软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007612] [CSTR: 32375.14.jos.007612]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于图表示学习的在线服务系统告警聚类方法*

陈 淼[1,2], 张弼铖[1,2], 张晨曦[3], 彭 鑫[1,2], 杨定裕[4], 李 伟[5], 钱泽林[5], 吴哲顺[5], 欧嘉煜[5], 钟坚锐[5]

[1](复旦大学 计算与智能创新学院, 上海 200438)

[2](上海市数据科学重点实验室 (复旦大学), 上海 200438)

[3](西安电子科技大学 计算机科学与技术学院, 陕西 西安 710126)

[4](区块链与数据安全全国重点实验 (浙江大学), 浙江 杭州 310027)

[5](阿里巴巴控股集团, 浙江 杭州 310023)

通信作者: 张晨曦, E-mail: zhangchenxi@xidian.edu.cn

摘 要: 在大型在线服务系统中, 由于各组件间错综复杂的依赖关系, 故障的发生常会引发大量相关告警, 形成告警风暴. 告警风暴不仅增加了值班工程师的工作负担, 也使得故障诊断和根因分析变得更加困难. 为了解决这一问题, 提出了 Alert-CM, 一种基于图表示学习的告警聚类方法, 将由同一故障引发的告警有效地聚类在一起, 从而减轻工程师的工作负担. 在告警管理过程中, 一条告警往往由故障发生时的多种底层系统数据共同汇总得到, 如异常相关应用、指标、日志、预警规则和应急场景等. Alert-CM 认为由同一个故障引发的多个告警通常在底层系统数据层面存在紧密关联, 并且当中的底层核心系统数据将更能代表当前告警对应的系统异常状态. 根据这一核心思想, Alert-CM 基于告警相关配置数据构建出细粒度的系统数据依赖图, 并将告警与图中各节点的依赖关系进行抽象和映射, 进一步扩展告警的特征空间. 基于系统数据依赖图, Alert-CM 搭建了图神经网络模型进行图表示学习, 挖掘核心底层系统数据对于告警的贡献强弱, 从而输出准确的告警向量表示. 最终, Alert-CM 使用 DBSCAN 算法实现告警聚类. 在真实工业数据集上对 Alert-CM 进行评估, 重点考察聚类的有效性和实时效率. 实验结果表明, Alert-CM 在告警聚类任务中的表现显著优于传统的告警聚合方法. 在评估中, Alert-CM 的 NMI 和 ARI 分别达到了 0.901 和 0.645, 相较于现有方法的平均值分别提升 31.7% 和 153.9%, 同时 Alert-CM 在在线实时聚类任务上也表现出良好的性能.

关键词: 告警; 聚类; 图表示学习; 图神经网络

中图法分类号: TP311

中文引用格式: 陈淼, 张弼铖, 张晨曦, 彭鑫, 杨定裕, 李伟, 钱泽林, 吴哲顺, 欧嘉煜, 钟坚锐. 基于图表示学习的在线服务系统告警聚类方法. 软件学报. http://www.jos.org.cn/1000-9825/7612.htm

英文引用格式: Chen M, Zhang BC, Zhang CX, Peng X, Yang DY, Li W, Qian ZL, Wu ZS, Ou JY, Zhong JR. Alert Clustering Method Based on Graph Representation Learning for Online Service Systems. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7612.htm

Alert Clustering Method Based on Graph Representation Learning for Online Service Systems

CHEN Miao[1,2], ZHANG Bi-Cheng[1,2], ZHANG Chen-Xi[3], PENG Xin[1,2], YANG Ding-Yu[4], LI Wei[5], QIAN Ze-Lin[5], WU Zhe-Shun[5], OU Jia-Yu[5], ZHONG Jian-Rui[5]

[1](College of Computer Science and Artificial Intelligence, Fudan University, Shanghai 200438, China)

[2](Shanghai Key Laboratory of Data Science (Fudan University), Shanghai 200438, China)

[3](School of Computer Science and Technology, Xidian University, Xi'an 710126, China)

[4](National Key Laboratory of Blockchain and Data Security (Zhejiang University), Hangzhou 310027, China)

* 收稿时间: 2024-12-10; 修改时间: 2025-06-05; 采用时间: 2025-12-09; jos 在线出版时间: 2026-04-01

[5](Alibaba Group, Hangzhou 310023, China)

Abstract: In large-scale online service systems, intricate dependencies among components often cause a single fault to trigger a massive number of correlated alerts, resulting in alert storms. Alert storms not only increase the workload of on-call engineers but also make fault diagnosis and root cause analysis more challenging. To address this issue, this study proposes Alert-CM, a graph representation learning-based alert clustering method that effectively groups alerts caused by the same fault, thereby alleviating engineers' workloads. In alert management, an alert is typically generated by aggregating various types of underlying system data at the time of a fault, including related applications, metrics, logs, alert rules, and emergency scenarios. Alert-CM assumes that multiple alerts triggered by the same fault usually exhibit tight correlations at the underlying system data level, and that the core underlying system data better represents the abnormal system state associated with the alerts. Based on this assumption, Alert-CM constructs a fine-grained system data dependency graph using alert-related configuration data, abstracting and mapping the dependency relationships between alerts and graph nodes to further expand the alert feature space. On top of the system data dependency graph, a graph neural network model is built to perform graph representation learning, in which the contributions of core underlying system data to alerts are automatically learned, producing accurate alert vector representations. Finally, the DBSCAN algorithm is applied to cluster alerts based on the learned representations. Alert-CM is evaluated on a real-world industrial dataset, with a focus on clustering effectiveness and real-time efficiency. The experimental results demonstrate that Alert-CM significantly outperforms traditional alert aggregation methods in alert clustering tasks. Specifically, Alert-CM achieves an NMI of 0.901 and an ARI of 0.645, corresponding to average improvements of 31.7% and 153.9% over existing methods, respectively. In addition, Alert-CM exhibits strong performance in online real-time clustering tasks.

Key words: alert; clustering; graph representation learning; graph neural network (GNN)

随着云计算的发展和微服务架构[1]的普及, 现代在线服务系统的业务功能不断扩展, 项目规模日益庞大, 内部业务逻辑也变得愈加复杂. 在这些规模较大且复杂的系统中, 故障的发生往往会导致严重的经济损失[2], 并对系统的安全性和企业信誉造成巨大的影响. 因此, 实时的异常检测和快速的故障恢复已成为保障系统稳定性的关键. 在企业的实际应用中, 可观测技术和告警管理[2–4]通常被结合使用, 以确保系统的持续健康监控. 可观测技术通常包括日志系统、分布式追踪系统与指标监控系统, 常见的工具有 Loki[5]、OpenTelemetry[6]、Prometheus[7]等, 其主要作用是根据配置对系统状态进行持续监控. 告警管理则是进一步在监控数据发生波动时判定是否达到告警触发条件. 若满足异常条件, 告警系统将实时收集异常信息, 生成告警并及时上报给值班工程师. 值班工程师根据告警信息对系统状态进行初步分析, 诊断故障原因并分派处理任务. 在此基础上, 相关系统工程师将对故障进行排查、定位和修复, 从而迅速解决问题, 防止故障进一步影响系统业务[4]. 然而, 在复杂的微服务系统中, 不同服务之间调用关系错综复杂, 由于级联效应[3], 一个微小的故障往往会在系统中快速传播, 并且引发相关服务的告警. 同时, 在由大量组件构成的系统中, 多个相对独立的故障也可能同时发生并导致告警数量剧增[8]. 在告警风暴[8–11]中, 值班工程师可能会被大量告警信息淹没, 导致无法及时准确地对每个告警进行分析和分派. 这不仅增加了处理的时间成本, 还可能因为告警的错误分派而导致后续故障处理的延误, 从而造成更大的系统风险.

为了避免告警风暴发生时的灾难情形, 告警聚合[8–12]技术应运而生. 告警聚合通过分析告警之间的关联性, 将来源于同一故障或高度重复的告警合并为一个告警组, 从而减轻值班工程师的负担, 并为故障分派和诊断提供便利. 现有的告警聚合技术通常以告警文本作为告警的主要特征, 并使用相关的应用拓扑结构和异常指标数据进一步扩充告警的特征空间. 告警聚合技术主要分为两类: 无监督学习方法和有监督学习方法. 无监督学习聚焦于挖掘异常状态下告警数据的内在模式和规律, 而有监督方法则通过机器学习技术, 利用历史人工聚合的告警信息引导模型训练, 输出更加精准的告警特征向量, 从而实现对告警的自动化精准聚合.

在告警管理过程中, 相较于直接依赖告警文本作为特征, 实际告警往往来源于底层系统中多个细粒度的数据, 这些底层系统数据通过告警的配置信息可以被拆解成多个方面, 如告警关联的应用、监控项、预警规则、应急场景、指标和日志等. 这些底层系统数据之间不仅存在层次化的依赖关系, 而且各层数据内部也存在复杂的关联. 当系统发生故障时, 异常状态会在这些底层系统数据之间传播, 最终引发相关告警. 因此, 由同一故障引发的多个告警必定在底层系统数据层面存在紧密的关联性, 且核心底层系统数据将更能准确反映当前的系统异常状态. 然而,

在实际应用中, 不同底层系统数据对于告警的影响强弱并不相同, 并且这种影响会根据不同的异常场景和故障类型发生动态变化. 例如, 某些系统数据可能在特定故障情况下对告警的生成起到更为关键的作用, 而在其他场景中, 这些数据可能对告警的影响较弱. 因此, 为了更有效地识别和聚类告警, 准确捕捉与告警相关的核心底层系统数据变得尤为重要. 根据这一核心思想, 本文对告警管理过程中的底层系统数据构建了层次化的系统数据依赖图, 并提出了基于图表示学习的告警聚类方法 Alert-CM (alert clustering graph representation learning model). 具体来说, Alert-CM 首先构建了一个层次化的系统数据依赖图, 将告警所关联的多种底层系统数据及其相互依赖关系抽象成一张异构图, 并将告警与图中各个节点的依赖关系进行提取. 在此基础之上, Alert-CM 采用图表示学习方法, 构建图神经网络, 利用历史告警的聚类信息引导模型训练, 从而有效区分出不同底层系统数据对告警的重要性, 进而准确输出告警对应特征向量表示. 最终, Alert-CM 使用 DBSCAN[13]方法输出告警聚类结果. 通过这一方法, Alert-CM 能够精准识别与告警相关的核心底层系统数据, 进而提升聚类的准确性和效果. 这种方法不仅提升了告警聚类的有效性和效率, 也为复杂故障诊断提供了更为可靠的支持.

为了验证方法的有效性与效率, 本文使用来自阿里巴巴控股集团的真实工业环境的告警数据集展开一系列实验研究. 实验结果表明, Alert-CM 在告警聚类任务中取得了良好的表现. 与现有的聚合方法相比, Alert-CM 在 NMI、ARI 指标上分别达到了 0.901 和 0.645, 提升了 31.7% 和 153.9%. 此外, Alert-CM 在在线告警聚类阶段表现出较高的效率, 很好地满足了实时聚类的需求.

综上所述, 本文的主要贡献如下.

• 根据告警管理过程将告警相关底层系统数据抽象为层次化的系统数据依赖图, 有效扩展了告警的特征空间, 为告警聚类提供了更加丰富的上下文信息.

• 提出一种基于图表示学习的告警聚类方法 Alert-CM, 在层次化系统数据依赖图的基础上引入图神经网络模型, 精确捕捉告警与核心底层系统数据之间的复杂关系, 显著提升了聚类的准确性和效率.

• 设计一系列实验验证了方法的有效性和效率, 并分析了一些重要参数配置的影响.

本文第 1 节介绍告警聚类的相关工作. 第 2 节介绍本文所需的基础知识. 第 3 节重点阐述本文基于系统数据依赖图和图表示学习技术的告警聚类方法. 第 4 节通过实验验证了本文方法的有效性与优越性. 第 5 节对本文内容进行总结, 并展望未来的工作.

1 相关工作

目前的告警聚合工作可以从使用的数据类型角度分为以下 3 类: 基于告警文本、基于异常指标和基于应用拓扑结构的聚合方法. 为了实现更为精确的告警聚合效果, 当前的研究通常结合多种数据类型进行综合分析.

