发表于 Scientific Reports，2025年11月5日，数字对象标识符：10.1038/s41598-025-27693-4

请引用本文的期刊版本。

arXiv:2512.23380v1 [cs.LG] 29 Dec 2025

基于协作式Transformer检测操作系统日志中点异常和集体异常的统一框架

Mohammad Nasirzadeh[1], Jafar Tahmoresnezhad[1*], Parviz Rashidi-Khazaee[1]

[1*]信息技术与计算机工程学院，Urmia University of Technology, Band，Urmia，57166-17165，西阿塞拜疆，伊朗。

*通讯作者。电子邮箱：j.tahmores@it.uut.ac.ir；贡献作者：nasirzadeh.moh@it.uut.ac.ir；p.rashidi@uut.ac.ir；

摘要

日志异常检测对于维护操作系统的安全至关重要。根据日志数据收集的来源，日志中会记录各种信息，这些信息可视为不同的日志模态。基于这一认识，单模态方法往往会忽略日志数据的不同模态，因此难以发挥作用；而多模态方法则无法处理这些模态之间的交互作用。我们将多模态情感分析应用于日志异常检测，提出了CoLog，这是一个利用多种模态协作编码日志的框架。CoLog 利用协作式Transformer和多头印象注意力，学习多个模态之间的交互作用，从而确保全面的异常检测。为处理这些交互作用所带来的异质性，CoLog 引入了模态适配层，用于适配不同日志模态的表示。该方法使 CoLog 能够学习数据中细微的模式和依赖关系，从而增强其异常检测能力。大量实验表明，CoLog 优于现有最先进的方法。此外，在检测点异常和集体异常方面，针对基于日志的异常检测，在七个基准数据集上，CoLog 的均值精确率达到 99.63%，均值召回率达到 99.59%，均值 F1 分数达到 99.61%。CoLog 全面的检测能力使其非常适用于网络安全、系统监控和运营效率提升。CoLog 代表了日志异常检测领域的一项重大进展，通过统一框架为点异常与集体异常检测提供了精密有效的解决方案，并应对复杂挑战

自动日志数据分析所带来的问题。我们还在
https://github.com/NasirzadehMoh/CoLog提供了 CoLog 的实现。

关键词：日志异常检测、多模态情感分析、点异常与集体异常检测、类别不平衡、深度学习

1 引言

系统在交互过程中发生的所有情况都会记录在系统日志中，包括带有时间戳的事件（例如交易、错误和入侵）。异常也称为异常值，是指日志数据中偏离系统常规行为的模式或事件。基于日志的异常检测用于识别和定位日志异常值[1]。然而，由于日志数据规模庞大、结构复杂且动态变化，手动分析日志或应用传统方法 [2–19] 会变得更具挑战性，因为这些方法主要依赖人工处理和规则设定 [20–23]。此外，源自不同系统的日志数据可能使用不同的术语。因此，基于深度学习的自动化日志异常检测对于及时应对威胁或挑战、设定响应阈值以及做出数据驱动的决策至关重要 [24]。

近年来，深度学习在自动化完成许多任务方面展现出巨大潜力[25–34]。自动化日志异常检测便是其中一例。鉴于深度学习技术在自动化日志异常检测方面取得了优异结果，将这些技术用于检测日志文件中的异常，可针对未识别的威胁、识别根本原因，以及缓解网络攻击、欺诈活动或系统故障等关键场景提供实时警报 [35–39]。因此，近年来，深度神经网络已成为自动化的基于日志的异常检测领域中占主导地位的建模方法。这一研究方向始于 DeepLog 在从日志文件中检测异常方面取得的突破性成果 [36]。多年来，基于深度学习的各种方法不断发展，在这一领域的影响力日益增强 [40]。自动化日志异常检测领域存在众多研究方向，例如多层感知机（MLP） [41–45]（作为其他深度学习架构的一部分）、卷积神经网络（CNN） [46–51], 、循环神经网络（RNN）[36, 38, 52–67], 、自编码器 [42, 68–72], 、生成对抗网络（GAN） [42, 52, 73–78], 、Transformer [21, 79–85], 、通常用于提升其他深度神经网络架构性能的注意力机制 [37, 41, 48, 86–89]、图神经网络（GNN） [90], 、演化粒状神经网络（EGNN） [91],以及大语言模型（LLM） [35, 92, 93]。

如图所示1，根据日志数据采集来源的类型，日志中可以存储各种各样的信息。实际上，这些数据通过不同的方面来表达，我们将这些方面称为模态。每个日志文件都包含两种主要模态：（1）记录序列，我们称之为序列模态；（2）每条记录的内容，我们称之为语义模态。

数字对象标识符： 10.1038/s41598-025-27693-4

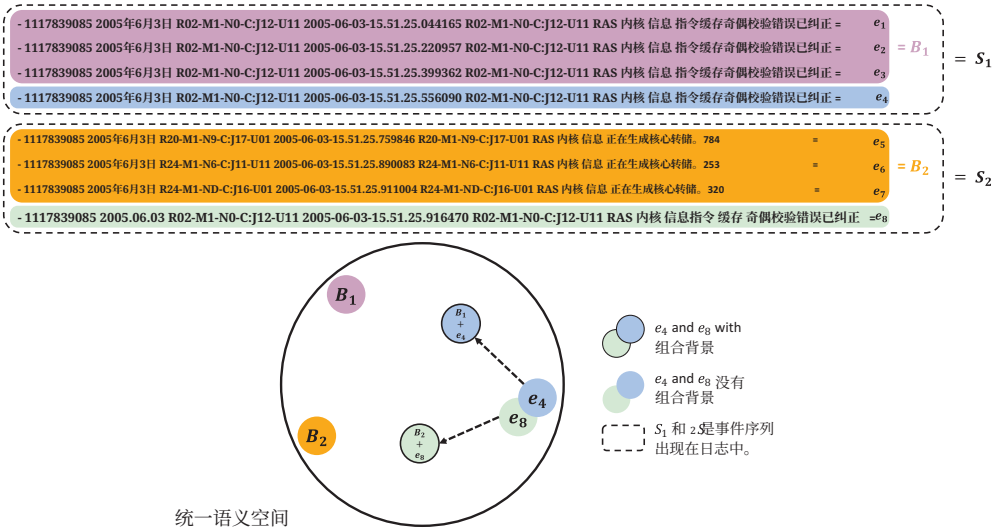


图 1 序列模态（B_1， B_2）如何为语义模态（e_4， e_8）补充更多信息，以及背景不同的相同样本如何在统一语义空间中分离的示意图。对于 e_4 和 e_8 事件向量，其背景，其具体而言是 B_1 和 B_2，或许可以用于以更有意义的方式描述 e_4 和 e_8 （需要注意的是， e_4 等同于 e_8）。据此，我们将 B_1 和 B_2 指定为序列模态，将 e_4 和 e_8 指定为语义模态。

一方面，事件序列特征可以从语义角度为每个日志事件提供信息。这意味着，日志文件中的各种信息都可以用于构建更准确的模型。图1 展示了序列模态和语义模态之间的关联，以及如何利用相应的序列向量区分相同事件的向量。另一方面，这些模态使模型能够从不同角度做出决策，从而在执行指定任务时提高结果的精确率。

另一方面，现有文献中的大多数方法都侧重于对日志进行单模态异常检测 [21, 35, 37, 38, 41, 43, 46, 47, 49, 50, 52–54、56–58、60、63、64、68、70–82, 87, 89, 90, 92, 93]。这些研究仅利用日志中记录的信息的某一特定方面。然而，忽略日志的其他模态会导致有价值的数据流失。事实上，仅依据日志文件信息的单一方面进行分析，可能导致模型无法识别更复杂的异常。另一方面，文献中提出的多模态方法采用早期 [51, 59, 61, 62, 83, 85, 86, 88, 91]，中间[42, 45, 48, 55, 65–67]，和晚期 [44] 融合机制，或为提取出的不同模态分别建立模型 [36, 69, 84]，从而带来各种挑战，如表1所示。拼接来自多个模态的未经处理的数据可能会产生维度极高的输入，因此难以处理，并带来高维度挑战。原始数据中的噪声可能影响模型，因为某一模态中的噪声可能影响整体信息，这体现了噪声敏感性挑战。拼接原始数据会增加复杂性，尤其是在数据类型和格式各异时，从而带来增强的数据复杂性挑战。由此得到的特征

数字对象标识符：10.1038/s41598-025-27693-4

来自多个模态的特征可能并不总是直接兼容或易于组合，这凸显了兼容性挑战。为每种模态部署不同的模型会增加系统的总体复杂性，从而加剧网络复杂性挑战。多个模态的贡献可能难以平衡，尤其是在某一模态提供的信息多于其他模态时，因此会带来贡献平衡挑战。在决策层合并结果或为每种模态使用不同模型，可能会忽略模态之间的重要交互作用，从而导致缺乏共享信息挑战。分别对每种模态执行处理可能造成冗余计算，尤其是在存在重叠特征时，因此会带来冗余挑战。由于各模态直到最终决策步骤才合并，提升性能的潜力可能受到限制，从而导致改进受限挑战。

表1 日志异常检测文献中所涵盖的多模态方法所面临的挑战。

	方法			
挑战	早期融合	中期融合	Late融合	分离模型
高维性	除.....外的所有 *MLog*[1]	*Mdfulog*[2], *Swisslog*[3], *WDLog*[4], *and FSMFLog*[5]	×	×
噪声敏感性	all	×	×	×
增强的数据复杂性	all	×	×	×
特征兼容性	all	*Mdfulog*, *Swisslog*, *WDLog*, *and FSMFLog*	×	×
网络复杂性加剧	×	all	all	all
平衡贡献	all	*Mdfulog*, *Swisslog*, *WDLog*, *and FSMFLog*	all	×
缺乏共享信息	×	×	all	all
冗余	×	×	all	all
改进有限	×	×	all	all

[1]Fu等人提出 *MLog* [2]Li 等人 提出 *Mdfulog* [3]Li 等人 提出 *Swisslog* [4]Niu等人 提出 *WDLog* [5]Niu等人 提出 *FSMFLog*

鉴于上述动机，我们提出一种基于多模态情感分析（MSA）[94–103] 的新型自动化的基于日志的异常检测方法，称为 CoLog，以克服上述挑战。CoLog 将日志中的异常视为负面情感。同理，正常样本被归类为正向情感。日志文件中最重要的两种模态是事件的语义特征，以及从事件序列中提取的特征。事件的背景和上下文可以构成序列模态。CoLog

通过构建语义模态和序列模态，利用两种模态之间的交互来深入学习异常。在此过程中，我们采用一种引导注意力（GA）机制 [104] ，以协作方式结合其他模态对每种模态进行编码。最后，为了通过融合模态实现预测目标，我们应用 Transformer 块，并将每种模态的输出特征向量转换到一个更高维的空间，即潜在空间，以进行有效融合。

此外，日志通常是记录点异常和集体异常的载体。检测日志记录中的点异常和集体异常至关重要，因为每种异常类型都揭示了系统行为的不同方面。点异常代表孤立的异常行为实例，通常预示着突发故障或攻击等即时问题。集体异常是指一段时间内出现的异常活动模式，可能意味着更深层次、更具系统性的问题，例如性能持续下降或长期存在的安全威胁。检测不同类型的异常有助于全面了解短期和长期问题，从而更有效地进行故障排查和预防性维护。然而，现有研究都只专注于识别一种日志异常类型。此外，据我们所知，在基于日志的异常检测领域之外，只有两种技术旨在利用统一框架识别点异常和集体异常。Li 等人 使用时序卷积网络（TCN）处理时间序列，并基于固定长度序列计算异常分数，以评估点数据或集体数据是否异常。然而，由于依赖预先设定的序列长度，该方法可能无法识别某些集体异常实例。此外，Liu 等人 使用改进局部离群因子（ILOF）检测可变长度的集体异常，其中使用已识别的点异常作为边界点，将残差序列划分为多个不同长度的子序列。每个子序列的异常分数均通过 ILOF 方法计算。该方法依赖异常分数的计算，因此可能导致误报，尤其是在较长的序列中。此外，这两种方法都没有明确学习不同模态之间的关系。因此，借助 Transformer [107] 架构，我们提出 CoLog，通过统一框架解决点异常和集体异常日志的检测问题。为此，我们提出使用协作式Transformer（CT）以端到端方式处理日志模态，使它们能够相互作用并学习彼此之间的关系。由于 CoLog 是一种有监督方法，因此不会面临误差累积的挑战。Th

e 摘要 o f 我们的贡 b献如下 follows.

1. 我们提出了 CoLog，这是首个将日志异常检测定义为多模态情感分析问题的框架，从而能够以统一的方式检测点异常和集体异常。2. 我们设计了一种具有印象注意力和模态适配层的协作式 Transformer 架构，用于捕捉语义模态与序列模态之间的细微交互作用。3. 我们引入了一种平衡机制，以自适应地对各模态的贡献进行加权，这有助于确保具有内在差异的模态能够在同一表示空间中表示。

4. 我们使用七个基准数据集对 CoLog 进行了评估，取得了最先进的性能，并在 GitHub 存储库中提供了 CoLog 的实现，以促进可复现性。5. 我们的研究成果和专业服务可在 Alarmif 网站上获取，从而在研究与实际应用之间建立直接联系。

本文的组织结构如下。第 2 节讨论该领域的相关研究。第 3 节定义任务，阐述Transformer对本工作的重要性、威胁模型以及相关假设。第 4 节将讨论基于MSA的日志异常检测中CT的设计与训练考量。评估结果见第 5 节；最后，我们总结本工作，并在第 6 节提出未来研究建议。

2 相关工作

众多研究工作对基于日志的异常检测进行了探讨 [1, 20, 22–24, 40, 108]。这些方法已经从正则表达式和基于规则的方法发展到应用基于深度学习的技术。传统上，传统技术采用正则表达式 [19], 、基于规则的方法 [5, 6, 8, 9, 15–18], 或机器学习（ML） [4, 12, 109–111] 来查找和提取异常事件。基于正则表达式的方法通常包括制定领域专用正则表达式。通过比较模式，利用这些正则表达式识别最近发现的漏洞。这些技术的一个主要缺点是可能忽略最新的异常事件。此外，当每个日志条目看似正常、但整个序列异常时，基于正则表达式的技术无法检测到任何异常活动。基于规则的方法通常使用预定义规则或特征码来检测日志数据中的异常模式。这些原则通常基于对正常和异常系统行为的观察。符合这些准则的数据会被归类为异常。基于规则的方法只能识别预定义的异常。基于机器学习的日志分析异常检测流程包括三个主要步骤：日志解析、特征提取和异常识别。基于机器学习的方法无法跟踪时序信息，因为它们没有记忆过去事件的机制。此外，由于需要人工从原始日志中提取特征，这些方法会严重影响模型输出。事实上，上述所有方法都需要广泛的领域专业知识。CoLog是一种灵活的方法，不依赖领域专业知识或预定义规则。CoLog在提取时序信息方面表现出显著的能力。

离群点检测应当实时进行。由于深度神经网络在实时文本分析方面表现出色，许多基于深度学习的方法将日志数据描述为自然语言序列，并有效地发现离群点 [108]。这开启了提出先进深度学习方法来识别日志异常的趋势。这些方法解决了前文所述方法存在的问题。

后续小节概述了基于深度学习的自动化日志异常检测相关文献。随后，我们回顾深度多模态情感分析（MSA）相关文献，并将我们的日志异常检测方法与深度学习方法进行比较。

2.1 基于深度学习的日志异常检测

基于深度学习的方法可追溯至 DeepLog 的问世 [36], ， 该方法在检测日志数据异常方面表现出良好的效率。由于这些技术在过去几年中迅速发展，如今已成为该领域建模的标准。因此，本小节主要聚焦于基于深度学习的模型。下面，我们将讨论单模态和多模态方法。

2.1.1 单模态方法

尽管目前已有多种用于日志异常检测的深度学习技术，本小节仍专门介绍用于日志异常检测的单模态模型。这些技术的核心理念源自自然语言处理（NLP）。此外，它们通常优先考虑日志数据的序列特性。常用的深度学习技术包括多层感知机 [41, 43], 卷积神经网络 [46, 47, 49, 50], 循环神经网络 [38, 52–54, 56–58, 60, 63, 64], 自编码器 [68, 70–72], 生成对抗网络 [52, 73–78], Transformer[21, 79–82], 注意力机制 [37, 41, 87, 89], GNN [90], 和大语言模型 [35, 92, 93],也常用于这些方法。除 [68], 前文提到的单模态方法可分为三大类：基于序列、基于语义和基于大语言模型的方法。Otomo 等人 提出了一种面向大规模网络的日志事件异常检测方法，该方法利用潜变量将多样化的日志数据嵌入隐藏状态，无需进行预处理或特征提取。其核心思路是首先将原始日志消息转换为每种日志类型对应的日志时间序列，然后使用条件变分自编码器，按天和日志类型将日志时间序列映射为潜变量。最后，采用聚类方法对潜变量进行聚类，并检测其偏离已识别簇的情况，将这些偏离视为异常。

