一种结合图卷积网络的微服务异常检测模型研究

Research on A Microservice Anomaly Detection Model Combining Graph Convolutional Networks

杨　晨，由志远，赵纯熙，童　博（中讯邮电咨询设计院有限公司，北京 100048）
Yang Chen，You Zhiyuan，Zhao Chunxi，Tong Bo（China Information Technology Designing & Consulting Institute Co.，Ltd.，Beijing 100048，China）

摘　要：
分布式部署的微服务系统中服务实例调用关系复杂，服务故障会随着调用链进行快速传播，给服务异常检测工作带来挑战。结合图卷积网络（GCN），提出一种先进的微服务异常检测模型（SADM-3）。模型采用eBPF技术对可观测性数据进行采集，构建服务调用有向图，并对节点和链路特征进行提取，使用GCN对异常服务节点和链路进行检测。最后，在TrainTicket和Sockshop数据集上对模型性能进行验证。实验结果表明，提出的SADM-3模型在异常检测准确率和检测速度方面均优于现有的基线模型。

关键词：
可观测性；特征提取；图卷积网络；服务异常检测
doi：10.12045/j.issn.1007-3043.2026.05.015
文章编号：1007-3043（2026）05-0079-07
中图分类号：TN915
文献标识码：A
开放科学（资源服务）标识码（OSID）：

Abstract：
In a distributed deployment microservice system，the service instance invocation relationships are complex，and service failures can quickly propagate along the invocation chain，posing challenges to service anomaly detection. It proposes an advanced microservice anomaly detection model（SADM-3）by combining graph convolutional networks（GCN）. The model first uses eBPF technology to collect observability data，constructs a service call directed graph，extracts node and link features，and uses GCN to detect abnormal service nodes and links. Finally，the performance of the proposed SADM-3 model is validated on the TrainTicket and Sockshop datasets，and the experimental results show that the model outperforms existing baseline models in terms of anomaly detection accuracy and speed.

Keywords：
Observability；Feature extraction；Graph convolutional network；Service anomaly detection

引用格式：杨晨，由志远，赵纯熙，等. 一种结合图卷积网络的微服务异常检测模型研究[J]. 邮电设计技术，2026（5）：79-85.

1 概述

1.1 背景

在微服务架构下，系统通常由分布式部署的多个独立的微服务组成，不同微服务之间相互调用，组成了系统的各个业务功能。每个微服务都可以在虚拟化环境中进行独立的开发、部署和扩展，增强了系统的敏捷性和可扩展性。

但由于微服务架构高内聚、低耦合和去中心化的特点，每个微服务均可进行多实例动态部署，产生了大量不同状态的服务实例，众多实例节点之间存在着复杂的调用关系[1]。当任意服务节点出现问题时，异常会沿着服务调用链进行传播，导致大量服务出现异常，这增加了服务异常检测的复杂性，成为微服务集群自动化运维工作的难点[2]。

1.2 现状

当前，在实际生产运维中，通常采用以下2种实现方式进行微服务的异常检测。一是通过对服务调用进行追踪，构造分布式服务调用拓扑图，通过对拓扑图中各节点的指标和日志数据进行监控分析，结合告警和运维经验实现对异常服务节点的定位[3-4]。二是基于多类型可观测性数据，将指标、链路、日志数据按

收稿日期：2026-03-18

照不同纬度进行组合，采用统计学方法、机器学习算法，对各服务节点进行分类或回归预测，作为服务节点异常的判断依据[5-6]。

本研究提出一种创新的微服务异常检测模型，该模型基于图卷积网络（GCN）实现微服务系统的服务实例级的异常检测。微服务异常检测模型由4个模块组成，具体如下。

a）数据采集和解析。基于eBPF技术实现对服务调用全链路数据的采集，通过在linux内核运行完成编译的、通过安全验证的eBPF代码，实现对内核流量的无侵入式采集。

b）服务调用有向图构建。设计一种新型的跨层级服务调用有向图的构建方法，基于指标、链路、日志数据，并结合服务调用时间序列，精确梳理整合多个服务调用请求。

c）特征提取和数据预处理。分别对有向图中的服务节点进行多模态特征提取，并采用Word2Vec嵌入模型，对特征中的字符串数据进行编码，减少噪声和特征波动对模型的干扰。

d）服务异常检测。构建基于图卷积网络的异常检测模型，对有向图中的节点特征进行学习，并采用有监督学习对模型进行训练，模型最终的输出结果为服务有向图中预测为异常的服务节点。