现有的工作中, 利用告警内容之间的相似度进行聚合是最为常见的思路. 由于工程师会预先定义告警模板, 并且告警管理系统会将触发告警时的系统异常状态实时汇总到告警模板中形成告警内容, 因此告警内容以规整的格式有效总结了系统具体异常状态的关键信息, 为告警聚合方法提供了极大的便捷性. 在对告警内容进行预处理与标准化之后, 告警内容中的停顿词、连接词和非语言符号等无意义信息被去除, 告警内容被整理成由多个关键词句构成的文本序列. 基于此, 告警聚合技术可以从文本和语义两个方面进一步挖掘告警内容的特征.

基于文本的告警聚合方法一般采用基于关键词重叠度的策略, 即两个告警的文本内容相似度越高, 它们对应的异常情况和故障主题也越相似, 聚类的相关性就越强. Zhao 等人[11]提出了一种利用词袋模型对告警文本进行分词, 并基于 Jaccard 系数计算词袋的重合度, 以此衡量告警文本之间的相似度的方法. 然而, 由于在线系统的复杂性, 利用告警文本相似度来判定告警关联存在局限性, 其仅适用于判别小规模系统中告警明显关联的情况. 而在复杂系统中, 不同的服务组件由不同的团队负责管理, 不同团队在制定告警时并不遵循统一的标准. 因此, 即使是被人工判断关联的两个告警在文本表达上也不一定高度相似. 更进一步地, 故障引发的告警往往在逻辑层面而非文本层面上具有内在关联, 比如“订单提交错误”告警与“订单页面前端渲染异常”告警常被关联. 因此, 越来越多的告警聚合技术开始探索告警在语义相似度层面的关联性, 进而提高聚合的准确性.

利用告警语义信息的聚合技术一般将离散的告警文本转化为连续的告警特征向量, 并可以进一步分为无监督方法和有监督方法两种. 常见的无监督学习方法通过使用预训练的词嵌入模型 (如 CBOW[14]、FastText[15]、Word2Vec[16]或 BERT[17]) 来获取告警上下文中的深层语义信息, 将告警文本转化为特征向量. GRLIA[8]采用随机游走策略从告警序列中采样, 训练 Word2Vec 模型并提取告警的语义信息. 阴振生等人[18]则使用表示学习技术获取告警在文本和拓扑角度的向量表示, 并基于流式聚类技术[19]实现告警压缩, 有效减少告警数量. 王维靖等人[20]提出了 NAP 方法, 通过使用预训练语言模型 ChineseBERT[21]结合 GRU 神经网络对告警文本进行编码并输入排序模型进行告警排序. 相比之下, 有监督学习方法则在无监督学习的基础上, 结合人工标注的历史告警聚合结果, 通过模型学习告警之间的关联性, 使得语义上相关的告警聚合效果更好. LiDAR[9]提出了基于图卷积网络的孪生编码网络, 将告警文本映射到更高维的语义空间中, 以提高语义信息的表达和聚合的准确性. 以上研究的卓越效果表明, 结合深度学习和图神经网络的语义信息提取技术在告警聚合中起到了至关重要的作用, 能够有效增强告警聚合的准确性和可靠性.

此外, 基于异常指标的告警聚合方法也得到了广泛的研究. 由于指标数据的时序性和波动性, 异常指标的特征提取通常较为复杂, 难以直接与告警文本结合进行聚合. 因此, 当前的研究主要利用异常指标进行辅助聚合, 缩小聚合范围, 简化聚合任务. 例如, GRLIA[8]使用极致理论[22] (EVT 算法) 利用告警数量对每个服务节点进行异常检测, 并通过动态时间规整[23] (DTW 算法) 对每两个异常服务节点计算指标趋势相似度, 以此判断两个异常服务节点是否属于同一故障影响范围, 进而将整个服务节点依赖图划分为多个故障影响图, 告警聚合范围则被缩小到各个故障影响图中. Zhao 等人[24]使用基于 LSTM 模型[25]的时间序列异常检测方法来提取指标序列的特征, 并进一步通过有监督的方式结合告警文本特征训练自定义排序模型, 实现告警的重要性排序. 此外, 部分告警聚合工作也尝试以指标波动刻画异常在系统中的动态传播过程, 并以指标相似度为切入点进行告警聚合. DyAlert[26]中就创新性地使用异构动态图模拟在故障的传播过程中系统异常指标和告警之间的动态交互过程, 其通过计算皮尔森相关系数来评估异常指标节点之间的相关性, 并使用有监督的方式来综合告警文本信息与相关指标信息生成告警的最终表示, 进而以端到端的方式预测告警间是否存在关联关系.

利用应用拓扑结构对于告警之间的关系进行建模也是当前告警聚合技术的一大研究热点. 最常用的拓扑结构是告警对应服务节点间的依赖关系图. 然而, 在复杂企业系统中, 服务之间的真实调用关系错综复杂, 服务依赖图常常是稠密且难以直接处理的, 为后续利用机器学习等技术进行建模带来困难. 因此, 如何从复杂的服务调用图中提取有用信息进行告警聚合成为了一个重要课题. GRLIA[8]通过计算告警重叠度和指标相似度来为服务节点间的依赖关系分配权重, 并使用 Louvain 算法[27]进行模块划分, 从而将庞大的服务依赖图拆分为多个故障影响子图. 此外, 为了增强服务节点之间的关联性, 减少无关节点的影响, LiDAR[9]中直接以历史上两个服务节点对应多个告警中是否有任意一对被人工关联过为依据建立服务依赖图, 确保服务依赖图中存在边的两个服务节点间必然存在某种关联关系. 此外, 部分工作也尝试挖掘告警相关其他拓扑结构. COT[28]中对于告警提取告警模板, 利用历史告警关联标签来构建历史告警模板关联图, 并结合服务依赖图通过有监督学习的方式进行告警对应根因服务预测. DyAlert[26]进一步对于指标与指标之间的关系、指标与告警之间的关系进行探索, 通过构建动态异构图来加强拓扑结构的表达能力.

在现有的告警聚合研究中, 虽然已经探索了多种方法来结合告警文本、指标数据和应用拓扑结构, 但大多数方法依然存在对告警关联性的提取过于依赖单一数据类型的问题, 忽视了在告警产生的过程中故障相关底层系统数据的多样性与复杂性, 而这些底层系统数据正是实现以同一故障为聚类目标的有效聚类方法的关键所在. 与现有方法相比, 本文方法通过构建层次化的系统数据依赖图, 将告警与底层系统数据的关联关系进行深度挖掘, 能够更精准地识别不同底层系统数据对告警特征的贡献程度. Alert-CM 利用图神经网络与图表示学习, 不仅考虑告警文本的语义信息, 还综合了多种底层系统数据的依赖关系, 实现了对告警的高效聚类. 通过这种方法, Alert-CM 能够更好地捕捉到告警与核心系统数据之间的紧密关联性, 实验结果表明, 本文提出的聚合方法显著提升了告警管理的精度和效率, 为复杂业务系统的稳定运行提供了更为有力的保障.

2 基础知识

本文针对告警管理过程中与告警相关的多种系统数据构建系统数据依赖图, 并提出了基于图表示学习的告警聚类方法, 下面就相关概念和基本知识予以介绍.

2.1 告警管理过程

在现代大型企业中, 为了实现复杂系统中异常情况的实时上报与服务故障的及时修复, 告警管理[3,4]成为不可或缺的一环. 告警管理过程在真实企业实践中一般包括监控指标配置、预警规则设置与汇总、应急场景分配、异常告警上报以及告警处理等一系列步骤. 下面就对这些步骤进行详细阐述.

2.1.1 监控指标配置

监控指标配置是告警管理的第 1 步. 在这一阶段, 系统工程师依托经验, 在告警系统中配置能够有效反映系统异常的各项指标. 根据监控内容的不同, 这些指标可分为系统指标和业务指标. 系统指标是用于衡量和评估应用程序性能、稳定性和资源使用情况的技术指标, 如 CPU 利用率、内存使用率等. 业务指标则更倾向于描述与业务场景相关的服务功能是否正常, 如请求提交成功率、页面加载响应时间等. 本文则主要针对配置更加复杂的业务指标进行研究.

在企业系统中, 业务指标的提取往往通过分析系统生成的日志来实现, 这一过程涉及日志解析、信息提取和特定指标计算等步骤. 在实际操作中, 业务指标的计算规则主要基于工程师的经验, 并在生产环境中经过长期的验证以确保准确性与稳定性. 具体来说, 通过解析相关日志行列, 可以对不同业务指标的行为特征进行建模. 特别地, 若两个指标源自同一日志文件的相关行列, 则二者在系统行为上的表现具有较高的相关性, 且可能指示出相似的系统异常状态. 在配置完成后, 告警系统基于设定的时间窗口对这些指标进行持续监控, 实时分析系统的健康状况与运行风险, 以确保在异常发生时及时响应.

2.1.2 预警规则设置与汇总

在监控指标配置完成后, 系统工程师需要定义一系列预警规则, 这些规则用于触发告警并汇总系统的异常状态. 考虑到系统状态的波动, 异常状态往往需要多个指标的数据来佐证, 因此, 一条预警规则通常与多个指标相关联, 并在规则名称中对异常状态进行概述. 在预警规则中, 系统工程师会根据经验为每个相关指标制定特定的触发条件, 触发条件包括对比方式和异常阈值. 对比方式是可选的, 其主要目的是通过将当前指标值与特定时间或时间段的指标值进行对比来更准确地表明异常程度. 例如, 常见对比方式有“当前之间的值与上周同比上升超过 20%”“最近 2 min 求和与上 2 min 求和的环比上涨超过 35%”等. 异常阈值则用于确定指标波动是否达到异常标准, 并且不同的阈值往往与不同的告警严重程度、告警上报策略等关联.

在企业实践中, 系统工程师往往会将多个相关的预警规则汇总到一个监控项中, 用以反映同一业务功能点的异常, 并且在监控项的名称中指明业务功能. 例如, 多个监控项可以汇聚成“酒店订单提交异常”的监控项, 确保能够全面描述订单提交过程中的异常状态. 因此, 监控项是告警管理过程中描述系统业务功能点是否正常的基本单位.

2.1.3 应急场景分配

为了进一步精确描述复杂企业系统中多个业务功能点之间的关联性与协作关系, 工程师团队会进一步分析业务功能的关键需求、提取多个复杂的粗粒度的业务场景, 拆解出多个相关的业务功能点, 并且将对应的监控项关联到此业务场景下, 从而构成应急场景. 因此, 关联到同一个应急场景的监控项往往服务于同一个业务功能, 并且更倾向于在此应急场景相关的异常发生时同时触发告警. 一个应急场景中一般包括应急场景名称、相关应用、多个关联监控项、场景生效时间等. 如后文表 1 所示, “酒店订单详情页请求成功量”“酒店订单提交成功量”“酒店订单支付成功率”这 3 个监控项被关联到“酒店交易订单异常故障场景”中, 描述的是与酒店订单提交相关的业务功能, 并且前两个监控项与酒店订单服务“order”相关, 后一个监控项与酒店订单支付服务“payOrder”相关, 此应急场景的生效时间是全天.

2.1.4 异常告警上报

在完成了指标配置和预警规则设置后, 告警系统会在指标异常并且触发预警规则时以文本形式生成并上报告

警. 告警管理系统会根据预先设定的告警模板收集异常时的相关系统信息并进行总结, 从而构成告警内容. 常见的告警内容一般包括告警标题、创建时间、告警概述、相关指标、相关应用、严重程度以及告警收敛次数等. 如表 2 所示, 根据应急场景和告警标题, 此告警表明的是酒店提交订单相关功能的异常, 其相关的异常监控项为“酒店提交订单相关错误排查”, 相关的异常指标共有两个, 分别为“订单提交成功率”和“订单提交成功量”, 对应的触发条件为“成功量当前时间的值与昨日同比下跌超过 50%”“成功量当前时间的值与上周同比下跌超过 50%”和“成功率最近 1 min 持续小于 95%”, 这 3 个条件均满足时即触发了预警规则“成功量下跌 1 min”. 此告警仅与应用 order 相关, 并且在短时间内重复告警了 5 次, 异常程度达到了 P3 级别. 告警生成之后会按照异常程度根据上报策略进行上报, 并将被汇总到值班工程师处进行统一处理与分派.