基于序列的单模态方法。 基于序列的建模 [21, 37, 41, 43,46、47、49、50、54、56、57、60、63、64、70–75、77–79, 81, 87, 89, 90]被广泛认为是基于日志的异常检测中的主导网络架构。通常，这些方法将日志数据视为自然语言序列。例如，Lu 等人从原始日志中提取日志键，并为其分配唯一的数值。较短的向量以零填充，较长的向量则截断，以确保每个日志键会话的长度相同。向量化后的日志被输入嵌入层，随后经过三个具有不同滤波器大小的一维卷积层。之后，使用最大池化、随机失活和全连接 Softmax 层对异常进行分类。Zhang 等人指出，由于日志记录语句不断演变，以及日志数据处理过程中存在噪声，日志数据具有不稳定性。他们提出了 LogRobust，通过从日志事件序列中提取语义特征向量来解决这一问题。随后，他们提出了一种基于注意力的双向长短期记忆网络（BiLSTM）模型，用于自动识别异常实例并学习各种日志事件的重要性。LogSpy [41] 是一种用于检测分布式系统中异常的方法。它采用名为频繁模板树（FT-Tree）的技术提取模板，并使用 Skip-gram 模型构建特征向量。该方法运用卷积神经网络架构和注意力机制来分析日志模板之间的关系，并提高实时日志异常检测的效率。LogBERT [79]

使用 Transformer 编码器对日志序列建模，以进行自监督日志异常检测。训练过程包含两个不同的任务：（1）掩码日志键预测，以准确识别正常序列中随机被掩码的日志键；（2）超球体体积最小化，使正常日志序列在嵌入空间中彼此靠近。LogBERT 对正常日志序列模式进行编码，并在训练后检测异常。LogGAN [74] 使用结合排列事件建模的日志级GAN。日志解析器将非结构化系统日志转换为结构化事件。排列事件建模降低了长短期记忆网络（LSTM）的序列依赖性问题，使其能够处理乱序到达的日志。对抗学习系统使用生成器生成合成日志，并由判别器对其进行验证。DeepCASE [87] 能够获取有关序列中日志事件之间相关性的知识。随后，解释器将相似事件归入相似的簇，以检测异常。系统管理员会收到异常样本报告，其决策由 DeepCASE 学习，并应用于未来出现的相同序列。Catillo等人提出了 AutoLog，这是一种半监督深度自编码器。其功能不依赖于日志行排序的概念。该方法通过提取数值评分向量来处理异构日志。在此过程中，AutoLog 不会纳入任何应用特定知识，也不会对日志中底层行的格式和顺序作出任何假设。Qi等人提出了一种名为 Adanomaly 的基于日志的异常检测方法。该方法使用双向生成对抗网络模型提取特征，并采用基于集成学习的分类技术检测异常。Adanomaly 借助双向生成对抗网络计算重构损失和判别损失，并将其作为特征。LogEncoder [43] 是一个利用带标签和无标签数据检测系统日志异常的框架。该系统包含三个主要组成部分：Log2Emb、Emb2Rep 和异常检测。Log2Emb 过程将离散日志事件转换为语义向量。Emb2Rep 过程采用基于注意力机制的LSTM模型，以区分正常序列与异常序列。该框架既适用于离线检测，也适用于在线检测。Zhang 等人提出了 LayerLog，这是一种用于检测日志序列异常的创新系统。LayerLog 使用名为“词-日志-日志序列”的层次结构来完成这一任务。该系统由三层组成：词、日志和日志序列。它能够从原始日志中提取语义信息而不造成任何损失。实证评估表明，LayerLog 在检测异常方面有效。LogFormer [37] 是一个包含预训练和微调两个阶段的流程，旨在检测日志异常并增强跨领域泛化能力。它提取日志序列的语义信息，并在特定领域上训练基于 Transformer的模型。通过采用基于适配器的微调，在大多数参数保持固定、仅更新少量参数的情况下，降低了训练成本。FastLogAD [77] 是一种用于检测日志数据中异常的无监督技术。它采用掩码引导异常生成器（MGAG）和判别式异常分离（DAS）模型。MGAG算法基于正常日志数据生成模拟异常，而DAS模型则区分正常日志序列与模拟异常，从而实现有效的实时检测。MLAD [81] 是一种利用以下技术进行日志异常检测的方法：

Transformer 和高斯混合模型（GMM），用于检测多个系统中的异常。MLAD 采用 SBERT [112] 将日志序列转换为语义向量。它利用稀疏自注意力机制捕捉词级依赖关系，并利用 GMM 突出“相同快捷方式”问题中与低频词相关的不确定性。 Zhang 等人 提出了 E-Log，这是一种面向数据库的细粒度弹性异常检测与诊断方法。Qiu 等人 引入了 LogAnomEX，这是一种基于 Electra 模型和门控双线性神经网络的无监督日志异常检测方法。Qiu 等人 提出了 FedAware，这是一种利用分形收缩自编码器的分布式物联网入侵检测方法。这些方法体现了有监督和无监督两种视角，进一步凸显了比较基线的重要性。

基于语义的单模态方法。 基于语义的单模态日志异常检测应用有限 [38, 52, 53, 58, 76, 80, 82]。一些基于语义的方法利用日志行提取语义信息，并通过无监督或半监督方法构建模型 [76, 80, 82]。其他技术则结合情感分析（SA）来检测日志异常 [38, 52, 53, 58]。这些方法以日志消息的极性为依据，学习并预测正常和异常日志事件。基于情感分析的日志异常检测方法将正常情况归为正向情感，将异常归为负面情感。例如，Farzad 和 Gulliver 提出了一种检测和分类日志消息中异常的方法，采用了三种不同的深度学习架构：Auto-LSTM、Auto-双向长短期记忆网络和 Auto-门控循环单元（GRU）。该方法包含两个主要阶段：分别使用自编码器从正向和负向类别中提取特征，然后拼接输出，生成单标签数据集，用于训练长短期记忆网络、双向长短期记忆网络和门控循环单元模型。分类模型采用 Softmax 激活函数、分类交叉熵损失和 Adam优化器进行训练[116]。 Farzad还提出了一种结合 GAN、自编码器和 GRU网络的日志消息异常检测方法。该方法利用提出的 SeqGAN 生成合成负向日志消息，对负向日志消息进行过采样以平衡数据集。自编码器从正向和过采样后的负向日志消息中提取特征，并为每个类别分别构建网络。该方法使用 GRU网络进行异常检测和分类，与未进行过采样的模型相比，准确率有所提高。Logsy [80] 是一种基于分类的非结构化日志数据异常检测方法。它使用自注意力编码器和超球面损失函数来学习紧凑的日志向量表示。该方法利用其他系统的辅助日志数据集来改善正常日志的表示并防止过拟合。学习得到的表示通过将异常分数设为日志与超球面中心的距离，帮助区分正常日志和异常日志。 Studiawan 等人 提出了 pylogsemtiment，这是一种基于情感分析、用于识别生产环境操作系统日志文件中异常的方法。该方法采用正面和负面情感分析，将负面情感检测视为与异常检测类似的任务。他们使用深度学习预测未见数据，并实现 GRU网络来判定情感。他们考虑到现实世界日志中负向消息出现频率较低所导致的类别不平衡问题，并提出使用 Tomek 链技术平衡两种情感类别。SentiLog 是 Zhang 等人 提出的另一种分析并行文件系统

日志并检测异常的方法。SentiLog 利用并行文件系统相关源代码集合，训练通用的情感自然语言模型。通过这种方式，SentiLog 学习开发者基于长短期记忆网络嵌入并行文件系统中的隐式语义信息。与 pylogsentiment 类似，SentiLog 表明，情感分析是对非结构化且难以解码的系统日志进行建模的一种有前景的方法。A2Log [82] 是一种针对日志数据的两步无监督异常识别方法，包括异常评分和决策。首先，使用自注意力神经网络为每条日志消息分配分数。此外，该方法通过扩充现有正常训练数据来设定决策边界，无需使用任何异常样本。LogELECTRA [76] 是一种专为非结构化系统日志设计的自监督异常检测模型。它无需日志解析器即可检查单条日志消息。系统通过训练判别器模型识别被替换的词元，从而理解正常日志消息的上下文。在推理过程中，LogELECTRA 会计算每条日志消息的异常分数，并实时识别单条异常，无需进行时间序列分析。

基于大语言模型的单模态方法。 在展示了大语言模型执行各种任务时独特且不断涌现的能力之后， [117], 近期研究提出利用大语言模型从日志数据中检测异常。例如，LogFiT [35] 是一种用于检测系统日志中异常的创新方法。该方法不依赖预先确定的模板或带标签数据，而是利用一种基于变换器的双向编码器表示（BERT）的语言模型来分析日志。系统通过使用正常日志数据预测掩码句子，获取语言模式知识。LogFiT 在推理过程中评估对前 k 个词元的预测准确率，以确定用于检测异常的阈值。LogFiT 专为轻松集成到现有自然语言处理框架而设计。LogPrompt [93] 利用大语言模型在在线场景中自动执行日志分析。该系统应对与理解日志异常检测结果的能力有限以及处理未见日志的能力相关的挑战。LogPrompt 采用三种先进的提示策略：自提示、思维链提示和上下文内提示。LogPrompt 无需大量训练数据，即可高效分析日志文件并识别异常模式。人工评估表明，LogPrompt 的解释具有参考价值且易于理解，有助于专业人员理解日志数据。Pan 等人 提出了 RAGLog，用于检测日志数据中的异常。RAGLog 将检索增强生成（RAG）模型与向量数据库和大语言模型结合使用。它无需解释日志，而是直接处理未处理的日志数据。大语言模型通过语义分析比较正常日志条目与查询条目，使 RAGLog 能够作为零样本分类器运行，无需使用异常日志样本进行训练。

2.1.2 多模态方法

本节重点考察利用多模态分析来发现日志数据中异常的模型。与日志分析中的单模态方法相比，这类方法相对少见。深度学习采用多种技术，例如

多层感知机 [42, 44, 45], 卷积神经网络 [48, 51], 循环神经网络 [36, 55, 59, 61, 62, 65–67], 自编码器[42, 69], 生成对抗网络 [42], Transformer [83–85], 注意力机制 [48, 86, 88], 以及 EGNN [91], 这些技术常用于上述方法中。前文提到的多模态技术主要分为三类：早期融合、 [51, 59, 61, 62, 83, 85, 86, 88, 91],中间融合 [42, 45, 48, 55, 65–67], 以及晚期融合 [44] 方法。该规则的例外包括 [36, 69, 84]。DeepLog [36] 是早期将深度学习用于日志异常检测的案例。它通过分析日志条目之间的序列关系来发挥作用，并将这些条目视为自然语言序列。接下来，模型学习正常日志模式，这些模式由日志键和表示每个日志条目的参数值向量构成。DeepLog 使用两个长短期记忆模型来学习日志键序列和参数值向量。此外，长短期记忆网络还用于预测未来的日志事件。随后，将这些预测与真实值进行比较，以识别异常。Qian 等人提出了 VeLog，这是一种利用变分自编码器（VAE）检测分布式系统中异常的方法。该流程包含两个不同阶段：离线训练和在线检测。在离线阶段，VeLog 收集并分析日志数据，生成表示日志执行情况的日志顺序矩阵和日志计数向量矩阵。VAE 模型基于这些矩阵学习正常的序列模式。在在线阶段，VeLog 分析并提取新日志的特征，生成可供比较的矩阵。通过将生成的矩阵与先前学得的模式进行比较，即可识别异常。该方法应用于大规模分布式系统时非常有效。He等人提出了 UMFLog，这是一种用于检测日志数据中异常的新型无监督模型。UMFLog 使用两个子模型有效捕获日志的语义特征和统计特征。第一个模型利用 BERT 从日志内容中提取语义特征。相比之下，第二个模型采用 VAE 学习统计特征，尤其关注日志中的词频。这种双特征方法提升了模型在无需带标签数据的情况下识别异常的能力，因此非常适合实际应用。

基于早期融合的多模态方法。 通常，这些方法会从日志数据的各个方面提取有意义的特征向量，并构建一个超向量。最后，将该超向量作为知识来源来训练模型。例如，Yu 等人在双向长短期记忆网络后添加注意力层，以评估先前事件在预测后续事件时的价值。他们将日志数据的序列特征和定量特征与语义特征融合，从而取得有效结果。提取序列、定量和语义特征向量后，将它们组合并输入模型，以理解日志事件之间的关系。该框架能够从事件日志中提取内在依赖关系。LogST [51] 是一种日志异常检测方法，它融合了从日志中提取的语义特征和主题特征。该过程使用 BERT 和奇异值分解（SVD）提取语义特征。此外，潜在狄利克雷分配（LDA）用于提取主题特征。随后，将组合后的特征输入改进的时间卷积网络模型，该模型利用加权残差连接进行异常检测。 Han等人 提出了 Log-MatchNet，这是一种用于检测日志数据中异常的方法，旨在处理非结构化且类别不平衡的日志数据。该方法采用日志解析，将日志内容信息转换为向量，并利用

BERT 模型进行通用特征提取。该方法采用匹配网络，学习输入向量与原型向量之间的相似度分数。原型向量代表少量带标签的异常日志。这些向量用于在少样本场景中对未知日志样本进行泛化。Yu 等人提出了 LogMS，这是一种日志异常检测技术，它利用多种信息来源并采用概率标签估计。该系统使用基于多源信息融合的长短期记忆网络处理语义、序列和定量信息，并使用基于概率标签估计的门控循环单元网络处理伪标注数据。当第一阶段未能识别异常时，系统会启动第二阶段以优化效率。即使日志数据随时间发生变化，LogMS 也能高效地识别异常。 Zhang 等人 提出了 MultiLog，这是一种面向分布式数据库的基于多变量日志的异常检测方法。MultiLog 首先处理每个数据库节点日志中的序列、定量和语义信息。该过程在独立估计模块中通过结合自注意力的长短期记忆网络模型完成。该系统利用带有元分类器的自编码器对节点概率进行归一化，并检测整个集群中的异常。

基于中间融合的多模态方法。 这类方法会独立提取特征并处理各个模态，直到某个阶段，再将特征图拼接起来，然后进行分类或决策。例如，Li 等人发现，先前的方法未能应对以下挑战：（1）积极开发和维护的软件系统会改变日志格式；（2）简单的监控工具可能无法发现性能问题的根本原因。因此，他们提出了 SwissLog，这是一种用于日志分析的稳健深度学习模型。SwissLog 使用 BERT 将语义信息嵌入嵌入向量中，并将时序信息存入另一个嵌入向量。SwissLog 将语义和时间嵌入向量的中期融合结果输入基于注意力的双向门控循环单元模型，以学习正常与异常日志序列之间的差异。Zhang 等人提出了 LogAttn，这是一种利用自编码器进行无监督日志异常检测的方法。LogAttn 首先将日志数据解析为结构化表示，并生成事件计数序列和语义向量序列。时序卷积网络和深度神经网络编码器分别学习日志数据中的时间相关性和统计相关性。解码器利用注意力机制形成的潜在表示重构日志序列，该机制会衡量特征重要性。通过将重构误差与预设阈值进行比较，基于完全使用正常日志数据训练的模型即可实现异常检测。Niu 等人提出了 FSMFLog，这是一种采用多特征融合来检测日志异常的方法。FSMFLog 利用前缀树结构从词列表中提取语义数据，而非依赖通常无法捕获关键语义的典型日志模板，从而克服了先前日志解析技术的局限。它首先通过预处理移除日志中的变量，随后采用启发式策略对日志语句进行分组。FSMFLog 使用经过注意力机制改进的双向门控循环单元模型。该模型利用语义、时间和类型特征。Yang 等人提出了一种将自注意力机制与双向门控循环单元模型相结合的日志异常检测方法。该方法使用两个双向门控循环单元模型：一个负责处理日志模板

序列，另一个专门处理日志模板频率向量。两个模型的结果经过拼接后输入自注意力层，该层融合特征，以预测后续日志模板的概率分布。当候选集合中缺少预期模板时，该方法便会检测到异常。Niu 等人提出了一种名为 WDLog 的稳健基于日志的异常检测框架，该框架融合宽度与深度学习，以应对演变中的日志数据所带来的异常检测挑战。该方法首先优化日志模板提取流程，在传统 Drain 算法中融入语义嵌入和聚类，从而有效降低模型对日志措辞变化的敏感度。随后，WDLog 从生成的日志模板中提取三类特征：时间特征、不变特征和统计特征。然后，结合使用带注意力机制的门控循环单元模型处理时间特征，并使用梯度提升决策树模型处理不变特征和统计特征，以进行异常检测。

基于后期融合的多模态方法。 在这些方法中，每种模态先独立处理和分类，随后再进行融合。接着，将各模态的结果合并，以作出最终决策。例如，Zhao 等人 提出了一种用于检测在线 Web 日志中网络扫描行为的方法，这对于识别潜在的网络攻击至关重要。该方法采用三个轻量级分类器，分析从 Web 流量日志中提取的不同特征，例如 HTTP 文本内容、状态码和请求频率。该方法根据源 IP 和目标 IP 对日志进行分类，并评估这些日志的特征，以区分扫描行为与正常行为。每个分类器处理特定的特征。因此，文本特征由多层感知机（MLP）分析，而 HTTP 状态码和请求频率则由支持向量机（SVM）评估。分类器生成表示扫描行为的概率，并通过决策策略将这些概率结合起来，以确定最终的分类结果。

2.2 用于多模态情感分析的深度学习

传统情感分析流程通常专注于从单一模态中提取用户情感。多模态情感分析将视觉和音频模态纳入基于文本的情感分析。与冗长的词语列表相比，音频和视觉特征能够更好地解释或描绘情感，因此会被加以利用。许多研究[94–103] 已在这一领域开展，其中大多数依赖深度学习理念。深度学习日益普及，原因在于它能够增强复杂处理能力，从而改进自动化流程。从多模态学习的角度来看，全面的综述 [118, 119] 阐述了多模态机器学习的基本原理、挑战和开放问题。这些研究强调了模态的系统整合、平衡与交互——这些都是 CoLog 所要解决的核心问题。此外，Qiu 等人 提出了一种用于多模态情感分析的链式交互注意力机制，进一步验证了情感领域开发的多模态融合策略能够为异常检测的进步带来启发。Delbrouck 等人 提出了一种 Transformer 编码器架构，可融合任意