2 模型

2.1 模型总体描述

在微服务架构中，单个服务的独立性和动态性体现了分布式调用链追踪的关键作用。由用户发起的业务请求通常会涉及多个微服务之间的相互调用，因此，采集服务调用链中的每个服务请求的可观测数据，对理解各服务的依赖关系和执行情况至关重要。

为实现在分布式微服务系统中，自动精确地检测和定位发生故障的根因服务，本研究概念化了微服务之间的请求调用，如图1所示。有向图中的每一个节点都代表一个服务实例，每一个边都代表一次服务请求，本研究将业务请求调用链路定义为多个服务请求序列，即trace={span i, $i\in\mathbb{N}+$}，其中每个span={P_m, $E(m, n, i)$, P_n}，表示一个业务请求从发起到结束的全部调用过程，调用过程中涉及的每个节点P，都由节点标识特征四元组进行标识，允许2个服务节点之间存在多条请求链路。

本研究将服务调用有向图表示为$G=(P, E, X_p, X_e)$，其中，P表示由$\{p_1, p_2, p_3, \ldots, p_n\}$组成的服务节点集，$E$表示由$\{e_1, e_2, e_3, \ldots, e_n\}$组成的服务链路集，每个节点都有一组节点特征向量$X_p \in R^{p\times d}$与之关联，$d$代表特征向量的纬度。本研究通过设计开发一个服务异常检测模型，将服务异常检测定义为服务调用有向图中节点的分类和预测任务，通过学习图中节点的特征向量，快速识别和检测微服务系统的异常。