表 1　应急场景示例

字段	内容
应急场景名称	酒店交易订单异常故障场景
相关应用	order、payOrder
关联监控项1	酒店订单详情页请求成功量
关联监控项2	酒店订单提交成功量
关联监控项3	酒店订单支付成功率
场景生效时间	任意时间生效

表 2　一条酒店订单提交相关的告警示例

字段	内容
告警标题	酒店提交订单相关错误排查_成功量下跌1 min
创建时间	2023-10-16 15:19
监控项	酒店提交订单相关错误排查
告警概述	[酒店提交订单监控项 成功量下跌1 min] 成功量当前时间的值与昨日同比下跌54.55% 成功量当前的值与上周同比下跌52.41% 成功率最近1 min持续小于95%
相关指标	订单提交成功率、订单提交成功量
相关应用	order
严重程度	P3
收敛次数	5
应急场景	酒店交易故障场景、酒店日历房交易下跌

2.1.5　告警处理

告警处理环节即为值班工程师根据告警内容进行初步诊断与分派, 随后将故障定位与解决的责任分配给相应的工程师. 故障修复后, 工程师通常会进行故障复盘, 分析根本原因并总结经验, 以防类似问题再次发生.

3　基于图表示学习的告警聚类方法

为了实现对告警的准确聚类, 本文提出了一种基于图表示学习的告警聚类方法. 如图 1 所示, 本文提出的方法主要分为 3 个阶段进行, 第 1 阶段为系统数据依赖图构建和与告警间的依赖关系提取, 第 2 阶段为基于轮廓系数的图表示学习模型训练, 第 3 阶段为告警聚类.

• 系统数据依赖图构建和与告警间的依赖关系提取. 对于稳定的告警系统, 这一阶段提取系统内组成告警的不同层次的底层数据节点, 分析同一层次内及不同层次间节点的关联关系, 构建出代表告警系统底层数据状态的异构多层次系统数据依赖图. 在此基础上, 进一步提取告警与系统数据依赖图中各节点间的关联关系. 多个告警如果存在关联或属于同一聚类, 其相关性将在底层数据节点中以不同程度的方式表现出来. 通过这种方式, 系统数据依

赖图为后续告警特征描述与告警聚类提供了强大的结构化数据支持.

• 基于轮廓系数的图表示学习模型训练. 基于系统数据依赖图和提取出的节点依赖关系, 本阶段搭建图神经网络模型进行图表示学习. 图神经网络能够有效捕捉节点之间的复杂依赖关系, 并将这些关系映射到低维空间中, 以生成节点的特征向量. 在此过程中, 本文采用基于轮廓系数的损失函数, 旨在优化节点之间的相似度, 使得同一聚类内的节点特征向量尽可能接近, 而不同聚类的节点特征向量之间尽可能远离. 本模型提供无监督和有监督两种训练方式, 能够灵活适应不同数据场景. 在有监督的模式下, 本文通过利用历史告警的聚类信息来引导模型训练, 使得图神经网络能够更精确地学习到告警与底层数据之间的依赖关系, 生成更加准确的告警特征向量. 这一阶段中的模型训练过程是离线的.

• 告警聚类. 对于每个告警, 经过训练的图表示学习模型将输出一个对应的准确特征向量. 在此基础上, DBSCAN 算法根据当前时间窗口内的所有告警向量进行在线聚类, 其能够有效识别密度相似的告警群体, 从而得到最终的实时告警聚类结果.

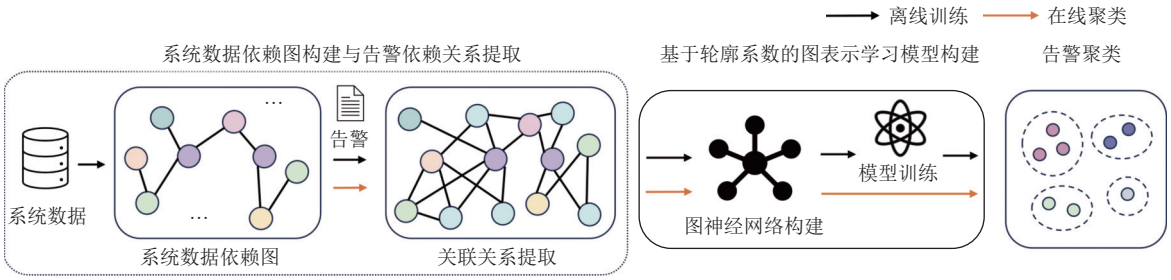

图 1 Alert-CM 流程图

3.1 系统数据依赖图构建和与告警间的依赖关系提取

3.1.1 系统数据依赖图构建

为了更系统化地建模告警数据, 本文首先构建了一个层次化的系统数据依赖图. 如图 2 所示, 图中的节点包括 7 种类型, 节点间的关联关系反映了业务系统中的多层次依赖.

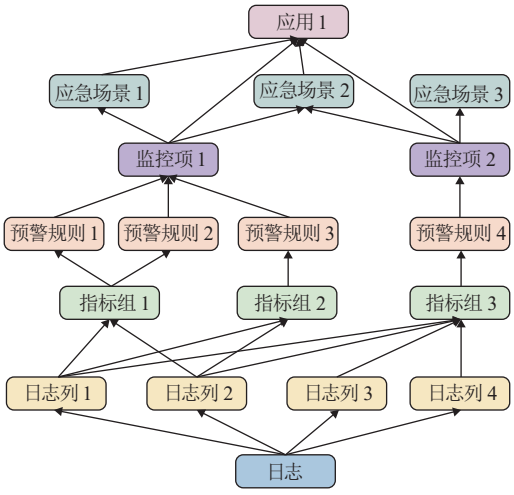

图 2 系统数据依赖图

(1) 监控项节点. 监控项是用于监控业务系统内某一个业务需求是否异常的监控条目. 一个监控项下往往会配置一条或多条预警规则, 一旦某一条预警规则达到异常条件则会产生告警. 每个监控项均具有监控项名称、关联

应用、所属产品线等属性.

(2) 预警规则节点. 每条预警规则中包含监控指标与预警触发条件两个部分. 预警触发条件会进一步指明指标异常的判定方法, 包括对比方法与异常范围, 如“成功量当前时间的值与上周同比下跌 50%”“总量最近 5 min 求和与上上周同比下跌 60%”等, 多个预警规则可以共用同一指标组节点, 反映对某一组指标的不同异常判断. 同时, 每条预警规则中也包含预警规则名称、生效时间等属性, 每条预警规则均会对应到某个监控

(3) 指标组节点. 业务监控中的指标均是由开发人员自定义配置并从对应日志中实时采样并计算得出的, 较好地反映了系统从开发角度的业务功能关注点, 使得监控系统可以有效捕获系统异常状态. 如图 3 所示, 开发人员在配置指标的过程中会指定采样的日志文件及其中对应的日志列, 并从日志行筛选规则设定、列值分组规则设定、统计列与统计方法选择这 3 个方面进行设置. 日志行筛选即为从日志文件中筛选出同时满足筛选规则的日志行, 每一条筛选规则设定日志列及对应满足条件的列值. 列值分组步骤将筛选出的日志行根据多条分组规则进行分组, 每条分组规则中设置日志列及用于分组的列值, 相当于 Pandas 中的 group_by 操作. 经过列值分组步骤之后, 原先的日志行被分割成多个日志行分组. 在此基础之上, 统计列与统计方式选择步骤设定了对于每一个日志行分组的统计方式, 如“求行数”“成功率”“对于列 A 求总和”等.

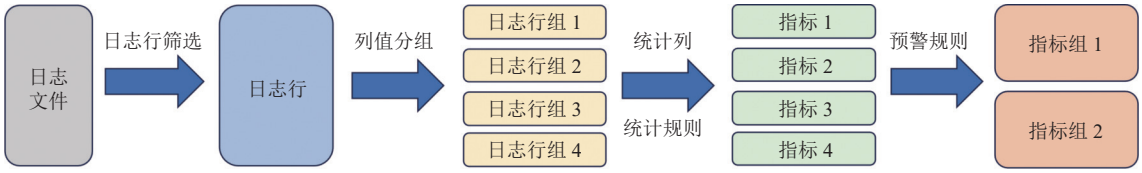

图 3 指标组的生成流程

为了进一步说明指标组的生成流程, 如图 4 所示, 在一个购物应用的初始日志中, 包含 8 条用户活动日志, 涉及 3 个用户、2 种请求类型 (如下单、支付)、2 种状态 (如成功、失败) 以及 3 种商品分类 (如食品、图书、服装). 在日志行筛选步骤中, 通过设定筛选条件为“请求类型=下单”且“状态=成功”, 系统保留了 4 条用户成功下单的相关日志. 接着, 在列值分组步骤中, 按照“商品种类”对这 4 条日志进行分组, 将其分为两组, 每组包含两条记录. 随后, 在统计列与统计方法选择步骤中, 设定统计列为“响应时间”, 统计规则为“求平均值”, 从而计算出每种商品的平均响应时间; 同时, 还可以通过设定统计规则为“求行数”来生成每种商品的下单总数. 通过以上操作, 原始日志文件中的 8 条记录被转化为 4 个关键指标, 包括每种商品的平均响应时间和下单总数. 开发人员通过以上 3 步从日志中提取了多条指标, 每条指标均有其对应的指标名称及相关的日志列、列值信息. 在此基础上, 预警规则对其感兴趣的一条或多条指标进行监控, 则按照相关的预警规则进行聚合后, 一条或多条指标即可形成一个指标组节点, 表明其满足相同的预警需求. 同时, 多条预警规则可以适用于同一个指标组, 实现对于这组指标从不同异常程度的监控.

(4) 日志列节点. 根据指标配置的流程, 日志列节点是日志文件中的多个日志列. 某个日志列节点可能和多个指标组关联, 多个指标组共享某个日志列节点体现了这些指标组节点之间可能存在某种关联关系, 共享日志列节点越多的指标组间的关联可能就越大. 日志列节点包含日志列名称、描述等文本信息, 进一步补充日志数据的业务用途.

(5) 日志节点. 日志节点是日志列节点所属于的日志文件.

(6) 应急场景节点. 为了方便应急人员快速定位故障对应业务域, 开发团队预先在告警管理系统中制定了大量的涉及多个应用的复杂故障场景, 每个故障场景下均配置了一个或多个监控项, 表明这些监控项服务于相同的业务需求并且之间存在较为紧密的关联关系. 每个应急场景节点均包含场景名称、相关应用、相关产品线等属性信息.

(7) 应用节点. 应用节点是系统中存在的真实服务节点, 应用节点与应用节点之间的关系即为 Trace 中存在的服务调用关系. 此外, 应用节点与监控项节点、应急场景节点、日志节点之间均存在依赖关系.