模态信息。该模型采用联合编码，同时对各个模态进行编码，并通过模块化协同注意力控制模态对自身的关注方式。Hu 等人 提出了一个名为 UniMSE 的框架。UniMSE 是一种用于统一多模态情感分析和对话情绪识别的新方法。该方法将任务重新表述为生成式格式，整合声学、视觉和文本特征，并采用模态间对比学习来最小化类内方差、最大化类间方差。Li 等人 提出了一种采用 Transformer 架构和软映射的多模态情感分析模型。Transformer 层利用注意力层映射各个模态，而软映射层则通过堆叠模块实现多模态信息融合。该模型通过考量多个模态之间的关系，解决了多模态情感分析中的数据交互问题。TETFN [97] 是一种多模态情感分析方法，通过整合文本、视觉和音频模态来增强视频情感识别。该方法使用预处理、特征提取、LSTM 网络和 TCN 对上下文信息进行编码，并生成单模态标签。TETFN 的核心是一个文本增强型 Transformer 模块，该模块采用面向文本的多头注意力（MHA）机制，将文本信息融入视觉和音频模态，从而实现有效的成对跨模态映射。Ahuja 等人 提出了一种针对含有文本信息图像的新型多模态情感分析方法。首先，该方法使用以高光学字符识别（OCR）准确率著称的 Google Cloud Vision API，从图像中提取文本。随后，对提取的文本进行预处理，并使用基于 RoBERTa 的模型开展文本情感分析。同时，视觉情感分析采用迁移学习方法来捕捉视觉特征。该方法通过加权融合策略整合两种分析的输出，从而提高整体情感分类的准确率。Liu 等人 提出了一种鲁棒的多模态情感分析模型，采用基于模态转换的方法处理不确定的缺失模态。该方法将视觉和音频数据转换为文本模态，对文本进行编码，并融合为缺失的联合特征（MJFs）。随后，Transformer 编码器模块对这些联合特征进行编码，以学习模态间的长期依赖关系。因此，情感分类基于 Transformer 编码器模块的输出进行。

2.3 超越日志特定方法

除了日志特定方法之外，多模态异常检测也已在视频理解等其他领域得到探索。例如，Su 等人提出了一种用于视频异常检测的语义驱动的双重一致性学习方法，该方法利用语义驱动表示对齐外观和运动特征，从而在弱监督下增强异常定位。同样，Su 等人提出了一种采用局部到全局专家混合的联邦弱监督视频异常检测方法，该方法在联邦学习框架中运用专家混合，以应对分布式和异构数据场景。

虽然这些方法针对视频模态，但在语义引导融合和跨模态一致性学习方面，它们与 CoLog 在概念上有相通之处。受这些原则的启发，CoLog 的协作式 Transformer 通过印象注意力整合语义和

序列日志模态，并借助模态适配层（MAL）实现对点异常和集体异常的统一检测。

2.4 我们的方法：更胜一筹

根据第 2.1 节引用的文献，研究者正逐步采用多种深度学习技术来检测日志记录中的异常。我们将这些技术分为两大类和六个子类。图 2 简要概述了基于日志的异常检测。CoLog 属于基于多模态的方法。

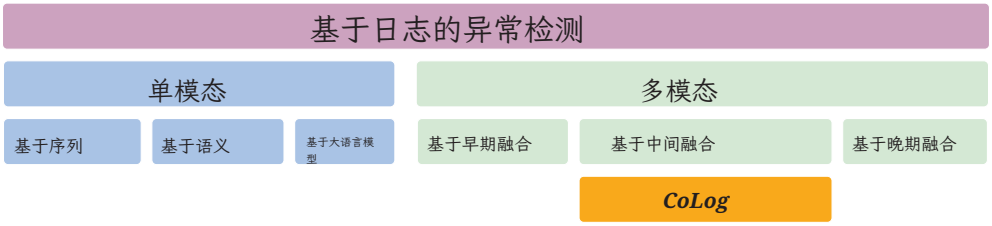


图 2 现有文献中关于识别日志数据异常方法的全面概述。

然而，我们的方法有所不同：我们旨在基于多模态情感分析和协作式Transformer，利用系统日志中各模态之间的冗余。协作式Transformer可用于补充不同的模态。当某个特定模态的数据不足时，便使用其他模态的数据。因此，这些Transformer能够从各模态中提取多种信息，并以交互方式对各模态进行编码。

正如[38, 52, 53, 58], 所称，基于情感的日志分析模型具有出色的性能和泛化能力，这使得经过充分训练的模型能够更好地适应日志语句的演变。我们提出了基于日志情感分析开发的CoLog框架。不过，与 [38, 52, 53, 58], 不同，我们的模型采用了更复杂的架构来进行基于多模态情感分析的异常检测。通过根据日志事件的背景评估其情感，CoLog能够分析点异常和集体异常。此外，如图1所示，CoLog能够利用事件发生背景中的辅助信息来区分相似事件。在接下来的章节中，我们将说明事件日志的上下文也可以加以利用。

与所评述的方法不同，CoLog依赖于日志内部的模态交互。它不会以任何方式将不同模态拼接成一个超向量。本文的挑战通过基于协作式Transformer的印象注意力加以解决。这种注意力有助于发现两种模态之间的相关性。由于深度学习模型性质复杂，超出人类的理解能力，因此通常被称为黑盒模型；理解每项决策背后的推理过程，是超出人类认知能力的任务。采用印象注意力机制有望提升我们的日志异常检测模型的可解释性。这是因为基于印象注意力计算出的注意力分数，

能够为模型的跨模态决策过程提供有价值的见解。CoLog将各模态的结果投射到Transformer模块上方的潜在空间中，以进行中期融合，并同时执行编码过程。由于不同模态本质上并不兼容，因此构建最优潜在空间至关重要。

此外，CoLog 架构能够通过统一框架识别单点和集体日志异常。据我们所知，CoLog 是首个执行此任务的方法。

3 预备知识

本节将讨论任务的定义，以及采用Transformer模型完成该任务的理由。此外，还将介绍用于评估潜在威胁的假设和威胁模型。

3.1 任务定义

此任务旨在进行日志分析，以检测系统运行期间出现的异常。分析范围包括分析日志信息以检测异常。我们将使用机器学习算法分析这些数据。通过测量准确率、精确率、召回率和 F1 分数等指标，评估异常检测方法的性能，以确保该方法不会产生过多误报。

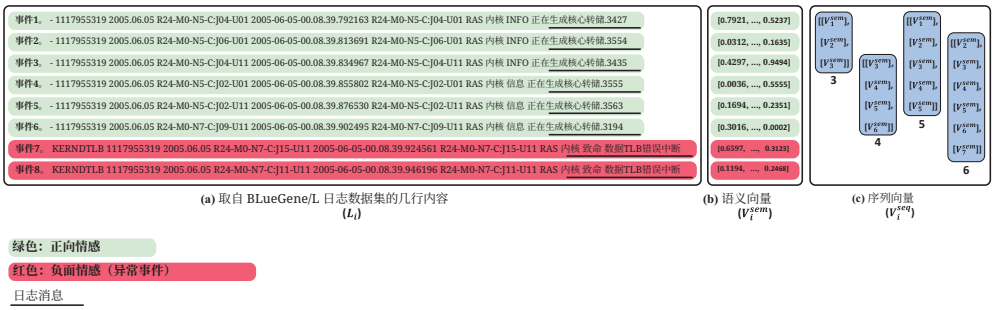


图3 (a) 从 BlueGene/L 日志数据集中提取的几行内容；(b) 语义模态：语义模态由从日志事件消息中提取的语义向量构建；(c) 序列模态：序列模态的构建方式是根据窗口大小为3、4、5和6，将语义向量追加到序列向量中。

在进行多模态情感分析时，研究人员会从不同视角（模态）考察对话，以判断参与者的情感状态。最常用的三种模态是文本、音频和视觉。同样，日志也具有类似人类对话的特性，能够从不同方面传递信息，例如统计、时间、语义以及基于事件序列的信息。不过，在日志中，最值得关注的是语义和序列模态。此外，为了检测日志中的异常，我们

可以采用情感分析流程来对情感进行分类，其中识别出负面情感即可提示存在异常情况。在图 3(a) 中，日志消息的负面和正面情感状态分别以红色和绿色表示。基于这些观察，我们提出将多模态情感分析应用于日志异常检测任务。图 3(b) 和图 3(c) 分别展示了语义和序列模态。这些图显示，日志事件的语义向量是根据日志消息生成的。随后，这些向量会根据窗口大小生成序列向量。在本例中，窗口大小决定了所生成序列向量的长度。这些向量将作为基于 Transformer 的多模态框架的输入，用于学习跨模态依赖关系并预测点异常和集体日志异常。

3.2 Transformer：为何选择?

Transformer 在处理序列时，展现出显著的速度优势。它们会更加关注关键组成部分，因此相比其他模型，处理速度更快。此外，Transformer 在多模态情感分析方面也有多项优势。Transformer 可以增强模型在多模态情感分析中的鲁棒性，这构成了另一项优势。它们还能够将印象注意力纳入架构，以从非对齐多模态数据中提取跨模态交互。此事之所以重要，是因为大多数现实世界信息都具有复杂性和多模态性。鉴于 Transformer 的这些优势，我们决定以 Transformer 为基础构建多模态模型。

3.3 假设

基于日志进行异常检测，很大程度上取决于日志的质量。存在多种可能未记录在系统日志中的攻击。如果某些可审计事件（例如登录尝试）未被妥善记录，那么 CoLog 很可能无法发现这类异常。另一种可能未记录在日志中的攻击是日志伪造或日志注入攻击。前者是一种中间人攻击，未经身份验证的用户介入应用程序与服务器之间。因此，我们假定对手试图修改系统日志记录行为是不可行的。与 pylogsentiment 类似，在训练期间，如果内核日志中的日志条目（如 *[*c010ce*54>]*mtrr wrmsr*+0* xf/0x2e）不提供人类可读的日志消息，我们就将其视为具有正面情感。需要澄清的是，我们将异常类比为“负面情感”，并不意味着它们与人类情感直接对应。相反，这是一种分析性抽象：异常是偏离预期行为的情况（负极性），而正常和良性事件则符合预期状态（正极性）。人类情感源自主观感知，而日志极性则由语义嵌入和序列模式结合上下文来确定。例如，“重启”事件在维护场景中可能是良性的，但若在意料之外发生，则可能是异常。

3.4 威胁模型

异常是偏离正常模式的数据实例。异常有多种类型，包括点异常、群体异常和上下文异常。点异常指的是与数据集总计模式显著偏离的单个数据点。一组数据实例可以形成一种异常模式，称为群体异常。上下文异常是指某个数据实例仅在特定上下文中表现异常，而在其他上下文中则被归类为正常。在特定上下文中考察时，点异常或群体异常也可能被归类为上下文异常。CoLog 基于协作式Transformer构建统一框架，以识别点异常和群体异常。该框架会根据日志条目发生时的背景或上下文评估其情感，并将任何表现出负面情感的条目标记为异常。

4 方法

本节介绍 CoLog 的架构和数据流，以及 CoLog 用于收集和检查日志数据以查找异常的具体流程。内容包括数据收集、标注和预处理的条件，规定如何执行异常检测任务的指导原则，以及用于基于 MSA 自动进行基于日志的异常检测的深度学习方法和技术。

4.1 总体架构

图4 展示了 CoLog 的训练框架，并概述了其架构。与大多数日志异常检测技术一样，CoLog 首先对原始日志进行预处理。首先，从原始值日志中提取语义和序列模态。将模态的嵌入向量输入模型之前，Tomek 链方法会处理类别不平衡问题。

将两个称为协作式Transformer的 Transformer 块应用于这些模态，并采用一种称为印象注意力的改进 GA 机制。CoLog 由 T 个相同的 CT 块堆叠而成，但每个块都有不同的训练参数。每个模态编码器都包含多头印象注意力（MHIA）、多层感知机、模态适配层（MAL）和层归一化（LNs）。输入模态被送入 CT 中相应的模态编码器。多层感知机采用非线性激活函数，并接收注意力层计算出的注意力分数。每个编码器在 MHIA、多层感知机和 MAL 层之后都设有残差连接和层归一化。编码块的结果被送入平衡层（BL），以判定日志消息的情感；在此过程中会进行软注意力和融合。请注意，负面情感与任何潜在异常行为相关联。

由于协作式Transformer（CT）中不同模态之间的信息共享，所提出的架构因此能够便捷地学习模态间的共享表示。

数字对象标识符： 10.1038/s41598-025-27693-4

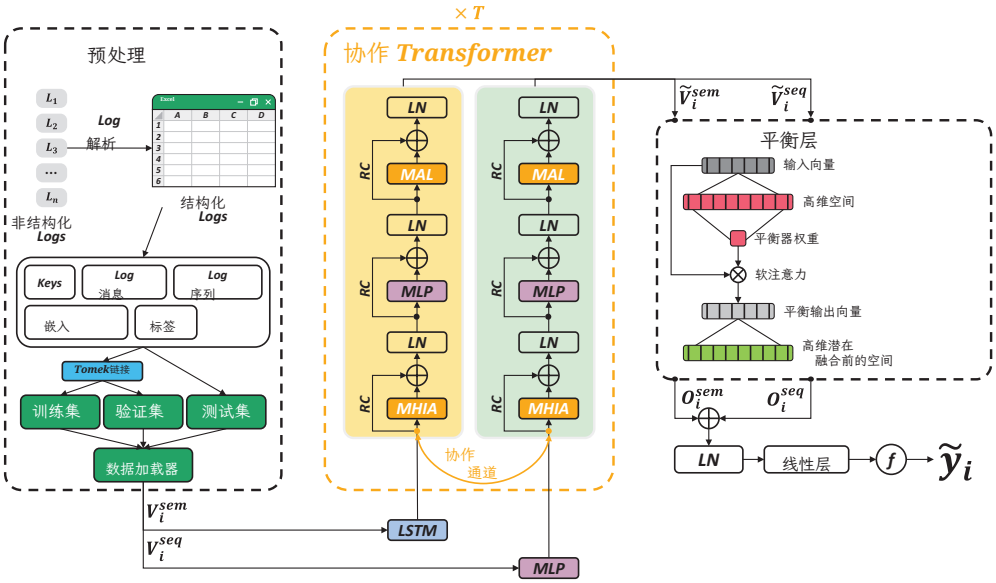


图 4 CoLog 概览。浅绿色和金色表示模态编码器。协作 Transformer 中的每个编码器均由多头印象注意力、MLP、MAL 和层归一化组成。多头印象注意力和 MAL 分别是多头印象注意力模块和模态适配层模块。预处理层将非结构化日志转换为模型易于理解的数据。平衡层旨在调节不同模态对最终结果的影响。

4.2 预处理

在预处理阶段，日志解析过程会从日志文件中提取一个特定元素，称为日志消息。随后，将每条日志消息分割为词元，并转换为小写。此过程基于训练数据集构建词汇表，生成包含 V_{size} 个不同词项的词典。之后，将每个词映射为一个 300 维向量。如果验证数据集或测试数据集中出现的词项不在我们提取的词汇表中，则用“UNK”词元替代。此外，单条日志消息的长度并不一致。因此，如果日志消息的长度低于指定阈值，就用零进行填充；反之，如果长度超过设定的阈值，则进行截断，使其缩短至预定义的消息长度。

4.2.1 日志解析

每个日志条目都包含多个彼此独立的部分。日志解析是将非结构化日志数据转换为结构化格式的过程，以便机器进行解释。每个日志文件都会经过解析，将其从非结构化形式转换为包含时间戳、服务名称、级别信息和日志消息等实体的结构化格式。我们必须将日志消息与日志条目分离，因为只有日志消息包含情感。此外，每条日志消息都会被转换为384维向量

使用SBERT生成，并与日志消息分开存储。在我们的实现中，我们采用nerlog-解析器 [123] 和 Drain [124] 日志解析器。实际上，每个数据集都会使用其对应的解析器进行处理和解析。

4.2.2 日志解析器：Drain

Drain 是一种具有代表性的日志解析算法。Drain 使用固定深度解析树，其中编码了专门创建的解析规则，以加快解析过程。Drain 首先使用正则表达式和用户定义的领域知识对日志进行预处理。其次，Drain 将预处理后的日志消息置于解析树的根节点。解析树的第一层节点表示日志组，其中的日志消息长度各不相同。Drain 使用一种称为词元相似度的距离概念来判断日志消息之间的相似程度。日志消息开头位置的词元用于选择后续的内部节点。随后，Drain 通过计算各日志组的日志消息与日志事件之间的相似度，判断是否将该日志消息添加到现有日志组中。Drain 的效果和准确性都相当出色。它还能够实时解析日志数据，因此非常适合日志异常检测。此外，
Drain 具有良好的可扩展性，适用于大规模环境。

4.2.3 日志解析器：nerlogparser

nerlogparser 使用命名实体识别（NER）技术，从文本中提取命名实体。
Nerlogparser 将日志文件中的命名实体识别为词或短语，其中包含日志条目中常见的字段，例如时间戳、主机名或服务名称。命名实体提取是识别日志条目中各个元素的过程。为执行 NER，nerlogparser 使用双向长短期记忆网络。nerlogparser 的主要优势在于能够使用预训练模型自动解析日志数据。因此，无需制定任何规则或正则表达式。nerlogparser 经过多种日志类型的训练，因此能够解析各种日志文件。

