9.61** |

### 3.3 敏感性分析

(1) 时间窗口的影响

我们在 4–18 的范围内, 以 2 为步长改变时间窗口的大小, 探究时间窗口的影响. 如图 3 所示, 实验结果表明, 当矩阵密度极低时, 较大的时间窗口有利于提供更多的时间调用信息, 更好地挖掘用户与服务之间隐含的非线性关系. 随着矩阵密度增加、时间窗口变大, 预测准确性有进一步提高的趋势. 我们认为这是因为

时序分解模块的使用在一定程度上揭示了周期性, 时间窗口越大, 可以获得的周期性信息越丰富.

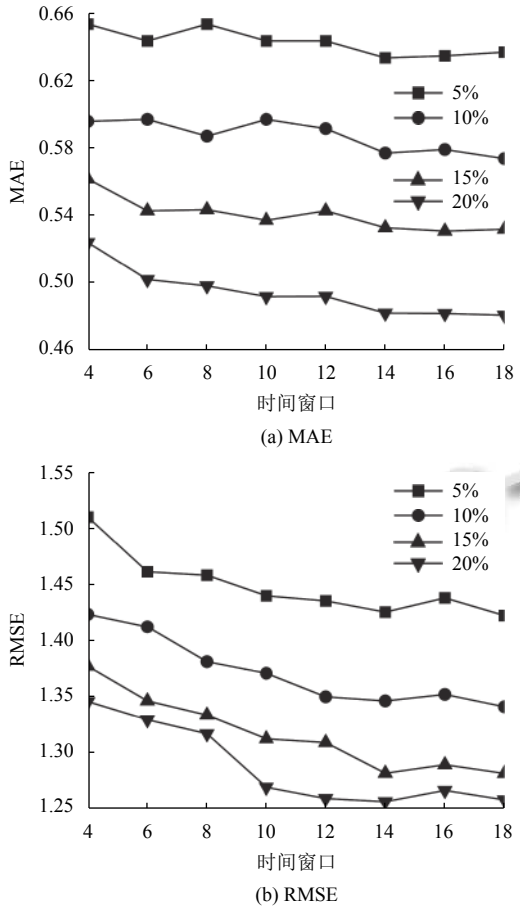


图 3 时间窗口的影响

(2) 时序分解模块性能比较

我们进行消融实验探究时序分解模块的有效性, 并与乘法分解模型进行对比. 如表 2 所示, 加法分解模型 QPSL-A (即 QPSL) 的预测性能略微强于乘法分解模型 QPSL-M, 我们分析认为加法模型分解后的时序信息更容易被卷积层用于交互产生周期性信息. 此外, 去除时序分解模块的 QPSL-W 模型预测精度出现下降, 这表明在 QPSL 方法中, 时序分解模块能有效提取时序数据中的趋势性和季节性信息, 实现方法对周期的初步感知, 从而为 QoS 预测提供辅助信息.

表 2 时序分解模块对预测表现 (MAE) 的影响

| QoS | Method | 5% | 10% | 15% | 20% |
|---|---|---|---|---|---|
| RT | QPSL-W | 0.6893 | 0.6135 | 0.5685 | 0.5028 |
| | QPSL-M | 0.6547 | 0.5923 | 0.5495 | 0.4914 |
| | QPSL-A | 0.6334 | 0.5768 | 0.5324 | 0.4816 |
| TP | QPSL-W | 12.1405 | 11.1135 | 10.2143 | 9.1624 |
| | QPSL-M | 11.6623 | 10.3455 | 9.7851 | 8.7335 |
| | QPSL-A | 11.1357 | 10.1052 | 9.2513 | 8.3323 |

(3) CNN 部分的性能比较

CNN 部分在整个 CNN-BiLSTM-Attention 层中有着重要作用, 它通过局部交互建模出周期性, 同时通过最大池化层找出重要的特征信息, 减少信息的冗余. 对于后续的 BiLSTM-Attention 处理有着重要影响, 我们采用控制变量的方式去除该部分得到 QPSL-C, 基于实验结果对比验证它的效果. 如表 3 所示, QPSL 的预测表现优于 QPSL-C, 这说明了 CNN 部分在提升方法的周期性感知能力上的重要地位.