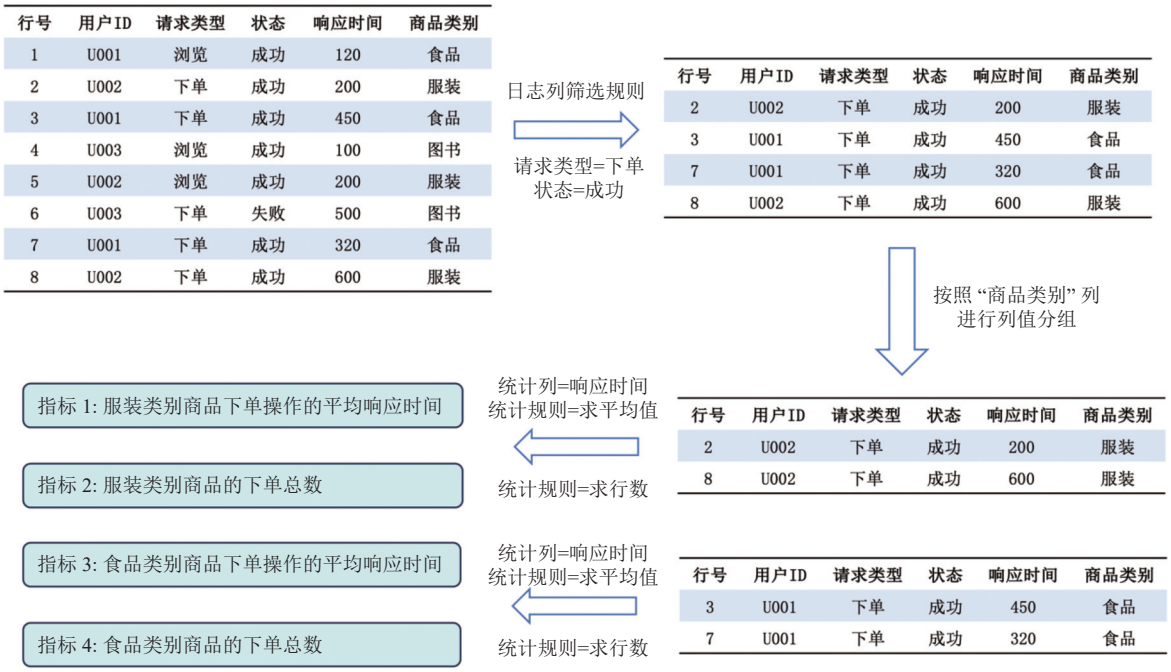

图 4　指标组生成示例

3.1.2　告警与系统数据依赖图中节点间的关联关系提取

当异常发生时, 告警管理系统收集各个异常节点的信息并生成告警. 正如前文所描述的, 每一条告警的文本信息中包括告警标题、对应的监控项、触发的预警规则信息、异常的指标组、相关的应急场景信息、异常发生的应用及对应的产品线信息. 结合系统数据依赖图, 告警文本可以进一步对应到图中的节点上. 如后文图 5 所示, 对于给出的告警示例, 其文本信息中的各组成部分可以进一步在系统数据依赖图中的各个层次中搜索到对应的系统数据节点, 这些系统数据节点相较于原始的告警文本包含更多的系统知识并且相互之间存在关联关系. 同时, 告警示例在去除与系统数据节点重复的冗余信息后被抽象为告警节点, 告警节点与搜索出的系统数据节点之间新建出关联关系, 从而更加立体直观地反映告警对应异常的影响范围.

在完成关联关系提取后, 本阶段对于系统数据依赖图中的节点和告警节点的属性信息进行预处理, 将其由自然语言属性表示转化为可处理的向量表示. 本文按照从前研究的经验将节点属性中的停顿词与非语言词汇移除, 后使用 jieba 分词工具对节点的文本信息进行分词处理, 将节点属性中的长句转化为多个词语的组合, 并构建全局词语集合. 在此基础之上, 本文使用 FastText 模型[15]对于提取出的词语集合进行编码, 将每个词语表示为一个向量, 并通过对词语向量进行加权平均, 获得每个节点的初始向量表示.

3.2　基于轮廓系数的图表示学习模型训练

在经过系统数据依赖图构建和关联关系提取后, 一个时间窗口内的告警聚类任务被转化为获取每个告警节点的准确特征表示和根据告警特征进行聚类两个阶段. 告警特征提取阶段的主要方法步骤如图 6 所示.

在单个告警节点的特征表示中, 既包括其自身属性, 也涵盖与之相关的多个系统数据节点的特征. 告警节点与不同系统数据依赖节点之间的依赖关系强弱不一, 这使得各系统数据依赖节点特征对告警节点特征的贡献程度各异. 此外, 属于同一聚类的告警往往由相同异常引起, 相应的告警节点所关联的底层系统数据节点之间通常具有较强的相关性, 例如, 共享同一节点或存在直接连接边等. 因此, 告警节点的向量表示应更加注重采集这些底层系统数据节点的特征, 以实现更高的相似度. 同时, 告警聚类结果也会反过来影响系统数据节点. 在初始的系统数据依赖图中, 边是无权的, 即所有系统数据节点间的关联强度被视为相等. 然而, 在实际系统中, 核心系统数据节点之间往往表现出更强的交互性, 从而展现出更高的相关性. 因此, 属于同一聚类的告警节点所对应的系统数据节点之间

的距离也应更近. 为了准确识别不同系统数据节点对告警节点影响力的强弱, 并动态调整系统数据节点之间的相关性, 以确保告警节点向量表示的提取精度, 本文提出了一种图表示学习模型 Alert-CM. Alert-CM 利用历史告警聚类结果作为训练数据, 通过持续调整网络参数以实现对节点间依赖关系的更精准提取, 从而更准确地对生成未来时间窗口的告警特征向量.

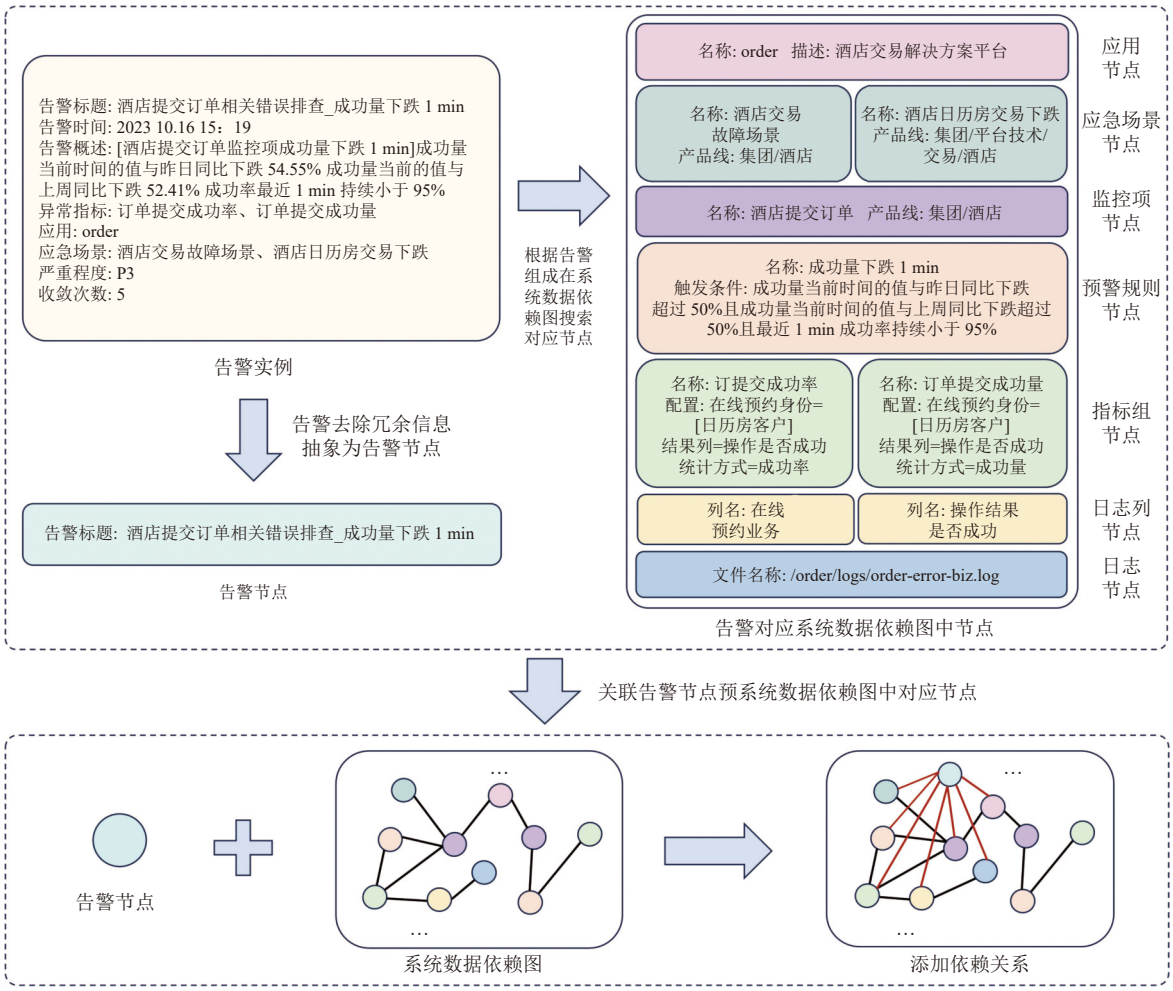

图 5 告警对应到系统数据依赖图中节点示例

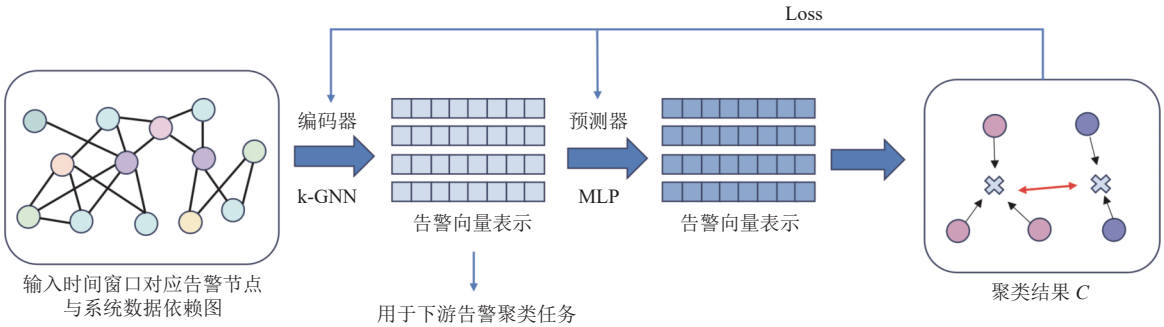

图 6 Alert-CM 中图表示学习模型的训练流程

在特征提取阶段, Alert-CM 采用异构图神经网络[29–31]编码器 k-GNN[32]作为核心组件. k-GNN 的设计理念结合了图神经网络的基本原理, 通过信息传递机制来有效捕捉节点间的结构关系. 针对系统数据依赖图中、告警与系统数据节点间存在的多种关联关系, k-GNN 配置了多组卷积核, 每组卷积核专门用于提取特定类型关系的特征. 这种设计使得模型能够针对不同的邻接关系进行灵活而细致的特征提取. 在特征聚合过程中, k-GNN 不仅考虑了每个节点自身的特征, 还整合了其邻居节点的特征, 从而在每一层中更新节点的向量表示. 这一过程遵循了 GNN 的层级传播机制, 通过多次迭代, 模型逐步捕捉到节点间的深层次依赖关系. 在聚类任务中, k-GNN 最终将聚焦于告警节点, 输出其经过多层特征聚合后的向量表示, 这一向量既反映了告警节点自身的特性, 也融入了其与相关系统数据节点间的关系特征. 其次, 模型使用一个 MLP 层作为预测器来对告警节点的向量表示进一步进行特征提取, 生成最终告警向量表示. 通过这种方式, Alert-CM 能够充分利用图结构数据的丰富性, 提升告警特征表示的准确性与有效性, 为后续的分析和聚类提供坚实基础.

在训练阶段, Alert-CM 在获取模型输出的告警向量表示后, 提供了无监督和有监督两种可选的训练方式, 以灵活适应不同的应用场景和数据需求. 具体而言, 在无监督训练模式[33,34]下, Alert-CM 无需依赖人工标注的历史聚类结果, 而是通过 DBSCAN 算法对模型输出的告警向量进行自动聚类, 随后利用这些聚类结果反向指导模型的训练过程. 在这一模式中, DBSCAN 算法基于告警向量在特征空间中的密度分布, 自主识别告警的聚类结构, 从而为训练过程提供动态反馈. 无监督训练模式尤其适用于标注数据不足或难以获取标注结果的实际应用场景, 其优势在于灵活性和适应性较强. 然而, 由于缺乏明确的监督信号, 模型输出的告警向量表示在初始阶段可能存在较大的准确性偏差. 因此, 这一模式更适合于那些属于同一聚类的告警在原始文本或特征表示层面本身具有较高相似度的情况. 在这种情况下, 无监督模式能够逐步优化模型对告警的表示能力, 提升聚类结果的准确性. 相较之下, 在有监督训练模式下, Alert-CM 利用历史告警的聚类结果作为监督信号, 直接指导模型的训练过程. 通过将历史数据中的告警聚类结果作为标签, Alert-CM 能够有效学习告警向量与底层数据之间的复杂依赖关系, 从而生成更贴合实际分类需求的高质量特征向量表示. 这一模式适用于已积累大量标注数据的场景, 其显著优势在于能够大幅提升模型的分类准确性和泛化能力. 特别是在大型在线服务系统中, 告警通常具有复杂的关联性与层次性, 有监督模式通过利用高质量的标注数据能够更好捕捉真实告警特征.