4.3 任务形式化

我们使用记号 $L_i^m, m \in \{sem,\ seq\}$ 表示从日志消息 $\{L_{i-w},\ L_{i-(w-1)},\ ...,\ L_{i-1},\ L_i\}$ 中提取的单模态原始值日志模态，其中w 为窗口大小，记号 $\{sem,\ seq\}$表示两种不同类型的模态：语义和序列。包含情感词的已解析原始日志消息用数学形式定义为 $L = \{L_1,\ L_2,\ \ldots,\ L_{|L|}\}$，其中从日志条目 i中提取的原始日志消息 L_i ， $|L|$ 表示日志消息总数。任务形式化统一了模型块的架构、模型参数和标签空间，其中两个 Transformer 块接收语义和序列模态。这些 Transformer 学习日志数据的表示。最后，我们的模型尝试预测整数值 $\tilde{y}_i \in \{0,\ 1\}$ ，以对日志条目 i进行分类。

任务制定分为两个小节。输入特征的制定描述了将原始日志模态转换为输入特征向量的过程，以供

模型使用。然而，标签制定将多模态情感分析应用于日志异常检测任务，将日志消息标签转换到一个表示情感状态的空间中。

4.3.1 构建输入特征

对事件情感的理解主要依赖于日志消息中的语义信息。基于这一观察，构建语义模态需要对每条原始日志消息进行预处理，并将其分割为来自 t_1 到 t_n 的词元，作为原始词。随后，将每个词元添加到一个列表中，该列表保留以分词文本表示的经过预处理的日志消息：

$$L_i^{sem} = [t_1,\ t_2,\ \ldots,\ t_n]. \tag{1}$$

其中，

$$n = number\ of\ tokens \tag{2}$$

之后，利用 word2vec [125] 为 L_i^{sem}中的词元创建嵌入向量，从而获取词嵌入向量。

此外，每条日志消息的原始序列特征都会转换为数值序列向量。日志序列向量可通过两种方式生成：背景和上下文。如图 5 所示，可以为每个日志条目生成背景事件列表和后续事件列表。通过拼接背景事件向量，可以生成背景序列向量。所得向量的维度为（$W \times k$），其中 W 表示窗口大小。基于上述概念，L_i^{seq} 可定义如下：

$$L_i^{seq} = [[L_j]]. \tag{3}$$

其中，

$$j \in \{i - W,\ i - (W-1),\ \ldots,\ i-1\} \tag{4}$$

类似地，上下文序列向量由背景事件向量和后续事件向量拼接而成，其维度大小为 $2W \times k$。

$$L_i^{seq} = [[L_j]]. \tag{5}$$

其中，

$$j \in \{i - W,\ i - (W-1),\ \ldots,\ i-1,\ i+1,\ \ldots,\ i+(W-1),\ i+W\} \tag{6}$$

是否使用公式 3 还是公式 5，取决于所采用的序列模态是否经过背景或上下文提取操作。

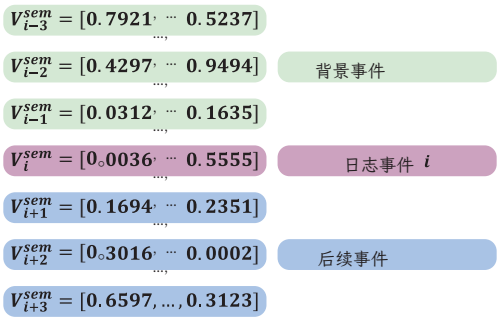


图 5 背景事件向量与后续事件向量之间差异的示意图。背景序列向量和上下文序列向量分别基于背景事件向量和后续事件向量构建。用数学术语表示， $B_i = [V_{i-3}^{sem}, V_{i-2}^{sem}, V_{i-1}^{sem}]$ 和 $C_i = [V_{i-3}^{sem}, V_{i-2}^{sem}, V_{i-1}^{sem}, V_{i+1}^{sem}, V_{i+2}^{sem}, V_{i+3}^{sem}]$，其中 V_i^{sem}是由 SBERT 从日志消息 i 中提取的语义向量， B_i 是日志消息 i的背景序列向量， C_i 是日志消息 i的上下文序列向量。

4.3.2 标签制定

CoLog 的目标是利用多模态情感分析检测日志异常，并预测日志消息 i 所反映的情感。在情感分析中，数值 1 通常用于表示正向情感，而 0 通常用于表示负面情感。识别负面情感的任务可视为等同于检测日志异常。因此，数据集中的每个实例都被赋予标签 1 或 0，分别对应其被分类为正常样本或异常样本。由此，通用标签 $y_i = \{y_i^p\}$ 包含日志消息 i 的极性，其中 $y_i^p \in \{0, 1\}$。

使用能够体现负面情感的词语为同一数据集中的未标注数据集添加标签。未标注数据集的负面情感词列表依据 pylogsentiment [38] 中提出的条件制定。

4.3.3 拉平竞争环境：类别不平衡

类别不平衡是机器学习中的普遍问题，其特点是训练数据中各类别的分布不均。这可能导致模型偏向多数类，从而在少数类上的性能不佳。为解决类别不平衡问题，人们提出了多种技术，包括数据层面方法，例如过采样方法 [126, 127] 和欠采样技术 [128, 129], ，以及算法层面方法，例如集成方法 [130, 131]。上述技术旨在提升机器学习模型处理不平衡数据集时的效率，具体做法是纠正数据不平衡，或调整学习算法以考虑类别分布。

为平衡类别，我们采用名为 Tomek链接[129] 的欠采样技术。Tomek链接用于欠采样，以生成多数类的新分布。日志数据中经常包含重复的多数类，可将其视为噪声。因此，我们选择了这种常用于消除噪声样本和边界多数类样本的方法。A

Tomek链接由属于不同类别且互为最近邻的两个数据项构成。假设整个数据集记为 $L = \{L_1, L_2, \ldots, L_{|L|}\}$，则平衡后的数据集 $L_{balanced}$ 可用数学公式定义如下：

$$L_{balanced} = tomeklink(L). \tag{7}$$

其中，$tomeklink(\cdot)$ 是一种旨在从数据集 L 中移除 Tomek链接（L_i，L_j）的函数。若不存在 L_k，使其与（L_i，L_j）中任一成员的欧几里得距离小于 L_i 与 L_j 之间的欧几里得距离，则（L_i，L_j）为一个 Tomek链接。移除 Tomek链接的过程会持续进行，直到每对最近邻向量都属于同一类别。需要特别注意的是，L_i 和 L_j 属于不同类别。

4.4 协作 Transformer

注意力层是 CoLog 最关键的组成部分。为了确定日志数据各模态之间的映射关系，我们通过修改两个相同 Transformer 块的注意力层来构建 CT，使其能够以端到端方式并行学习数据表示。因此，在从一种模态学习信息时，我们利用 MHIA机制，将另一种模态中的信息作为指导。

单模态Transformer编码器的注意力机制定义如下：

$$attention(Q,\ K,\ C) = softmax(\frac{QK^T}{\sqrt{d}})C. \tag{8}$$

其中，Q、K和 C 分别是查询、键和上下文。我们使用 $Q = K = C$实现自注意力机制。查询、键和上下文向量使模型能够理解完整序列上下文中所有词的相对重要性。运算 QK^T 会生成一个平方注意力矩阵，其中包含输入矩阵 V^m各行之间的相关性；如果该输入的大小为 $N \times k$，则 $m \in \{sem,\ seq\}$。表达式 $\sqrt{d}$ 是用于调整缩放的因子，其中 $d = k$。

将多个自注意力机制堆叠起来，使其在不同位置关注来自不同表示子空间的数据，这一概念称为多头注意力（MHA），如下所示：

$$MHA(Q,\ K, C) = concat(h_1,\ h_2,\ \ldots,\ h_h)W_o. \tag{9}$$

其中，

$$\begin{aligned} h_i &= attention(QW_i^Q,\ KW_i^K,\ CW_i^C) \\ i &\in \{1,\ 2,\ \ldots,\ h-1,\ h\} \end{aligned} \tag{10}$$

其中，W^Q、W^K、$W^C \in R^{d\times d}$。我们堆叠 h 层自注意力来执行多头注意力，以学习事件之间的复杂模式。经过Transformer块编码后，将每个Transformer块的输出用于投影到用于分类的平衡层。

如图 4所示，在 CT 中集成 MHIA，需要将每个模态编码器中的 MHA 替换为 MHIA，图中以浅绿色和金色表示

。MHIA 的架构如图 6所示。MHIA 要求对所用模态进行并行编码。此外，MHIA 的实现还包括利用当前模态的 Q 向量，以及通过多头注意力机制从次要模态获取的 K 和 C 向量，计算当前模态的注意力分数。CT 架构中的每个模态编码器 Transformer都包含相应的 MHIA模块。

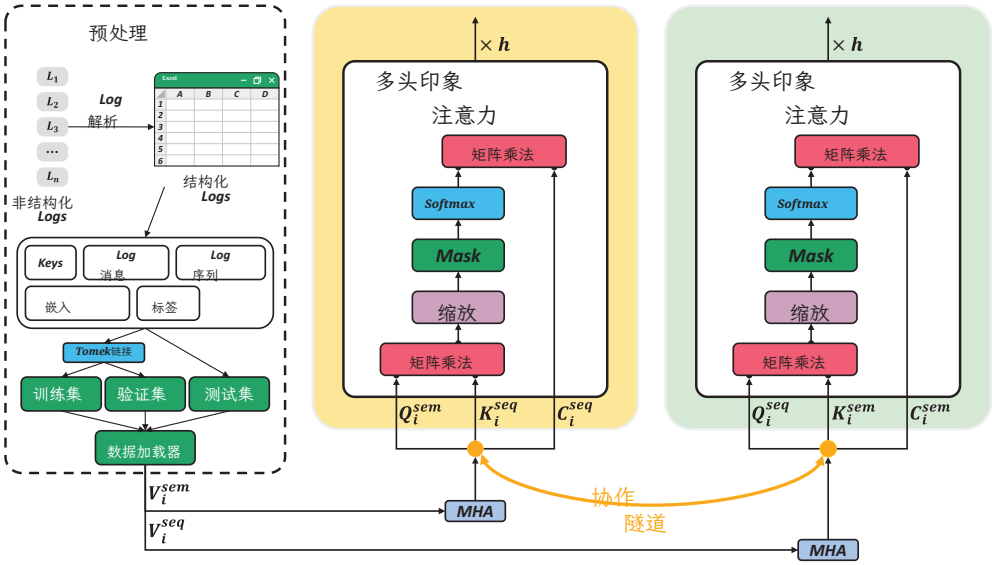


图 6 多头印象注意力层的架构。MHIA 过程涉及通过多头注意力机制，利用当前模态的 Q 向量以及次要模态的 K 和 C 向量，计算当前模态的注意力分数。根据多头印象注意力架构，各种模态会并发编码。

4.5 并发编码

并发编码过程指对每种模态同时进行编码。如上所述，这意味着在转向另一种模态之前，不会先将用于特定模态的编码块展开。平衡层通过 Transformer 编码块完成编码后，再将数据投影到潜在空间以进行分类。

4.6 模态适配层

由于不同模态在 CT 架构中协作编码，因此必须确保提取的表示不含不必要的信息。日志包含多种模态，利用其他模态的知识对每种模态进行编码，对于获得更优且信息更丰富的该模态表示至关重要。然而，由于不同模态之间存在固有差异，协作编码可能导致每种模态的表示出现不一致和噪声

。MAL 被提出作为应对这些挑战的一种解决方案。MAL 由 N 软注意力层组成，其对应的输出会堆叠在一起。每种模态 V^m， $m \in \{sem, seq\}$ 在进入 MAL 时，首先被转换到一个新的高维表示空间 V^m_{high} 。此转换通过以下公式 11 计算：

$$V^m_{high} = V^m W^m_{high}. \tag{11}$$

其中， $V^m \in R^{N\times K}$ 是 CoLog 执行 MAL 的输入矩阵，而 $W^m_{high} \in R^{k\times \mathbf{2}k}$ 是一个变换矩阵，用于将 V^m 嵌入到所有 MAL 共享的更高维空间中。

在第二阶段，必须从获取的高维空间中生成序列中每个节点的权重，以实现软注意力，并提取序列中每个节点纯粹且全局的表示。为此，我们需要 w^{node}_i、$i \in [1,\ N]$。每个 w^{node}_i的大小必须为 $1 \times 2k$，而向量集合 $\{w^{node}_i\}$则因输入序列中的节点而异。每个节点的权重根据以下概念，使用公式 12计算得出。

$$w^m = softmax(V^m_{high}(W^m_{low})^T). \tag{12}$$

其中，

$$W^m_{low} = concat(w^{node}_i). \tag{13}$$

其中， W^m_{low} 是一个变换矩阵，用于从高维空间中提取每个节点的权重。

公式 14 根据上一阶段获得的权重，按如下方式计算输入矩阵的软注意力：

$$\widetilde{V}^m_i = softattention(V^m) = \sum_{j=0}^{N} w^m[i][j] \odot V^m[j]. \tag{14}$$

其中， $\widetilde{V}^m_i$表示序列中单个节点的计算输出， $\odot$ 表示哈达玛积。

对于序列长度为 N 的输入矩阵 V^m ，MAL 的输出$\widetilde{\ }$可定义为按照以下关系堆叠所有 V^m_i 向量：

$$\widetilde{V}^m_{out} = layernorm(\widetilde{V}^m + V^m). \tag{15}$$

其中，

$$\widetilde{V}^m = stacking(\widetilde{V}^m_1,\ \widetilde{V}^m_2,\ \ldots,\ V^m_N). \tag{16}$$

变换矩阵 W^m_{high}在所有节点间共享，将输入序列中的每个节点映射到高维空间，如图7所示。softmax函数和向量 w^{node}_i为每个节点分配权重 $w^m[i]$ ，且每个节点的权重各不相同 i。通过计算所有节点的加权和得到最终输出，随后将其输入平衡层和分类层，以作出最终

数字对象标识符： 10.1038/s41598-025-27693-4

决策。这样做是为了使每种模态都能识别相应的跨节点表示，并消除每种模态中的噪声。完成此任务后，每个节点向量都能从其他节点各自的表示中提取重要信息，从而有助于确定每个节点的全局情感极性。

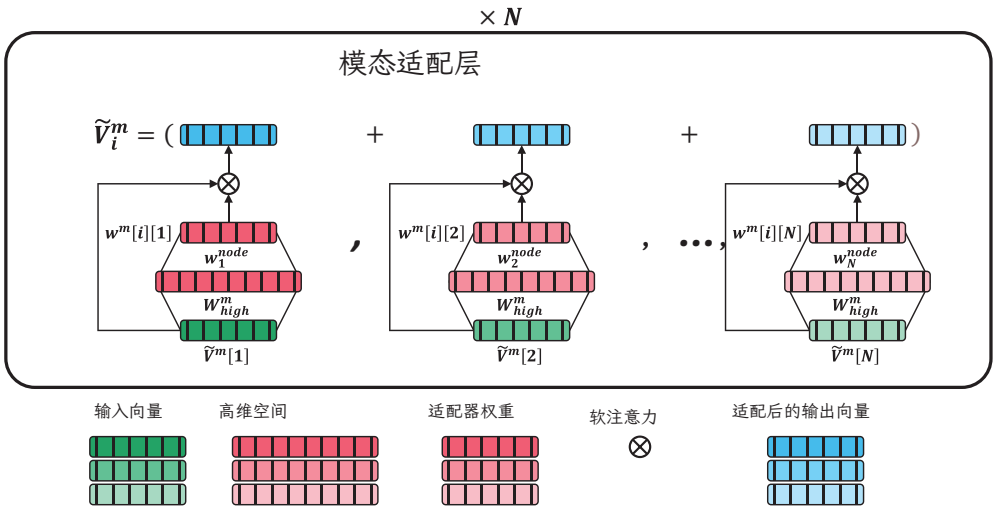


图 7 模态适配层的架构。MAL 通过为输入序列中的每个节点分配权重，为每种模态生成整体表示。它还可以去除以协作方式编码的模态噪声。

4.7 平衡层

在模态 Transformer 块的末尾，我们添加了一个平衡层，将模态投影到新的表示空间中。由于各种日志模态之间存在固有差异，CoLog 会在相同的语义空间中汇总每种模态的特征向量。因此，如图 4所示，每个模态编码器学习得到的结果都会被投影到潜在空间中，以实现有意义的融合；在这个高维空间中，CoLog 对每种模态的特征向量进行处理。此外，由于不同模态对判定输入的正常性或异常性可能具有不同的重要性，我们从各自的高维表示中提取每种模态的平衡器权重。根据平衡器权重，我们可以在融合之前通过软注意力平衡各模态的贡献。平衡层与大小为 1 的 MAL 相同，不同之处在于，所得向量会被投影到潜在空间中，以确保实现有意义的融合。在平衡层中引入潜在空间，是因为各模态的性质不同。

