面向云平台弹性伸缩的多头注意力-残差修正容器伸缩行为学习模型

赵楠楠[1],杨 帆[1],王 浩[1],张佳萌[1],吴若非[1],赵彤轩[2],
师雨露[1],张 晓[1],赵晓南[1],黄志杰[1],韩淑捷[1]
(1. 西北工业大学计算机学院,陕西西安 710129;2. 西北工业大学自动化学院,陕西西安 710129)

摘 要: 云平台和微服务架构的快速发展,使弹性伸缩成为保障性能与资源效率的核心机制. 现有研究虽在负载预测与组合建模方面取得了进展,但大多仍以 CPU、内存等资源利用率为预测目标,再通过阈值规则或控制逻辑间接触发副本调整. 这种"预测-控制解耦"的设计带来多重不足:一方面,预测误差在规则映射过程中被放大,难以保证伸缩动作的准确性;另一方面,真实调度器中的滞回、冷却与离散动作机制难以刻画,使得预测结果难以直接落地. 因此,直接对伸缩行为进行学习,即建模副本数的动态变化规律,成为一种更具控制导向和部署价值的思路. 针对这一问题,本文提出一种多头注意力-残差修正容器伸缩行为学习模型. 该模型首先利用自回归滑动平均模型(Auto Regressive Integrated Moving Average,ARIMA)对副本数序列进行趋势分解,再以双向长短期记忆(Bidirectional Long-Short Term Memory,BiLSTM)神经网络对残差序列建模,并引入多头注意力机制(Multi-Head Attention,MHA)自动捕获关键时序特征,通过残差修正(Residual Correction)提升预测精度与健壮性. 基于 Alibaba 公开的 cluster-trace-microservices-v2022 真实数据集,系统对比了 PETformer、SparseTSF、TFEGRU、GRU、Transformer、Seq2Seq-LSTM、Seq2Seq-GRU、Seq2Seq-Transfomer、GRU-LSTM、CNN-LSTM、CNN-LSTM-GRU 等主流模型. 实验结果表明,所提 ARIMA-BiLSTM-MHA 组合模型在均方误差(Mean Squared Error,MSE)、均方根误差(Root Mean Squared Error,RMSE)、平均绝对误差(Mean Absolute Error,MAE)、平均绝对百分比误差(Mean Absolute Percentage Error,MAPE)、决定系数(Coefficient of Determination,R^2)等核心指标上,较各类基线方法分别取得了 1.57%~71.56%(MSE)、0.72%~46.67%(RMSE)、1.57%~59.10%(MAE)、1.97%~60.48%(MAPE)、0.27%~15.70%(R^2)的相对百分比性能提升,其中 R^2 最高提升至 0.954 3. 进一步地,在基于 DeathStarBench socialNetwork 基准应用构建的容器副本伸缩控制实验中,行为学习驱动的伸缩策略相较于 CPU 阈值型 HPA 策略,在将平均 P99 延迟降低 2.11% 并有效抑制负载跃迁阶段尾延迟尖峰的同时,成功将平均副本数减少约 17%,显著缓解了资源过度配置问题. 结果证明,该方法能够更准确、稳定地学习伸缩动作并输出直接可用的控制语义,为云平台自动扩缩容提供高效、可靠的决策支持.

关键词: 云平台;弹性伸缩;容器伸缩行为;多头注意力机制;残差修正;混合模型;深度学习;时序预测
基金项目: 国家自然科学基金(No.62202382,No.62272394);广东省基础与应用基础研究基金(No. 2021A1515110080)
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2025)12-4408-21
电子学报 URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20250831

A Hybrid Multi-Head Attention and Residual Correction Model for Elastic Scaling Behavior Learning in Cloud Platforms

ZHAO Nan-nan[1], YANG Fan[1], WANG Hao[1], ZHANG Jia-meng[1], WU Ruo-fei[1], ZHAO Tong-xuan[2],
SHI Yu-lu[1], ZHANG Xiao[1], ZHAO Xiao-nan[1], HUANG Zhi-jie[1], HAN Shu-jie[1]
(1. *School of Computer Science, Northwestern Polytechnical University, Xi'an, Shaanxi* 710129, *China*;
2. *School of Automation, Northwestern Polytechnical University, Xi'an, Shaanxi* 710129, *China*)

Abstract: The rapid development of cloud platforms and microservice architectures has made elastic scaling a critical mechanism for ensuring both performance and cost efficiency. Although prior studies have advanced workload forecasting and hybrid modeling, most approaches still focus on predicting resource utilization (e.g., CPU or memory) and then map-

收稿日期:2025-09-24;录用日期:2025-12-08;责任编辑:覃怀银

ping forecasts to scaling actions through threshold rules or controller logic. This forecast-control decoupling amplifies prediction errors and fails to capture practical mechanisms such as hysteresis, cooldown, and discrete scaling steps, thereby limiting deployment feasibility. To overcome these limitations, we directly learn scaling behaviors, modeling replica count dynamics as autoscaler control actions. We propose a hybrid model, ARIMA-BiLSTM-MHA, that integrates ARIMA for long-term trend extraction, BiLSTM for residual sequence modeling, multi-head attention for capturing critical temporal dependencies, and residual correction for improving robustness against bursty and non-stationary workloads. We conduct extensive experiments on the real-world Alibaba cluster-trace-microservices-v2022 dataset, where we systematically compare our method with baselines including PETformer, SparseTSF, TFEGRU, GRU, Transformer, Seq2Seq-LSTM, Seq2Seq-GRU, Seq2Seq-Transfomer, GRU-LSTM, CNN-LSTM and CNN-LSTM-GRU. Our results demonstrate that our approach consistently outperforms existing methods, achieving relative improvements of 1.57%~71.56% (MSE), 0.72%~46.67% (RMSE), 1.57%~59.10% (MAE), 1.97%~60.48% (MAPE), and 0.27%~15.70% (R^2), with R^2 reaching up to 0.954 3. Furthermore, we conduct container replica autoscaling experiments based on the DeathStarBench socialNetwork benchmark. We show that the behavior learning-driven strategy, compared with the CPU-threshold HPA strategy, successfully reduces the average replica count by approximately 17% while lowering the average P99 latency by 2.11% and effectively suppressing tail-latency spikes during load transitions, thereby significantly mitigating resource over-provisioning. We show that our model can more accurately and stably learn and forecast scaling actions, providing forward-looking decision support for autoscaling in practical cloud environments.

Key words: cloud platforms; elastic scaling; scaling behavior learning; multi-head attention; residual correction; hybrid models; deep learning; time series forecasting

Foundation Item(s): National Natural Science Foundation of China (No. 62202382, No. 62272394); Guangdong Basic and Applied Basic Research Foundation (No.2021A1515110080)

1　引言

云计算与微服务架构已成为现代数字基础设施的技术基石. 弹性伸缩(Elastic Scaling)作为云平台的核心机制,要求系统能够根据业务负载的动态变化自动调整容器副本数量,以实现服务的高可用性与资源利用率的最优化. 目前主流平台(如 Kubernetes[1,2])普遍采用基于阈值的横向伸缩控制器(如Horizontal Pod Autoscaler,HPA),动态调节容器副本数量;同时,纵向伸缩控制器(Vertical Pod Autoscaler,VPA)可用于自动调整容器副本的资源请求和限制,以优化资源分配. 然而在面对频繁突发流量、多尺度负载波动以及控制时延敏感等复杂业务场景时,传统阈值方法往往表现出响应滞后、频繁抖动等问题,难以兼顾服务稳定性与资源成本. 由此可见,提升容器副本伸缩控制的智能化水平,构建可部署、可解释且具备控制导向的学习机制,已成为云平台智能管理的重要课题.

已有研究尝试从不同角度探索副本控制的预测辅助机制,大致可归纳为三类范式:第一类是基于资源状态量(如CPU、内存)的预测方法,适用范围广,但预测目标与伸缩控制语义解耦,需要额外的阈值规则或分析器才能触发副本调整,误差在映射过程中易被放大;第二类是融合统计建模与深度学习的组合方案,如趋势-残差分离、注意力增强或多子网协同建模,在拟合精度和多尺度建模方面取得进展,但预测目标大多仍是资源状态量,难以直接驱动扩缩容动作;第三类是面向伸缩行为/控制语义的探索,开始尝试将预测结果用于调度决策,并在容器级进行闭环验证. 然而,现有多数方法依旧以资源指标为输出,需要借助规则映射为副本动作,本质上仍属于"预测-控制解耦",能够直接以副本数为端到端预测目标并无缝对接 HPA 的研究仍十分有限.

基于此,本文选择一种全新的研究思路,即直接对伸缩行为进行学习,具体体现为建模容器副本数的动态变化规律,而不是预测资源使用率. 本文假设公开trace中配置的弹性伸缩策略是合理的,并将副本数的变化轨迹视作真实控制动作的外化结果. 通过对该动态过程的学习,能够间接近似调度器的控制语义,突破传统"预测-控制解耦"范式,形成更具控制导向和部署价值的预测方案. 这一思路不仅能避免规则映射带来的误差放大问题,还能更好地刻画调度器中固有的滞回、冷却和离散动作机制,从而为云平台弹性伸缩提供更贴近实际的智能化支撑.

针对这一目标,本文在"趋势提取+残差建模"的主线基础上,提出一种经多头注意力(Multi-Head Attention,MHA)与残差修正(Residual Correction)增强的ARIMA-BiLSTM-MHA组合模型. 该模型首先利用自回归滑动平均模型(Auto Regessive Integrated Moving Average,ARIMA)对副本数序列进行全局趋势刻画,捕捉周期性与线性变化模式;随后采用双向长短期记忆(Bidirectional Long-Short Term Memory,BiLSTM)神经网络

对残差序列进行非线性建模,学习高频扰动与滞后依赖;并引入 MHA 以突出关键时间步的特征贡献,通过残差修正进一步提升模型在非平稳与突发负载场景下的预测精度与健壮性. 该结构兼具统计建模的可解释性与深度学习的强表达力,在保持模型简洁性的同时,能够高效捕捉复杂的伸缩行为序列,并输出具备控制语义的伸缩动作.

为验证所提方法的实用性与性能优势,在 Alibaba cluster-trace-microservices-v2022 真实微服务 trace 数据集上,针对伸缩变动率最高的微服务 MS_28578 的容器副本数量时序,设计并实施了多步长伸缩行为学习评估,系统性对比了包括 PETformer、SparseTSF、TFEGRU、GRU、Transformer、Seq2Seq-LSTM、Seq2Seq-GRU、Seq2Seq-Transfomer、GRU-LSTM、CNN-LSTM、CNN-LSTM-GRU 等主流统计模型、深度学习模型以及融合结构在内的多种基线方法. 各模型均在统一的数据划分和评测指标(MSE、RMSE、MAE、MAPE、R^2)下进行评估. 实验结果表明,所提出的 ARIMA-BiLSTM-MHA 组合模型在所有评价指标上均显著优于上述各类基线方法,充分验证了其在复杂负载场景下提升伸缩动作学习精度与稳定性. 本文创新点与贡献包括:

(1)提出一种以伸缩行为学习为导向的研究框架,以副本数轨迹作为真实控制动作学习依据,超越以往文献"状态预测→规则映射"的解耦范式,闭合"预测-调度"链路.

(2)设计 ARIMA-BiLSTM-MHA 组合模型,结合 ARIMA 的趋势提取能力与 BiLSTM 的残差建模优势,并通过多头注意力机制自动聚焦关键时序信息,在复杂伸缩场景下实现伸缩行为学习性能的稳定提升.

(3)在真实微服务 trace 数据上与主流模型进行系统性对比,结果表明本方法能够更准确地学习与预测伸缩动作,为云平台弹性伸缩调度的优化与部署提供了高效、可靠的决策支持.

(4)基于 DeathStarBench socialNetwork 基准应用构建容器伸缩控制实验,将行为学习模型直接嵌入伸缩决策链路,与 CPU 阈值驱动的 HPA 策略进行对比. 结果显示,行为学习策略能够显著降低平均副本数,并有效抑制负载跃迁阶段的尾延迟尖峰,从控制效果和部署可行性两个维度验证了所提方法的实用价值.

2 相关工作

云平台资源预测与弹性伸缩控制是云计算系统智能管理的核心议题. 现有研究在方法范式与落地目标上已形成三条较为清晰的谱系:其一是面向状态量(如 CPU/内存/请求到达率)的预测,通过阈值或分析器映射为伸缩动作;其二是融合/组合建模,以趋势-残差分离、注意力增强或多子网协同等机制提升对复杂时序的拟合能力;其三是面向伸缩行为/控制语义的探索,尝试让模型输出能够直接承载扩缩容动作、服务于调度决策. 下文按此脉络归纳代表性工作,并据此明确本文的切入点:在控制语义导向下,直接学习伸缩动作(副本变化)的生成规律,以闭合"预测-控制"链路.

2.1 统计时序模型与传统方法

传统的统计建模与机器学习方法在早期研究中构建了云平台资源预测的理论基础. 文献[3]提出以 ARIMA 为核心并结合排队分析推导资源规模的研究,验证了"预测驱动的主动扩缩容"相较静态/阈值策略在服务质量(Quality of Service,QoS)与成本上的综合优势. 面向多应用与强噪声场景,文献[4]中采用结合 Savitzky-Golay 滤波、小波分解以及多 ARIMA 分量建模的去噪多尺度重构流程,显著提升了统计模型在非平稳负载上的适用性. 文献[5]提出从"三支决策"视角对负载序列进行状态划分并选取二次移动平均(Double Moving Average,DMA)、支持向量回归(Support Vector Regression,SVR)等模型的决策融合路线.

本类方法的目标变量多为状态量(如 CPU 利用率、内存利用率、请求到达率等),对副本数等控制变量缺乏直接建模,且受限于线性/弱非线性假设与单变量输入,对突发、异质与多模态时序的健壮性不足,难以支撑云原生微服务的高动态场景.