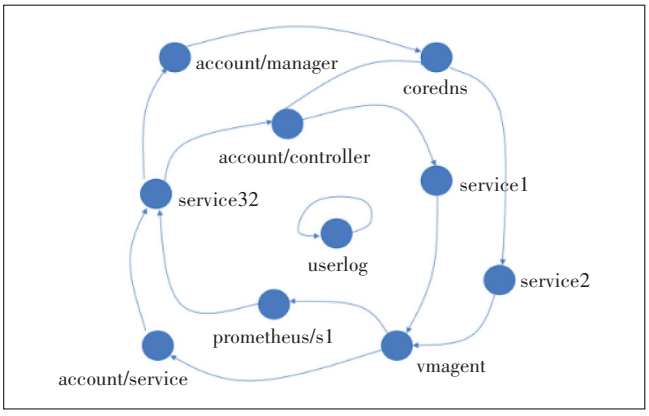


图1　微服务调用有向图

本研究提出的微服务故障异常检测模型（SADM-3）的工作流程如图2所示，模型由数据采集和解析、服务调用有向图构建、特征提取、服务异常检测4个阶段组成。

2.2 数据采集和解析

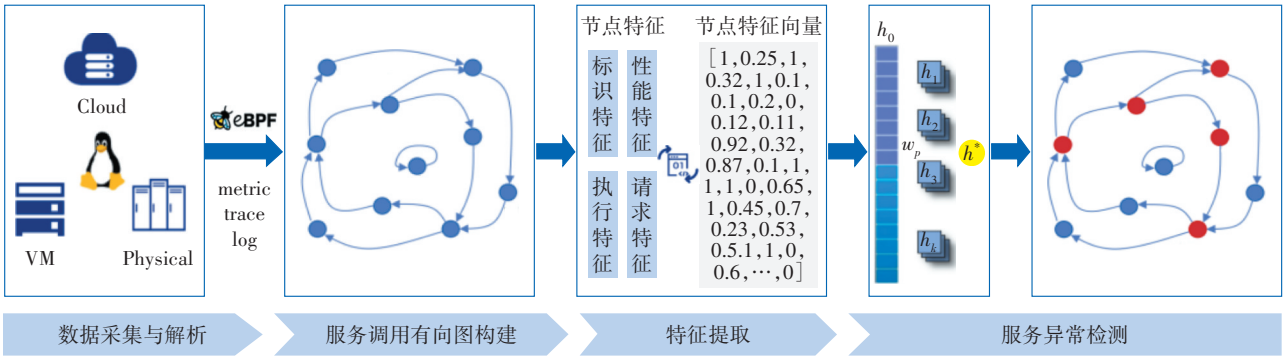
本研究基于eBPF技术，在分布式部署的微服务系统中，实现对指标、链路、日志等可观测性数据的无侵入式采集和解析。针对单个服务节点，采集、解析、构建了98维的指标数据、112维的链路数据、85维的日志数据。通过提取服务调用的trace ID、request ID、instance ID等关键数据类型，为服务调用有向图的构建和数据特征的提取提供充分的数据来源。

2.3 服务调用有向图构建

为准确构建微服务系统中各服务实例之间的调用关系，本研究以每个服务实例作为服务调用有向图的节点，将采集到的链路数据、指标数据、日志数据与服务实例进行关联。服务调用有向图的构建包括以下3个步骤。

a）创建服务节点。通过分析链路数据，识别出每次服务调用中执行的具体事务，从中提取执行各类事务的节点信息，构建服务节点的四维特征，即每个节点都有用来标识节点唯一性的四元组特征数据。

b）分析节点之间的调用关系。记录各服务节点



注：服务调用有向图与服务异常检测中有向图的节点与图1一致

图2　微服务故障异常检测模型（SADM-3）

之间的调用时间戳，通过分析服务调用发生的时序，梳理不同业务所触发的服务调用链路之间的顺序和依赖关系，进一步整合明确微服务系统的服务边界。

c）连接节点形成有向图。结合服务节点四元组中的时序数据，通过整合服务之间的请求响应关系，构建时序化的系统服务调用拓扑，动态反映微服务系统中每个时刻的服务交互行为，提升服务故障异常检测的可解释性。

2.4 特征提取和数据预处理

在使用图神经网络进行服务异常检测的过程中，需要对服务调用有向图中各服务节点的关键特征进行提取，作为异常检测模型的输入。对于服务节点，共提取了标识特征（mark）、性能特征（performance）、执行特征（execute）、请求特征（request）这4种不同类型的数据特征。

针对服务节点的标识特征和请求特征中字符串数据类型的特征，本研究采用Word2Vec嵌入模型[7]来训练和编码特征数据集中的字符串，最终将服务节点标识特征编码为64维向量。针对服务节点特征中的数值特征值，本研究用0填充数据集中的缺失数值，并采用数据归一化的方式，将相同类型的特征数值归一化到[0,1]。

2.5 服务异常检测

本研究选用图卷积网络（GCN）来构建服务异常检测模型，使用服务调用有向图中的每个服务的节点特征来训练模型。最终，模型以d维特征向量的形式，输出被模型识别为异常的服务节点。

模型使用DGL（2023）框架进行实现。通过卷积运算，GCN能够获取节点与邻居节点的相关性信息。对服务调用有向图$G=(P,X_p)$，在图卷积操作的初始状态下，模型的输入特征$H^{(0)}=x_p(\forall x_p \in X_p)$，使用式（1）生成节点特征矩阵$H^{(1)}$：

$$H^{(1)}=\sigma(W_p^{(1)}\cdot(H^{(0)}+H_{N(p)}^{(1)})) \tag{1}$$

$$H_{N(p)}^{(1)}=\mathrm{Agg}(H_m^{(0)}),\ \forall m \in N(p) \tag{2}$$

其中，σ表示GCN的激活函数；W_p表示可学习优化的节点权重矩阵；Agg表示平均聚合函数，用来计算节点特征的平均值；N表示选择函数，用来从节点的邻居集中选择相邻的节点。对叠加多个卷积层的GCN模型来说，可以通过卷积操作整合相邻节点和链路的特征信息。模型第k-1层的特征输出$H^{(k-1)}$是第k层的输入特征，使用式（3）生成第k层的特征矩阵$H^{(k)}$：