语义模态和平衡层的序列模态输出分别用符号 O_i^{sem} 和 O_i^{seq} 表示。

4.8 分类

最后，在投影到潜在空间后，将各模态的平衡特征向量相加、归一化，并转换为二维向量，以应用激活函数。根据图 4，在成功计算完所有 Transformer 块后，与各模态对应的向量将进行逐元素求和。随后，通过多层感知机和层归一化使用这些向量预测每个输入样本，如下所示：

$$y_i = W_{classification}(layernorm(O_i^{sem} + O_i^{seq})). \tag{17}$$

其中，$W_{classification}$ 表示多层感知机的学习矩阵。

4.9 异常检测

如果在单个变量中向模型提供多个 CoLog 输入，则 CoLog 输入可以表示为 $I = \{keys, V^{sem}, V^{seq}, embeddings, labels\}$。CoLog 接收 I，并尝试预测标签集合 y。异常检测阶段与训练过程相似。值得注意的是，CoLog 可以通过两种不同的方式检测异常：优先仅检测点异常，或同时检测点异常和集体异常。

4.9.1 点异常检测

基于第 4 节所述的方法论，CoLog 接收输入 I，并在点异常检测过程中尝试对其进行预测。此过程旨在让 CoLog 接收单个日志条目，与其背景或上下文协作进行分析，并确定其极性。

4.9.2 点异常与集体异常检测

通过统一框架进行点异常与集体异常检测，需要 CoLog 接收输入，并尝试确定单个日志事件的极性，以及其背景或上下文的极性。由此可推断，在这种情况下，CoLog 执行的是 4 类分类。这些类别包括：事件异常、事件及其背景或上下文均异常、仅事件的背景或上下文异常，以及事件及其背景或上下文均正常。因此，在这种情况下，必须标注这些事件的背景或上下文。

如果日志事件序列中有一条日志消息反映出负面情感，$L = \{L_1, L_2, \ldots, L_{|L|}\}$ 则该序列异常。基于此，还可以为每个日志事件标注背景或上下文。最后，根据第4节提供的方法论，CoLog 接收输入 I，并在点异常与集体异常检测过程中生成预测。该过程旨在让 CoLog 接收单条日志条目，结合其背景或上下文进行协作分析，并判定其极性以及背景或上下文极性。

5 实验

本节将首先介绍操作系统日志数据集。随后，我们将讨论 CoLog 的实验设置以及所提出模型的评估结果。

5.1 日志数据集

实验在基准操作系统日志数据集上进行，包括公开可用的 BlueGene/L、Hadoop 和 Zookeeper 数据集。表2 全面汇总了用于评估 CoLog 性能的所有数据集的统计信息。

表2 公开可用的操作系统日志数据集概要。

数据集	行数	正样本记录数	负样本记录数	数据大小
Spark[1]	33, 236, 604	31, 513, 147	1, 723, 457	2.75 GB
蜂蜜5[2]	124, 386	67, 798	56, 588	12.6 MB
Windows[2]	25, 000	18, 599	6, 401	3.48 MB
Casper	11, 086	9, 874	1, 212	930 KB
Jhuisi	11, 737	9, 063	2, 674	0.98 MB
Nssal	107, 093	91, 349	15, 732	8.53 MB
Honey7	8, 712	8, 162	550	734 KB
Zookeeper	74, 380	25, 873	48, 507	9.95 MB
Hadoop	394, 308	382, 870	11, 438	48.61 MB
BlueGene/L	4, 747, 963	4, 399, 486	348, 477	708.76 MB

[1]所提供的数据集用于评估 CoLog 的鲁棒性。[2]所提供的数据集用于评估 CoLog 的泛化能力。

Spark [132], Honey5 [133], 和 Windows [132] 数据集未用于 CoLog 训练。这些数据集用于测试 CoLog 的泛化能力和鲁棒性。Spark 是一种用于处理大量数据的管理解决方案。从 32 台主机获取的日志中，既记录了正常的 Spark 系统操作，也记录了异常的 Spark 系统操作。Honey5 源自蜜网项目于 2010 年举办的取证挑战赛 5。该数据集提供的是一个已遭到入侵的 Linux操作系统实例。最后一个数据集是 Windows 数据集，它汇总了香港中文大学实验室中运行 Windows 7 的计算机所产生的多份日志。Windows操作系统采用基于组件的服务（CBS）机制，以便安全、受控地安装程序。

本研究使用的这组数据集中，其余七个数据集中的第一个数据集用于模型训练。该数据集取自 Digital Corpora 提供的磁盘映像，并命名为 nps-2009-casper-rw [134]。研究人员使用可启动USB创建了 ext3文件系统的转储。该数据集包含一台运行 Linux操作系统的机器所产生的操作系统日志。数字取证研究研讨会（DFRWS）是一场年度安全会议，它提供了我们用于训练 CoLog 的第二和第三个系统日志数据集。该研讨会于 2009 年举办了一场操作系统日志取证挑战赛。DFRWS 2009年取证挑战赛通过研究日志文件，试图找出一名非法传送机密资料的入侵者。该案例涉及 jhuisi [135] 和 nssal [135] 主机。两台基于 Linux 的

Sony PlayStation 3 主机。其他操作系统日志取自蜜网取证挑战赛 7（Honey7） [136]。Honey7 包含一台被入侵的Linux服务器的磁盘映像。所有数据集的克隆磁盘映像中都包含 /var/log/ 目录。随后提取了身份验证日志、内核日志和系统日志。Zookeeper 用于管理分布式系统。香港中文大学实验室收集了来自 32 台主机、为期 26 天的 Zookeeper 日志 [132] 。大数据工具 Hadoop 可在多台机器间分发作业。一套由五台机器组成、共计 46 个核心的 Hadoop集群生成了 Hadoop 日志数据集 [137]。在该数据集中，机器停机和网络断连属于异常。劳伦斯利弗莫尔国家实验室（LLNL）的 BlueGene/L超级计算机配备 131,072 个处理器和 32,768 GB 内存，并提供了一个开放数据集 [138]。记录分为警报和非警报两类，其中警报表示存在异常行为。

5.2 实验设置

本节介绍我们用于运行实验的系统硬件和软件配置。随后，我们讨论 CoLog 的超参数以及用于评估 CoLog 的指标。

5.2.1 系统配置

我们使用一台 Colabpro机器，其中央处理器（CPU）配备 12 个核心、83.5 GB 随机存取存储器（RAM）以及一块 NVIDIA A100 40GB 图形处理器（GPU），用于执行 CoLog 和其他方法的实验。

我们使用 Python 3.10.12 开发 CoLog，并使用 PyTorch [139] 2.0.1+cu118 作为后端。我们使用 imbalanced-learn [140] 库实现 Tomek链接，并将其与其他类别平衡技术进行评估。我们利用 Python 中的 Ray库 [141] 确定模型的最优参数。我们还使用 sentence-transformers 库 [112] 从日志消息中提取句子嵌入向量。

5.2.2 超参数

Ray Python 库为 CoLog 选择最优超参数。使用上述库对 CoLog 的 108 种不同配置进行了分析。此外，还分别考察了窗口大小、类别不平衡和训练比例，以确定各项超参数的最佳取值。CT层数为 2，注意力头数为 4。批量大小为 32。CoLog 的学习率设为 5e-5，学习率衰减设为 0.5，并以此衰减率将学习率降低三次。采用 Adam优化器。由于 Adam 对 RAM 的需求极低，因此被认为适用于这种情况。训练过程最多进行 20 个训练轮次，提前停止准则设为 5。如果连续 5 个训练轮次没有进一步提升，训练过程将终止。模型预热过程包含 5 个训练轮次。丢弃率设为 0.1。语义模态和序列模态的长度均为 60，词向量的嵌入维度为 300。CoLog 模型的隐藏层维度设为 256，投影向量维度也设为 2048。

序列模态类型设为上下文，窗口大小设为 1。序列模态的维度设为 384。我们采用留出验证方案。根据以下比例将数据集分为三部分：60% 用于训练，20% 用于验证，20% 用于测试。对于 BGL 数据集，由于资源有限，上述比例设为 10%、45% 和 45%。此外，采用 Tomek 链技术解决数据不平衡问题。最后，在每个日志数据集上训练 CoLog，并在同一数据集上进行测试以生成结果。

5.2.3 指标

CoLog 的性能通过多种指标进行评估，包括精确率、召回率、F1 分数和准确率。这些指标根据真正例（TP）、假阳性（FP）、真负例（TN）和假负例（FN）的值计算。使用 scikit-learn 库 [142]，并通过“宏平均”选项计算指标。针对每个数据集，分别在单个文件上计算评估指标，然后汇总所有文件的结果并求平均值。

精确率。精确率衡量 CoLog 生成的所有正类预测中被正确识别的实例所占的比例。精确率公式如下：

$$Precision = \frac{TP}{TP + FP}. \tag{18}$$

其中，TP是正确预测的正类实例，FP是错误预测的正类实例。

召回率或敏感度。召回率用于量化 CoLog 正确检测出的实际正例所占的比例。召回率的数学表达式如下：

$$Recall = \frac{TP}{TP + FN}. \tag{19}$$

其中，FN 表示被错误预测为负例的实例。

F1 分数。F1 分数用于评估精确率与召回率之间的平衡。F1 分数的数学表达式如下：

$$F1\text{–}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2 \times TP}{2 \times TP + FP + FN}. \tag{20}$$

准确率。准确率用于量化 CoLog 所做正确预测的比例。准确率的数学表达式如下：

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}. \tag{21}$$

其中，TN 是被正确预测为负类的实例。

数字对象标识符：10.1038/s41598-025-27693-4

5.3 结果

本节讨论在 CoLog 上开展的实验，其中使用了若干图表，包括混淆矩阵、归一化混淆矩阵、受试者工作特征（ROC）曲线和精确率-召回率（PR）曲线。公开可用日志数据集上的 CoLog 综合结果见表 3。此外，这些结果的图示见图 8 至图 14。各小节安排如下。在接下来的分析中，我们将 CoLog 应用于上述数据集，并与其他日志异常检测方法进行比较。随后，我们将讨论超参数调优的结果、训练比例的影响、解决类别不平衡问题的各种方法所产生的影响、窗口大小的作用，以及 CoLog 得出的向量的展示。接下来，我们将详细考察 CoLog 的泛化能力或鲁棒性。随后，我们将通过消融研究讨论 CoLog 的重要设计组成部分。接下来，我们开展开创性研究。随后，我们将把我们的贡献与其他日志异常检测技术进行比较。

CoLog 应用于各种数据集时展现出卓越的通用性，并始终保持高水平的性能。该方法的有效性体现在其对所有测试数据集均具有很高的分类准确率，CoLog 的分类得分接近完美或达到完美。在 Casper、Jhuisi、Honey7 和 Zookeeper 数据集上，CoLog 的分类表现无懈可击，没有任何假阳性或假阴性。同样，CoLog 在 Nssal 和 Hadoop 数据集上也表现出色，假阳性和假阴性的数量很少，并且检测出了 Hadoop 数据集中的所有异常。即使应用于规模庞大的 BGL 数据集，CoLog 仍保持卓越性能，准确识别出 1,964,265 个真正例和 156,775 个真负例，仅出现少量错误，其中只有 1 个假阴性。所有数据集的 ROC 曲线和 PR 曲线的曲线下面积（AUC）值始终接近 100 或 100，进一步证明了 CoLog 的强大性能。

CoLog 的卓越表现可归因于几项重要贡献。首先，在 CoLog 中，语义日志模态与序列日志模态相互协作。除了语义特征之外，利用序列特征还扩充了可用信息，并提升了模型识别异常的能力。此外，每种模态均使用不同的神经网络，使 CoLog 能够控制模态之间的交互作用，从而获得更可靠的结果。最后，CoLog 利用潜在空间融合各模态。这一点至关重要，因为不同模态具有不同的特性，需要采用复杂的融合机制才能实现有意义的整合。这些贡献体现了 CoLog 的适应性，并证明它是在各种场景中检测日志异常的可靠工具。

表3 综合 CoLog 在公开可用日志数 据集上的结果。

数据集	平均值	类别	# 样本	精确率	召回率	F1 分数	准确率
Casper		0：异常	243	100	100	100	100
		1：正常	1976	100	100	100	
	宏		2219	100	100	100	
Jhuisi		0：异常	536	100	100	100	100
		1：正常	1814	100	100	100	
	宏		2350	100	100	100	
Nssal		0：异常	3147	99.936	99.841	99.889	99.967
		1：正常	18271	99.972	99.989	99.981	
	宏		21418	99.955	99.915	99.935	
Honey7		0：异常	110	100	100	100	100
		1：正常	1633	100	100	100	
	宏		1743	100	100	100	
Zookeeper		0：异常	9702	100	100	100	100
		1：正常	5176	100	100	100	
	宏		14878	100	100	100	
Hadoop		0：异常	2289	100	99.913	99.956	99.994
		1：正常	33893	99.994	100	99.997	
	宏		36182	99.997	99.956	99.977	
BlueGene/L		0：异常	156807	99.999	99.980	99.989	99.998
		1：正常	1964266	99.998	100	99.999	
	宏		2121073	99.999	99.990	99.994	

5.3.1 CoLog 与业界的对比研究

我们将 CoLog 与以下工具箱中提供的一些日志异常检测算法进行了比较：loglizer [111] 和 deeploglizer [24] 。loglizer 用于机器学习算法，而 deeploglizer 则用于基于深度学习的方法。这些工具箱包含五种有监督的检测方法，包括 loglizer 工具箱中的逻辑回归（LR） [143], 支持向量机（SVM） [109], 和决策树（DT）[109]，以及 deeploglizer 工具箱中的注意力双向长短期记忆网络 [62] 和卷积神经网络 [45]。此外，还包括四种无监督技术：loglizer 中的孤立森林（IF） [144] 和主成分分析（PCA） [12] 算法，以及 deeploglizer 中实现的长短期记忆网络 [36] 和 Transformer [80] 。此外，我们还评估了 CoLog 与 pylogsentiment 相比的性能。pylogsentiment 是一种用于操作系统日志情感分析的工具。它使用门控循环单元来识别异常，并根据情感对日志条目进行分类。这有助于检测潜在问题并评估系统的整体状态。

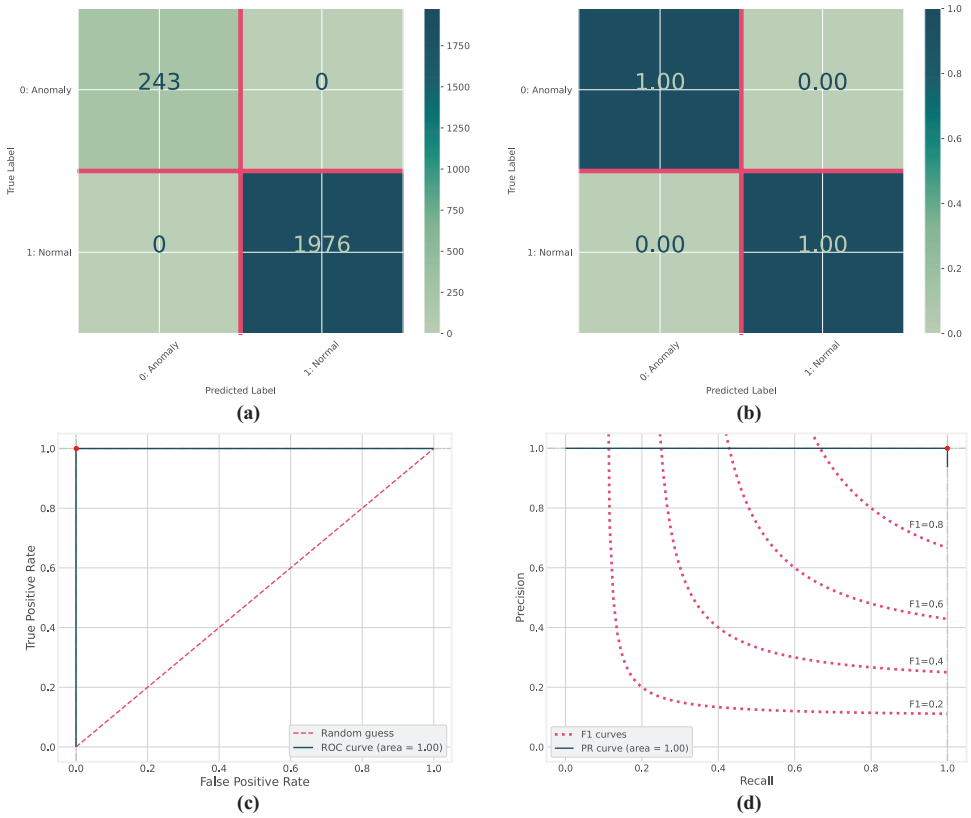


图 8 将 CoLog 应用于 Casper 日志数据集时生成的输出可视化结果。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，以及 (d) 精确率-召回率曲线

在将日志数据集输入 loglizer 或 deeploglizer 模型之前，我们使用 Drain 和 nerlogparser 工具分离所有日志记录并提取日志消息。模型的超参数设置为 loglizer 和 deeploglizer 工具箱推荐的相同配置。此外，我们采用滑动分区来执行日志分区。随后，使用 loglizer 和 deeploglizer 中的算法处理这些日志事件序列。表 4 至表 10展示了 CoLog 与其他异常检测方法的性能评估。

表11展示了所有技术在均值层面的 F1 分数。所有方法中，pylogsentiment 的性能最优，因为它在两个情感类别之间实现了平衡。这使其深度学习模型能够更有效地可靠识别少数类中的异常活动。pylogsentiment 的平均 F1 分数为 99.135。其他基于深度学习的模型，包括 Transformer、卷积神经网络、注意力双向长短期记忆网络和长短期记忆网络，在所有数据集上的表现相近。例如，它们的平均 F1 分数分别为 97.845、97.812、97.661 和 96.765。这些方法得益于采用深度学习进行异常检测，因而性能有所提升。深度学习擅长