受到 CARL-G[35]的启发, Alert-CM 根据不同的训练模式, 将一个时间窗口内的告警划分至对应的关联类, 并为图中每个告警节点计算其轮廓系数 (silhouette index)[36–38], 轮廓系数衡量了簇内距离与其最近相邻簇的簇间距离之比, 返回值范围为 $[-1,1]$. 对于图 $G=(V,E)$, V 是节点集合, $|V|=n$ 为节点数目, $E\subseteq V\times V$ 是边的集合. 设 C 为聚类集合, $C_i\subseteq V$ 表示第 i 个聚类中的节点集合, 且 $c=|C|$ 为聚类的数量. 为了简化符号表示, 设 $U\in[1,c]^n$ 为聚类分配集合, 其中 U_u 表示节点 u 的聚类分配. 对于图中的每个节点 u, 其均具备对应的向量表示 h_u. 如公式 (1), 轮廓系数为当前时间窗口中每个告警节点 u 的轮廓系数 $s(u)$ 的平均值.

$$Sil(C)=\frac{1}{n}\sum_{u\in V}s(u) \tag{1}$$

其中, 如公式 (2), $s(u)$ 由凝聚度 $a(u)$ 和分离度 $b(u)$ 共同计算得到. 具体来说, 对于一个聚类中的告警节点 u, 如公式 (3), 凝聚度 $a(u)$ 表示它与同一聚类中其他告警节点之间的平均距离, $a(u)$ 越小, 表示 u 越接近其同聚类中的其他节点. 而分离度 $b(u)$ 则如公式 (4), 被定义为节点 u 与距离它最近的其他聚类中节点的平均距离, $b(u)$ 衡量了节点 u 与最接近的异类节点之间的距离.

$$s(u)=\frac{b(u)-a(u)}{\max\{a(u),b(u)\}} \tag{2}$$

$$a(u)=\frac{1}{\left|C_{U_u}\right|-1}\sum_{v\in\left(C_{U_u}-\{u\}\right)}DIST(h_u,h_v) \tag{3}$$

$$b(u)=\min_{i\neq U_u}\frac{1}{|C_i|}\sum_{v\in C_i}DIST(h_u,h_v) \tag{4}$$

根据以上分析, 较大的轮廓系数表明同一个聚类内的样本相异度较小, 不同聚类之间的样本相异度较大, 聚类

结果的可区分性越强. 因此轮廓系数越大, 当前聚类结果越好. 因此, Alert-CM 的训练思路为使得每个时间窗口的轮廓系数更接近最佳值. 由于轮廓系数的理论最佳值为 1, 基于轮廓系数的损失函数定义如公式 (5) 所示. 轮廓系数的计算过程中融合了对比学习的思想, 基于轮廓系数的损失函数可以较好地拉近属于同一个聚类的相似样本之间的距离, 同时将不属于同一个聚类的样本尽可能远离. 完成损失计算后, 模型通过反向传播更新编码器和预测器的参数, 进而更新系统数据节点的特征表示, 使告警节点的表示更加准确地迭代更新.

$$L_{\text{sil}} = |1 - Sil(C)| \tag{5}$$

3.3 告警聚类

在训练阶段完成后, Alert-CM 将通过 k-GNN 编码器学习得到的告警节点向量用于后续的告警聚类任务, 同时舍弃预测层以避免过拟合并提高模型的泛化能力. 告警聚类任务通常无法预先确定聚类簇的个数, 因此, 本文采用了 DBSCAN 聚类方法, 该方法能够自动识别聚类数, 并根据数据的分布自适应调整聚类结构.

在在线告警聚类任务中, 当一个新的告警产生时, Alert-CM 会实时提取该告警与系统数据依赖图中各节点的关联关系, 并将其输入到已经训练好的模型中, 以输出该告警对应的特征向量. 该特征向量会与当前时间窗口内所有告警的特征向量一起, 作为 DBSCAN 聚类方法的输入. DBSCAN 根据这些特征向量对告警进行聚类, 自动识别告警之间的关系并输出聚类结果. 最终, 聚类结果将被及时发送给值班工程师, 为其提供告警分派和故障诊断的依据. 通过实时聚类, Alert-CM 能够有效地减轻工程师的工作负担, 并确保故障的及时定位和处理.

4 实验分析

为了评估方法的有效性, 本文围绕以下 4 个问题开展了实验验证, 并与相关方法进行了对比.

- RQ1: 本文方法在在线服务系统中告警聚类任务的有效性如何?
- RQ2: 本文方法在模型训练和在线服务系统中告警聚类任务中的效率如何?
- RQ3: Alert-CM 中系统数据依赖图内不同类型的节点对于告警聚类任务有效性的贡献如何?
- RQ4: 相关超参数对实验结果有何影响?

4.1 实验设置

本文使用 Python 3.7、PyTorch 1.13.1[39]和 PyTorch Geometric 2.3.1[40]来实现模型. 所有实验均在配备了 Intel Xeon Platinum 8163 处理器、368 GB 内存以及 32 GB GPU 内存的 Tesla V100 服务器上进行. 模型的超参配置如下: 图中节点的特征向量的维度为 256. 异构图 k-GNN 网络的隐藏层层数为 2, 卷积核个数为 14 个分别对应异构图中的每一种边. MLP 层的隐藏层维度为 512, 用于捕获更加复杂的告警信息. 后续使用 DBSCAN 方法进行聚类的邻域半径大小为 0.4, 最小样本数为 1.

4.1.1 实验数据集

本文使用的告警数据集来自阿里巴巴控股集团的真实工业环境, 其中涉及 10 个主要服务产品线, 其中包含超过 600 个服务应用, 涉及约 2 000 个监控项、1 500 个应急场景、4 900 条预警规则与 4 300 个有效指标组. 基于此工业环境搭建的系统数据依赖图中拥有共计 16 146 个节点与 44 114 条边. 本文收集了 2023 年 10 月 1 日–11 月 1 日的共计约 5 000 条告警. 根据应急工程师的处理经验, 该段时间故障发生更加频繁, 故障造成的影响面较大, 为告警聚类任务提供了较为复杂的系统环境. 在统计了历史告警处理过程中人工打标的告警聚类组的平均时间跨度平均值与相邻告警的时间间隔的平均值基础之上, 本文选择设定时间窗口的跨度为 2.5 h, 时间窗口移动步长为 30 min. 在去除不包含告警或仅仅包含 1 个告警的时间窗口, 告警数据共计被划分到 1 388 个时间窗口中. 时间窗口中告警数量的最小值为 3 个, 最大值为 160 个, 平均值为 19 个.

对于有监督方法而言, 告警聚类标签的准确性和可靠性对于方法效果至关重要. 本文的数据收集自实际系统, 为了进一步补全值班工程师在处理紧急情况时对于告警聚类的遗漏, 本文邀请了 4 名阿里巴巴控股集团的资深工程师手动分析时间窗口中告警间的关联性并将其进一步分为多个聚类. 这一过程旨在确保数据的合理性, 从而为后续分析提供坚实基础. 在此基础之上, 本文将告警数据集分为训练集和测试集两部分, 为了提高模型的普适性和

避免过拟合, 本文根据时间窗口的起始时间进行划分, 使用 2023 年 10 月 1 日–20 日的时间窗口作为训练集, 2023 年 10 月 21 日–31 日的时间窗口作为测试集.

4.1.2 评估指标

由于本文的任务是针对告警进行聚类, 因此本文选用聚类任务在监督学习场景常用的 NMI 与 ARI 指标来评估聚类结果与真实标签之间的相似度.

• NMI (normalized mutual information)[41]: 是基于互信息的概念, 其衡量聚类产生的簇划分跟真实标签之间的信息共享程度. NMI 的值域为 $[0,1]$, 一个高 NMI 值意味着聚类结果与真实标签高度吻合.

• ARI (adjusted Rand index)[42]: 兰德指数 (RI) 的改进版本, 其校正了随机性, 并调整了 RI 在不同聚类场景下不提供常量基线的问题. ARI 的值域为 $[-1,1]$, ARI 值接近 1 表示聚类结果与真实分类高度相似, 而接近 0 的值表示聚类效果与随机聚类相当, 负 ARI 值表明聚类性能低于随机性, 通常表示数据被错误划分到其他聚类中.

4.1.3 对比方法

• Alert-CM unsupersived: Alert-CM 的无监督版本, 采用自监督的方式进行训练. Alert-CM unsupersived 在经过图神经网络得到告警对应的向量表示后采用 DBSCAN 方法进行聚类, 并将聚类结果作为损失函数的输入.

• DBSCAN: 基于密度的聚类算法, 输入为一个时间窗口中的所有告警的特征向量, 输出为多个聚类后的告警簇结果. 在直接使用 DBSCAN 进行聚类作为对比方法时, 告警仅基于其文本内容利用 jieba 进行分词并基于 FastText 模型得出特征向量.

• LiDAR[9]: 构建了一个用于识别告警事件关联的有监督学习框架. 该框架采用深度学习模型从历史上人工关联的事件的文本描述中学习相似的语义表示. 同时, 其基于历史事件关联性来构建组件依赖网络并实施 node2vec 算法[43]来对告警事件的结构信息进行编码. 对于未来产生的告警事件, LiDAR 在提取语义信息和结构表示后, 通过余弦距离计算事件之间的语义相似性和结构相似性, 并与预定义阈值相比较来实现事件关联的识别.

• UHAS[11]: 在识别告警风暴的基础之上使用无监督的方式进行告警聚类任务. 其从文本相似度和拓扑相似度两个维度两个告警事件之间的相似度. 文本相似度通过计算两个告警文本在通过词袋模型进行分词之后的 Jaccard 距离来计算. 在获取系统的服务调用图的基础之上, 拓扑相似度为计算两个告警对应服务在图上的最短路长度来表示. 最终, 在得到时间窗口中每两个告警事件之间的相似度构成相似度矩阵之后, UHAS 使用 DBSCAN 算法输出聚类结果.

• GRLIA[8]: 一种基于图表示学习的无监督告警事件聚合框架. 该框架通过以下流程实现告警聚合: 首先, 结合告警事件与系统监控数据 (如关键性能指标 KPI) 构建故障影响图; 其次, 利用 Deepwalk[44]和 Word2Vec 算法融合告警的文本信息与拓扑结构, 生成告警的向量表示; 最终, 通过计算向量间的相似度来判定告警事件的相关性.

以上 5 种方法中, LiDAR 和 GRLIA 的主要目标是进行告警事件的关联. 为了更好地适配本文的告警聚类任务, 对于这两个方法, 本文在对于告警关联结果的基础之上利用并查集进行聚类, 将处理之后的结果作为最终聚类结果.

4.2 与现有方法在告警聚类有效性方面的对比 (RQ1)

后文表 3 展示了 Alert-CM 与其他对比方法在前文提到的数据集上的实验效果. 第 1 列为方法名称, 第 2 列为训练方式, 第 3 列为各个方法输出结果的 NMI 指标值, 第 4 列为 ARI 指标值. 正如表 3 中所示, Alert-CM 在 NMI 和 ARI 指标方面均表现优异, 分别达到了 0.901 和 0.645, 显著超过现有的对比方法. 具体来说, Alert-CM 相比现有方法 (LiDAR、UHAS、GRLIA) 的平均 NMI 提高了 31.7%, ARI 提高了 153.9%. 同时, 与 Alert-CM 的无监督训练版本方法相比, Alert-CM 在 NMI 和 ARI 指标上分别提高了 17.2% 和 164.3%.