2.2 单一深度学习模型

随着深度学习在时序建模中的广泛应用,研究者开始探索长短期记忆网络(Long-Short Term Memory,LSTM)、门控循环单元(Gated Recurrent Unit,GRU)、卷积神经网络(Convolutional Neural Network,CNN)、Informer 等网络在云平台负载预测中的表现. 文献[6]中提到随着深度时序建模发展,LSTM、GRU、CNN、Informer 等在云平台负载预测中广泛应用,该文献通过 Zoneout 正则化与门控改造(如 LSTM-Z)来强化记忆与泛化. 文献[7]、文献[8]中提到通过稀疏注意力与编码器改造提升长序列建模(如 v-Informer 与改进 Informer). 文献[9]中提到利用双向卷积结构(Bidirectional Convolution,BC)增强对波动负载的表征能力(如 BC-LSTM). 文献[10]提出的 PETformer 模型通过将占位符增强技术(Placeholder Enhance Technique,PET)引入到 Transformer 结构,在长序列预测中显式重构时间片划分与通道交互方式,在多数据集上取得了优于传统 Transformer 与线性基线方法的性能. 文献[11]则以 SparseTSF 为代表,利用稀疏建模(Sparse Modeling)与轻量化结构,在保持长期预测精度的同时显著降低参数量与推理开销,体现了轻量且健壮的单模型趋势.

该路线普遍提升了非线性与长依赖拟合能力,但

大多仍以CPU/内存等资源状态量为预测目标,缺少面向副本控制的输出接口. 输入特征多为单源资源序列,与弹性控制器(如HPA/VPA)对接不足.

2.3　融合与组合建模

为突破单一模型性能瓶颈,近年来研究者广泛探索融合式与组合式预测架构,其核心思路是在结构、特征或决策层引入多元协同机制,以提升对复杂时序的建模能力. 相关研究大致可归纳为以下三类主线:

第一类是"统计+深度"的两阶段组合模型,通常以ARIMA、三次指数平滑(Triple Exponential Smoothing, TES)、经验模态分解(Empirical Mode Decomposition, EMD)、变分模态分解(Variational Mode Decomposition, VMD)等方法提取全局趋势,再辅以LSTM、GRU、最小二乘支持向量机(Least Square Support Vector Machine, LSSVM)等深度网络建模残差波动. 此类方法代表如文献[12~16]提出的EMD-TCN、VMD-ISSA-LSSVM、ARIMA-LSTM等架构,有效提升了在非平稳负载下的拟合精度与泛化能力. 文献[17]进一步融合门控CNN与Transformer,面向容器负载构建嵌套组合结构. 但这类方法的预测目标普遍聚焦于CPU、内存等资源状态量,未对接副本控制语义.

第二类是多子网协同建模路径,通过并行或串联多个深度子网络(如GRU、CNN、BiLSTM)以捕捉多尺度依赖与异质特征. 文献[18~21]构建了如GRU-LSTM串联、双通道时序融合、结合多元变分模态分解(Multivariate Variational Mode Decomposition, MVMD)与MHA的BiLSTM、多变量Transformer等结构,在长/短期关联建模与输入异构性适配方面表现突出. 文献[22]提出的时频增强门控循环单元(Time-Frequency Enhanced Gated Recurrent Unit, TFEGRU)模型则进一步将时频分析、门控循环单元与注意力机制结合,通过在频域增强负载序列表征,并在时域上引入注意力加权的GRU结构,有效提升了云工作负载预测在多频段波动场景下的精度与稳定性. 文献[23]、文献[24]均引入粒子群优化(Particle Swarm Optimization, PSO)机制,分别用于调优输入经Savitzky-Golay滤波处理的LSTM模型与门控循环单元-循环神经网络组合模型(Gated Recurrent Unit-Recurrent Neural Network, GRU-RNN)的结构参数,以提升负载预测精度与模型对复杂场景的适应能力.

第三类关注容器行为的结构/语义组合建模. 文献[25]提出基于高效监督学习的深度神经网络(efficient supervised learning-based Deep Neural Network, esDNN)模型,通过GRU、1D-CNN与多变量时间序列预测的滑动窗口构造(Sliding window for Multivariate Time series Forecasting, S-MTF)协同建模容器多维负载. 文献[26]提出了GROUP模型,该模型创新性地将STL(Seasonal and Trend decomposition using Locally estimated scatterplot smoothing)分解与容器组聚类引入到CNN-BiGRU多步预测组合模型. 文献[27]提出的COIN框架通过显式分解工作负载中的共性与个性模式,引入双通道GRU模型来建模集群级容器行为变动. 这些方法增强了对容器动态与结构特征的刻画能力,在实际微服务场景中具有一定部署潜力,但其预测目标仍集中于资源状态量,缺乏面向副本控制的输出接口.

另有少数研究尝试将组合模型融入弹性控制系统中进行闭环验证. 文献[28]以CNN-LSTM-GRU多结构融合为基础,结合混合贝叶斯优化(Hybrid Bayesian Optimization, HBO)调参,以内存负载需求为预测目标,在CloudSim模拟器上实现了虚拟机级内存资源伸缩控制. 尽管该工作验证了组合模型的系统性能潜力,但其控制对象为虚拟机,预测变量为状态量,且实验环境难以支撑容器级副本控制建模,因而难以直接迁移至微服务弹性伸缩场景.

综上,融合与组合模型在多尺度表达与非线性建模方面取得了显著进展,成为当前主流路线之一. 然而,当前绝大多数工作仍停留于资源状态预测层面,但在目标变量选择、结构通用性与控制语义输出等方面仍存不足,难以直接驱动副本粒度的弹性决策,亟需简洁、可部署且具控制导向的预测方案予以补位.

2.4　面向伸缩行为/控制语义的方法

随着研究逐渐从资源状态预测转向系统级控制语义,学界开始探索如何让预测结果能够直接承载扩缩容动作,以实现"预测-控制闭环". 在文献[29]中的工作负载学习框架以序列到序列(Seqence to Sequence, Seq2Seq)与注意力机制(Attention)为核心,并配合平滑与优化模块改进伸缩稳定性,体现了"预测-调度一体化"的系统取向. 文献[30]中提出了基于CNN-BiGRU-Attention组合预测模型的容器调度方案(Container Scheduling Strategy based on a Load Prediction Model, CSSLPM),分别在CloudSim与DeathStarBench Train-Ticket基准应用上实现了容器层级构建预测与调度一体化策略的闭环验证,有效地评估了节点扩缩容与容器迁移效果.

在少量工作中,预测目标已经从资源状态前移到副本数本身. 文献[31]提出的ADApt框架面向边缘计算场景,首先利用K-means聚类算法监控边缘设备的资源利用率,旨在识别物理设备的过载异常;仅当检测到设备异常时,系统才触发梯度提升回归模型来预测缓解该异常所需的副本数量. 尽管该工作与本文都关注副本数这一控制变量,但在建模范式与适用场景上存在显著差异. 一方面,ADApt属于"设备异常驱动"模式,其伸缩动作是被动触发的,旨在解决底层资源瓶

颈;而本文模型属于“负载需求驱动”模式,通过对副本轨迹的连续时序建模,主动适配业务层的负载波动,从而确保上层服务的性能稳定;另一方面,ADApt的运行严格依赖于物理设备的连续监控数据以计算异常分数,而本文在数据特征上则聚焦于容器级别的资源利用率指标与副本伸缩控制结果. ADApt在输出层显式涉及副本数,较传统“资源预测→阈值映射”的方案更接近控制语义,对“直接建模伸缩行为”的探索具有代表性. 此外,该框架的侧重点在于边缘设备过载场景下的单次副本配置决策,其建模范式仍以基于时间窗口的静态回归为主,主要回答“当前状态下应配置多少副本”,并未显式刻画副本轨迹的多步演化、调度滞后与冷却/离散步长等控制特性. 这种伸缩控制逻辑属于设备异常驱动的被动响应模式,与本文旨在学习的负载需求驱动的伸缩行为控制存在本质差异. 相比之下,本文面向云平台微服务场景,将完整的副本数时间序列作为预测对象,注重伸缩控制器在时间轴上的行为轨迹,与ADApt框架在应用场景和研究层级上具有一定的互补性.

需要强调的是,判别“是否面向控制行为”的标准在于输出是否直接表达扩缩容语义. 本文将输出可直接映射为scale-out/scale-in的离散动作或目标副本数视为“控制语义直接表达”. 若仍需经阈值/分析器二次映射,则归为“预测-控制解耦”. 现有多数实现仍以资源指标为输出,并依赖阈值/分析器映射为副本动作,本质上依旧是“预测-控制解耦”. 能够直接以伸缩动作(副本变化)为学习目标,并能够对接HPA的工作仍然有限,且多停留在单步决策与边缘场景. 对多尺度依赖、突发波动以及调度器固有的滞回、冷却、离散步长的刻画亦不充分,导致控制稳定性与部署可用性受限. 相较之下,本文的伸缩行为学习模型以副本轨迹作为核心输出,通过趋势-残差分离与深度序列建模显式捕捉伸缩动作的时间相关性与控制特性,在目标层级与控制语义表达上与上述工作形成互补,这正是本文切入的缺口.

2.5 优化增强与结构创新

跨越上述路线,亦有研究从训练-推理机制提出增强策略. 文献[32]中m-gap预测(为调度器留出反应窗口)与基于聚类的分组建模,在不改动主干结构的前提下,提升调度友好性与异质任务适配. 文献[33]从系统层面对云平台负载预测进行梳理,覆盖统计、深度、混合与集成等主流模型及增强手段,并在多数据集下进行测试分析,以总结性能与代价权衡. 然而,该文献所覆盖的大量工作大多以CPU/内存等资源状态量为核心目标,鲜有面向副本数这一控制变量的端到端建模与可部署输出.

综上所述,针对上述空白,本文在控制语义导向下提出伸缩行为学习模型:以副本轨迹作为真实控制动作,基于“趋势-残差分离”的ARIMA-BiLSTM主干并引入多头注意力与残差修正以强化关键滞后依赖与突发波动刻画;模型输出直接具备scale-out/scale-in判定语义,可与HPA直接对接,直接推动HPA控制策略中副本数的调整决策,从而闭合预测-调度链路. 对于VPA,其实现则需要进一步的映射步骤:即从副本数变化到资源请求变化量的转化,以完成对资源利用率(如CPU和内存)变化的响应. 与既有方法相比,本文在目标层级(控制量而非状态量)与输出可部署性(无需规则映射即可驱动伸缩)上形成了清晰差异,并以简洁、可实施的结构为微服务弹性伸缩提供针对性支撑.

3 问题定义与技术背景

3.1 问题定义

在容器化云原生系统中,弹性伸缩的本质是调度器根据负载变化触发扩容/缩容动作,其外化表现为副本数量的动态调整. 与传统CPU、内存利用率预测不同,本研究的目标并非仅预测资源状态量,而是直接学习容器的伸缩行为. 这一问题本质上是一个控制语义学习问题:副本数量的演化轨迹可视作调度器动作的外部表征,对其进行建模即可近似捕捉“预测-调度”链路的隐含机制.

在数据层面,伸缩行为最终外化为副本数随时间的动态序列. 进一步地,副本数变化往往与系统的外部资源特征紧密耦合,如平均CPU利用率、内存利用率或调用上下文负载模式. 因而,本文将伸缩行为学习形式化为一个多源时序建模问题. 设时刻t的副本数为$y(t)$,对应的外部资源特征向量为$\boldsymbol{x}(t)\in R^d$,历史观测序列可有如下表示.

设历史时刻t的容器副本数为$y(t)$,历史序列Y如式(1)所示:

$$Y=\left\{y(\tau),x(\tau)\middle|\tau=1,2,\cdots,t\right\} \tag{1}$$

其中,$x(\tau)$表示在时刻τ的外部特征量集合,可以是一维特征(如单一资源指标)或多维矩阵化特征(如多类资源、链路流量、调用上下文),具有可扩展性和开放性. 基于长度为p的多源输入序列,目标是预测未来H个时间步的伸缩控制语义输出(即副本数),其预测过程可形式化为式(2)~(4)所示:

$$\begin{aligned}\hat{y}(t+1)=&f_\theta\big(y(t),y(t-1),\cdots,y\big(t-p+1\big),\\&x(t),x(t-1),\cdots,x\big(t-p+1\big)\big)\end{aligned} \tag{2}$$

$$\begin{aligned}\hat{y}(t+2)=&f_\theta\big(y(t+1),y(t),\cdots,y\big(t-p+2\big),\\&x(t+1),x(t),\cdots,x\big(t-p+2\big)\big)\end{aligned} \tag{3}$$

$$\cdots$$

$$\hat{y}(t+H)=f_\theta\big(y(t+H-1),y(t+H-2),\cdots,y(t-p+H),\\ x(t+H-1),x(t+H-2),\cdots,x(t-p+H)\big) \tag{4}$$

其中,$y(t)$表示时刻t的副本数(观测值),$\hat{y}(t+i)$表示对未来第$t+i$时刻副本数的实值预测($i=1,2,\cdots,H$),p为历史观察窗口长度,$f_\theta(\cdot)$为过去连续p个时刻的副本数到未来第i步副本数的参数化伸缩行为学习函数(映射关系),即$f_\theta:R^{p(1+d)}\to R$,H为预测步长.