数字对象标识符：10.1038/s41598-025-27693-4

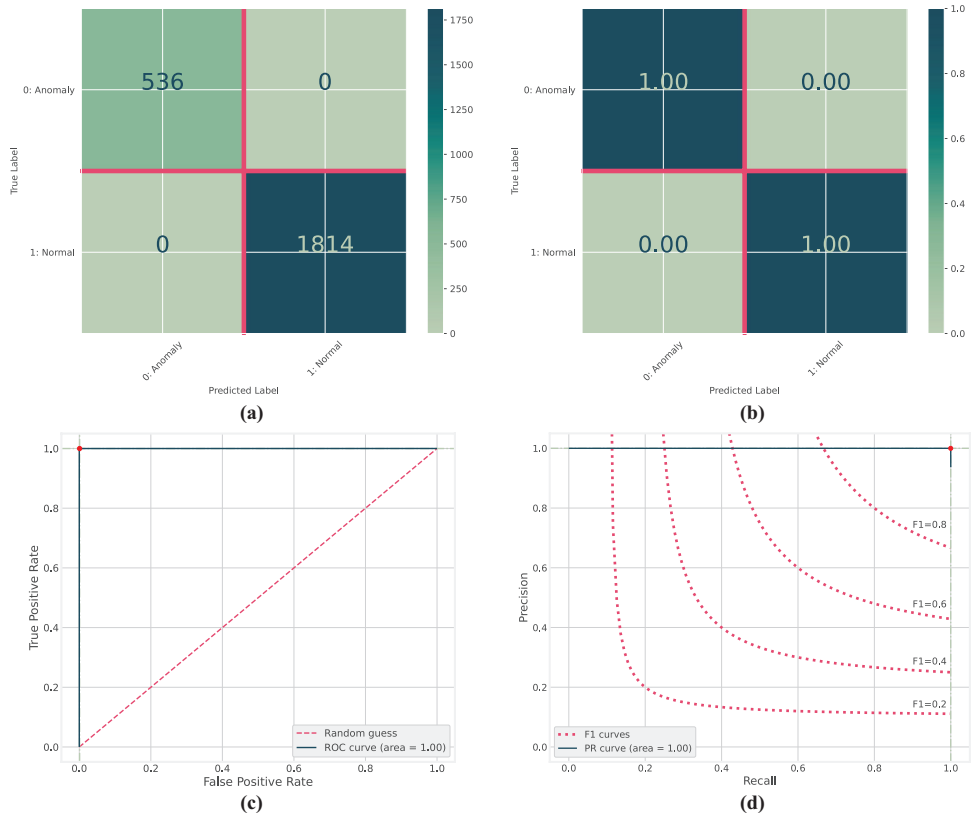


图9 CoLog 应用于 Jhuisi 日志数据集时生成的输出可视化结果。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，以及 (d) 精确率-召回率曲线

检测日志中的异常，因为它能够处理非结构化数据并自主提取特征。它可以有效处理大量数据，并且在利用日志语义时，能够适应随时间演变的异常。Transformer 的结果更好，因为它使用词级表示来表示日志条目。在表11中，孤立森林的平均 F1 分数为 49.372，而主成分分析算法的平均 F1 分数为 47.273。总体而言，机器学习方法在日志异常检测方面往往面临困难，这是因为日志数据复杂且非结构化，分布偏移会引入未见过的异常，同时还需要复杂的特征工程。此外，这些模型可能对超参数设置较为敏感，因此需要进行大量微调。

表 4 至表 10 表明，与其他技术相比，CoLog 在所有数据集上的性能均更优。此外，表 11 清楚地说明，CoLog 显著提升了异常识别效果。与排名第二的 pylogsentiment 相比，CoLog 在所有数据集上的平均 F1 分数更高。鉴于 pylogsentiment 的平均 F1 分数为 99.135，要在所用基准数据集上实现完美的异常检测，还需进一步提升 0.865。

数字对象标识符： 10.1038/s41598-025-27693-4

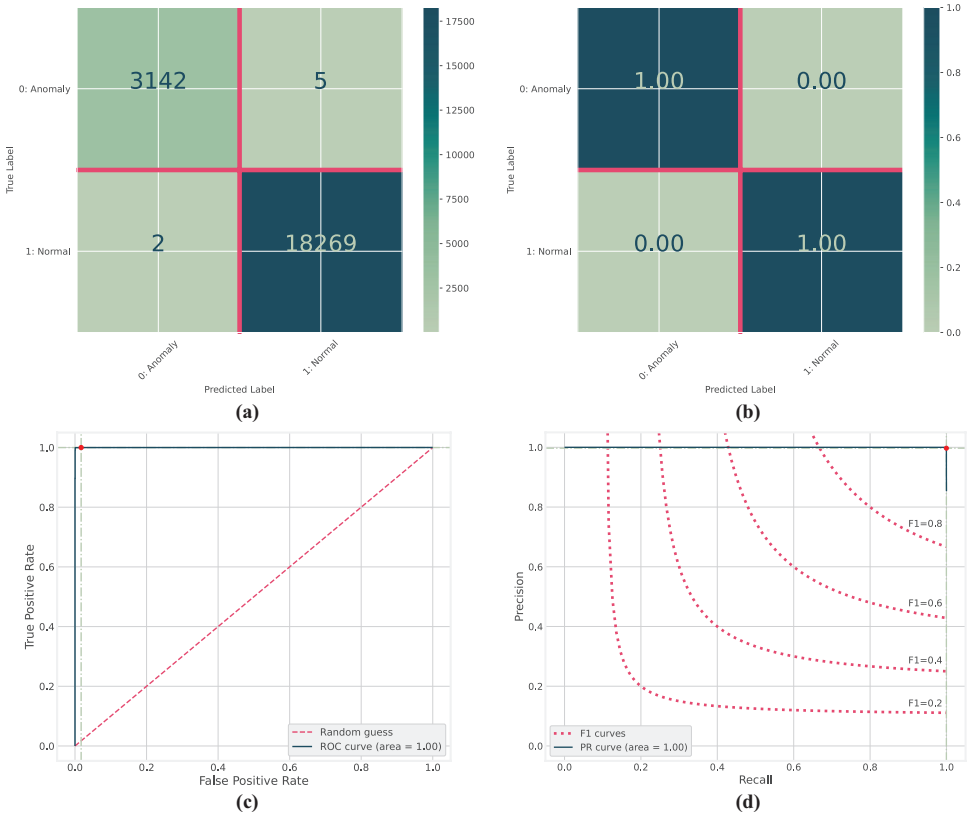


图 10 将 CoLog 应用于 Nssallog 数据集时生成的输出可视化结果。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，(d) 精确率-召回率曲线

CoLog 将该值降至 0.013。这意味着，在可提升结果的空间中，预计已改善 98.50%。

在许多情况下，在日志异常检测中，无监督方法的性能往往不如监督技术。这是因为缺少带标签数据，限制了对复杂模式的理解并降低了准确率。无监督方法也往往会产生更多假阳性。监督方法通过持续使用带标签数据进行训练来提高准确率。在本研究中，我们将日志异常检测定义为一项多模态情感分析任务，将问题从传统的无监督异常检测转变为监督分类问题。传统的异常检测方法通常将异常视为罕见或离群事件，而不为其明确标注；由于日志数据高度不平衡且变化多端，这限制了模型有效学习判别性特征的能力。通过将日志消息中的异常解读为负面情感，我们的方法运用情感分析技术为日志条目添加情感标签，从而实现监督学习。这一转变使模型能够利用带标签

数字对象标识符： 10.1038/s41598-025-27693-4

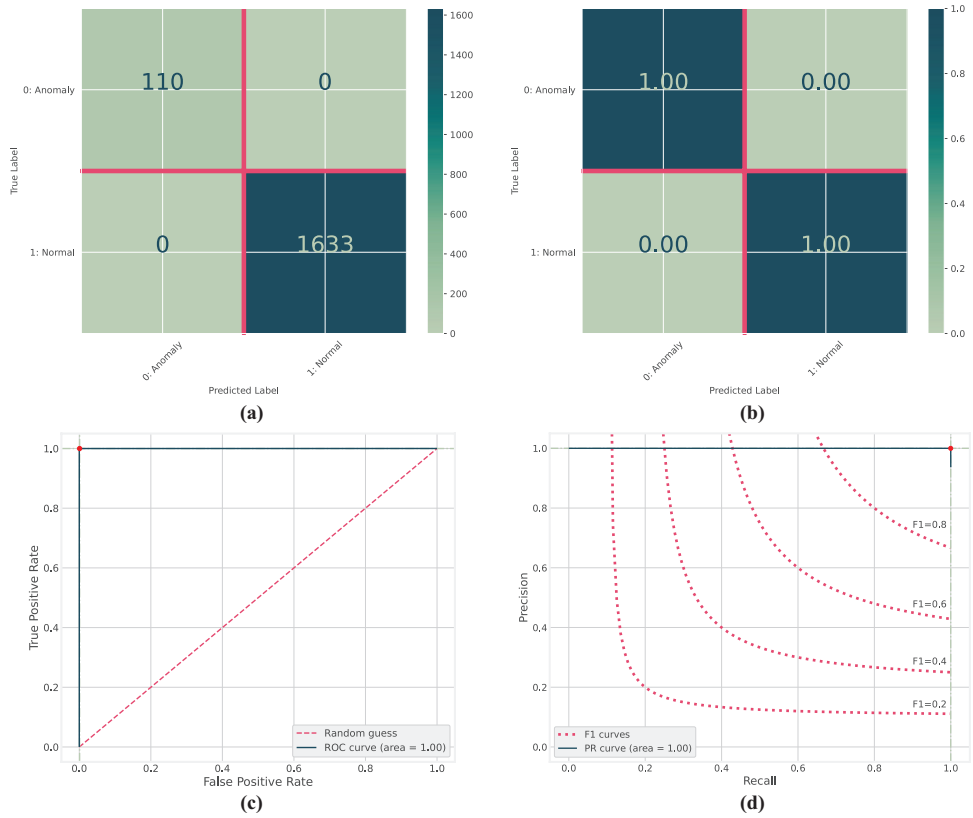


图 11 CoLog 应用于 Honey7日志数据集时生成的输出视觉结果汇总。（a）混淆矩阵，（b）归一化混淆矩阵，（c）受试者工作特征曲线，以及（d）精确率-召回率曲线

数据进行训练，通过捕捉表明异常系统行为的细微语义模式和上下文细微差异，提高检测准确率。因此，与纯粹的无监督方法相比，我们的多模态情感分析框架兼具日志丰富的语义表示和监督学习模型强大的分类能力，从而提升了检测性能。

此外，CoLog 在异常检测方面表现卓越，这归功于其通过序列和语义两种模态学习日志数据的新颖方法。与可能只关注日志数据某一方面的传统方法不同，CoLog 融合多种模态，以生成更全面的日志数据表示。这种双模态学习使 CoLog 能够更深入地理解日志数据，从而发现原本可能被忽略的信息。提升 CoLog 效能的另一项重要因素，是其获取日志表示的协作方法。不同于以往可能孤立地分析日志，或仅在不同数据模态之间进行有限交互的方法，CoLog 确保序列信息和语义信息能够同步学习。这种协作方法能够产生

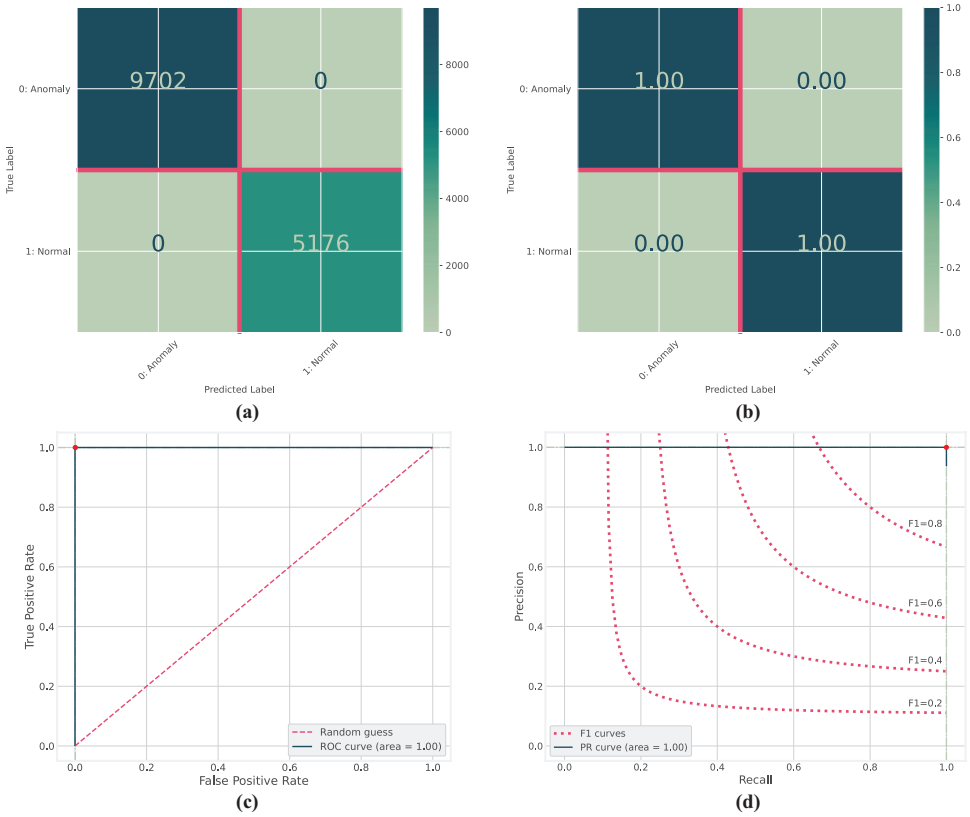


图 12 CoLog 应用于 Zookeeper 数据集时生成的可视化结果集合。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，以及 (d) 精确率-召回率曲线

更全面、更精细的日志数据表示，从而增强模型区分正常活动与异常活动的能力。联合分析日志有助于提升模型对各种日志上下文和场景的适应性。这种灵活性对于实际应用至关重要，因为日志的形式和内容可能存在显著差异。此外，在许多传统方法中，整合不同的数据模态可能会引入噪声和不一致性，最终降低性能。CoLog 的一项突出特性是其模态适配层，该层通过去除不同模态组合时可能产生的噪声，在优化性能方面发挥着关键作用。该层充当过滤器，确保只融合相关且互补的特征。通过消除冲突或冗余信息，MAL 可防止因未经过审慎考量便组合模态而导致的性能下降。CoLog 过滤这些噪声的能力确保生成的日志表示更加干净可靠，从而提高分析的总体准确率。因此，CoLog 更能应对日志数据固有的多变性和复杂性，成为领域内管理员更有效的工具。CoLog 还利用潜在

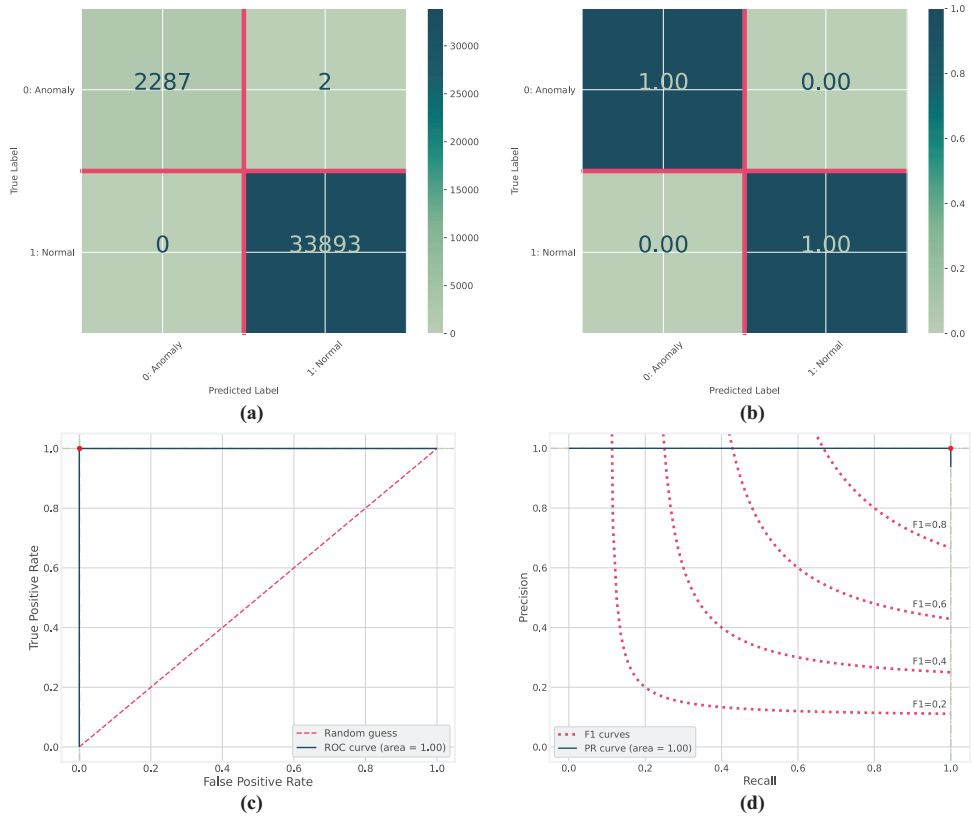


图 13 将 CoLog 应用于 Hadoop 日志数据集后生成的输出可视化结果。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，以及 (d) 精确率-召回率曲线