4.2.1 案例分析

为了深入理解 Alert-CM 方法的工作机制并解释上述实验结果的合理性, 本文首先选取了一个真实工业环境中由应急工程师人工聚类的告警组案例进行研究, 以更加清晰地识别引导告警聚类的隐藏关键要素, 并论证 Alert-CM 中融入的告警相关底层系统数据对于提升告警聚类任务有效性的原理.

表 3 不同方法的告警聚类有效性比较

方法	训练方式	NMI	ARI
Alert-CM	有监督	0.901	0.645
Alert-CM unsupervised	无监督	0.769	0.244
DBSCAN	无监督	0.525	0.149
LiDAR	有监督	0.813	0.481
UHAS	无监督	0.487	0.093
GRLIA	无监督	0.753	0.189

表 4 展示了一组人工聚类的告警实例, 包含两个聚类: 一是描述火车票订单处理流程异常的告警 A、B、C、D, 二是描述会员订单相关业务异常的告警 E. 其中, 告警 A 和 E 来自应用 P1, 告警 B 和 C 来自应用 P2, 告警 D 来自应用 P3. 应用 P1 与 P3、P2 与 P3 之间存在调用关系.

表 4 一组人工聚类的告警实例

告警	时间	告警标题	告警文本描述	聚类结果
A	2023/12/12 10:05	火车票提交订单异常	[12-12 10:05 火车票提交订单_成功量下跌 1 min] 成功量 最近 3 min求和与昨天同比下跌73.75%>50% 当前值为 79; 成功率 当前1 min持续小于 95 当前值为 83.16; 应用: P1	聚类 1
B	2023/12/12 10:07	火车票支付成功异常	[12-12 10:07 火车票支付成功量_成功量下跌] 成功量 最近 3 min求和与上3 min求和的环比下跌 55.10%>50% 当前值为 273; 成功量 最近 3 min 与昨天同比连续下跌 69.04%>30%当前值为74; 应用: P2	聚类 1
C	2023/12/12 10:08	火车票出票异常	[12-12 10:08 火车票出票成功率_出票成功量 3 min_白天] 成功量 最近 3 min求和与昨天同比下跌 36.94%>0% 当前值为 367; 成功量 最近 3 min求和与上周同比下跌 31.27%>20% 当前值为367; 应用: P2	聚类 1
D	2023/12/12 10:10	各行业生单成功率下跌异常	[12-12 10:10 创建订单不分业务_成功率下跌 2 min] 成功率 最近 2 min持续小于 85 当前值为2.08; 应用: P3	聚类 1
E	2023/12/12 10:15	会员订单优惠信息查询异常	[12-12 10:15 会员系统订单提交折扣错误_成功量下跌 3 min] 成功量 最近 3 min求和与昨天同比下跌 25.74%>15% 当前值为 167; 应用: P1	聚类 2

现有告警聚类工作常利用告警文本、相关拓扑关系等数据来刻画告警特征并基于此进行聚类. 然而, 这些信息不足以有效地区分真实工业场景中多个告警是否来源于同一故障, 容易造成误判或遗漏. 主要原因包括以下两个方面.

• 告警文本相似度的局限性: 由于业务系统的复杂性和告警描述缺乏统一标准, 属于同一聚类的告警在文本描述层面并不总是相似. 例如, 表 4 中的告警 A、B、C 虽然均与火车票业务相关, 但其关键的服务功能描述并不相同, 而是形成更深层的逻辑关系, 如订单的 “提交-支付-出票” 流程. 此外, 告警 D 与告警 A、B、C 在文本层面的相似度较低, 且相似部分主要来自告警模板中的通用性文字 (如 “成功率”“同比”“环比” 等). 因此, 告警文本层面的相似度对于告警相关性的贡献是不稳定且有限的. 告警聚类任务不能仅仅依赖文本相似度, 而需要进一步挖掘告警文本在逻辑层面的语义相似度, 同时排除通用描述性文字对相似度的干扰, 精准提取对告警对应服务功能起关键作用的词汇.

• 应用拓扑结构的复杂性: 在复杂的业务系统中, 告警对应应用的拓扑结构往往过于复杂, 服务调用图非常稠密. 根据调研, 在本文所使用的某国内大型在线业务系统中, 约 80% 的应用对间的距离在 5 跳之内. 因此, 现有工作中使用告警对应应用间的最短路径长度或自监督模型提取应用特征的方法粒度过大, 往往会夸大告警在拓扑角度的相似度. 此外, 在复杂系统中, 一个应用节点通常承载多个较大的业务功能点, 因此, 即使两个告警产生于同一个应用, 其之间也并不一定存在关联. 例如, 表 4 中的告警 A 和告警 E 均产生于应用 P1 且都与订单相关, 但两者并不属于同一个聚类. 因此, 告警聚类任务需要对告警相关系统数据进行更细粒度的分析, 从而更深层次地挖掘属

于同一聚类的告警对应的关键细节特征

基于上述分析可以看出, 仅依赖告警文本或拓扑信息进行告警聚类存在局限性, 容易遗漏与故障相关的关键信息. 以表 4 为例, 通过深入分析底层系统状态发现, 告警 A、B、C、D 所对应的监控项均与同一应急场景相关, 这验证了它们在业务功能层面的关联性. 相比之下, 告警 A 和 E 虽来自同一应用 P1, 但其对应的指标分属不同日志, 表明两者功能相对独立, 关联度较低.

以上示例说明现有告警聚类方法过度依赖单一数据类型, 未能充分系统化利用告警管理过程中与故障相关的丰富底层系统数据. 为此, 本文通过系统分析告警管理流程, 将告警相关系统数据解构为多个细粒度的关键维度(如相关应用、监控项、预警规则、指标、日志和应急场景等), 并建立这些维度间的依赖关系图, 揭示故障传播的完整路径. 这种对底层数据的精细拆解与其间关联关系的建模显著扩展了告警特征空间, 有助于挖掘出更精准的告警特征表示. 同时, 为了识别底层系统数据中的关键信息, 本文提出 Alert-CM, 通过有监督或无监督方式训练图神经网络模型来挖掘不同底层数据节点对告警特征的贡献程度, 从而提升特征生成的准确性, 进而优化告警聚类效果.

4.2.2 与 Alert-CM 的无监督训练版本方法在告警聚类有效性方面的对比

在 Alert-CM unsupervised 中, 图神经网络通过综合系统数据依赖图中与告警相关的节点信息, 计算并输出告警的向量表示. 随后, DBSCAN 方法基于这些向量表示进行聚类, 并将该聚类结果作为损失函数的输入进行优化. 根据前文, 基于轮廓系数的损失函数将会使得同一聚类中告警节点的距离更近, 不同聚类中告警节点的距离更远. 实验表明, Alert-CM unsupervised 在 NMI 和 ARI 指标上的效果优于基准无监督方法 UHAS 和 GRLIA, 分别为 0.769 和 0.244, 显示了其在提取准确告警特征并进行聚类任务的有效性. 然而, 从整体来看, Alert-CM unsupervised 在本文所使用的复杂工业数据集上的表现仍然存在局限性.

相较于 Alert-CM unsupervised, Alert-CM 在 NMI 和 ARI 指标上分别提高了 17.2% 和 164.3%. 这一显著提高充分体现了引入有监督信息对于告警聚类任务的重要性. 在实际复杂工业系统中, 告警和系统依赖图中的节点之间的关系错综复杂, 节点的特征对告警的影响也具有多样性和层次性. 例如, 某些节点可能对告警特征的贡献较强, 而其他节点的影响较弱, 这种依赖关系在不同场景下也会发生变化. 这意味着即便是告警经过图神经网络初步计算得到的向量相似, 也不代表这些告警必然属于同一聚类. 若没有引入有监督的信息来引导聚类模型的学习, DBSCAN 输出的聚类结果可能无法准确反映这些复杂的依赖关系. 这会导致图神经网络在训练过程中未能精确刻画节点与告警之间的真实依赖关系, 进而导致聚类结果与真实的告警分类存在较大的偏差. 因此, 无监督训练模式更适合应用于属于同一聚类的告警在原始文本或特征表示层面本身具有较高相似度的场景.

针对上述问题, Alert-CM 通过引入有监督学习模式, 利用历史告警的聚类结果作为标签, 不仅增强了图神经网络在建模底层系统数据与告警依赖关系方面的能力, 还通过有效利用标签信息引导了模型的优化过程. 这种方式显著提升了聚类性能, 使聚类结果更贴近真实分类, 进一步验证了 Alert-CM 在复杂工业数据集中的优越性.

4.2.3 与现有基准方法在告警聚类有效性方面的比较

在其他现有各类基准方法中, 直接对告警特征向量使用 DBSCAN 聚类是最简单的实现方案. 实验结果显示, 该方法的 NMI 和 ARI 分别为 0.525 和 0.149, 在所有对比方法中处于较低水平. 不过, 由于 DBSCAN 直接处理连续型特征向量, 相比基于离散统计的 UHAS 方法, 其 NMI 和 ARI 指标仍略有提升.

在基准对比方法中, UHAS 和 GRLIA 属于无监督学习方法. 这两种方法均结合了告警文本与其对应应用之间的真实服务调用拓扑图. GRLIA 还引入了告警相关的关键异常指标相似度, 作为增强聚类效果的另一个衡量指标. 与 UHAS 采用简单的词袋模型对告警文本进行分词并基于词语重复度评估文本相似度不同, GRLIA 的优势在于使用 Word2Vec 将告警文本的词汇映射到高维向量空间, 从而更有效地捕获上下文信息, 挖掘文本语义层面的相似性. 因此, GRLIA 的 NMI 和 ARI 指标均高于 UHAS. 然而, 无监督方法在特征提取的灵活性上存在局限, 难以根据任务的不同和变化进行相应调整, 导致 GRLIA 和 UHAS 在处理相似但属于不同聚类的告警时表现不佳, 最终输出的结果往往将不同聚类的告警归类于同一聚类, 从而造成较低的 ARI 值.

与 GRLIA 和 UHAS 相比, Alert-CM 和 LiDAR 为有监督学习方法. 它们利用神经网络模型对告警聚类任务或

关联任务进行建模, 并基于历史告警聚类或关联信息进行训练, 以进一步强化有效特征的提取. 正因如此, Alert-CM 和 LiDAR 的实验效果优于无监督方法. 得益于有监督学习对特征差异的更好捕捉, 这两种方法在 ARI 指标上显示出明显的提升. 具体而言, LiDAR 中的深度神经网络主要聚焦于提取告警文本的语义特征, 而对告警对应应用间的历史关联角度的拓扑关系则相对简单地进行采样, 并采用 node2vec 进行编码, 最终, 告警的特征向量表示为文本相似度与拓扑相似度的加权和, 权重由超参数控制. 相比之下, Alert-CM 不仅扩展了对告警的描述, 纳入应急场景、预警规则、指标和日志等数据, 还通过 k-GNN 将所有数据整合进特征提取过程, 以更有效地调整与综合各类数据对告警节点的特征贡献. 此外, Alert-CM 通过引入预测层来避免模型过拟合, 从而提升泛化能力, 因此 Alert-CM 在 NMI 和 ARI 指标上表现更为优异.

此外, 在 Alert-CM 与对比方法中, UHAS 采用相似度矩阵刻画告警之间的相似关系, 而 Alert-CM、GRLIA 和 LiDAR 则引入特征向量表示告警. GRLIA 和 LiDAR 通过计算告警间的余弦相似度并与设定的阈值比较, 以判断告警之间的相关性, 从而进行聚类合并; 而 UHAS 和 Alert-CM 则使用 DBSCAN 算法自动识别聚类数量并直接输出聚类结果. 相较于基于阈值的比较方案, DBSCAN 的自适应性使其在处理复杂聚类问题时展现出更佳的性能, 在处理聚类问题时表现更佳.

4.3 与现有方法在告警聚类效率方面的对比 (RQ2)

表 5 展示了 Alert-CM 与基准对比方法在训练和测试阶段的平均时间, 其中每个方法的训练过程均为 100 个轮次, 测试时间为对测试集样本取平均值的结果. 由于 UHAS 是基于统计的无监督学习方法, 且不涉及模型训练, 因此不在时间比较的范围内.