需要指出的是,虽然式(2)~(4)的输出形式上是副本数预测,但其差分形式如式(5)所示:

$$\Delta\hat{y}(t+i)=\hat{y}(t+i)-\hat{y}(t+i-1) \tag{5}$$

直接对应调度器的伸缩动作(scale-out/scale-in).因此,本研究的问题定义不仅涵盖了时间序列预测,更是一次对伸缩行为语义的直接学习,为闭合"预测-调度"链路奠定了基础.工程落地时,为将实值预测转化为可执行的离散控制,定义后处理算子为式(6)、式(7)所示:

$$\begin{aligned}\Delta\tilde{y}(t+i)&=\Gamma\big(\hat{y}(t+i)\big)\\&=\mathrm{clip}(\mathrm{round}(\hat{y}(t+i)),y_{\min},y_{\max})\end{aligned} \tag{6}$$

$$a(t+i)=\mathrm{clip}\big(\Delta\hat{y}(t+i),-\kappa_{\downarrow},+\kappa_{\uparrow}\big) \tag{7}$$

其中,$\Gamma(\cdot)$集成了取整、边界约束与最大步长限制,$-\kappa_{\downarrow}$以及$+\kappa_{\uparrow}$可结合实际冷却/滞回策略设置.在线部署时,$\Delta\hat{y}(t+i)$或$a(t+i)$可直接映射到HPA控制接口.与传统状态量预测相比,该问题的挑战性体现在:副本序列具有离散性、非平稳性和策略驱动性,其背后包含调度器的滞回、冷却与突发响应机制,远难于平滑资源曲线的建模.同时,引入外部资源特征集合作为多源输入,不仅提升了问题的抽象层次,也为捕捉复杂调度逻辑提供了更丰富的上下文,使预测结果更具控制导向性与工程价值.多源输入所需要的跨尺度依赖与语义耦合要求模型同时具备趋势刻画、非线性记忆、关键片段聚焦与局部误差补偿能力.

3.2　技术背景

根据3.1节讨论的问题定义,本文将伸缩行为学习形式化为一个以容器副本数量为目标预测输出的多源时间序列建模问题.为支撑这一任务,本文选择并融合了四类具有互补优势的技术:ARIMA、LSTM、多头注意力与残差修正.它们分别对应副本数演化中的长期趋势、非线性依赖、关键时刻捕捉与误差补偿,构成后续模型设计的理论与方法基础.

3.2.1　ARIMA模型

ARIMA模型是一类广泛应用于时间序列分析与预测的统计建模方法,特别适用于具有明显平稳性或可通过差分操作转化为平稳序列的数据场景.在云计算与弹性容器服务的副本数预测任务中,ARIMA模型能够兼顾理论严谨性与实践适用性,对历史副本数序列的动态演化进行高效建模.

如图1所示,ARIMA建模流程主要包括数据采集、平稳性检验、差分转换、定阶分析(ACF/PACF)、参数估计、模型诊断与预测输出等步骤.

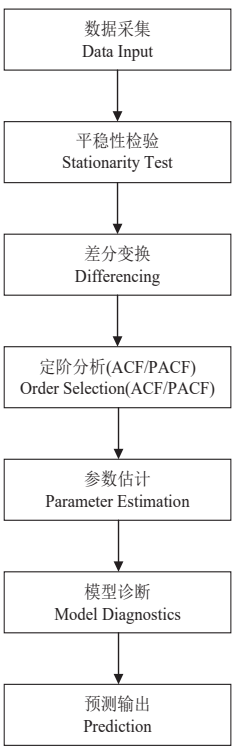


图1　ARIMA模型建模流程

ARIMA模型通过引入自回归(AR)、差分(I)和滑动平均(MA)三类参数,有效捕捉序列的长期趋势、短期扰动及噪声影响.其一般数学表达如式(8)所示:

$$y_t=\sum_{i=1}^{p}\phi_i y_{t-i}+\sum_{j=1}^{q}\theta_j\varepsilon_{t-j}+\varepsilon_t \tag{8}$$

其中,y_t表示时刻t的副本数(预测目标);ϕ_i是AR部分的系数,反映历史副本数的影响权重;p为自回归项的阶数;θ_j是MA部分的系数,捕捉历史扰动(白噪声)对当前预测的影响;q为滑动平均项的阶数;ε_t是白噪声扰动项,满足零均值、有限方差.

针对非平稳时间序列,ARIMA通过d阶差分操作消除趋势成分,实现序列平稳化,记作式(9):

$$y_t^{(d)}=(1-B)^d y_t \tag{9}$$

其中,B为滞后算子;d为差分阶数;$y_t^{(d)}$表示对原始序列y_t进行d次差分后的结果.通过调节p、d、q参数,ARIMA(p,d,q)可适配各种具有自相关结构的时间序列数据.

结合本文背景,ARIMA在模型应用流程中,首先对

原始副本数序列进行平稳性检验(如ADF检验),若非平稳则通过差分操作转化为平稳序列.随后,通过自相关函数(ACF)与偏自相关函数(PACF)图形分析,初步确定p和q的取值,如图2所示.模型训练阶段,采用历史数据估计参数,最终利用拟合模型对未来副本数序列进行递推预测.

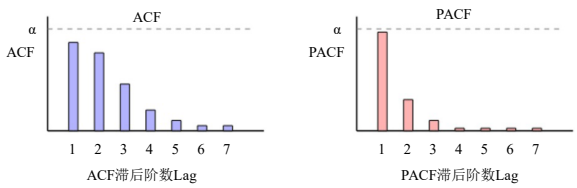


图2 ACF与PACF图例

理论上,ARIMA模型能够有效分离副本数量的长期演化趋势与短时扰动影响,适合建模具有明显周期性或缓慢波动行为的序列.在实践中,差分操作与参数调优共同提升了模型的泛化能力与预测精度,为容器副本数的自动扩缩容提供了数据驱动的决策支持.

3.2.2 LSTM长短期记忆模型

LSTM模型是一种经典的循环神经网络结构,旨在解决传统循环神经网络模型(Recurrent Neural Network,RNN)在长序列学习过程中存在的梯度消失与爆炸问题.其本质在于引入可学习的门控机制,对历史信息实现选择性记忆与遗忘,从而有效捕捉时间序列中的长期依赖特征.如图3所示,在LSTM单元内部,记忆状态C_t贯穿整个序列信息流,决定了每一时刻信息的遗留与更新.以时间步t为例,LSTM单元的输入包括当前输入z_t以及上一时刻的输出y_{t-1},二者拼接后依次流经遗忘门、输入门和输出门,以调控信息的存储与释放.

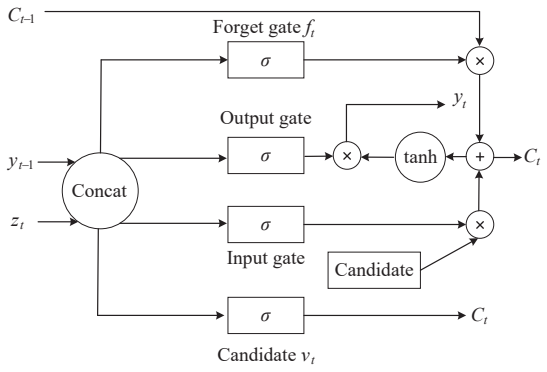


图3 LSTM单元结构示意图

首先,遗忘门的输出f_t控制前一时刻记忆C_{t-1}有多少被保留到当前单元,其计算如式(10)所示:

$$f_t=\sigma\left(\boldsymbol{W}_f\cdot\left[y_{t-1},z_t\right]+\boldsymbol{b}_f\right) \tag{10}$$

其中,$\sigma(\cdot)$表示sigmoid激活函数,$\boldsymbol{W}_f$与$\boldsymbol{b}_f$分别为遗忘门的权重矩阵与偏置向量.输入门i_t决定当前新输入写入记忆单元的强度,候选记忆v_t通过式(11)、式(12)计算得到:

$$i_t=\sigma\left(\boldsymbol{W}_i\cdot\left[y_{t-1},z_t\right]+\boldsymbol{b}_i\right) \tag{11}$$

$$v_t=\tanh\left(\boldsymbol{W}_v\cdot\left[y_{t-1},z_t\right]+\boldsymbol{b}_v\right) \tag{12}$$

其中,$\boldsymbol{W}_i$、$\boldsymbol{W}_v$、$\boldsymbol{b}_i$、$\boldsymbol{b}_v$分别为输入门和候选记忆的参数矩阵与偏置向量.当前记忆状态的更新由遗忘门与输入门加权融合历史记忆和新信息,具体表达为式(13)所示:

$$C_t=f_t\odot C_{t-1}+i_t\odot v_t \tag{13}$$

其中,$\odot$表示Hadamard诸元素乘积.输出门o_t负责控制当前记忆单元对外部输出的程度,其计算方式如式(14)所示:

$$o_t=\sigma\left(\boldsymbol{W}_o\cdot\left[y_{t-1},z_t\right]+\boldsymbol{b}_o\right) \tag{14}$$

最终,当前单元的输出如式(15)所示:

$$y_t=o_t\cdot\tanh\left(C_t\right) \tag{15}$$

其中,z_t表示当前时刻输入观测(如历史副本数),y_{t-1}为上一时刻网络输出,C_{t-1}为上时刻记忆状态,$\boldsymbol{W}_o$、$\boldsymbol{b}_o$均为需学习的模型参数.LSTM通过上述门控结构,不仅可灵活记忆或遗忘不同时刻的重要信息,还能有效防止长序列训练中梯度的衰减或爆炸,因此在复杂时间序列建模领域具有极高的表达力.理论上,LSTM能够近似任意复杂的非线性动态系统,其对长期与短期依赖建模的能力已被广泛证实,尤其适合应用于具有周期性、趋势性和突发性的时间序列预测等场景.在实际应用中,LSTM模型的性能依赖于输入序列窗口长度、网络层数、隐藏单元规模等关键超参数的合理选择,并可与残差修正、注意力机制等结构模块协同设计,以进一步提升建模灵活性与泛化能力.

3.2.3 MHA

注意力机制最早源于神经机器翻译领域,旨在模拟人类在处理信息时的“聚焦”能力,即在给定输入序列的不同位置赋予不同的权重.传统单头注意力模型通过一次查询-键-值(Query-Key-Value)操作实现上下文建模,然而其表达能力有限,难以捕捉多层次、不同子空间的信息交互.Vaswani等人[34]在Transformer架构中提出了MHA,显著提升了模型的表示能力与泛化能力.

图4展示了多头注意力机制的基本结构.给定输入序列的表示$X\in R^{T\times d}$model,X首先通过线性映射分别获得多个查询(Query)、键(Key)、值(Value)矩阵,具体第h个头的计算方式如式(16)~(18)所示:

$$Q_h=X\boldsymbol{W}^{Q_h} \tag{16}$$

$$K_h=X\boldsymbol{W}^{K_h} \tag{17}$$

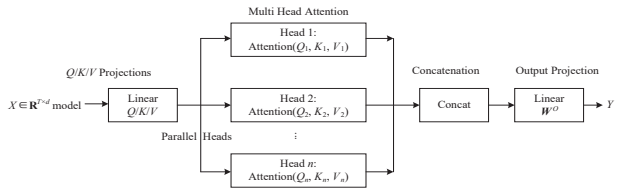


图4　多头注意力机制基本结构

$$V_h = XW^{V_h} \tag{18}$$

其中,W^{Q_h}、W^{K_h}、$W^{V_h} \in \mathbf{R}^{d_{\text{model}} \times d_k}$ 为第 h 个注意力头的可训练参数,$d_k = d_v = d_{\text{model}}/H$,$H$ 为注意力头数. 对于每个头,独立计算其缩放点积注意力如式(19)所示:

$$\begin{aligned} \text{Head}_h &= \text{Attention}\left(Q_h, K_h, V_h\right) \\ &= \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right) V_h \end{aligned} \tag{19}$$

随后,将所有头的输出拼接后再次映射,得到式(20):

$$\text{MultiHead}(X) = \text{Concat}\left(\text{Head}_1, \text{Head}_2, \cdots, \text{Head}_h\right) W^O \tag{20}$$

其中,$W^O \in \mathbf{R}^{d_{\text{model}} \times d_{\text{model}}}$ 是最终线性变换的参数矩阵. 多头机制的核心优势在于:每个注意力头可独立建模序列中的不同位置依赖与语义特征,进而提升模型在时间序列、语言建模、图结构建模等任务中的泛化性能与稳定性. 在本研究中,本文引入多头注意力模块作为可选组件嵌入到LSTM或其他时序模型中,以增强模型对不同时间尺度、不同副本行为模式的感知能力. 相比单头注意力,其在捕捉周期性变化、延迟突变及局部波动等复杂动态特征方面表现出更优的适应性.

3.2.4　残差修正

在时间序列预测任务中,复杂的数据生成机制往往包含趋势、周期、扰动和异常等多种成分,不同建模结构在拟合这些特征方面存在各自的偏好与局限. 尽管主干预测模型(如LSTM、Transformer等)已具备强大的建模能力,但在实际应用中,仍可能存在对短期突变、局部高频变化响应不及时,或对残留误差收敛不足的现象. 尤其在副本数时间序列预测问题中,这种问题表现为:模型在整体趋势把握准确的前提下,仍存在对临时副本波动响应偏缓、短期调整误差放大的问题. 这种"主预测误差残留"现象在微服务等动态系统中极易导致资源分配滞后或浪费.

残差修正机制(Residual Correction)正是在上述背景下被引入的一种结构性增强策略. 如图5所示,其核心思想是:不试图将所有动态特征交由主模型一并建模,而是将预测误差(残差)显式地作为独立建模对象,通过引入轻量级补偿模块,对主模型输出结果进行细粒度校正,从而提升整体预测性能. 这一设计延续了传统统计建模中的"主模型+剩余项"的思想,强化了系统对不可预期扰动与难建模模式的响应能力.

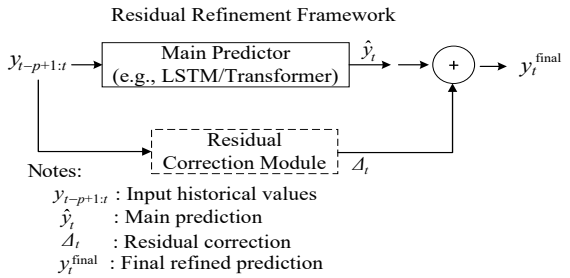


图5　残差修正示意图

设主干模型对时间步 t 的预测结果为 $\hat{y}_t$,残差修正模块输出为 Δ_t,则最终输出值为式(21):

$$y_t^{(\text{final})} = \hat{y}_t + \Delta_t \tag{21}$$

其中,Δ_t 表示基于历史状态特征 H_t(可能包含 y_{t-1}, $y_{t-2}, \cdots, z_t$)所建模的残差映射,即式(22):

$$\Delta_t = \phi\left(H_t\right) \tag{22}$$

$\phi(\cdot)$可由简单的前馈网络或可学习的注意力模块构成,用以提取主模型难以捕捉的残差模式并进行结构性补偿.