用于有意义地融合不同日志数据模态的空间。CoLog 并非简单地拼接各模态的特征或分别处理它们，而是探索一个共享的潜在空间，使这些特征能够得到有效整合。通过将日志数据映射到共同的潜在空间，CoLog 能够捕捉对于准确异常检测至关重要的复杂关系和依赖性。

表 12 通过记录在 Hadoop 数据集上进行训练和测试所耗费的时间，评估不同模型的效率。与 pylogsentiment 和 CoLog 相比，LSTM、注意力双向长短期记忆网络、卷积神经网络和 Transformer 方法的训练与推理所需时间明显更少。这是因为这些方法使用日志键，而不是日志的语义特征；前者所需的计算资源要少得多。CoLog 的训练时间少于 pylogsentiment，但其推理阶段的表现较差。

数字对象标识符：10.1038/s41598-025-27693-4

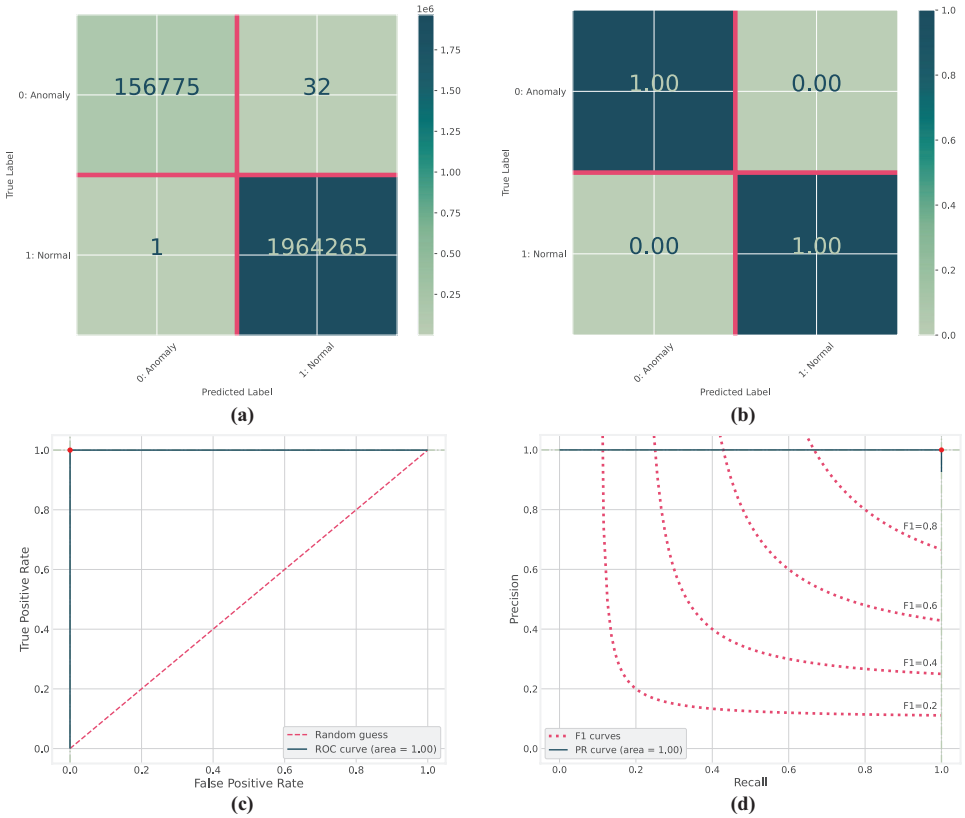


图 14 CoLog 应用于 Blue-Gene/L 日志数据集时生成的输出可视化图集。(a) 混淆矩阵，(b) 归一化混淆矩阵，(c) 受试者工作特征曲线，以及 (d) 精确率-召回率曲线

5.3.2 提升成功率：超参数调优研究

超参数调优在深度学习中至关重要，因为它会直接影响模型性能和泛化能力 [145]。这一过程可以显著提升模型的准确率和效率 [146]。适当的调优能够应对过拟合或欠拟合等挑战，这对于噪声较多且易发生突变的日志数据至关重要。此外，它还有助于检验各种模型配置，从而获得更具鲁棒性和可靠性的结果。鉴于调优在日志异常检测中的重要性，我们使用 Python 中的 Ray 软件包对 CoLog 进行调优。在调优过程中，我们使用 Casper、Jhuisi 和 Honey7 三个数据集，并在这些数据集上应用 CoLog 的 324 种不同配置。根据 Casper 、 Jhuisi 和 Honey7 数据集的调优结果，每个数据集的超参数选择标准是在最短时间内达到最佳准确率。总计最优配置由所有数据集的最优配置共同得出。由于篇幅限制，本文仅介绍每个数据集 108 次运行中效率最高的 27 次运行

表4 CoLog 与其他日志异常检测方法在 Casper 数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	66.959	60.863	59.700	90.993
支持向量机 [109]	58.614	60.757	59.482	90.967
决策树 [110]	83.466	77.037	79.488	94.998
注意力双向长短期记忆网络 [63]	99.766	99.872	99.819	99.834
卷积神经网络 [46]	99.766	99.872	99.819	99.834
pylogsentiment [38]	99.487	99.413	99.449	99.459
无监督方法				
孤立森林 [144]	52.407	50.650	49.926	88.149
主成分分析 [12]	51.205	50.282	49.362	87.480
长短期记忆网络 [36]	97.973	98.843	98.380	98.505
Transformer [80]	98.409	99.100	98.738	98.837
CoLog[1]	**100**	**100**	**100**	**100**

[1]CoLog 是一种有监督方法。

表 5 CoLog 与其他日志异常检测方法在 Jhuisi数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	68.373	66.127	64.886	79.182
支持向量机 [109]	63.643	66.506	64.051	80.914
决策树 [110]	91.534	89.769	90.550	93.313
注意力双向长短期记忆网络 [63]	95.123	97.270	96.169	99.341
卷积神经网络 [46]	97.139	94.885	95.982	99.341
pylogsentiment [38]	98.867	98.761	98.813	98.850
无监督方法				
孤立森林 [144]	57.519	52.774	51.700	74.707
主成分分析 [12]	44.828	49.299	45.014	75.448
长短期记忆网络 [36]	96.879	92.385	94.508	99.121
Transformer [80]	96.879	92.385	94.508	99.121
CoLog[1]	**100**	**100**	**100**	**100**

[1]CoLog 是一种有监督方法。

数据集（完整的微调结果可在 CoLog 的 GitHub 代码库中查看）。所选超参数详见第 5.2.2。

图15 从左到右展示了 CoLog 在 Casper、Jhuisi 和 Honey7 数据集上的前 27 次执行结果。浅绿色柱表示 CoLog 在相应配置下的准确率，玫瑰色柱则表示达到该准确率所需的总时间。因此，配置 3384、6587 和 2868 分别在最短时间内为 Casper、Jhuisi

数字对象标识符： 10.1038/s41598-025-27693-4

表 6 CoLog 与其他日志异常检测方法在 Nssal 数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	85.133	74.728	76.476	97.604
支持向量机 [109]	80.206	74.935	76.474	97.655
决策树 [110]	94.791	87.700	89.470	98.063
注意力双向长短期记忆网络 [63]	96.750	98.805	97.754	99.813
卷积神经网络 [46]	96.703	98.243	97.460	99.789
pylogsentiment [38]	97.170	96.050	96.602	99.020
无监督方法				
孤立森林 [144]	65.504	57.352	56.101	80.967
主成分分析 [12]	52.642	53.827	49.505	80.614
长短期记忆网络 [36]	96.148	97.669	96.896	99.742
Transformer [80]	96.304	99.354	97.778	99.813
CoLog[1]	**99.955**	**99.915**	**99.935**	**99.967**

[1]CoLog 是一种有监督方法。

表 7 CoLog 与其他日志异常检测方法在 Honey7数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	68.143	70.000	69.042	96.286
支持向量机 [109]	68.143	70.000	69.042	96.286
决策树 [110]	93.260	83.307	86.359	97.471
注意力双向长短期记忆网络 [63]	**100**	**100**	**100**	**100**
卷积神经网络 [46]	**100**	**100**	**100**	**100**
pylogsentiment [38]	99.970	99.107	99.535	99.943
无监督方法				
孤立森林 [144]	47.509	50.948	48.064	85.966
主成分分析 [12]	60.871	58.898	55.943	88.279
长短期记忆网络 [36]	96.212	98.810	97.429	98.155
Transformer [80]	96.923	99.048	97.932	98.524
CoLog[1]	**100**	**100**	**100**	**100**

[1]CoLog 是一种有监督方法。

和 Honey7 数据集实现了最佳结果。图15 还展示了每个数据集最优配置对应的标签。

表 8 CoLog 与其他日志异常检测方法在 Zookeeper 数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	98.369	98.464	98.416	98.562
支持向量机 [109]	97.987	97.769	97.877	98.078
决策树 [110]	98.599	98.880	98.737	98.851
注意力双向长短期记忆网络 [63]	95.783	92.928	94.300	98.387
卷积神经网络 [46]	95.121	92.662	93.850	98.252
pylogsentiment [38]	99.722	99.898	99.810	99.973
无监督方法				
孤立森林 [144]	32.607	50.000	39.473	65.215
主成分分析 [12]	79.011	51.209	42.121	66.021
长短期记忆网络 [36]	95.825	91.887	93.750	98.252
Transformer [80]	99.890	99.416	99.652	99.361
CoLog[1]	**100**	**100**	**100**	**100**

[1]CoLog 是一种有监督方法。

表 9 CoLog 与其他日志异常检测方法在 Hadoop 数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	48.523	50.000	49.250	97.046
支持向量机 [109]	48.523	50.000	49.250	97.046
决策树 [110]	48.523	50.000	49.250	97.046
注意力双向长短期记忆网络 [63]	97.640	97.955	97.792	97.902
卷积神经网络 [46]	99.719	99.847	99.783	99.955
pylogsentiment [38]	99.886	99.732	99.809	99.905
无监督方法				
孤立森林 [144]	47.702	50.000	48.824	54.034
主成分分析 [12]	49.995	49.996	49.996	58.214
长短期记忆网络 [36]	99.850	97.397	98.589	99.715
Transformer [80]	97.280	99.833	98.518	99.685
CoLog[1]	**99.997**	**99.956**	**99.977**	**99.994**

[1]CoLog 是一种有监督方法。

5.3.3 比例的影响：训练比例影响考察

训练数据不足且不平衡（例如日志数据）可能导致过拟合，即模型在训练数据上表现出色，却无法处理新数据 [147]。此外，采用合适训练测试比例训练的模型通常会表现出更优异的性能 [148]。因此，确定最佳训练测试数据比例至关重要。训练数据的百分比会影响模型的抗噪鲁棒性以及

表10 CoLog与其他日志异常检测方法在 BlueGene/L 数据集上的比较。粗体数字表示优异结果。

异常检测技术	精确率	召回率	F1 分数	准确率
监督方法				
逻辑回归 [143]	54.028	51.852	52.092	90.368
支持向量机 [109]	46.314	50.000	48.087	92.628
决策树 [110]	60.576	50.998	50.303	92.348
注意力双向长短期记忆网络 [63]	97.640	97.955	97.792	97.902
卷积神经网络 [46]	97.640	97.955	97.792	97.902
pylogsentiment [38]	99.892	99.963	99.928	99.980
无监督方法				
孤立森林 [144]	53.081	50.047	51.519	47.389
主成分分析 [12]	51.168	54.260	38.970	48.487
长短期记忆网络 [36]	97.414	98.296	97.806	97.902
Transformer [80]	97.640	97.955	97.792	97.902
CoLog[1]	**99.999**	**99.990**	**99.994**	**99.998**

[1]CoLog 是一种有监督方法。

表11 CoLog 与其他日志异常检测方法的排名。
粗体数字表示优异结果。

异常检测技术	Mean F1 分数
主成分分析 [12]	47.273
孤立森林 [144]	49.372
支持向量机 [109]	66.323
逻辑回归 [143]	67.123
决策树 [110]	77.737
长短期记忆网络 [36]	96.765
注意力双向长短期记忆网络 [63]	97.661
卷积神经网络 [46]	97.812
Transformer [80]	97.845
pylogsentiment [38]	99.135
CoLog	**99.987**

日志数据中大量存在的数据突变。经过优化选择的比例能够使模型学习数据中的潜在模式，同时保持对异常值和噪声的鲁棒性。这在实际应用中尤为重要，例如基于日志的异常检测，因为数据可能存在噪声且不可靠。我们通过在 Casper、Jhuisi 和 Honey7 数据集上应用 CoLog，评估了不同训练比例的影响；这些数据集经过预处理，采用的训练比例分别为 0.4、0.5、0.6、0.7、0.8 和 0.9。图16 展示了分析不同训练数据比例的结果。当 CoLog 应用于 Jhuisi 和 Honey7 数据集且训练比例为 0.6 时，结果最佳。Casper 数据集的最佳比例为 0.8。根据上述比例，Casper 数据集的 F1 分数为 100，

表 12 Hadoop 数据集上深度学习模型的效率。 集。

异常检测技术	训练时间（秒）		推理时间（秒）	
	总计	每个样本	总计	每个样本
长短期记忆网络 [36]	7.72	0.00112	0.14	0.00002
注意力双向长短期记忆网络 [63]	25.71	0.00372	0.21	0.00003
卷积神经网络 [46]	14.85	0.00215	0.07	0.00001
Transformer [80]	9.05	0.00131	0.09	0.00001
pylogsentiment [38]	2615.29	0.01138	94.78	0.00262
CoLog	2479.02	0.01079	124.00	0.00343

Jhuisi数据集的比例为99.879，Honey7数据集的比例为100。在日志异常检测中，找到合适的训练比例至关重要。理想的训练数据比例会显著影响深度学习模型的收敛时间，而这在此情境下至关重要。合适的比例能确保模型在训练过程中获得多样且具有代表性的样本，从而加快学习进程。比例失衡可能导致收敛缓慢，或使模型陷入局部最小值。

5.3.4 平衡探索：类别不平衡解决方法比较研究

数据不平衡在深度学习中起着至关重要的作用，会深刻影响模型的性能和泛化能力。训练数据不平衡会使模型偏向多数类，导致其在少数类上的性能不佳[149, 150]。这给自动化日志异常检测带来了巨大挑战，因为少数类往往代表着关键实例 [38]。缓解数据不平衡有助于深度学习模型对所有不平衡类别做出准确预测，从而提升其在现实场景中的实用性。人们提出了许多方法来减轻数据不平衡的影响，重点在于均衡类别分布，或调整学习过程以优先关注少数类 [126–131]。

CoLog 由于多数类中众多日志条目之间具有相似性，因此采用欠采样技术。我们使用 Tomek 链方法在所有数据集上解决类别不平衡问题。图 17 展示了 Tomek 链相较于各种类别平衡技术的有效性，包括压缩最近邻规则 [151], NearMiss [128], 邻域清理规则 [152], 单侧选择 [153], 以及应用于 Casper、Jhuisi 和 Honey7 数据集的随机欠采样。

图17 表明，在所有数据集中，Tomek链接在精确率、召回率、F1 分数和准确率方面均取得了最高均值，分别达到99.908%、99.972%、99.940%和99.957%。在Honey7数据集中，Tomek 链方法的结果略优于其他技术。然而，压缩最近邻规则和NearMiss算法在

数字对象标识符：10.1038/s41598-025-27693-4



图 15 CoLog 在 (a) Casper、(b) Jhuisi 和 (c) Honey7 数据集上进行超参数微调的图示汇总。选择最优配置的标准是在最短时间内达到最高准确率。根据这一标准，结果从左到右排序。每个数据集对应的示意图中也展示了其最优配置。

该数据集上的性能较差。在Jhuisi数据集中，除NearMiss外，所有方法均取得了良好结果。Tomek链接和邻域清洗规则技术在Jhuisi数据集中也产生了相同的结果。值得注意的是，Tomek 链方法在Casper数据集中取得了100%的完美结果。

5.3.5 窗口大小的奥妙：窗口大小影响研究

我们在所有数据集中均采用窗口大小为 1、序列类型为上下文的设置。表 13 至表 15 展示了在 Casper、Jhuisi 和 Honey7 数据集上，基于上下文的窗口大小为 1 时的性能，并将其与不同窗口大小和序列类型的结果进行了比较。