$$H^{(k)}=\sigma(W_p^{(k)}\cdot(H^{(k-1)}+H_{N(p)}^{(k)})) \tag{3}$$

$$H_{N(p)}^{(k)}=\mathrm{Agg}(H_m^{(k-1)}),\ \forall m \in N(p) \tag{4}$$

通过聚合有向图中相邻节点和前一层节点的特征，服务节点异常检测模型使用式（5）输出检测为异常的服务节点特征。

$$O_p=H_{N(p)}^{(k)} \tag{5}$$

3 实验

3.1 实验目标

为对提出的服务故障异常检测模型进行全面的评估，本研究设计了一系列实验场景，并通过对实验结果进行对比分析，回答以下3个研究问题。

问题1：基于服务调用有向图提取的服务特征和路径特征，各类型特征对服务异常检测的准确性有多大影响（第3.3.1节）？

问题2：与其他服务异常检测的方法相比，本研究训练的基于图卷积神经网络的模型在准确性和鲁棒

性方面是否具备优势(第3.3.2节)?

问题3:与其他服务异常检测的方法相比,本研究训练的基于图卷积神经网络的模型在数据量依赖和时间开销方面是否具备优势(第3.3.3节)?

3.2 实验设置

3.2.1 实验环境

实验环境分为微服务集群环境和模型训练环境,其中,服务集群环境作为微服务系统的运行环境,部署了可观测数据采集工具,来获取服务请求调用数据;模型训练环境则用来对提出的异常检测模型进行训练和验证。

微服务集群环境基于Kubernetes进行搭建,由12台虚机组成,其中3台作为master节点,其余9台虚机分别构建成3个node节点,各节点的虚机的配置摘要如表1所示。

表1 微服务集群环境配置摘要

名称	操作系统	CPU	内存	数量
master1	CentOS 7.0	4核8线程	64 G	3台
node1.1	CentOS 7.0	12核24线程	1 T	3台
node2.1	CentOS 7.0	12核24线程	1 T	3台
node3.1	CentOS 7.0	12核24线程	1 T	3台

在模型算法的实现过程中,本研究使用机器学习框架PyTorch来实现图卷积网络的训练以及其他基线模型的训练和验证。Python环境集成在Anaconda3中,具体的模型训练环境配置如表2所示。

表2 模型训练环境配置

名称	配置
操作系统	CentOS 7.0
处理器	Intel Corei7-4790 3.60 GHz
内存	1 T
显卡	NVIDIA RTX 3090
环境库	Python3.8、Pytorch、Numpy、Pandas

3.2.2 实验数据集

为对本研究所提出的微服务异常检测模型的性能进行评估,实验选用了2个开源的微服务系统Sockshop和TrainTicket。Sockshop作为一个电子商务系统,包含8个不同的微服务,而TrainTicket是一个火车票预订系统,集成了45个微服务。实验通过在微服务集群环境中部署2个系统,为故障异常检测模型的训练和验证提供数据基础。

同时,在微服务架构的系统环境下,由资源配置问题而引发的故障尤为广泛。资源配置故障通常由环境中的资源限制配置引起,会导致环境中部分服务实例无法获取足够的资源来处理请求,常见的资源配置故障有内存溢出、CPU过载、网络时延过高等类型。因此,实验基于上述2个微服务系统,通过故障注入的方式,将资源密集型操作集成到请求处理代码中,用于生成资源配置故障,并利用自动化测试工具来模拟用户请求并生成相应的数据。