残差修正机制在建模中的优势主要体现在三个方面:首先,其对主模型形成了一种"结构外部监督",有效缓解了模型对复杂扰动模式的拟合瓶颈;其次,该机制在参数开销有限的前提下提升了模型容量,增强了模型表达的多样性与泛化能力;最后,残差路径具有良好的可控性与可解释性,便于调试与部署,适合以模块化方式嵌入现有模型框架.

综上所述,容器副本数量的动态演化并非平滑的资源曲线,而是调度器在业务负载、依赖拓扑与系统约束下所展现出的离散伸缩行为. 这种行为具有显著的非平稳性、离散性与突发性,单一模型往往只能捕捉长期趋势,却难以精准响应扩缩容动作中的瞬时波动与局部异常. 为此,本文提出了一种多层次、互补式的建模框架:在主干层面,采用ARIMA与BiLSTM的协同结合. ARIMA负责稳定提取长期趋势与周期模式,BiLSTM则通过双向循环并行建模前后上下文,强化对周期极值与突发事件的敏感性. 在离线训练中可充分利用双向依赖提升表达力,而在上线推理中仍按时间正向滚动,不依赖未来观测,兼顾可部署性与预测精度.

在此基础上,引入多头注意力机制以实现跨时间尺度的关键片段聚焦,使模型能够自适应捕捉突发负载与延迟依赖. 同时设计残差修正模块,对主干难以覆盖的高频扰动与局部异常进行细粒度补偿,显式提供了一条"结构化外部监督"路径,进一步提升模型的健壮性与控制语义逼近能力.

在上述组合下，模型不仅在数值预测精度上实现了显著提升，更重要的是将预测任务从“状态拟合”提升为“行为刻画”：通过对副本演化轨迹与动作差分的联合建模，间接捕捉调度器在复杂场景下的决策逻辑．并结合后处理算子$\Gamma(\cdot)$的取整、约束与步长控制，将输出直接映射为可对接HPA的可执行伸缩语义．由此，本文在不改变实验评测口径的前提下，提出了一种可解释、可部署且工程友好的方案，以闭合“预测-调度”链路．

4 系统设计与实现

本章围绕构建容器伸缩行为学习模型展开，从数据构建与行为度量出发，完成到“趋势-残差-控制语义”的端到端实现．与传统“负载或状态预测”不同，本文将副本数演化视为弹性控制器在业务压力、依赖拓扑与平台约束下的离散决策轨迹，并据此设计可部署的学习-映射链路．

4.1 系统整体架构

系统设计架构示意如图6所示，从系统视角看，容器伸缩行为学习与预测系统由三大核心功能模块构成：

(1)数据采集与分析模块．负责从云平台监控系统中读取原始微服务运行指标数据(包含CPU利用率、副本数量等)，对多副本实例进行清洗与聚合，统计伸缩活跃度和相关性，并按统一格式构建训练、验证和测试数据集，为后续建模提供高质量时序样本．

(2)组合预测模型模块．在数据样本基础上，构建ARIMA-BiLSTM-MHA组合预测模型，对容器副本数序列进行趋势-残差联合建模，实现多步副本数前瞻预测，是整个系统的核心学习与推理组件．

(3)评估与持久化模块．对模型输出进行误差评估与结果可视化，计算MSE、MAE、MAPE、R^2等评估指标，并将预测轨迹、绘图文件以及模型参数统一持久化，支持后续复现与部署，最终完成未来副本数序列与评估指标结果的输出．

4.2 数据采集与分析

4.2.1 数据来源与数据清洗

本文使用Alibaba发布的微服务集群公开数据集cluster-trace-microservices-v2022[21,35]中的部分数据，通过官方脚本fetchData.sh下载MSMetrics文件集中的部分文件，即MSMetrics_0至MSMetrics_669共670个CSV文件，覆盖了28 611个微服务在前335 h内的详细运行数据．每个CSV文件记录了对应时间窗口内所有微服务副本实例的资源利用率和运行状态，数据记录的采样间隔为60 000 ms(即1 min)，每个CSV包含30个采样点，即覆盖30 min的记录．表1展示了MSMetrics数据表的核心字段及其含义．

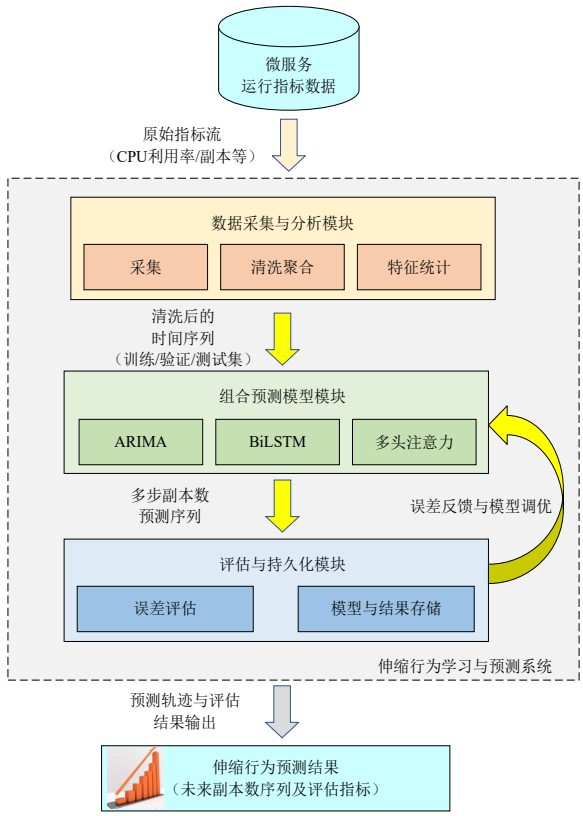


图6 系统设计架构示意图

表1 MSMetrics数据集字段构成

字段名称	类型	含义
timestamp	整数	时间戳(ms)
msname	字符串	微服务逻辑唯一标识
msinstanceid	字符串	微服务容器副本实例逻辑唯一标识
nodeid	字符串	计算节点唯一标识
cpu_utilization	浮点	CPU利用率
memory_utilization	浮点	内存利用率

表2给出了MSMetrics单条数据记录的样例，记录表示：在采样的第60 s(时间戳60 000 ms)，微服务MS_28578的容器副本POD_373正在节点NODE_41756上运行，其在采样时刻的CPU利用率约为79.44%，内存利用率约为44.40%．整个数据集通过对不同副本、节点、时间的联合观测，为后续建模与特征工程提供了高质量、细粒度的时序数据基础．

表2 MSMetrics数据集记录举例

字段名称	值
timestamp	60 000
msname	MS_28578
msinstanceid	MS_28578_POD_373
nodeid	NODE_41756
cpu_utilization	0.794 385 000 005 364 6
memory_utilization	0.444 004 058 837 890 6

在本研究中,容器副本数量 replica_count 被定义为:同一时间戳下、同一微服务(msname)所出现的不同副本(msinstanceid)记录的总数,如式(23)所示:

$$\text{replica\_count}(t,m) = \left|\left\{\text{msinstanceid}\left(\text{timestamp}=t,\text{msname}=m\right)\right\}\right| \tag{23}$$

即,统计每一对(timestamp,msname)出现的唯一 msinstanceid 数量,作为该微服务此刻的副本数. 进一步地,cpu_avg 和 mem_avg 分别表示同一时间戳下、同一微服务所有副本 CPU 利用率与内存利用率的算术平均值,反映该服务整体的资源使用情况,如式(24)、式(25)所示,其中 $N_{t,m}$ 是通过统计当前时间戳 t 和微服务 m 对应的所有唯一 msinstanceid 数量得到的,即容器副本的数量.

$$\text{cpu\_avg}(t,m) = \frac{1}{N_{t,m}}\sum_{k=1}^{N_{t,m}}\text{cpu\_utilization}_{t,m,k} \tag{24}$$

$$\text{mem\_avg}(t,m) = \frac{1}{N_{t,m}}\sum_{k=1}^{N_{t,m}}\text{mem\_utilization}_{t,m,k} \tag{25}$$

为保证数据分析的准确性,本文采用如算法 1 所示的数据清洗流程.

算法 1　数据清洗算法

输入:原始 MSMetrics 记录集 R

输出:清洗后的聚合表 T

1. 初始化:T
2. 对 R 中每个唯一的(timestamp, msname)组重复步骤 3~7
3. S←选取所有满足 timestamp=t 且 msname=m 的记录
4. replica_count←S 中不同 msinstanceid 数量
5. cpu_avg←S 中所有 cpu_utilization 的均值
6. mem_avg←S 中所有 memory_utilization 的均值
7. 将(t, m, replica_count, cpu_avg, memory_avg)添加到 T
8. 返回 T

4.2.2　微服务伸缩活跃度分析

为系统刻画集群中各微服务的弹性伸缩行为,本文对副本数变化类型做出如下界定:当连续两个观测时刻副本数量增加时,记为一次扩容;减少时,记为一次缩容;若无变化,则视为稳定. 据此,定义总伸缩次数为扩容与缩容次数之和,总观测次数为全部相邻观测对的总数. 进一步,本文用伸缩变动率(Replica Change Rate,RCR)衡量微服务副本数动态变化的活跃度,其计算如式(26)所示:

$$\text{RCR} = \frac{N_{\text{out}}+N_{\text{in}}}{N_{\text{obs}}} \tag{26}$$

其中,N_{out} 为扩容次数(scale-out count),N_{in} 为缩容次数(scale-in count),N_{obs} 为总观测次数(total observation count). 该指标反映了微服务在给定观测窗口内弹性调节的频率和强度.

如图 7 所示,对前 335 h 内的运行数据进行统计发现,绝大多数微服务的伸缩变动率极低,即副本数长期保持稳定,仅有极少数服务表现出显著的弹性变化.

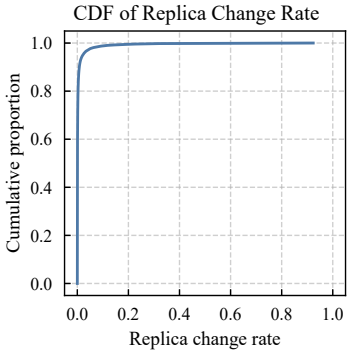


图 7　容器副本数伸缩变动率累积分布(CDF)

表 3 进一步给出了伸缩变动率排名前五的微服务,其活跃度、伸缩行为等特征远高于平均水平,显示出强烈的动态弹性需求. 由表 3 和图 7 可见,MS_28578 等极少数服务在观测期间表现出极高的伸缩活跃度,其副本数变动率远超其他微服务. 因此,本文选取 MS_28578 作为弹性副本数预测模型的研究对象,具有显著的代表性和挑战性.

表 3　伸缩变动率 Top-5 微服务

微服务 ID	扩容次数	缩容次数	总伸缩次数	总观测次数	伸缩变动率/%
MS_28578	9 790	8 772	18 562	20 099	92.35
MS_30502	9 076	9 149	18 225	20 099	90.68
MS_6945	8 887	8 895	17 782	20 099	88.47
MS_26991	8 416	8 415	16 831	20 099	83.74
MS_58286	7 413	7 387	14 800	20 099	73.64

为了直观展示高活跃服务的时间特征,图 8 给出了 MS_28578 在前 335 h 内的副本数随时间变化曲线. 曲线呈现出明显的周期性起伏与陡峭的上/下切换,既存在稳定平台段,又伴随密集的扩缩容事件. 综合 CDF 的总体结论、Top-5 排序及时间序列形态,MS_28578 既具有充足的伸缩事件(有利于训练与检验),又具有强烈的非平稳性与突发性(对模型具有挑战性),因此被选定为本文后续弹性副本数预测与方法验证的代表性目标.