表 5　不同方法的告警聚类效率比较 (s)

方法	训练耗时	测试耗时
Alert-CM	2 127.19	0.024
Alert-CM unsupervised	3 170.71	0.027
LiDAR	1 475.69	0.064
UHAS	—	0.323
GRLIA	425.32	0.031

从表 5 中数据可以看出, Alert-CM 的训练时间为 2 127.19 s, 高于 LiDAR 的 1 475.69 s 和 GRLIA 的 425.32 s. 而由于 Alert-CM unsupervised 需要在训练过程中针对模型输出的告警向量表示使用 DBSCAN 进行聚类, 因此其相较于 Alert-CM 花费更长的训练时间. 这一结果符合预期, 原因在于 Alert-CM 使用了更多维度的告警相关数据, 并结合了更为复杂的图神经网络架构对告警特征进行更全面的学习和深度建模, 从而导致了更长的训练时间. 然而, 考虑到 Alert-CM 采用的是离线训练方式, 较长的训练时间在实际应用中是可以接受的, 特别是在需要高准确率和全面特征提取的场景中.

在测试时间方面, Alert-CM 的每个时间窗口的平均测试结果输出时间为 0.024 s, 较 LiDAR 的 0.064 s 和 GRLIA 的 0.031 s 更为高效. 尽管 Alert-CM 的训练时间较长, 但其测试时间在同类方法中表现优异, 能够满足值班工程师对实时聚类输出的需求, 从而有效支持故障诊断与事件分派. 该效率优势使得 Alert-CM 在实际应用中能够平衡准确性与实时性, 提供高效且可靠的告警聚类结果.

4.4 系统数据依赖图中不同类型节点对告警聚类有效性的影响 (RQ3)

为了进一步验证系统数据依赖图及其内部不同类型的节点对于告警聚类任务的有效性和合理性, 本实验针对系统数据依赖图中的节点类型进行消融, 得出基于不同结构的系统数据依赖图的 Alert-CM 模型在告警聚类任务上的有效性结果.

如前文所述, 本文提出的系统数据依赖图中包括应用节点、应急场景节点、监控项节点、预警规则节点、指标组节点、日志列节点和日志节点. 具体来说, 在针对系统数据依赖图中节点类型的消融实验中, 本文采用层次化的消融策略. 需要注意的是, 由于监控项节点作为整个系统数据依赖图构成的入口, 且每个告警都存在对应的监控

项, 监控项节点及相关边将在系统数据依赖图中予以保留, 不参与消融. 消融实验设置如下.

・Alert-CM_only_app: 仅保留应用节点, 模型主要利用告警文本和应用间拓扑关系, 与传统告警聚类方法使用的数据类型一致.

・Alert-CM_no_log: 移除日志列节点、日志节点及相关边, 评估业务系统底层日志数据对告警聚类的贡献.

・Alert-CM_no_metric: 移除指标组节点、日志列节点、日志节点及相关边, 评估指标节点的作用. 由于指标组节点基于日志动态计算得出, 故同时移除底层日志相关节点.

・Alert-CM_no_rule: 移除预警规则节点、指标组节点、日志列节点、日志节点及相关边, 评估系统数据依赖图中所有告警相关底层细粒度数据对于告警聚类的影响. 因预警规则基于指标构建, 故同时移除底层指标组节点和日志相关节点.

和前文所述的实验一致, 消融实验过程遵循完整的方法流程. 所有消融后的方法变体均采用 Alert-CM 的有监督训练模式, 训练轮数为 100 轮, 学习率设为 0.003. 告警聚类有效性评估指标采用 NMI 和 ARI.

表 6 展示了在系统数据依赖图中消融不同类型节点后 Alert-CM 方法在告警聚类有效性方面的评估结果. 由表可见, 随着系统数据依赖图中节点类型和其间依赖关系的减少, NMI 和 ARI 指标均呈现明显的下降趋势. Alert-CM_only_app 变体因仅保留应用节点而效果最为不佳, NMI 仅为 0.640, ARI 仅为 0.352. 这一结果明确表明仅依靠告警文本信息和相关拓扑信息无法提供足量且准确的告警特征来有效完成告警聚类任务, 这种性能的显著下降揭示了告警聚类过程中对多维度信息的依赖性. Alert-CM_no_scenario 变体仅去除了应急场景节点, 其效果下降幅度较小, NMI 基本持平的同时 ARI 下降了 0.062, 说明应急场景节点提供的宏观信息有助于增强方法对细致聚类需求的处理能力, 减少误判与漏判. 此外, 从 Alert-CM_no_log、Alert-CM_no_metric 和 Alert-CM_no_rule 的结果可见, 随着系统数据依赖图中监控项向下底层数据节点的逐层消融, 模型效果呈现逐步下降的趋势. 其中, Alert-CM_no_rule 变体由于去除了监控项节点向下的所有底层系统数据节点而在 3 种方法中表现最差, NMI 和 ARI 分别降至 0.693 和 0.421, 进一步表明告警相关底层细粒度系统数据节点能为告警聚类任务提供更精确的告警特征信息.

表 6　系统数据依赖图中消融不同类型节点后告警聚类有效性比较

方法	消融节点类型	NMI	ARI
Alert-CM	保留所有类型节点	0.901	0.645
Alert-CM_only_app	仅保留应用节点	0.640	0.352
Alert-CM_no_scenario	去除应急场景节点	0.889	0.583
Alert-CM_no_log	去除日志节点、日志列节点	0.838	0.58
Alert-CM_no_metric	去除指标组节点、日志节点、日志列节点	0.749	0.474
Alert-CM_no_rule	去除预警规则节点、指标组节点、日志节点、日志列节点	0.693	0.421

4.5 超参数对于实验效果的影响 (RQ4)

在聚类阶段, Alert-CM 采用了 DBSCAN 算法, DBSCAN 需要额外设置两个超参数: 邻域半径 ε 和最小邻居数 MinPts, 这两个参数对聚类效果具有直接影响. 邻域半径 ε 决定了聚类的范围, 而最小邻居数 MinPts 则控制每个聚类中所需的最小样本数, 二者的取值会显著影响聚类结果的质量.

图 7 展示了邻域半径 ε 对 NMI 和 ARI 指标的影响. 实验表明, 当 ε 值较小时, 聚类的范围会缩小, 导致聚类数量增多, 每个聚类中的样本数减少. 在这种情况下, NMI 和 ARI 指标显著下降, 因为只有高度相似的告警能够被归入同一聚类. 相反, 当 ε 值增大时, 聚类的范围扩大, DBSCAN 可能将更多的样本点合并进现有聚类, 甚至可能将所有样本归为一个单一聚类. 这种情况同样会导致 NMI 和 ARI 指标的显著下降. 实验结果表明, 当 ε 取 0.4 时, NMI 和 ARI 达到了最大值, 表明此时的邻域半径能够有效平衡聚类数量和样本归类的准确性, 从而实现优化的聚类效果.

图 8 展示了最小样本数 MinPts 对 NMI 和 ARI 的影响. MinPts 的设定决定了 DBSCAN 在聚类过程中如何处理噪声节点. 当 MinPts 不为 1 时, DBSCAN 会将不属于任何聚类的节点视为噪声节点, Alert-CM 将这些噪声节点

作为独立的聚类处理. 随着 MinPts 值的增大, 较小规模的聚类可能会被忽略, 更多的节点将被 DBSCAN 视为噪声节点, 导致 ARI 分数的骤降. 尽管如此, 较大规模聚类中的节点仍能保持较高的互信息, 因此 NMI 的下降趋势较为平缓. 在实际应用中, 直接忽略不属于任何聚类的告警可能会错失一些潜在的安全隐患, 因此将 MinPts 设为 1 能够在保持较高聚类有效性的基础上, 更好地符合实际应用需求.

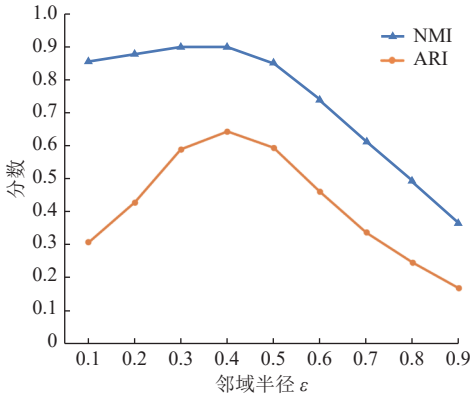

图 7 邻域半径 ε 对于告警聚类有效性的影响

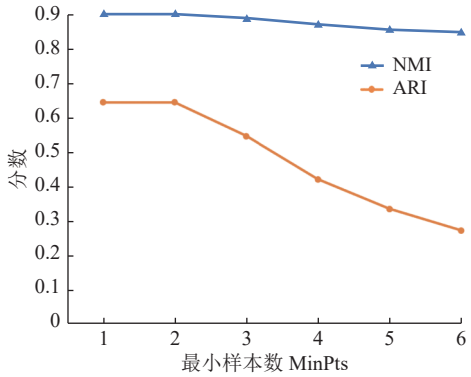

图 8 最小样本数 MinPts 对于告警聚类有效性的影响

4.6 方法局限性

尽管 Alert-CM 在告警聚类任务中展现出显著优势, 但仍然存在一些局限性需要进一步优化. 首先, Alert-CM 采用离线训练模式, 这意味着模型在训练完成后无法根据新采集的底层系统数据及其关联关系进行动态调整和优化. 然而, 在实际生产环境中, 告警管理系统的配置信息、业务逻辑及系统拓扑可能会随着时间不断演变, 静态训练的模型可能无法及时适应这些变化, 进而影响告警聚类的准确性和鲁棒性. 因此, 未来的研究方向之一是引入在线学习机制, 使 Alert-CM 能够持续从新增数据中学习, 并动态调整其参数, 以增强模型对系统变化的适应能力, 提高告警聚类结果的实时性和精准度.

其次, 由于 Alert-CM 依赖于复杂的图结构表示和深度神经网络进行特征提取与聚类建模, 其训练过程相对耗时, 尤其在处理大规模系统数据时, 训练效率可能成为模型实际部署的瓶颈. 虽然对于离线训练而言, 较长的训练时间在一定程度上是可以接受的, 但在业务需求快速变化的工业环境中, 如何提升训练效率仍然是一个亟待优化的问题.

5 总 结

本文提出并实现了 Alert-CM, 一种基于图表示学习的告警聚类方法. 由于由同一故障引发的告警往往在底层数据层面存在紧密关联, Alert-CM 构建了层次化的系统数据依赖图, 并将告警与系统中各节点的依赖关系进行抽象和映射, 有效扩展了告警的特征空间. Alert-CM 不仅充分利用了告警与底层数据之间的依赖关系, 还通过无监督学习或有监督学习引导图神经网络模型捕捉关键的底层系统数据特征, 区分不同系统数据对告警的影响强度, 从而提升了告警聚类的准确性与可靠性. 本文在复杂的真实工业数据集上评估了 Alert-CM 的有效性和效率. 实验结果表明, 与传统的告警聚合技术相比, Alert-CM 在准确性上具有显著提升, 并且在在线实时聚类中能够高效响应, 具备较好的实际应用性能.

除了告警聚类, Alert-CM 还具有拓展到故障根因分析领域的潜力. 我们计划进一步开发和完善 Alert-CM, 利用系统数据依赖图中的结构信息来预测故障的根本原因, 从而为工程师提供更加精准的故障诊断支持. 这一扩展不仅有助于提高故障恢复的效率, 还能够在系统维护过程中提供决策支持, 提升整体系统的可靠性和稳定性. 同时, 我们还将持续不断地在更多在线真实工业系统中评估 Alert-CM 的效果.