为采集服务实例相关的性能指标,本研究使用基于eBPF的采集器,直接从linux内核获取服务调用的性能数据,并采用分布式跟踪框架Apache SkyWalking(Apache 2023)来收集服务请求调用数据。实验在采集器部署并开始进行性能采集和链路追踪后,随机选择多个服务实例进行故障注入。采集器以1 s为采集时间间隔,获取系统从正常状态到发生故障后,各服务实例的性能指标和调用链路时序数据。最终,实验选取故障注入前1 h和后1 h(共计2 h)内,共7 200组的时序性能数据,分别获得了服务节点和服务链路的时序数据集合,服务节点和服务链路数据集概述如表3和表4所示。对于每个数据集,实验以8:2的比例将数据集划分为训练集和测试集。

3.2.3 对比实验设计

为了对第3.1节中提出的关键问题进行验证,设计如下对比实验。

针对问题1,实验在对基于图卷积网络的异常检测模型进行训练时,选取全量特征、部分特征分别对

表3 服务节点数据集概述

	Sockshop数据集	TrainTicket数据集
服务节点(单组)/个	98	583
服务节点(共计)/个	705 600	4 197 600
正常节点(共计)/个	661 500	3 950 683
异常节点(共计)/个	44 100	246 917
正常/异常比例	16:1	17:1

表4 服务链路数据集概述

	Sockshop数据集	TrainTicket数据集
服务链路(单组)/条	74	428
服务链路(共计)/条	532 800	3 081 600
正常链路(共计)/条	497 280	2 900 330
异常链路(共计)/条	35 520	181 270
正常/异常比例	15:1	17:1

异常识别模型进行训练，以验证使用不同数量的特征数据进行训练对模型异常识别准确性的影响程度，进一步验证各类型特征对模型的贡献程度。

针对问题2，实验选用了5种异常检测方法，与本研究构建的异常检测模型进行对比实验。选用的对比方法分为2类，一是基于统计学的异常检测方法WAS[8]（Workflow-Aware approach with Statistics），二是基于机器学习的异常检测方法，包括局部离群因子模型[9]（Local Outlier Factor，LOF），增强孤立森林算法[10]（Enhance Isolation Forest，EIF），频率增强条件变分自编码器[11]（Frequency enhanced Conditional Variational AutoEncoder，FC_VAE）和多模态LSTM[12]（Multimodal-LSTM），以对各类异常检测方法的总体性能进行验证和综合分析。

针对问题3，实验选用上述5种基线模型，分别使用不同数量的训练数据对模型进行训练，以验证各模型对数据量的依赖程度，并对各模型在训练和测试验证的时间开销进行统计和评估，对异常检测模型的检测速度进行分析。

3.2.4 评价指标

异常检测模型的评价指标基于4种基本分类，即真阳性（TP，表示预测正确的阳性样本）、真阴性（TN，表示预测正确的阴性样本）、假阳性（FP，表示预测错误的阳性样本）和假阴性（FN，表示预测错误的阴性样本），分类原理如表5所示。

表5　评价指标混淆矩阵

	Positive	Negative
True	真阳性（TP）	真阴性（TN）
False	假阳性（FP）	假阴性（FN）

实验基于4种基本分类，采用以下指标对异常检测模型的性能进行量化评估。

a）准确率（accuracy）：表示预测正确的样本占总样本的比例，用来对所有样本的预测进行评估，计算如式（6）所示。

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

b）精确率（precision）：表示预测正确的阳性样本占所有预测为阳性样本的比例，即所有预测为异常的样本中，真正为异常的样本所占的比例，计算如式（7）所示。

$$\text{precision} = \frac{TP}{TP + FP} \tag{7}$$

c）召回率（recall）：表示所有预测正确的阳性样本占所有真正为阳性样本的比例，即所有异常的样本中，被预测为异常的样本所占的比例，计算如式（8）所示。

$$\text{recall} = \frac{TP}{TP + FN} \tag{8}$$

d）调和平均数（F1-score）：表示精确率和召回率的调和平均数。F1-score值只有在精确率和召回率均较高的情况下才能实现，因此F1-score值可以较好地对异常检测的准确率进行评价，其计算如式（9）所示。

$$\text{F1-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{9}$$

3.3 实验分析

3.3.1 特征关键性分析

在本节中，实验对基于TrainTicket数据集提取的服务节点特征进行裁剪，并使用裁剪后的特征对异常检测模型进行训练。结合异常检测模型评价指标，量化评估各类型特征对异常检测模型检测准确性的影响，训练后的模型在测试集上的表现如图3所示。