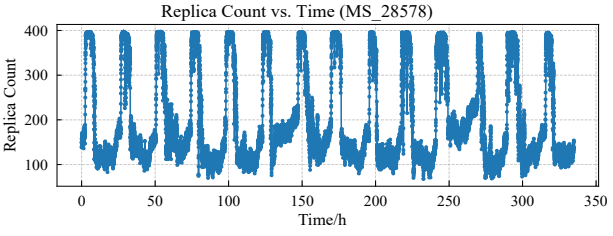


图 8　MS_28578 容器副本数随时间变化曲线

4.2.3 数据相关性分析

在建立伸缩行为学习模型前，本文需要对主要变量做相关性分析，以厘清“资源状态-伸缩动作”的统计关系．因此，有必要定量分析各特征之间的相关性，以指导输入变量的选择．相关性分析不仅能够揭示伸缩结果即副本数（replica_count）与主要资源利用指标（cpu_avg、mem_avg）之间的统计关系，还能为后续回归建模的合理性提供理论依据．本文采用皮尔逊相关系数（Pearson Correlation Coefficient）衡量不同特征间的线性相关性，其定义如式（27）所示：

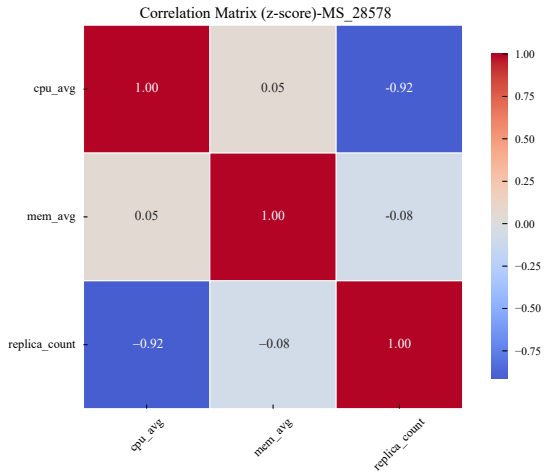
$$r_{X,Y}=\frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\overline{Y})^2}} \tag{27}$$

其中，X_i和Y_i分别为第i个观测样本，$\overline{X}$和$\overline{Y}$分别为X和Y的均值，n表示样本数量．$r_{X,Y}$的取值范围为$[-1,1]$，绝对值越大表示线性相关性越强，正值为正相关，负值为负相关．

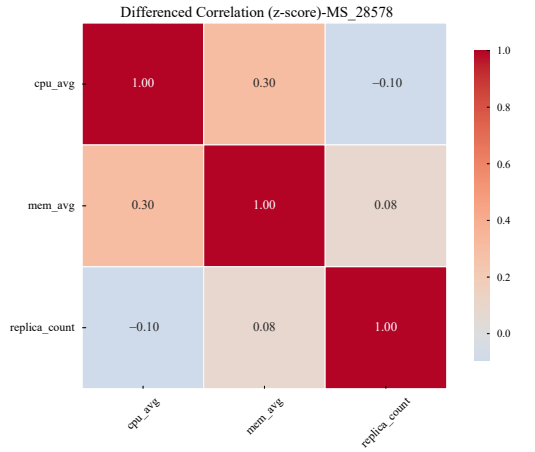
图9（a）展示了经过标准化（z-score）处理后的MS_28578服务的原始序列相关矩阵．可以观察到，副本数与CPU利用率之间呈现极强的负相关（Pearson系数为–0.92），而与内存利用率的相关性仅为–0.08，几乎可以忽略．这一结果表明，CPU利用率与副本变化之间存在显著的统计联系，虽然这种联系在数值上体现为扩容后CPU被均摊的效应，但它依然能够反映伸缩动作发生前后的资源压力差异．进一步地，图9（b）给出了同一服务序列经一阶差分处理后的相关性热力图，可以看到所有相关系数的绝对值均急剧下降，副本数与CPU、内存的差分相关性分别为–0.10和0.08，二者基本不再相关，说明CPU的瞬时变化并不是副本调整的直接线性驱动因素．

综合来看，副本数的历史序列固然是建模伸缩行为的核心输入输出特征，但CPU指标作为外部上下文，能够补充刻画扩缩容触发时资源利用的变化背景．为此，本文在后续建模中采用副本数与CPU利用率的联合输入：副本数序列用于学习自回归趋势与行为轨迹．CPU序列用于提供伸缩触发的上下文信息．该设计既保证了模型对伸缩行为的敏感性，又避免了单变量建模可能带来的信息缺失．

综上，本文后续的建模与实验将采用副本数（replica_count）与CPU利用率（cpu_avg）的联合输入，其中副本数历史序列作为核心自回归特征，捕捉伸缩行为的长期演化轨迹．CPU利用率则作为外部上下文特征，补充反映扩缩容触发前后的资源压力差异．通过这种多源输入设计，模型既能保持对伸缩行为本身的敏感性，又避免了单一变量可能带来的信息不足，从而更好地支撑弹性伸缩场景下的行为建模与可部署预测．



（a）原始数据回归分析



（b）一阶差分回归分析

图9 MS_28578数据相关性热力图

4.3 组合预测模型

为提升伸缩行为学习的精度和泛化能力，本文提出一种融合ARIMA长期趋势建模、双向LSTM短时残差回归、多头注意力机制与残差修正的组合预测模型．位于图6系统架构中的组合预测模型的基本原理如图10所示．

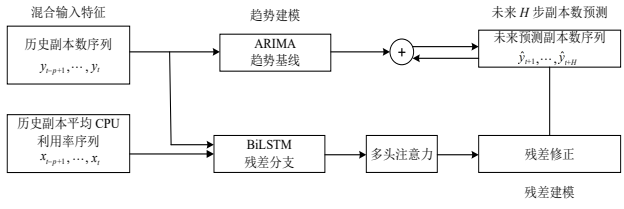


图10 组合预测模型模块原理示意图

组合预测模型结构由两条并行分支组成：主干ARIMA分支用于捕捉副本数序列的平稳趋势与周期性成分，残差修正分支则采用BiLSTM网络结合多头注意

力模块,针对ARIMA难以拟合的高频扰动和短时波动进行细粒度建模,最终通过加法融合实现副本数的多步前瞻预测.

在模型输入端,首先将历史窗口长度为p的副本数序列$\{y_{t-p+1},y_{t-p+2},\cdots,y_t\}$送入ARIMA模块,获得未来$h$步的主趋势预测$\{\hat{y}_{t+1}^{\mathrm{ARIMA}},\hat{y}_{t+2}^{\mathrm{ARIMA}},\cdots,\hat{y}_{t+h}^{\mathrm{ARIMA}}\}$. 同时,以副本数与副本平均CPU利用率的联合历史序列作为残差分支的输入,计算残差如式(28)所示:

$$\Delta y=y-\hat{y}^{\mathrm{ARIMA}} \tag{28}$$

并以此为目标,将序列特征编码为三维张量后送入多层双向LSTM网络. BiLSTM输出经多头自注意力机制增强,显式建模局部高频扰动与短期动态. 残差分支的输出$\{\Delta\hat{y}_{t+1},\Delta\hat{y}_{t+2},\cdots,\Delta\hat{y}_{t+h}\}$与主干ARIMA输出相加,得到最终预测值如式(29)所示:

$$\hat{y}_{t+h}=\hat{y}_{t+h}^{\mathrm{ARIMA}}+\Delta\hat{y}_{t+h} \tag{29}$$

其中,$\hat{y}_{t+h}^{\mathrm{ARIMA}}$为ARIMA分支输出,$\Delta\hat{y}_{t+h}$为BiLSTM-MHA分支的残差校正项.

本模型的训练过程如算法2所示. 首先基于所有训练样本按维度独立拟合ARIMA模型,并在主干预测基础上以MSE损失优化BiLSTM残差分支. 推理阶段,ARIMA输出作为基线,BiLSTM分支动态补偿,确保对副本数短时突变、非线性扰动具有良好响应. 多头注意力机制进一步提升了残差建模的表达力,残差修正路径为整个系统提供结构性外部监督和更强的泛化能力.

算法2　ARIMA-BiLSTM-MHA训练算法

输入:历史副本平均CPU利用率序列$x=x_{t-p+1},x_{t-p+2},\cdots,x_t$;历史副本数序列$y=y_{t-p+1},y_{t-p+2},\cdots,y_t$

输出:未来H步副本数预测$y_{t+1},y_{t+2},\cdots,y_{t+H}$

1. 拟合ARIMA基线模型ARIMA,得到趋势预测$\hat{y}_{t+1}^{\mathrm{ARIMA}},\hat{y}_{t+2}^{\mathrm{ARIMA}},\cdots,\hat{y}_{t+H}^{\mathrm{ARIMA}}$;
2. 计算训练集残差$r=y-f_{\mathrm{ARIMA}}(y)$;
3. 输入历史序列x,y到双向LSTM,提取时序特征h;
4. 对h施加Multi-Head Attention,生成上下文增强表示c;
5. 通过残差预测头,输出未来H步残差$\Delta\hat{y}_{t+1},\Delta\hat{y}_{t+2},\cdots,\Delta\hat{y}_{t+H}$;
6. 计算最终预测$\hat{y}_{t+h}=\hat{y}_{t+1}^{\mathrm{ARIMA}}+\Delta\hat{y}_{t+h},h=1,2,\cdots,H$;
7. 返回$\hat{y}_{t+1},\hat{y}_{t+2},\cdots,\hat{y}_{t+H}$.

为进一步提升对副本动态时序依赖的建模能力,残差分支采用双向LSTM结构. 相较于单向LSTM,双向LSTM网络通过正反两个方向的循环单元,能够同时捕捉历史与未来窗口内的序列特征,从而对副本数的趋势变化、周期极值和突发事件具有更高的敏感性和表达能力. 因此,BiLSTM相比于单向LSTM在伸缩行为学习场景下更具结构优势.

4.4　评估与持久化

为保障所提方法的可复现实验性与工程落地性,本文在Python 3.10.13环境下基于PyTorch与pmdarima实现了ARIMA-BiLSTM-MHA组合预测模型,并在此基础上构建了评估与持久化模块. 该模块主要完成两类功能:一是对多步预测结果进行定量评估和可视化分析;二是对模型参数及评估产物进行统一持久化管理,为后续复现实验和在线部署提供支撑.

在评估子模块中,系统首先将组合预测模型模块输出的多步副本数预测序列与测试集中的真实副本数序列对齐,按时间步计算误差. 针对整个测试集,分别统计MSE、RMSE、MAE、MAPE以及R^2等指标,用于综合刻画模型在不同误差度量下的拟合程度与泛化能力. 与此同时,评估子模块还自动生成关键时间区间的"预测-真实"对比曲线、误差分布曲线以及随预测步长变化的误差曲线,为第5节的实验结果分析提供可视化依据.

在持久化子模块中,系统对重要中间产物和最终结果进行结构化保存,包括:清洗与切分后的训练、验证和测试数据集元信息(如窗口长度、预测步长、标准化方式等);经多轮调优后确定的ARIMA参数和BiLSTM-多头注意力网络权重;各评估指标的数值结果以及对应的绘图文件.

评估与持久化模块位于整个系统流水线的末端,一方面接收组合预测模型模块输出的多步副本数预测序列,另一方面将评估结果与持久化信息向外输出为"伸缩行为预测结果(未来副本数序列及评估指标)". 同时,模块通过误差统计和指标比较,将"误差反馈与模型调优"信息回传至组合预测模型模块,用于指导后续的超参数调整和结构改进. 通过这种"前向预测+反向反馈+统一持久化"的设计,第4节所提出的三个功能模块在逻辑上构成了一条完整的、可部署的伸缩行为学习与预测链路:数据采集与分析模块负责构建高质量时序样本,组合预测模型模块完成趋势-残差联合建模与多步预测,评估与持久化模块则实现性能度量、结果归档与模型迭代闭环.

5　实验评估与讨论

5.1　实验环境

本文涉及的所有模型开发和评估实验工作均在如表4所示的软硬件平台上完成.

就数据集与训练任务配置而言,如表5所示,本文所使用的数据集来源于cluster-trace-microservices-v2022中MS_Metrics文件集中的部分文件(涵盖前335 h的运行数据),根据4.1节的分析结果,选择目标微服务为MS_28578,依据timestamp/msname聚合每个采样点的副本平均CPU资源利用率cpu_avg以及副本数量replica_count,并采用StandardScaler对全部特征做均值方差归一化. 样本按7∶1.5∶1.5比例分为训练集、验证集与测试集,所有基线与对比方法均采用一致的训练任

表4 硬件与软件环境

硬件环境	
CPU	Intel(R) Core(TM) Ultra 7 265KF (3.90 GHz)
内存	64 GB (32 GB DDR5-6000 SDRAM*2)
GPU	NVIDIA GeForce RTX 3060
外部存储	Acer Predator GM7 4 TB NVMe SSD (PCIe 4.0)
软件环境	
操作系统	Windows 11 Enterprise 24H2 Windows Subsystem for Linux 2.5.10
编程语言	Python 3.10.13
机器学习环境	PyTorch 2.5.1 (with CUDA 12 support)
容器环境	Docker Engine 26.1
主要依赖	pmdarima, numpy, pandas, matplotlib, seaborn

务、数据处理与标准化策略.

表6给出了本文所提出的ARIMA-BiLSTM-MHA组合预测模型的具体超参数配置.

5.2 评估标准

为客观衡量伸缩行为学习模型的误差规模、相对偏差与拟合优度,本文在测试集上采用五类回归指标:MSE、RMSE、MAE、MAPE与R^2. 除R^2越大越好外,其余指标值越小表示性能越优. 若不另作说明,本文默认以预测窗口内逐点误差展开后统一统计.

MSE为残差平方的算术平均,能够度量预测残差的二阶矩,能突出较大的偏差,因此对异常点更敏感,常用于训练与模型选择,其计算如式(30)所示:

$$\mathrm{MSE}=\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2 \tag{30}$$

RMSE为MSE的平方根,与原量纲一致,更便于直观理解误差量级,其计算如式(31)所示:

$$\mathrm{RMSE}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2} \tag{31}$$

MAE为相对误差绝对值的平均百分比,能够度量平均偏差的绝对值,相比RMSE对极端误差更稳健(L1度量),其计算如式(32)所示:

$$\mathrm{MAE}=\frac{1}{n}\sum_{i=1}^{n}\left|y_i-\hat{y}_i\right| \tag{32}$$

MAPE为总离差中被模型解释的比例,提供了与量纲无关的相对误差,便于跨场景比较. 为避免分母为0带来的不稳定,若数据出现$y_i=0$,则采用平滑$\max(|y_i|,\varepsilon)$($\varepsilon=10^{-6}$),其计算如式(33)所示:

$$\mathrm{MAPE}(\%)=\frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i-\hat{y}_i}{\max\left(\left|y_i\right|,\varepsilon\right)}\right| \tag{33}$$

R^2衡量模型解释真实值方差的比例. $R^2=1$表示完美拟合;$R^2=0$等价于总是预测均值;$R^2<0$表示劣于均值基线,其计算如式(34)和式(35)所示:

$$R^2=1-\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2} \tag{34}$$

$$\bar{y}=\sum_{i=1}^{n}y_i \tag{35}$$

其中,对于本节所述的6个公式,y_i与$\hat{y}$分别为第i个样本(或时间点)的容器副本数的真实值与预测值,n为统计样本总数,$\bar{y}$为真实值均值,ε为数值稳定性常数.