表 3 CNN 部分对预测表现 (MAE) 的影响

| QoS | Method | 5% | 10% | 15% | 20% |
|---|---|---|---|---|---|
| RT | QPSL-C | 0.6621 | 0.5975 | 0.5502 | 0.4970 |
| | QPSL | 0.6334 | 0.5768 | 0.5324 | 0.4816 |
| TP | QPSL-C | 11.5371 | 10.6284 | 9.7152 | 9.0041 |
| | QPSL | 11.1357 | 10.1052 | 9.2513 | 8.3323 |

(4) BiLSTM 部分的实验对比

表 4 对比了在时序学习部分使用 LSTM 或 BiLSTM 的效果. 结果表明, 相对于使用 LSTM 的 QPSL-L 方法, 使用 BiLSTM 的 QPSL 方法的预测精度有一定提升. 这是因为 LSTM 只学习前向的序列特征之间的依赖, 而 BiLSTM 可以同时学习前向和后向的依赖, 从而实现对周期感知能力的加强, 能够有效提升预测精度.

表 4 LSTM 和 BiLSTM 对预测表现 (MAE) 的影响

| QoS | Method | 5% | 10% | 15% | 20% |
|---|---|---|---|---|---|
| RT | QPSL-L | 0.6535 | 0.5930 | 0.5493 | 0.4882 |
| | QPSL | 0.6334 | 0.5768 | 0.5324 | 0.4816 |
| TP | QPSL-L | 11.3215 | 10.2835 | 9.4104 | 8.4035 |
| | QPSL | 11.1357 | 10.1052 | 9.2513 | 8.3323 |

(5) Attention 部分的影响

为了验证 Attention 机制的影响, 我们将 QPSL 模型与去除 Attention 机制的 QPSL-AT 模型进行比较, 实验结果如表 5 所示. 可以看出 QPSL 的预测表现始终优于 QPSL-AT. 我们将其归结为, 注意力机制可以根据过往时刻的周期信息以较高的准确性构造出预测时刻的周期信息, 在高维空间中反映出预测时刻服务的相关信息, 提升预测准确性.

表 5 Attention 部分对预测表现 (MAE) 的影响

| QoS | Method | 5% | 10% | 15% | 20% |
|---|---|---|---|---|---|
| RT | QPSL-AT | 0.6784 | 0.6015 | 0.5537 | 0.4912 |
| | QPSL | 0.6334 | 0.5768 | 0.5324 | 0.4816 |
| TP | QPSL-AT | 11.7451 | 10.8704 | 9.8034 | 9.1534 |
| | QPSL | 11.1357 | 10.1052 | 9.2513 | 8.3323 |

## 4 总结

实时的 QoS 预测对于用户即时选择合适的服务具有重要意义. 针对边缘计算中用户频繁移动导致的服务负载动态波动和预测模型周期性感知不足, 本文提出一种基于服务负载实时预测 QoS 的深度神经模型 (QPSL). 它对服务的负载状况进行新的特征表示, 使用时序分解模块和 CNN-BiLSTM-Attention 层实现对服务负载状况和周期性的感知. 在真实的融合数据集上进行了实验, 实验结果表明, QPSL 方法在响应时间和吞吐量任务上分别使 MAE 平均提升了 12.8% 和 10.92%, 优于现有的时间感知 QoS 预测方法.

在未来的工作中, 我们计划研究自适应时间窗口的动态 QoS 预测模型以支持更动态的场景.

## 参考文献


1 Filali A, Abouaomar A, Cherkaoui S, *et al*. Multi-access edge computing: A survey. IEEE Access, 2020, 8: 197017–197046. [doi: 10.1109/ACCESS.2020.3034136]

2 Zhang YL, Zheng ZB, Lyu MR. WSPred: A time-aware personalized QoS prediction framework for Web services. Proceedings of the 22nd IEEE International Symposium on Software Reliability Engineering. Hiroshima: IEEE, 2011. 210–219.