References

[1] Lewis J, Fowler M. Microservices: A definition of this new architectural term. 2014. https://www.martinfowler.com/articles/microservices.html

[2] Chen JJ, He XT, Lin QW, Xu Y, Zhang HY, Hao D, Gao F, Xu ZW, Dang YN, Zhang DM. An empirical investigation of incident triage for online service systems. In: Proc. of the 41st IEEE/ACM Int'l Conf. on Software Engineering: Software Engineering in Practice (ICSE-SEIP). Montreal: IEEE, 2019. 111–120. [doi: 10.1109/ICSE-SEIP.2019.00020]

[3] Chen ZB, Kang Y, Li LQ, Zhang X, Zhang HY, Xu H, Zhou YF, Yang L, Sun J, Xu ZW, Dang YN, Gao F, Zhao P, Qiao B, Lin QW, Zhang DM, Lyu MR. Towards intelligent incident management: Why we need it and how we make it. In: Proc. of the 28th ACM Joint Meeting on European Software Engineering Conf. and Symp. on the Foundations of Software Engineering. ACM, 2020. 1487–1497. [doi: 10.1145/3368089.3417055]

[4] Lou JG, Lin QW, Ding R, Fu Q, Zhang DM, Xie T. Software analytics for incident management of online services: An experience report. In: Proc. of the 28th IEEE/ACM Int'l Conf. on Automated Software Engineering. Silicon Valley: IEEE, 2013. 475–485. [doi: 10.1109/ASE.2013.6693105]

[5] Loki: Like Prometheus, but for logs. 2025. https://github.com/grafana/loki/

[6] Grafana Labs. Grafana Tempo. 2025. https://grafana.com/oss/tempo/

[7] Prometheus. IO. Prometheus. 2025. https://prometheus.io/

[8] Chen ZB, Liu JY, Su YX, Zhang HY, Wen XM, Ling X, Yang YQ, Lyu MR. Graph-based incident aggregation for large-scale online service systems. In: Proc. of the 36th IEEE/ACM Int'l Conf. on Automated Software Engineering. Melbourne: IEEE, 2021. 430–442. [doi: 10.1109/ASE51524.2021.9678746]

[9] Chen YJ, Yang X, Dong H, He XT, Zhang HY, Lin QW, Chen JJ, Zhao P, Kang Y, Gao F, Xu ZW, Zhang DM. Identifying linked incidents in large-scale online service systems. In: Proc. of the 28th ACM Joint Meeting on European Software Engineering Conf. and Symp. on the Foundations of Software Engineering. ACM, 2020. 304–314. [doi: 10.1145/3368089.3409768]

[10] Chen J, Wang P, Wang W. Online summarizing alerts through semantic and behavior information. In: Proc. of the 44th Int'l Conf. on Software Engineering. Pittsburgh: ACM, 2022. 1646–1657. [doi: 10.1145/3510003.3510055]

[11] Zhao NW, Chen JJ, Peng X, Wang HL, Wu XY, Zhang YZ, Chen ZK, Zheng XZ, Nie XH, Wang G, Wu Y, Zhou F, Zhang WC, Sui K, Pei D. Understanding and handling alert storm for online service systems. In: Proc. of the 42nd ACM/IEEE Int'l Conf. on Software Engineering: Software Engineering in Practice. Seoul: ACM, 2020. 162–171. [doi: 10.1145/3377813.3381363]

[12] Lin D, Raghu R, Ramamurthy V, Yu J, Radhakrishnan R, Fernandez J. Unveiling clusters of events for alert and incident management in large-scale enterprise it. In: Proc. of the 20th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM, 2014. 1630–1639. [doi: 10.1145/2623330.2623360]

[13] Ester M, Kriegel HP, Sander J, Xu XW. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996. 226–231.

[14] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv:1301.3781, 2013.

[15] Mikolov T, Grave E, Bojanowski P, Puhrsch C, Joulin A. Advances in pre-training distributed word representations. arXiv:1712.09405, 2017.

[16] Mikolov T, Sutskever I, Chen K, Corrado G, Dean J. Distributed representations of words and phrases and their compositionality. In: Proc. of the 27th Int'l Conf. on Neural Information Processing Systems. Lake Tahoe: Curran Associates Inc., 2013. 3111–3119.

[17] Cui YM, Che WX, Liu T, Qin B, Wang SJ, Hu GP. Revisiting pre-trained models for chinese natural language processing. In: Findings of the Association for Computational Linguistics: EMNLP 2020. ACL, 2020. 657–668. [doi: 10.18653/v1/2020.findings-emnlp.58]

[18] Yin ZS, Chen J, Wang P, Wang W. Alert stream compression method based on representation learning. Computer Applications and Software, 2024, 41(7): 34–41 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-386x.2024.07.006]

[19] Aggarwal CC, Han JW, Wang JY, Yu PS. A framework for clustering evolving data streams. In: Proc. of the 29th Int'l Conf. on Very Large Data Bases. Berlin: VLDB Endowment, 2003. 81–92.

[20] Wang WJ, Chen JJ, Yang L, Hou DJ, Wang XK, Wu FD, Zhang RZ, Wang Z. Network-side alert prioritization method based on multivariate data fusion. Ruan Jian Xue Bao/Journal of Software, 2024, 35(8): 3610–3625 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/7118.htm [doi: 10.13328/j.cnki.jos.007118]

[21] Sun ZJ, Li XY, Sun XF, Meng YX, Ao X, He Q, Wu F, Li JW. ChineseBERT: Chinese pretraining enhanced by glyph and pinyin information. In: Proc. of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th Int'l Joint Conf. on Natural Language Processing (Vol. 1: Long Papers). ACL, 2021. 2065−2075. [doi: 10.18653/v1/2021.acl-long.161]

[22] Siffer A, Fouque PA, Termier A, Largouet C. Anomaly detection in streams with extreme value theory. In: Proc. of the 23rd ACM

SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. Halifax: ACM, 2017. 1067–1075. [doi: 10.1145/3097983.3098144]

[23] Keogh E, Ratanamahatana CA. Exact indexing of dynamic time warping. Knowledge and Information Systems, 2005, 7(3): 358–386. [doi: 10.1007/s10115-004-0154-9]

[24] Zhao NW, Jin PS, Wang LX, Yang XQ, Liu R, Zhang WC, Sui KX, Pei D. Automatically and adaptively identifying severe alerts for online service systems. In: Proc. of the 2020 IEEE Int'l Conf. on Computer Communications. Toronto: IEEE, 2020. 2420–2429. [doi: 10.1109/INFOCOM41043.2020.9155219]

[25] Sutskever I, Vinyals O, Le QV. Sequence to sequence learning with neural networks. In: Proc. of the 28th Int'l Conf. on Neural Information Processing Systems (Vol. 2). Montreal: MIT Press, 2014. 3104–3112.

[26] Chen YR, Zhang CX, Dong Z, Yang DY, Peng X, Ou JY, Yang H, Wu ZS, Qu XJ, Li W. Dynamic graph neural networks-based alert link prediction for online service systems. In: Proc. of the 38th IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE). Luxembourg: IEEE, 2023. 79–90. [doi: 10.1109/ASE56229.2023.00177]

[27] Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008, 2008(10): P10008. [doi: 10.1088/1742-5468/2008/10/P10008]

[28] Wang YH, Li GZ, Wang ZJ, Kang Y, Zhou YF, Zhang HY, Gao F, Sun J, Yang L, Lee P, Xu ZW, Zhao P, Qiao B, Li LQ, Zhang X, Lin QW. Fast outage analysis of large-scale production clouds with service correlation mining. In: Proc. of the 43rd IEEE/ACM Int'l Conf. on Software Engineering (ICSE). Madrid: IEEE, 2021. 885–896. [doi: 10.1109/ICSE43902.2021.00085]

[29] Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. IEEE Trans. on Neural Networks, 2009, 20(1): 61–80. [doi: 10.1109/TNN.2008.2005605]

[30] Hamilton WL, Ying R, Leskovec J. Inductive representation learning on large graphs. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 1025–1035.

[31] Zhang ZW, Cui P, Zhu WW. Deep learning on graphs: A survey. arXiv:1812.04202, 2020.

[32] Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, Grohe M. Weisfeiler and leman go neural: Higher-order graph neural networks. In: Proc. of the 33rd AAAI Conf. on Artificial Intelligence. Honolulu: AAAI, 2019. 4602–4609. [doi: 10.1609/aaai.v33i01.33014602]

[33] Hassani K, Khasahmadi AH. Contrastive multi-view representation learning on graphs. In: Proc. of the 37th Int'l Conf. on Machine Learning. PMLR, 2020. 4116–4126.

[34] Park J, Lee M, Chang HJ, Lee K, Choi JY. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In: Proc. of the 2019 IEEE/CVF Int'l Conf. on Computer Vision. Seoul: IEEE, 2019. 6518–6527. [doi: 10.1109/ICCV.2019.00662]

[35] Shiao W, Saini US, Liu YZ, Zhao T, Shah N, Papalexakis EE. CARL-G: Clustering-accelerated representation learning on graphs. In: Proc. of the 29th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining. Long Beach: ACM, 2023. 2036–2048. [doi: 10.1145/3580305.3599268]

[36] Rousseeuw PJ. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 1987, 20: 53–65. [doi: 10.1016/0377-0427(87)90125-7]

[37] Lenssen L, Schubert E. Clustering by direct optimization of the medoid silhouette. In: Proc. of the 15th Int'l Conf. on Similarity Search and Applications. Bologna: Springer, 2022. 190–204. [doi: 10.1007/978-3-031-17849-8_15]

[38] Hruschka ER, de Castro LN, Campello RJGB. Evolutionary algorithms for clustering gene-expression data. In: Proc. of the 4th IEEE Int'l Conf. on Data Mining (ICDM 2004). Brighton: IEEE, 2004. 403–406. [doi: 10.1109/ICDM.2004.10073]

[39] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin ZM, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai JJ, Chintala S. PyTorch: An imperative style, high-performance deep learning library. In: Proc. of the 33rd Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2019. 8026–8037.

[40] Fey M, Lenssen JE. Fast graph representation learning with PyTorch Geometric. arXiv:1903.02428, 2019.

[41] Vinh NX, Epps J, Bailey J. Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In: Proc. of the 26th Annual Int'l Conf. on Machine Learning. Montreal: ACM, 2009. 1073–1080. [doi: 10.1145/1553374.1553511]

[42] Hubert L, Arabie P. Comparing partitions. Journal of Classification, 1985, 2(1): 193–218. [doi: 10.1007/BF01908075]

[43] Grover A, Leskovec J. node2vec: Scalable feature learning for networks. In: Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. San Francisco: ACM, 2016. 855–864. [doi: 10.1145/2939672.2939754]

[44] Perozzi B, Al-Rfou R, Skiena S. DeepWalk: Online learning of social representations. In: Proc. of the 20th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM, 2014. 701–710. [doi: 10.1145/2623330.2623732]

附中文参考文献

[18] 阴振生, 陈佳, 王鹏, 汪卫. 基于表示学习的告警数据流压缩算法. 计算机应用与软件, 2024, 41(7): 34–41. [doi: 10.3969/j.issn.1000-386x.2024.07.006]

[20] 王维靖, 陈俊洁, 杨林, 侯德俊, 王星凯, 吴复迪, 张润滋, 王赞. 基于多元数据融合的网络侧告警排序方法. 软件学报, 2024, 35(8): 3610–3625. http://www.jos.org.cn/1000-9825/7118.htm [doi: 10.13328/j.cnki.jos.007118]

作者简介

陈淼, 硕士生, 主要研究领域为云原生, 智能化运维.

张弼铖, 硕士生, 主要研究领域为云原生, 智能化运维.

张晨曦, 博士, 副教授, CCF 专业会员, 主要研究领域为智能化运维, 云原生, 软件测试.

彭鑫, 博士, 教授, 博士生导师, CCF 杰出会员, 主要研究领域为智能化软件开发, 云原生与智能化运维, 泛在计算软件系统.

杨定裕, 博士, 研究员, CCF 专业会员, 主要研究领域为数据库与 AI, 软硬件性能优化, 云原生智能运维.

李伟, 本科, 主要研究领域为智能化运维, 可观测技术.

钱泽林, 硕士生, 主要研究领域为智能化运维, 可观测技术.

吴哲顺, 硕士生, 主要研究领域为智能化运维, 可观测技术.

欧嘉煜, 硕士生, 主要研究领域为智能化运维, 可观测技术.

钟坚锐, 本科生, 主要研究领域为应急技术与管理.