表5 数据集与训练任务配置

数据集	
名称	cluster-trace-microservices-v2022
本文所使用的文件名集合	MSMetrics_0.csv ~ MSMetrics_669.csv
列名集合	timestamp, msname, msinstanceid, nodeid, cpu_utilization, memory_utilization
总时长	335 h
采样周期	60 000 ms
微服务总数	28 611
数据清洗	
目标微服务	MS_28578
列名集合	timestamp, msname, cpu_avg, mem_avg, replica_count
标准化	StandardScaler,均值方差归一化
数据切分	训练集0.70/验证集0.15/ 测试集0.15
训练任务	
输入数据列	cpu_avg, replica_count
输出数据列	replica_count
输入步长	10
输出步长	5

表6　ARIMA-BiLSTM-MHA组合预测模型配置

ARIMA配置	
p	1
d	1
q	1
趋势项	n
搜索方式	步进式AIC/BIC搜索
阶数选择标准	aicc
最大拟合迭代次数	200
小窗口最大迭代次数	40
BiLSTM配置	
隐单元数量	32
层数	2
Dropout	0.2
Attention层配置	
类型	nn.MHA
头数	4
通用超参数	
批大小	16
最大训练轮数	3 000
早停容忍最大轮数	300
优化器	adam
初始学习率	0.001
损失函数	MSE

5.3　对比实验

为全面评估本文所提出的ARIMA-BiLSTM-MHA组合模型的性能,本节选取了11类主流基线方法,包括Transformer(仅使用编码器)、GRU、CNN-LSTM、GRU-LSTM、CNN-LSTM-GRU、Seq2Seq-LSTM、Seq2Seq-GRU、Seq2Seq-Transformer、PETformer、SparseTSF与TFEGRU.各模型均在同一测试集上评估,相关指标结果见表7,均按MSE从小到大排序.

在各项指标上,本文提出的ARIMA-BiLSTM-MHA模型在所有回归指标上均展现出显著的综合优势.在MSE指标上,模型在全部对比对象中实现了平均相对降低39.85%,具体相对降低幅度介于1.39%(相对于PETformer)至71.56%(相对于Seq2Seq-Transformer)之间;RMSE平均相对降低24.27%,相对降低幅度介于0.72%(相对于PETformer)至46.67%(相对于Seq2Seq-Transformer);MAE平均相对降低31.03%,相对降低幅度介于1.57%(相对于PETformer)到59.10%(相对于Seq2Seq-GRU);MAPE平均相对降低30.64%,相对降低幅度介于1.97%(相对于PETformer)至59.10%(相对于Seq2Seq-GRU)之间;在R^2方面,平均相对提升5.77%,相对提升幅度从0.27%(相对于PETformer)到15.70%(相对于Seq2Seq-Transformer).

图11展示了本文提出的ARIMA-BiLSTM-MHA模型在测试集上的副本数量真实值与预测值对比曲线.可以看出,所提方法能够精确拟合伸缩行为直接结果即副本数的多种动态变化特征,无论是突发扩容、极值回落还是平台期,都实现了较高的趋势跟踪和极值捕捉能力.特别是在拐点和剧烈变化区间,模型的响应滞后和幅度误差均显著低于其他主流方法,高频扰动阶段的预测曲线也更加平滑,未出现误差积累或异常放大现象,进一步印证了多头注意力机制对时序噪声与局部极值的有效抑制作用.

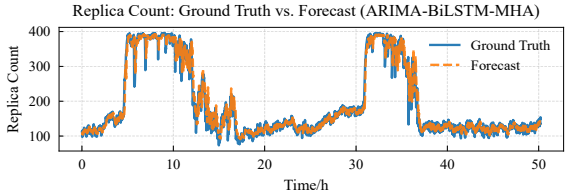


图11　ARIMA-BiLSTM-MHA模型在测试集上的伸缩行为学习拟合曲线

表7　各模型在MS_28578测试集上的伸缩行为学习性能指标对比

模型	MSE	RMSE	MAE	MAPE/%	R^2
ARIMA-BiLSTM-MHA	489.41	22.12	13.75	7.96	0.954 3
PETformer	496.33	22.28	13.97	8.12	0.951 7
SparseTSF	521.86	22.84	14.35	8.43	0.948 8
TFEGRU	540.94	23.26	14.64	8.29	0.949 5
BiGRU-MHA	642.69	25.35	16.39	8.95	0.940 0
CNN-BiLSTM-MHA	746.79	27.33	19.64	11.15	0.930 3
GRU-BiLSTM-MHA	799.31	28.27	19.66	11.46	0.925 4
Seq2Seq-BiLSTM-MHA	1 043.95	32.31	25.52	15.16	0.902 5
CNN-BiLSTM-GRU-MHA	1 339.61	36.60	29.88	18.44	0.874 9
Transformer	1 483.76	38.52	30.54	17.06	0.848 9
Seq2Seq-BiGRU-MHA	1 576.93	39.71	33.62	20.14	0.852 8
Seq2Seq-Transformer	1 720.81	41.48	22.58	12.67	0.824 8

图12给出了训练损失随轮次的收敛过程. 实验结果表明,模型在初始几轮内损失快速下降,15轮内总体下降幅度超过59%,随后训练损失趋于平稳,并在2 939轮时达到训练全局最低值0.026 71. 这充分说明了所提组合模型具备良好的优化收敛性和泛化能力,并且在实际部署中能够提供稳定、可靠的伸缩行为预测能力.

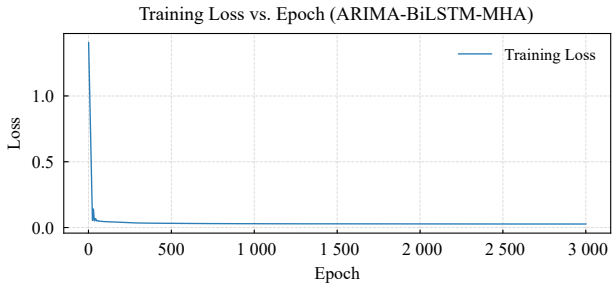


图12 训练损失随训练轮次曲线

综上所述,本文提出的ARIMA-BiLSTM-MHA方案通过趋势建模与残差学习的解耦、双向时序上下文建模以及多头注意力对关键区间的显式聚焦,有效提升了多步伸缩行为学习的预测精度与稳定性. 在误差控制与拟合优度方面均实现了对现有主流模型的全面超越,为微服务弹性伸缩等实际场景提供了高精度、可落地的预测工具.

5.4 消融实验

为了评估各个组件对伸缩行为学习组合模型性能的具体贡献,本节设计了消融实验,比较了ARIMA-BiLSTM-MHA模型与去除不同模块后的版本. 通过对比以下四个模型的性能,分析各个模块在容器伸缩行为学习中的作用.

(1)ARIMA-BiLSTM-MHA:该模型包括ARIMA用于趋势建模、BiLSTM用于残差学习和多头注意力机制,用以增强模型的特征捕捉能力.

(2)ARIMA-BiLSTM-MHA-NoResidualCorrection:保留ARIMA与BiLSTM-MHA的网络结构,但不再采用"趋势分解-残差修正"的二阶段方式,即不再对ARIMA预测残差进行学习与回加,而是直接输出深度子网络的预测结果,以评估残差修正机制的必要性.

(3)ARIMA-LSTM-MHA:去除BiLSTM,保留ARIMA、LSTM以及多头注意力机制,以检验BiLSTM在伸缩行为学习中的作用.

(4)ARIMA-BiLSTM:去除多头注意力机制,保留ARIMA与BiLSTM部分,以考察多头注意力机制对伸缩行为捕捉精度的贡献.

(5)ARIMA:仅使用ARIMA对副本数序列进行统计建模,不包含任何深度网络成分,以进一步检验BiLSTM与多头注意力机制在伸缩行为学习中的作用.

(6)BiLSTM-MHA:仅保留BiLSTM-MHA,对原始副本数序列进行端到端预测,以检验ARIMA趋势分解在伸缩行为学习中的作用.

(7)ARIMA-LSTM:去除BiLSTM和多头注意力机制,仅保留ARIMA与LSTM部分,作为原始基线,以综合评估BiLSTM和多头注意力机制在伸缩行为学习中的重要性.

如表8所示,本文所提出的ARIMA-BiLSTM-MHA在各项指标中均取得了最优性能,其MSE为489. 41,RMSE为22.12,MAE为13.75,MAPE为7.96%,R^2为0.954 3. 模型的性能均优于其他变体,表明ARIMA-BiLSTM-MHA能更好地捕捉 伸缩行为中的复杂时序模式.

表8 不同ARIMA-LSTM模型在MS_28578测试集上的预测性能指标对比

模型	MSE	RMSE	MAE	MAPE/%	R^2
ARIMA-BiLSTM-MHA	489.41	22.12	13.75	7.96	0.954 3
ARIMA-LSTM- MHA	492.77	22.20	13.96	8.04	0.953 6
ARIMA-BiLSTM	498.75	22.33	13.83	8.00	0.953 4
ARIMA-LSTM	507.17	22.52	13.98	8.05	0.952 6
ARIMA-BiLSTM-MHA-NoRedisualCorrection	538.26	23.20	14.32	8.13	0.949 7
ARIMA	1 039.56	32.24	20.17	11.58	0.902 9
BiLSTM-MHA	1 106.09	33.26	21.14	11.75	0.896 8

相比于本文模型,去除残差修正机制后的模型(ARIMA-BiLSTM-MHA-NoResidualCorrection)预测性能出现了明显恶化. 具体而言,其MSE由489.41升高到538.26,RMSE由22.12升高到23.20,MAE从13.75升高到14.32,MAPE也由7.96%增加到8.13%,R^2则从0.954 3降至0.949 7. 两者的网络容量基本一致,仅在"是否基于ARIMA预测结果进行显式残差修正"这一点上存在差异,但性能却出现了可观差距,这表明趋势分解与深度残差学习之间的协同是必要的:在缺乏明确残差校正的情况下,深度子网络被迫同时拟合长期趋势与局部细节,更容易在强趋势、强非平稳序列上陷入次优解,难以同时兼顾整体走势和局部结构的精确刻画.

相比于本文模型,去除BiLSTM后的模型(ARIMA-LSTM-MHA)性能有所下降. 具体而言,MSE增加至

492.77, RMSE 增加至 22.20, MAE 为 13.96, MAPE 为 8.04%, R^2为 0.953 6. 这表明,BiLSTM 在学习伸缩行为时序依赖和短期波动中具有关键作用. BiLSTM 通过正向和反向两个方向同时编码序列,对“某一时刻的副本决策如何依赖于过去一段时间的负载轨迹以及紧随其后的调整结果”进行联合建模,能够显式利用左右两侧的上下文信息. 去除 BiLSTM 后,模型只能依靠单向记忆累积来近似这些模式,对复杂负载形态和多步预测窗口的拟合能力相应下降,最终体现在多项误差指标上的小幅但稳定劣化.

相比于本文模型,去除多头注意力机制后的模型(ARIMA-BiLSTM)在 MSE 上升至 498.75, RMSE、MAE 和 MAPE 也均有所增加,R^2降至 0.953 4. 这表明多头注意力机制对提升模型在伸缩行为学习中的精度和健壮性具有重要作用,尤其是在捕捉关键时序点和异常波动方面. 在伸缩行为中,扩缩容拐点、突发负载区间以及长平台前后的边界点,对后续多步副本预测的影响显著高于其余平稳时段. BiLSTM 在时间上对各个位置的“重要性”权重相对均衡,难以及时放大这些关键位置的贡献. 多头注意力通过在多个子空间中并行学习“短期尖峰”“中期趋势转折”“周期性波动”等不同模式,并对关键时间步赋予更高权重,使得模型能够在强噪声、强扰动的残差序列中突出真正决定伸缩行为的关键信号,抑制偶发波动对预测轨迹的干扰. 因此,当去掉注意力层之后,模型在拐点和异常波动区间的拟合能力下降,高频噪声更容易折射到多步预测结果中,引发误差指标的一致变差.

ARIMA-LSTM 作为去除 BiLSTM 和多头注意力机制后的原始基线,其 MSE 达到 507.17, RMSE 为 22.52, MAE 为 13.98, MAPE 为 8.05%, R^2降至 0.952 6. 这进一步验证了 BiLSTM 和多头注意力机制对伸缩行为学习中时序捕捉能力和预测精度的提升作用.

最后,将完整模型与 ARIMA 和 BiLSTM-MHA 这两个“单分支极端模型”对比,可以更直接地观察趋势分解与残差学习的互补性. 单独使用 ARIMA 时, MSE 提高到 1 039.56, R^2下降到 0.902 9;单独使用 BiLSTM-MHA 时, MSE 为 1 106.09, R^2进一步下降到 0.896 8. 与完整模型相比,这两种极端情形的误差大幅增加、拟合优度明显降低:仅依赖趋势分解会在拐点和高频扰动区间出现较大偏差,而仅依赖深度网络端到端建模则需要同时拟合长期趋势和局部噪声,训练难度显著增大,也更容易在负载模式发生变化时出现不稳定.

整体而言,相较于其他消融变体模型,ARIMA-BiLSTM-MHA 在各项指标上均取得了稳定且幅度可观的改进. 具体而言,ARIMA-BiLSTM-MHA 在 MSE 指标上的相对降低百分比平均值为 20.63%,且相对降低幅度的范围为 0.68%(与 ARIMA-LSTM-MHA 相比)到 55.75%(与 BiLSTM-MHA 相比). 在 RMSE 指标上,ARIMA-BiLSTM-MHA 相较于其他模型,平均降低了 12.10%,最差降低幅度为 0.34%(与 ARIMA-LSTM-MHA 相比),最优降低幅度为 33.49%(与 BiLSTM-MHA 相比). 在 MAE 指标上,ARIMA-BiLSTM-MHA 相较于其他模型,平均降低为 12.44%,最差降低幅度为 0.63%(与 ARIMA-BiLSTM 相比),最优降低幅度为 34.97%(与 BiLSTM-MHA 相比). 在 MAPE 指标上,ARIMA-BiLSTM-MHA 相较于其他模型,平均降低了 11.34%,最差降低幅度为 0.47%(与 ARIMA-BiLSTM 相比),最优降低幅度为 32.25%(与 BiLSTM-MHA 相比). 在 R^2指标上,ARIMA-BiLSTM-MHA 相较于其他模型,平均提升了 2.15%,最差提升幅度为 0.03%(与 ARIMA-LSTM-MHA 相比),最优提升幅度为 6.41%(与 BiLSTM-MHA 相比).