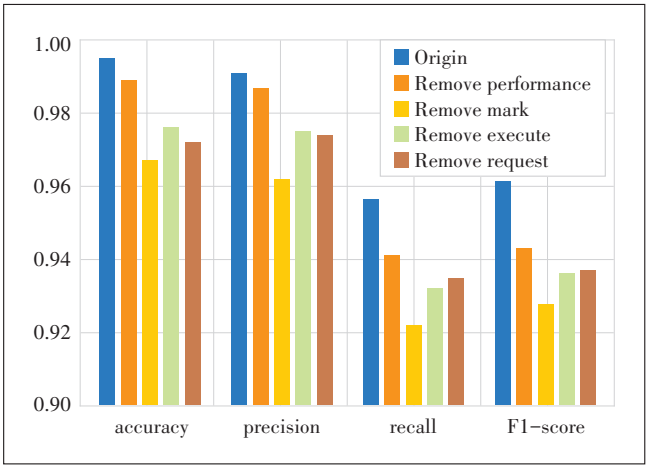


图3　异常检测模型SADM-3在TrainTicket数据集的实验结果

从图3可以看出，与完整的节点特征相比，在使用去除性能特征后的节点特征进行模型训练时，模型的召回率和F1值分别下降了1.57%和1.87%，但对准确度和精准度没有较大影响。但去除节点标识特征后，模型的召回率和F1值分别下降了3.56%和3.43%，对服务异常检测的成功率产生了较大影响，且检测的准确度和精准度也有明显下降，可以说明，节点标识特征对提升异常检测的有效性起到了关键作用。在使用去除执行特征后的节点特征进行模型训练时，模型的召回率和F1值分别下降了2.51%和2.60%，在使用

去除请求特征后的链路特征进行模型训练，模型的召回率和F1值分别下降了2.20%和2.50%，同时，异常检测的准确度和精准度也都发生了一定的下降。

特征关键性实验结果表明，本研究提取的每个节点特征对提升异常检测模型的整体性能都有着独有的贡献，特别是节点的标识特征和执行特征，对异常检测模型的训练至关重要。

3.3.2 异常检测准确性分析

为验证异常检测模型的综合性能，实验基于Sock-shop数据集和TrainTicket数据集，分别对选取的5个对比模型以及本研究构建的异常检测模型进行训练，各模型在测试集上的表现如表6所示。

传统基于统计学的WSA模型无法对微服务环境下复杂的服务调用关系进行学习和理解，导致其在测试数据集上的表现并不出色。与LOF、EIF、FC_VAE、Mul-LSTM相比，SADM-3在SockShop数据集上的F1值分别提高了12.2%、18.0%、5.19%、3.70%，在Train-Ticket数据集上的F1值分别提高了10.6%、17.7%、4.72%、5.70%，表明SADM-3在理解和检测不同微服务系统异常调用方面具有较强的鲁棒性。与对比模型相比，SADM-3在对2个不同微服务系统进行异常检测时，均取得了较高的F1分值，这一结果得益于SADM-3可以有效聚合服务调用有向图中相邻节点之间的特征，理解微服务调用之间的复杂关系。同时，SADM-3的准确度和精准度尤为出色，该模型在Sock-Shop和TrainTicket数据集中的准确度分别为0.997 6和0.995 2，精准度分别为0.989 4和0.991 2，较高的检测准确度和精准度对减少异常检测的误报至关重要。

异常检测模型的实验结果表明，与选取的对比模型相比，SADM-3模型表现出了最佳的检测性能，体现出SADM-3模型具备良好的适应性和学习效率。

3.3.3 异常检测效能分析

为验证SADM-3模型在不同数量训练数据下的有效性和鲁棒性，研究分别使用不同比例的SockShop数据和TrainTicket数据对模型进行训练，训练后的模型在测试集上的表现如表7所示。