3 Chen YP, Zhang YQ, Xia H, *et al*. A hybrid tensor factorization approach for QoS prediction in time-aware mobile edge computing. Applied Intelligence, 2022, 52(7): 8056–8072. [doi: 10.1007/s10489-021-02851-z]

4 Yan C, Zhang YK, Zhong WY, *et al*. A truncated SVD-based ARIMA model for multiple QoS prediction in mobile edge computing. Tsinghua Science and Technology, 2022, 27(2): 315–324. [doi: 10.26599/TST.2021.9010040]

5 Cai WJ, Wang YF, Ma JH, *et al*. CAN: Effective cross features by global attention mechanism and neural network for ad click prediction. Tsinghua Science and Technology, 2022, 27(1): 186–195. [doi: 10.26599/TST.2020.9010053]

6 Xiong RB, Wang J, Li ZQ, *et al*. Personalized LSTM based matrix factorization for online QoS prediction. Proceedings of the 2018 IEEE International Conference on Web Services. San Francisco: IEEE, 2018. 34–41.

7 Li BZ, Ye CY, Yu XZ, *et al*. QoS prediction based on temporal information and request context. Service Oriented Computing and Applications, 2021, 15(3): 231–244. [doi: 10.1007/s11761-021-00322-4]

8 Zou GB, Li TF, Jiang M, *et al*. DeepTSQP: Temporal-aware service QoS prediction via deep neural network and feature integration. Knowledge-based Systems, 2022, 241: 108062. [doi: 10.1016/j.knosys.2021.108062]

9 Wang SG, Zhao YL, Huang L, *et al*. QoS prediction for service recommendations in mobile edge computing. Journal of Parallel and Distributed Computing, 2019, 127: 134–144. [doi: 10.1016/j.jpdc.2017.09.014]

10 Zhang HX, Dong YH, Yang YJ. Mobility-aware personalized service recommendation in mobile edge computing. Eurasip Journal on Wireless Communications and Networking, 2021, 2021(1): 196. [doi: 10.1186/s13638-021-02068-1]

11 张鹏程, 金惠颖. 一种移动边缘环境下面向隐私保护 QoS 预测方法. 计算机学报, 2020, 43(8): 1555–1571. [doi: 10.11897/SP.J.1016.2020.01555]

12 Yin YY, Zhang WP, Xu YS, *et al*. QoS prediction for mobile edge service recommendation with auto-encoder. IEEE Access, 2019, 7: 62312–62324. [doi: 10.1109/ACCESS.2019.2914737]

13 Wang SG, Ma Y, Cheng B, *et al*. Multi-dimensional QoS prediction for service recommendations. IEEE Transactions on Services Computing, 2019, 12(1): 47–57. [doi: 10.1109/TSC.2016.2584058]

14 Ngaffo AN, El Ayeb W, Choukair Z. Service recommendation driven by a matrix factorization model and time series forecasting. Applied Intelligence, 2022, 52(1): 1110–1125. [doi: 10.1007/s10489-021-02478-0]

15 张雅倩. 基于时间和位置信息的服务质量预测方法研究 [ 硕士学位论文 ]. 西安: 西安邮电大学, 2022.

16 熊伟, 李兵, 吴钊, 等. 一种时空敏感的 QoS 预测方法. 计算机学报, 2019, 42(4): 772–785. [doi: 10.11897/SP.J.1016.2019.00772]

17 陈慢慢. 时间感知的 Web 服务 QoS 预测方法研究 [ 硕士学位论文 ]. 杭州: 杭州电子科技大学, 2022.

18 Cleveland RB, Cleveland WS, McRae JE, *et al*. STL: A seasonal-trend decomposition procedure based on LOESS. Journal of Official Statistics, 1990, 6(1): 3–73.

19 Mnih V, Heess N, Graves A, *et al*. Recurrent models of visual attention. Proceedings of the 27th International Conference on Neural Information Processing Systems. Montreal: MIT Press, 2014. 2204–2212.

20 Wang SG, Guo Y, Zhang N, *et al*. Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach. IEEE Transactions on Mobile Computing, 2021, 20(3): 939–951. [doi: 10.1109/TMC.2019.2957804]

21 Zheng ZB, Zhang YL, Lyu MR. Distributed QoS evaluation for real-world Web services. Proceedings of the 2010 IEEE International Conference on Web Services. Miami: IEEE, 2010. 83–90.

22 Xiong W, Wu Z, Li B, *et al*. A learning approach to QoS prediction via multi-dimensional context. Proceedings of the 2017 IEEE International Conference on Web Services. Honolulu: IEEE, 2017. 164–171.


(校对责编: 孙君艳)