结合以上实验分析,BiLSTM 和多头注意力机制在本任务中相辅相成,前者增强了对副本数序列时序依赖的建模能力,后者通过赋予模型对关键时序点的自动聚焦能力,进一步提升了伸缩行为学习的稳定性和精度. 去除任何一个模块后,模型的伸缩行为预测能力和健壮性都会下降,尤其在处理高频扰动和异常波动时.

综上所述,ARIMA-BiLSTM-MHA 模型相较于其他消融变体,提升了对伸缩行为的预测能力和健壮性,证明了每个模块在这一任务中的不可或缺的作用. BiLSTM 对时序捕捉的增强能力以及多头注意力机制对关键时序点的自动聚焦能力,确保了模型不仅能精确伸缩行为结果(即预测副本数),同时还能有效地模拟和理解伸缩行为的动态过程.

5.5　容器伸缩控制实验

为了进一步检验所提出 ARIMA-BiLSTM-MHA 方案在实际伸缩决策环节中的控制效果,本节选取了开源微服务基准测试套件 DeathStarBench[36]中的一个典型社交网络微服务应用 socialNetwork[37],构建容器伸缩控制实验,对比分析不同伸缩控制策略在同一阶梯负载下的动态伸缩行为、资源利用率以及服务质量.

在应用层,本文选取其中的 ReadHomeTimeline 请求作为研究对象. 该请求用于获取用户首页时间线,需要聚合多条社交关系和帖子元数据,是系统中较为典型且对尾延迟敏感的读请求. 本文选取该请求微服务调用链上的 home-timeline-service 作为容器伸缩目标微服务,通过调整其容器副本数量来吸收请求负载变化,最大可用副本数设置为 20. 实验总时长为 600 s,采用由 wrk2 工具生成的动态变化负载,负载在实验过程中多次升高和回落,以模拟实际业务场景中常见的波动过程. 整个实验期间,系统按固定时间间隔记录 home-timeline-service 的副本数、单副本平均 CPU 利用率以及

ReadHomeTimeline 请求的 P99 延迟.

实验对比了如下两种容器伸缩控制策略:

(1)行为学习模型驱动的策略. 基于本文提出的建模方法,利用离线采集的 home-timeline-service 微服务的历史负载数据进行模型训练,构建 ARIMA-BiLSTM-MHA 模型作为伸缩控制器. 控制器的采样间隔为 10 s,即每 10 s 收集一次过去 10 个时间点的副本平均 CPU 利用率与副本数,将其输入到组合模型中预测未来 5 个时间步的副本数序列,并取第一个预测值作为当前的伸缩决策. 由于模型推断需要积累完整的历史窗口特征,前 100 s 为冷启动阶段,此时副本数固定为 10.

(2)容器副本平均 CPU 资源利用率驱动的 HPA 策略. 该策略旨在模拟资源驱动型的副本伸缩策略,同样以 10 s 为采样周期,当观测到副本平均 CPU 利用率持续低于下限时,HPA 逐步缩减副本;当高于上限时,按照比例扩容,单次调整幅度受到限制. 副本数的初始值同样为 10.

图 13 给出了两种策略在实验期间的副本数量随时间变化曲线. 可以看到,在冷启动结束之后,行为学习策略给出的副本数随负载变化呈现出较为规则的"阶梯"形走势:在负载上升阶段能够及时增加副本,负载回落阶段也能够较快收缩,没有出现明显的扩缩容滞后或长时间"拖尾",整体调整轨迹平滑而稳健. 这种抖动较小、变化合理的特性与组合模型的结构设计是一致的. 一方面,ARIMA 部分显式刻画了伸缩行为的长期趋势,将平稳部分和周期性变化从原始序列中剥离,使后续神经网络主要关注残差中的短期扰动与高频变化;另一方面,BiLSTM 通过双向时序建模同时利用历史上下文,能够捕捉副本数在"即将扩容"或"即将缩容"前的细微迹象,而不只是依赖当前一个时间点的 CPU 利用率. 多头注意力机制在此基础上对关键转折位置进行加权聚焦,将对尾延迟敏感的区间赋予更高权重. 三者结合,使得行为学习控制器在给定最近一段时间的 CPU 利用率与副本数后,可以直接预测未来多个时间步的副本需求,其输出本质上是在逼近"满足当前性能需求所需的合理副本水平",从而在图 14 中表现为平滑的阶梯形曲线.

与之形成对比的是图 14 中 CPU 阈值 HPA 策略的副本轨迹. 整体来看,HPA 曲线在多个阶段明显高于行为学习策略,并在较长时间内接近最大副本上限,这说明该策略在稳态阶段往往倾向于"多开副本"以换取较低 CPU 利用率. 在负载上升阶段,HPA 曲线通常先保持上一阶段的副本数,在经历一个或多个采样周期之后才开始扩容,呈现出典型的控制滞后;负载下降阶段则相反:行为学习策略可以较快缩容,而 HPA 由于需要 CPU 利用率长期低于下阈值才触发缩容,副本数下降缓慢,甚至在负载已经明显回落时仍然处于"过度配置"状态.

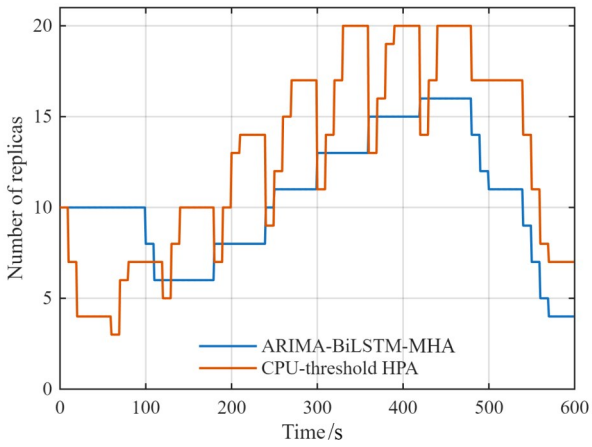


图 13 两种策略的副本数量对比曲线

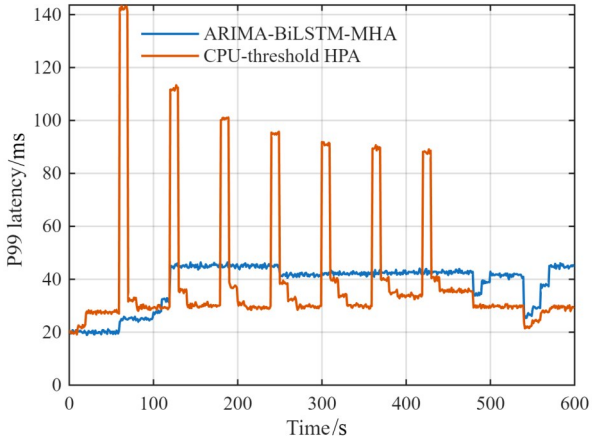


图 14 两种策略的 P99 尾部响应延迟对比曲线

造成这一现象的根本原因在于,CPU 阈值 HPA 采用的是"单点反馈+区间阈值+离散步长"的控制逻辑. 控制器每隔固定采样周期仅观察当前时刻的平均 CPU 利用率,并将其与预设的上下阈值区间进行比较:若超过上限则按比例增加一定数量的副本,若低于下限则减少一定数量的副本;在 CPU 仍位于区间内部时,则保持不变. 首先,由于采样周期有限,当负载在采样点之间发生突增时,HPA 无法立即感知,必然在至少一个采样周期之后才做出扩容决策,这在图 14 中体现为负载突变后的一段"水平延迟段". 其次,副本数量的调整是以固定步长进行的,而非直接跳转到更合理的副本水平;在 CPU 反馈噪声和负载抖动的共同作用下,HPA 往往在"略少副本"和"略多副本"之间来回摆动,既难以精确落在合适的阶梯上,又为了避免频繁伸缩不得不选择偏保守的副本数. 最后,CPU 利用率本身只是尾延迟的一个间接代理,不同负载形态下 CPU 指标与 P99 延迟之间的映射关系并不完全一致,使得基于固定目

标区间的HPA很难在各个阶段都找到真正“恰当”的副本配置,这也进一步加剧了其滞后性和不精确性.

为了评估上述不同伸缩策略对服务性能的影响,图14展示了两种策略下ReadHomeTimeline请求P99延迟随时间的变化曲线. 可以观察到,在行为学习策略下,请求的P99延迟在冷启动结束后大多落在38~42 ms的区间内,波动较小且形态平滑;相比之下,HPA在多数稳态时间段的P99延迟也保持在40 ms左右,但在每次负载突升后的若干秒内都会出现非常突出的尾延迟峰值,峰值高度介于90~140 ms之间. 这是因为在负载突升瞬间,HPA的副本数仍然停留在突升前阶段的配置水平,需要经过若干采样周期才开始扩容,导致短时间内出现副本短缺,从而在P99延迟上表现为明显的性能尖峰. 这说明在负载快速变化场景下,基于CPU阈值的策略由于本身的调整滞后性,难以及时弥补短期欠配,容易产生显著的尾延迟抖动;而行为学习策略通过对趋势和残差的联合建模,能够在负载变化前后更快速地给出合理的扩缩容动作,使整体尾延迟曲线更加平滑.

图15给出了两种策略在整个实验期间的单副本平均CPU利用率随时间变化曲线. 可以看到,在行为学习策略下,单副本平均CPU利用率在冷启动阶段略低,在进入稳定阶段后大部分时间处于70%~85%之间,资源利用率较高且没有出现长期过载;HPA则由于在多个阶段过度扩容,在绝大部分时间里都处于70%以下,仅在负载骤增而尚未完成扩容的瞬间出现接近90%的高利用率峰值. 这一现象表明,HPA为了降低稳定阶段的延迟,更多依赖增加副本来“摊薄”负载,导致总体资源利用率偏低,而行为学习策略在保证性能的前提下更接近于“高利用率但不过载”的状态.

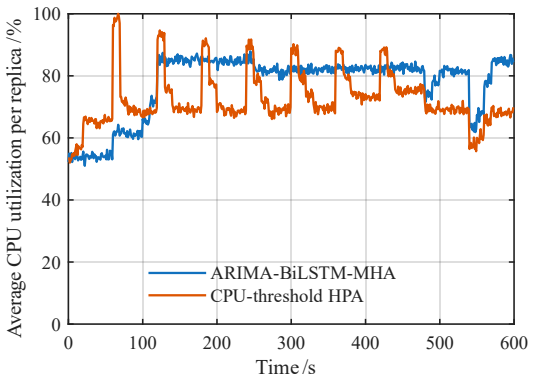


图15　两种策略的副本平均CPU利用率对比曲线

表9汇总了两种伸缩控制策略在成本、资源利用率以及性能方面的综合结果. 可以看到,从副本数量来看,行为学习策略的平均副本数为10.50,而HPA的平均副本数达到12.65,行为学习策略相对减少约17%的副本数量;从单副本CPU利用率来看,行为学习策略的平均值为77.35%,HPA为72.26%,行为学习策略比HPA高约5.1个百分点,进一步印证了其在资源成本上的优势;从P99延迟来看,两种策略的平均P99延迟分别约为38.42 ms和39.25 ms,行为学习策略相对减少约2.11%,且HPA在多个负载跳变阶段产生明显的P99尖峰,而行为学习策略整体延迟更为平滑.

表9　两种伸缩控制策略的实验结果

伸缩策略	平均副本数	平均单副本CPU利用率/%	平均P99延迟/ms
行为学习	10.50	77.35	38.42
HPA	12.65	72.26	39.25

综合副本数量、CPU利用率和P99延迟三个维度的结果可以看出,在相同应用和硬件条件下,基于ARIMA-BiLSTM-MHA的行为学习伸缩控制器能够在动态负载场景中生成更为合理和平滑的伸缩轨迹,在保持不高于HPA的尾延迟水平的同时,显著降低了相对于HPA的副本冗余,并提升了整体CPU资源利用率. 结合前文在真实追踪数据上的预测指标和消融分析,本节实验进一步表明,将容器副本伸缩行为作为直接学习目标,可以在云平台弹性伸缩场景中同时兼顾预测精度、控制效果和资源配置效率.

6　总结与展望

本文围绕云平台弹性伸缩中的伸缩行为学习问题,提出了一种融合多头注意力与残差修正机制的ARIMA-BiLSTM组合预测模型. 针对现有统计与深度学习方法在复杂动态场景下的建模瓶颈,所提方法充分发挥了ARIMA的长期趋势分解能力、BiLSTM的时序特征捕捉优势,以及多头注意力机制对关键时序片段的自适应建模能力. 实验部分基于Alibaba公开的微服务数据集cluster-trace-microservices-v2022,并与多种主流模型(如PETformer、SparseTSF、TFEGRU、GRU、Transformer、Seq2Seq系列、GRU-LSTM、CNN-LSTM、CNN-LSTM-GRU等)进行了系统对比. 实验结果表明,本文提出的模型在MSE、RMSE、MAE、MAPE等核心评估指标上,分别平均相对降低39.85%、24.27%、31.03%、39.02%以及5.77%的相对R^2提升,其中MSE相对Seq2Seq-Transformer的最大提升幅度为71.56%. 此外,模型的预测曲线与训练损失收敛过程也表现出优秀的拟合度、良好的收敛性和稳健的泛化能力,进一步验证了其在伸缩行为学习任务中的有效性.

在伸缩控制层面,本文进一步将所提模型嵌入开源微服务基准套件DeathStarBench的socialNetwork应用,构建容器伸缩控制实验,对比“行为学习模型驱动策略”和“CPU阈值型HPA”两种控制方案. 实验结果表

明,在同一动态负载条件下,两种策略均能将ReadHomeTimeline请求的P99延迟控制在较低水平,但行为学习策略的延迟曲线更加平滑,显著减弱了负载突升阶段的尾延迟尖峰;同时,在平均P99延迟相对降低2.11%的前提下,其平均副本数相较HPA约减少17%,单副本平均CPU利用率提升约5.09个百分点,体现出更高的资源利用效率. 该闭环实验表明,将容器伸缩行为作为直接学习目标的组合模型,不仅在离线预测精度上优于多种基线方法,也能够在实际控制场景中生成更加平稳、经济的伸缩轨迹,在服务性能与资源成本之间取得更优折中.