实验结果表明，尽管训练数据量有限，SADM-3模型也可保持较高的检测性能。在只有5%训练数据的情况下，SADM-3模型在SockShop和TrainTicket数据集上的F1分值分别为0.932 9和0.889 5，表明SADM-3模型在较少训练数据的情况下依然可以获得较好的异常检测能力。随着训练数据增加到10%和50%，模型的各项指标都得到了较大的提升。与使用全量训练数据训练的对比模型相比，使用少量训练数据的SADM-3模型表现出的检测性能仍然具有竞争力。

本研究提出的SADM-3异常检测模型与基线模型在2个数据集上训练和测试的时间开销如表8所示。

实验结果表明，尽管基于统计学的WSA模型的训练时间较短，但其异常检测的性能表现较差，而其他基线模型在训练时花费的时间开销没有明显差异。本研究提出的SADM-3模型由于充分学习了服务之间的调用关系，虽然花费了相对较长的训练时间，但模

表6　各异常检测模型在2个数据集上的测试结果

Dataset Method	SockShop				TrainTicket			
	accuracy	precision	recall	F1-score	accuracy	precision	recall	F1-score
WSA	0.818 9	0.792 3	0.742 3	0.851 1	0.871 6	0.857 6	0.729 8	0.796 0
LOF	0.851 5	0.848 1	0.855 2	0.853 3	0.832 3	0.825 1	0.867 1	0.859 3
EIF	0.736 3	0.709 8	0.817 5	0.797 8	0.719 8	0.706 3	0.802 3	0.791 0
FC_VAE	0.917 8	0.907 3	0.919 8	0.922 3	0.923 5	0.936 5	0.901 2	0.915 9
Mul-LSTM	0.975 2	0.961 5	0.929 8	0.936 9	0.963 1	0.969 3	0.892 8	0.906 5
SADM-3	0.997 6	0.989 4	0.969 0	0.972 8	0.995 2	0.991 2	0.956 2	0.961 3

表7　不同比例数据所训练的SADM-3模型在测试数据集上的表现

Dataset Coverage	SockShop				TrainTicket			
	accuracy	precision	recall	F1-score	accuracy	precision	recall	F1-score
5%	0.985 8	0.981 2	0.883 4	0.932 9	0.988 6	0.982 1	0.890 1	0.889 5
10%	0.991 2	0.984 7	0.930 9	0.956 3	0.991 2	0.986 7	0.931 1	0.912 9
50%	0.995 6	0.982 5	0.942 1	0.969 2	0.993 2	0.989 9	0.948 2	0.952 3
100%	0.997 6	0.989 4	0.969 0	0.972 8	0.995 2	0.991 2	0.956 2	0.961 3

表8　各异常检测模型进行训练和测试的时间开销

Dataset	Time	WSA	LOF	EIF	FC_VAE	Mul-LSTM	SADM-3
Sock-Shop	Training time/min	9	14	16	18	24	32
	Test time/s	12	21	16	18	36	24
Train-Ticket	Training time/min	58	82	86	89	102	118
	Test time/s	72	118	102	109	132	124

型异常检测的性能最好。结合模型测试验证时间开销，训练后的各模型在实际异常检测中所需的检测时间差异并不明显。

异常检测模型的实验结果表明，在训练数据有限的情况下，SADM-3模型仍然获得较高性能表现；与基线模型相比，SADM-3的模型训练时间开销相对较大，但模型进行异常检测的时间开销影响并不明显。