尽管如此,本文的研究仍然存在一些局限性. 首先,当前模型主要依赖副本数的历史轨迹和单一的资源利用率特征(如副本的平均CPU利用率),未能充分考虑微服务间的上下游依赖关系、RPC链路流量、HTTP请求模式等多源时序特征,这些因素对复杂伸缩行为的影响尚未得到全面挖掘. 未来的研究将致力于扩展特征空间,融合多维度时序特征,以更全面地刻画微服务的弹性行为,特别是业务级联效应对副本数动态变化的影响. 其次,本文的训练与评估主要基于Alibaba公开数据集,该数据集是当前少数同时提供微服务实例级信息与资源监控指标、适合伸缩行为学习的生产级trace. 由于其他云服务厂商的微服务trace要么尚未公开,要么缺乏副本级"资源-伸缩动作"联合时序记录,本文暂未在多云环境下系统评估模型在不同调度策略与平台配置下的跨平台泛化性. 随着未来更多厂商逐步开放细粒度的微服务运行数据,后续工作将优先将本文方法迁移到更多类型的trace上,分析在"跨云平台、跨调度策略"场景下的迁移性能,并探索结合少量目标域样本进行轻量级微调或领域自适应的方案. 再次,尽管离线实验与在线控制实验验证了方法的优越性,但在真实云平台和开源弹性伸缩系统中的工程落地尚需进一步推进. 未来的研究将集中于推动模型与实际弹性决策系统的深度集成,特别是在典型业务波动和节假日等突发场景下,评估模型在资源利用与成本优化方面的实际效果. 进一步地,模型的推理时延、自适应能力和部署友好性仍是未来优化的重点,以确保模型能够在复杂微服务生态中实现高效的实时推理与可扩展应用.

综上所述,本文所提出的ARIMA-BiLSTM-MHA组合预测模型为云平台提供了一种高精度、可落地的伸缩行为学习解决方案,并展示了其在弹性伸缩决策支持中的潜在应用价值. 未来的研究将进一步拓展模型的功能与应用场景,推动其在实际云平台中的广泛应用,为分布式微服务容器系统的弹性伸缩提供更智能、更高效、可持续的解决方案.

参考文献

[1] NGUYEN T T, YEOM Y J, KIM T, et al. Horizontal pod autoscaling in kubernetes for elastic container orchestration[J]. Sensors, 2020, 20(16): 1-18.

[2] VERGADIA P. Visualizing Google Cloud: 101 Illustrated References for Cloud Engineers and Architects[M]. Hoboken: John Wiley & Sons, Inc., 2022.

[3] CALHEIROS R N, MASOUMI E, RANJAN R, et al. Workload prediction using ARIMA model and its impact on cloud applications' QoS[J]. IEEE Transactions on Cloud Computing, 2015, 3(4): 449-458.

[4] BI J, YUAN H T, LI S, et al. ARIMA-based and multiapplication workload prediction with wavelet decomposition and savitzky-golay filter in clouds[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2024, 54(4): 2495-2506.

[5] 杨阳, 姜春茂, 李志聪. 三支决策视角下的云平台负载预测研究[J]. 小型微型计算机系统, 2020, 41(7): 1363-1370.
YANG Y, JIANG C M, LI Z C. Cloud platform load forecasting from the perspective of three-way decisions[J]. Journal of Chinese Computer Systems, 2020, 41(7): 1363-1370. (in Chinese)

[6] 谢同磊, 邓莉, 曹振, 等. 基于LSTM-Z的云平台主机负载预测方法[J]. 计算机工程与设计, 2023, 44(9): 2561-2568.
XIE T L, DENG L, CAO Z, et al. Host load prediction method based on LSTM-Z in cloud[J]. Computer Engineering and Design, 2023, 44(9): 2561-2568. (in Chinese)

[7] 尤文龙, 邓莉, 李锐龙, 等. 基于v-Informer的云平台资源负载预测方法[J]. 计算机科学, 2024, 51(12): 147-156.
YOU W L, DENG L, LI R L, et al. Load prediction method of cloud resource based on v-informer[J]. Computer Science, 2024, 51(12): 147-156. (in Chinese)

[8] 李浩阳, 贺小伟, 王宾, 等. 基于改进Informer的云计算资源负载预测[J]. 计算机工程, 2024, 50(2): 43-50.
LI H Y, HE X W, WANG B, et al. Cloud computing resource load prediction based on improved informer[J]. Computer Engineering, 2024, 50(2): 43-50. (in Chinese)

[9] 朱金灿, 邓莉, 梁晨君, 等. 云平台主机资源负载预测分析研究[J]. 小型微型计算机系统, 2021, 42(12): 2538-2544.
ZHU J C, DENG L, LIANG C J, et al. Analysis and prediction of host resource load in the cloud[J]. Journal of Chinese Computer Systems, 2021, 42(12): 2538-2544. (in Chinese)

[10] LIN S S, LIN W W, WU W T, et al. PETformer: Long-term time series forecasting via placeholder-enhanced transformer[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2025, 9(2): 1189-1201.

[11] LIN S S, LIN W W, WU W T, et al. SparseTSF: Lightweight and robust time series forecasting via sparse modeling[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2026, 48(1): 170-183.

[12] 史爱武, 罗良杰, 何凯. 基于EMD-TCN的云资源预测研究[J]. 计算机应用与软件, 2023, 40(7): 85-90.
SHI A W, LUO L J, HE K. Cloud resource prediction based on EMD-TCN[J]. Computer Applications and Software, 2023, 40(7): 85-90. (in Chinese)

[13] 杨哲兴, 谢晓兰, 李水旺. 基于VDM-ISSA-LSSVM的云资源短期负载预测模型[J]. 实验室研究与探索, 2023, 42(6): 117-124.
YANG Z X, XIE X L, LI S W. Short term load prediction model for cloud resources based on VMD-ISSA-LSSVM[J]. Research and Exploration in Laboratory, 2023, 42(6): 117-124. (in Chinese)

[14] 林涛, 冯竞凯, 郝章肖, 等. 基于组合预测模型的云计算资源负载预测研究[J]. 计算机工程与科学, 2020, 42(7): 1168-1173.
LIN T, FENG J K, HAO Z X, et al. Cloud computing resource load prediction based on combined prediction model[J]. Computer Engineering & Science, 2020, 42(7): 1168-1173. (in Chinese)

[15] 姚军, 刘明. 基于凌日搜索算法优化的组合预测模型[J]. 计算机应用, 2025, 45(12): 3925-3930.
YAO J, LIU M. Combined prediction model optimized by transit search algorithm[J]. Journal of Computer Applications, 2025, 45(12): 3925-3930. (in Chinese)

[16] 徐江, 张晨飞, 王富强, 等. 基于ARIMA-LSTM的容器云资源预测方法[J]. 重型机械, 2022(6): 6-14.
XU J, ZHANG C F, WANG F Q, et al. Container cloud resource prediction method based on ARIMA-LSTM[J]. Heavy Machinery, 2022(6): 6-14. (in Chinese)

[17] ZHANG L K, XIE Y L, JIN M P, et al. A novel hybrid model for docker container workload prediction[J]. IEEE Transactions on Network and Service Management, 2023, 20(3): 2726-2743.

[18] 贺小伟, 徐靖杰, 王宾, 等. 基于GRU-LSTM组合模型的云计算资源负载预测研究[J]. 计算机工程, 2022, 48(5): 11-17, 34.
HE X W, XU J J, WANG B, et al. Research on cloud computing resource load forecasting based on GRU-LSTM combination model[J]. Computer Engineering, 2022, 48(5): 11-17, 34. (in Chinese)

[19] 王艺霏, 于雷, 滕飞, 等. 基于长-短时序特征融合的资源负载预测模型[J]. 计算机应用, 2022, 42(5): 1508-1515.
WANG Y F, YU L, TENG F, et al. Resource load prediction model based on long-short time series feature fusion[J]. Journal of Computer Applications, 2022, 42(5): 1508-1515. (in Chinese)

[20] 史爱武, 罗于, 李林逸, 等. 基于MVMD-MHAT-BiLSTM的云资源负载预测方法[J]. 软件导刊, 2024(12): 18-26.
SHI A W, LUO G, LI L Y, et al. Cloud resource load prediction method based on MVMD-MHAT-BiLSTM[J]. Software Guide, 2024(12): 18-26. (in Chinese)

[21] ZHANG M, AN C Y, YANG C H. Multivariate workload aware correlation model for container workload prediction[C]// 2023 IEEE 29th International Conference on Parallel and Distributed Systems. Piscataway: IEEE, 2024: 972-979.

[22] ZHAO F Y, LIN W W, LIN S S, et al. TFEGRU: Time-frequency enhanced gated recurrent unit with attention for cloud workload prediction[J]. IEEE Transactions on Services Computing, 2025, 18(1): 467-478.

[23] CHEN L, ZHANG W W. A deep learning-based approach with PSO for workload prediction of containers in the cloud[C]//2021 13th International Conference on Advanced Infocomm Technology. Piscataway: IEEE, 2022: 204-208.

[24] 胡应钢, 郭翔, 赵海燕, 等. 基于粒子群优化GRU-RNN组合模型的云计算资源负载预测[J]. 内蒙古民族大学学报(自然科学版), 2023, 38(4): 315-321.
HU Y G, GUO X, ZHAO H Y, et al. Cloud computing resource load prediction based on particle swarm optimization GRU-RNN combination model[J]. Journal of Inner Mongolia Minzu University (Natural Sciences), 2023, 38(4): 315-321. (in Chinese)

[25] XU M X, SONG C H, WU H M, et al. esDNN: Deep neural network based multivariate workload prediction in cloud computing environments[J]. ACM Transactions on Internet Technology, 2022, 22(3): 1-24.

[26] FENG B B, DING Z J. GROUP: An end-to-end multi-step-ahead workload prediction approach focusing on workload group behavior[C]//Proceedings of the ACM Web Conference 2023. New York: ACM, 2023: 3098-3108.

[27] DING Z J, FENG B B, JIANG C J. COIN: A container workload prediction model focusing on common and individual changes in workloads[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(12): 4738-4751.

[28] 徐海洋, 刘海龙, 陈先, 等. 基于组合负载预测模型的多租户数据库弹性伸缩方法[J]. 软件学报, 2025, 36(3): 981-994.

XU H Y, LIU H L, CHEN X, et al. Elastic scaling method for multi-tenant databases based on hybrid workload prediction model[J]. Journal of Software, 2025, 36(3): 981-994. (in Chinese)

[29] LUO S T, XU H L, YE K J, et al. The power of prediction: Microservice auto scaling via workload learning[C]// Proceedings of the 13th Symposium on Cloud Computing. New York: ACM, 2022: 355-369.

[30] WANG L, GUO S D, ZHANG P L, et al. An efficient load prediction-driven scheduling strategy model in container cloud[J]. International Journal of Intelligent Systems, 2023, 2023: 5959223.

[31] MEHRAN N, NIKOLOV N, PRODAN R, et al. ADApt: Edge device anomaly detection and microservice replica prediction[C]//2025 IEEE 9th International Conference on Fog and Edge Computing. Piscataway: IEEE, 2025: 6-10.

[32] GAO J C, WANG H Y, SHEN H Y. Machine learning based workload prediction in cloud computing[C]//2020 29th International Conference on Computer Communications and Networks. Piscataway: IEEE, 2020: 1-9.

[33] SAXENA D, KUMAR J, SINGH A K, et al. Performance analysis of machine learning centered workload prediction models for cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2023, 34(4): 1313-1330.

[34] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[EB/OL]. (2023-08-02)[2025-10-10]. https://arxiv.org/abs/1706.03762.

[35] NIEWUYA. Cluster-trace-microservices-v2022[EB/OL]. (2023-06-27)[2025-08-27]. https://github.com/alibaba/clusterdata/tree/master/cluster-trace-microservices-v2022.

[36] GAN Y, ZHANG Y Q, CHENG D L, et al. An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems[C]// Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. New York: ACM, 2019: 3-18.

[37] DEATHSTARBENCH. Social network microservices benchmark[EB/OL]. (2019-01-01) [2025-11-29]. https://github.com/delimitrou/DeathStarBench/tree/master/socialNetwork.

作者简介

赵楠楠 女,1987年10月出生于山东省肥城市. 现为西北工业大学计算机学院副教授、硕士研究生导师. 主要研究方向为云存储、分布式系统、容器虚拟化等.
E-mail: nannanzhao@nwpu.edu.cn

张佳萌 女,2003年3月出生于河北省承德市. 现为西北工业大学计算机学院硕士研究生. 主要研究方向为云存储和分布式计算等.
E-mail: zhangjiameng@mail.nwpu.edu.cn

杨 帆 男,2000年12月出生于陕西省西安市. 现为西北工业大学计算机学院硕士研究生. 主要研究方向为面向机器学习的云存储、容器系统分布式技术等.
E-mail: one@mail.nwpu.edu.cn

吴若非 男,2004年3月出生于江苏省盐城市. 现为西北工业大学计算机学院本科生. 主要研究方向为云计算和数据存储等.
E-mail: wuruofei2022@mail.nwpu.edu.cn

王 浩 男,2001年10月出生于山西省临汾市. 现为西北工业大学计算机学院硕士研究生. 主要研究方向为容器系统和容器化应用等.
E-mail: kingshulk989@mail.nwpu.edu.cn

赵彤轩 男,2001年5月出生于陕西省西安市. 现为西北工业大学自动化学院硕士研究生. 主要研究方向为分布式自动控制系统、多传感器融合、鲁棒滤波等.
E-mail: npuzhaotongxuan@mail.nwpu.edu.cn