4 总结

为解决分布式微服务系统中服务实例调用关系复杂、故障传播迅速所导致的服务异常监测困难的问题，本研究提出了一种结合图卷积网络（GCN）的微服务异常检测模型（SADM-3），研究的主要贡献如下。

a）在服务调用有向图的构建方面，本研究基于eBPF技术，实现对可观测性数据的无侵入式采集，实现了一种跨层级的服务调用有向图构建方法，能够精确梳理和整合多个服务调用请求，动态反映微服务系统的服务交互行为。

b）在服务异常检测模型方面，本研究通过对服务节点的多模态特征进行提取和编码，使用GCN对服务调用有向图中的节点特征进行学习，有效聚合相邻节点的特征信息，提升异常检测的准确性和鲁棒性。

c）在模型性能验证方面，本研究在搭建的开源微服务系统下进行了多角度的性能验证实验。实验结果表明，本研究提出的SADM-3模型在准确率、召回率和F1-score等指标上均优于现有的基线模型，具有较强的实用性和推广价值。

尽管SADM-3模型在异常检测性能上表现出色，但其训练时间相对较长，未来将探索更高效的训练算法或分布式训练策略，以缩短模型的训练时间。同时，当前的模型主要关注异常检测，未来可以结合故障根因分析技术，进一步定位异常的根本原因，为微服务系统的自动化运维提供更全面的支持。

参考文献：

[1] 李亚晓，李青山，王璐，等. AmazeMap：基于多层次影响图的微服务故障定位方法[J]. 软件学报，2024，35(7)：3115-3140.
[2] 方浩天. 基于深度学习的微服务性能异常检测与根因定位方法研究[D]. 武汉：华中科技大学，2023.
[3] CHEN J，LIU F G，JIANG J，et al. TraceGra：a trace-based anomaly detection for microservice using graph deep learning[J]. Computer Communications，2023，204：109-117.
[4] LIU D W，HE C，PENG X，et al. MicroHECL：high-efficient root cause localization in large-scale microservice systems [C]//2021 IEEE/ACM 43rd International Conference on Software Engineering：Software Engineering in Practice (ICSE-SEIP). Madrid：IEEE，2021：338-347.
[5] PANAHANDEH M，HAMOU-LHADJ A，HAMDAQA M，et al. ServiceAnomaly：an anomaly detection approach in microservices using distributed traces and profiling metrics[J]. Journal of Systems and Software，2024，209：111917.
[6] CHEN Z B，LIU J Y，SU Y X，et al. Adaptive performance anomaly detection for online service systems via pattern sketching[C]//Proceedings of the 44th international conference on software engineering. New York：Association for Computing Machinery，2022：61-72.
[7] MIKOLOV T，CHEN K，CORRADO G，et al. Efficient estimation of Word representations in vector space [EB/OL]. [2026-01-26]. https://arxiv.org/abs/1301.3781.
[8] WANG T，ZHANG W B，XU J W，et al. Workflow-aware automatic fault diagnosis for microservice-based applications with statistics[J]. IEEE Transactions on Network and Service Management，2020，17(4)：2350-2363.
[9] XU Z K，KAKDE D，CHAUDHURI A. Automatic hyperparameter tuning method for local outlier factor，with applications to anomaly detection[C]//2019 IEEE International Conference on Big Data (Big Data). Los Angeles：IEEE，2019：4201-4207.
[10] HARIRI S，KIND M C，BRUNNER R J. Extended isolation forest[J]. IEEE Transactions on Knowledge and Data Engineering，2021，33(4)：1479-1489.
[11] KINGMA D P，WELLING M. Auto-encoding variational bayes[EB/OL]. [2025-08-26]. https://arxiv.org/abs/1312.6114.
[12] NEDELKOSKI S，CARDOSO J，KAO O. Anomaly detection from system tracing data using multimodal deep learning[C]//2019 IEEE 12th International Conference on Cloud Computing (CLOUD). Milan：IEEE，2019：179-186.

作者简介：
杨晨，工程师，硕士，主要从事云内监控产品的研发设计工作；由志远，高级工程师，硕士，主要从事网络技术与数智化技术融合创新研究及应用研发工作；赵纯熙，工程师，硕士，主要从事网络创新产品、云内监控产品的研发设计工作；童博，高级工程师，硕士，主要从事新型IP网络技术、SDN及云网创新产品的研究、研发工作。