数字对象标识符： 10.1038/s41598-025-27693-4

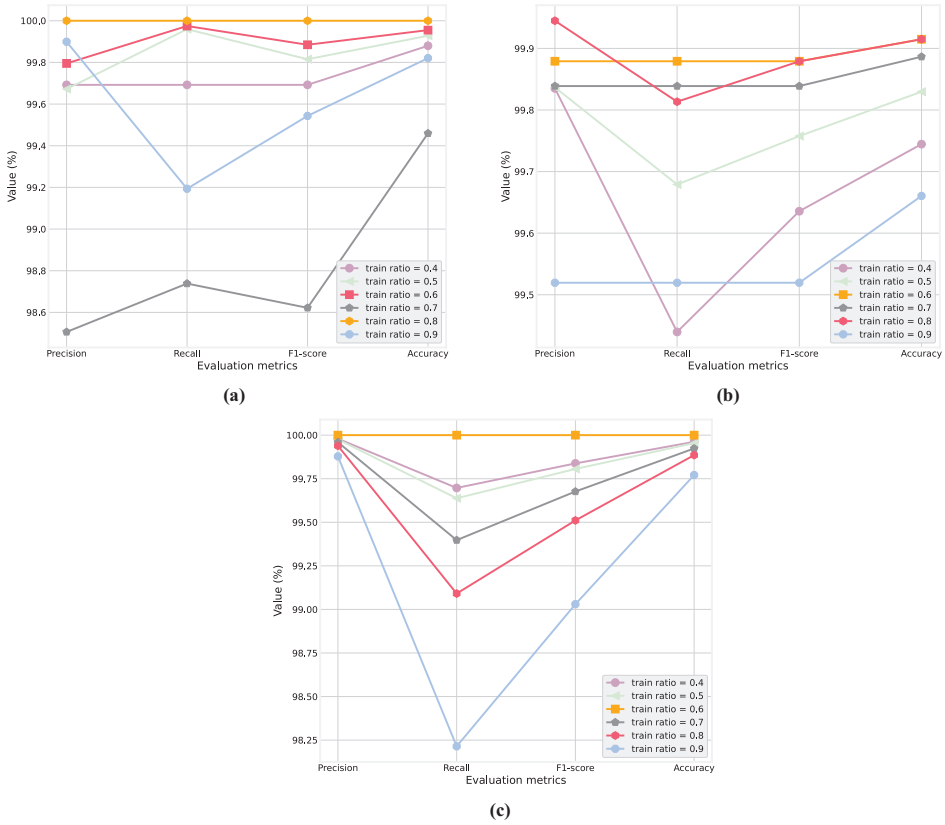


图 16 CoLog 在 (a) Casper、(b) Jhuisi 和 (c) Honey7数据集上执行的各种训练-测试配置的视觉示意图。

如表 11 至表 13 所示，在所有采用不同窗口大小进行评估的数据集中，基于上下文的窗口大小为 1 时，以最低成本取得了最优结果。在 Casper 数据集（即表 13）中，除基于背景的窗口大小为 6，以及基于上下文的窗口大小为 1、2 和 12 外，其他窗口大小的结果均可接受且相同。上述例外配置的评估指标均达到 100%。在 Jhuisi 数据集（即表 14）中，所有结果均具有独立显著性，其中基于上下文的窗口大小为 1 时结果最优。对于 Honey7 数据集（即表 15），基于背景的窗口大小为 2、3、6 和 9 时，所有指标均达到 100%；基于上下文的窗口大小为 1 和 3 时也达到 100%。其余配置的精确率、召回率、F1 分数和准确率分别为 99.97%、99.55%、99.76% 和 99.94%。

5.3.6 日志景观：CoLog 输出向量的可视化

在本实验中，我们收集了预训练 CoLog 学到的日志消息向量表示。图18 展示了以下数据集上的日志向量表示

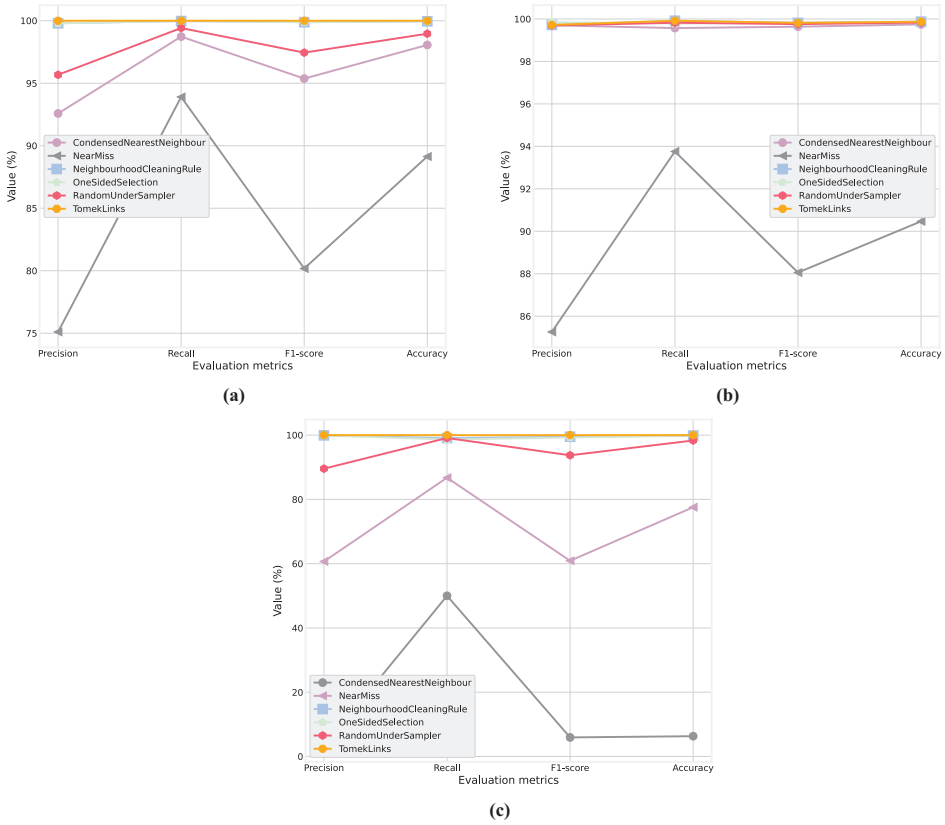


图 17 CoLog在(a) Casper、(b) Jhuisi和(c) Honey7数据集上使用Tomek链接与其他类别不平衡解决方法的比较。

Casper、Jhuisi 和 Honey7 数据集，展示了通过主成分分析降维技术得到的测试集划分的低维表示。我们证明，日志向量表示得到了准确分类。正常样本紧密聚集在一起。大多数异常分布得比正常样本更远，并聚集在一起，因而错误较少。因此，应用 argmax 预测函数可获得最佳性能。

5.3.7 韧性：泛化与鲁棒性研究

我们考察 CoLog 在识别此前未训练过的数据集中的未知异常方面的鲁棒性和泛化能力。这些数据集包含 Spark、Honey5 和 Windows 日志。尽管如此，如表 16 所示，CoLog 在识别已知异常的同时，也持续展现出识别未知异常的优异效果。CoLog 在 Honey5 和 Windows 数据集上分别取得了 98.83% 和 99.12% 的 F1 分数。CoLog 已在多个数据集上接受训练，因此能够处理未识别的异常。主要有三种

数字对象标识符： 10.1038/s41598-025-27693-4

表13 Casper 数据集中不同窗口大小和序列类型下 CoLog 性能的比较分析。粗体数字表示优异结果。

	序列类型									
	背景					上下文				
窗口大小	1	2	3	6	9	1	2	3	6	9
精确率	99.80	99.80	99.80	**100**	99.80	**100**	**100**	99.80	99.80	99.80
召回率	99.97	99.97	99.97	**100**	99.97	**100**	**100**	99.97	99.97	99.97
F1 分数	99.88	99.88	99.88	**100**	99.88	**100**	**100**	99.88	99.88	99.88
准确率	99.95	99.95	99.95	**100**	99.95	**100**	**100**	99.95	99.95	99.95

表1 4 在 Jhuisi数据集上，对 CoLog 在不同窗口大小和序列类型下的性能进行比较分析。粗体数字表示优异结果。

	序列类型									
	背景					上下文				
窗口大小	1	2	3	6	9	1	2	3	6	9
精确率	99.97	99.76	99.94	99.88	99.78	**100**	99.92	99.52	99.85	99.75
召回率	99.91	99.76	99.81	99.88	99.25	**100**	99.72	99.52	99.79	99.16
F1 分数	99.94	99.76	99.88	99.88	99.51	**100**	99.82	99.52	99.82	99.45
准确率	99.96	99.83	99.91	99.91	99.66	**100**	99.87	99.66	99.87	99.62

表 15。Honey7数据集中不同窗口大小和序列类型下 CoLog 性能的比较分析。加粗数字表示优异结果。

	序列类型									
	背景					上下文				
窗口大小	1	2	3	6	9	1	2	3	6	9
精确率	99.97	**100**	**100**	**100**	**100**	**100**	99.97	**100**	99.97	99.97
召回率	99.55	**100**	**100**	**100**	**100**	**100**	99.55	**100**	99.55	99.55
F1 分数	99.76	**100**	**100**	**100**	**100**	**100**	99.76	**100**	99.76	99.76
准确率	99.94	**100**	**100**	**100**	**100**	**100**	99.94	**100**	99.94	99.94

种解释。首先，我们使用 word2vec 词嵌入来表示每条日志消息中的词。word2vec 能够有效捕捉相似词语的含义，例如“warning”和“stopping”。其次，我们使用 sentence-transformers 为相似的日志消息生成相应的句子嵌入。第三，我们以协作方式采用有监督 MSA 模型，使其在已知和新数据集上均取得优异性能。

此外，我们在 Spark 数据集上进行了鲁棒性实验。我们使用原始 Spark 数据集训练并评估 CoLog，具体而言，训练集占总数据的 10%，验证集占 45%。随后，我们在数据集剩余的 45% 上测试训练好的模型。该合成数据集以 0%、5%、10%、15% 和 20% 的注入比例注入新的日志事件，

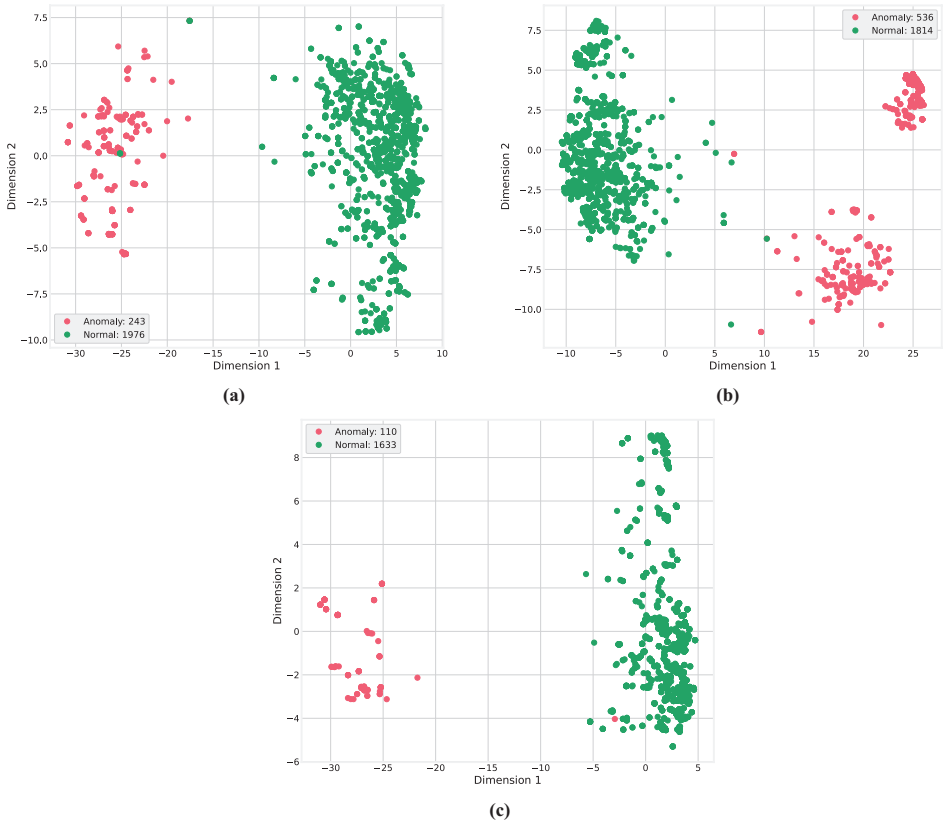


图 18 使用主成分分析对 CoLog 在 (a) Casper、(b) Jhuisi 和 (c) Honey7数据集上的输出向量进行的视觉展示。

这些替代数据集。实验结果如图 19所示。CoLog 展现出出色的召回率指标。尽管不稳定日志事件的注入比例不断上升，CoLog 仍能始终保持较高的召回率，即使注入比例较高也不例外。例如，注入比例为 20% 时，CoLog 的召回率仍可达到 98.51%。这验证了我们的方法论能够充分适应不稳定的日志事件。CoLog 将所有日志事件都表示为语义向量，因此能够检测语义相近的不稳定日志事件。因此，CoLog 可以高效分析新的日志事件，并能够持续处理包含未见过或不稳定日志事件的数据集。

5.3.8 揭示影响：消融研究

我们在 Casper、Jhuisi 和 Honey7 数据集上进行了消融研究，结果汇总于表 17。

表16 CoLog 在此前未见过的数据集上的评估 集。

数据集	平均值	类别	# 样本	精确率	召回率	F1 分数	准确率
Honey5		0：异常	56588	98.375	99.731	99.049	99.129
		1：正常	67798	99.773	98.625	99.196	
	宏		124386	99.074	99.178	99.122	
Windows		0：异常	6404	99.855	96.705	98.255	99.119
		1：正常	18567	98.876	99.952	99.411	
	宏		24971	99.365	98.328	98.833	

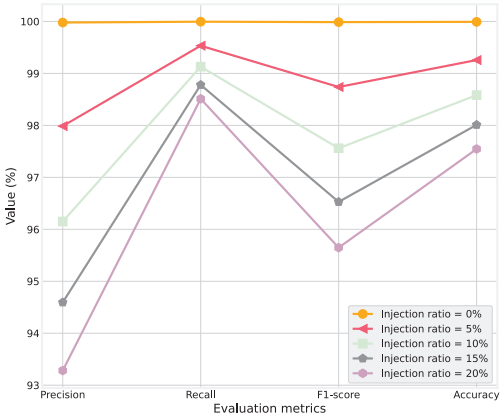


图19 在 Spark 数据集上，不同比例注入不稳定日志事件时 CoLog 结果的可视化示意图。

为进行消融研究，我们从多模态信息中移除序列模态，以评估事件背景或上下文对模型性能的影响。移除序列模态会导致性能下降，这表明序列信号对于基于日志的异常检测至关重要，也说明日志事件与事件背景或上下文之间具有互补性。此外，我们从 CoLog 中移除 MHIA 模块，并用多头注意力替换它。我们还移除了 MAL 和旨在实现各种模态融合的平衡层，结果所有性能指标均有所下降。这些结果证明，MHIA、MAL 和平衡层对于学习用于检测日志中异常的表示是有效的。最后，我们评估了不同模块之间的独特关联。结果表明，CoLog 的改进并非简单叠加而来，而是源于其各个模块之间的协同组合。

5.3.9 未知领域：突破性研究

一种用于检测日志中的点异常与集体异常的统一框架，可优化流程并确保分析的一致性。这种全面的方法

数字对象标识符： 10.1038/s41598-025-27693-4

表17 CoLog消融研究。Seq和Sem分别表示序列和语义。此外，CT、MHIA、MAL和BL分别表示协作Transformer、多头印象注意力、模态适配层和平衡层。

	模态		模块			指标			
数据集	Seq	Sem	MHIA	MAL	BL	精确率	召回率	F1 分数	准确率
Casper	×	✓	×	×	×	98.80	99.85	99.31	99.73
	✓	×	×	×	×	98.41	99.10	98.74	98.84
	✓	✓	✓	×	✓	99.39	99.92	99.65	99.86
	✓	✓	✓	×	×	99.33	99.51	99.42	99.77
	✓	✓	✓	✓	×	99.33	99.51	99.42	99.77
	✓	✓	×	✓	×	99.97	99.79	99.88	99.95
	✓	✓	×	✓	✓	99.36	99.72	99.54	99.82
	✓	✓	×	×	✓	99.56	99.74	99.65	99.86
	✓	✓	✓	✓	✓	100	100	100	100
Jhuisi	×	✓	×	×	×	99.88	99.88	99.88	99.92
	✓	×	×	×	×	96.88	92.39	94.51	99.12
	✓	✓	✓	×	✓	99.61	99.54	99.58	99.70
	✓	✓	✓	×	×	97.98	98.29	98.13	98.68
	✓	✓	✓	✓	×	99.22	99.09	99.15	99.40
	✓	✓	×	✓	×	99.42	99.49	99.46	99.62
	✓	✓	×	✓	✓	99.77	99.51	99.64	99.74
	✓	✓	×	×	✓	99.63	99.04	99.33	99.53
	✓	✓	✓	✓	✓	100	100	100	100
Honey7	×	✓	×	×	×	99.97	99.55	99.76	99.94
	✓	×	×	×	×	96.92	99.05	97.93	98.52
	✓	✓	✓	×	✓	99.97	99.55	99.76	99.94
	✓	✓	✓	×	×	99.94	99.09	99.51	99.89
	✓	✓	✓	✓	×	99.97	99.55	99.76	99.94
	✓	✓	×	✓	×	100	100	100	100
	✓	✓	×	✓	✓	99.97	99.55	99.76	99.94
	✓	✓	×	×	✓	99.51	99.51	99.51	99.89
	✓	✓	✓	✓	✓	100	100	100	100

减少了孤立检查每种异常类型可能产生的盲点，提高了检测系统的准确率和可靠性。采用统一框架能够更高效地部署资源并处理异常，从而增强系统的总体韧性。统一框架支持应用不同模态，以实现更完善且信息更丰富的点异常与集体异常检测。正如第4.8节所述，CoLog 为日志消息分配标签，并

日志事件的背景或上下文。随后，在收到输入 I 后，模型旨在学习并预测与每个输入相关联的标签。此方法包含四个类别。其中，**0** 表示两种模态均为负，**1** 表示只有日志事件是异常，2 表示只有日志事件的背景或上下文是异常，3 则表示输入为正常实例。Casper、Jhuisi、Nssal、Honey7、Zookeeper、Hadoop 和 BlueGene/L 数据集上的突破性研究实验结果如表 18 所示。可以看出，CoLog 在这项任务中表现出色。CoLog 在所有数据集的各项指标上都取得了出色且稳定的结果。例如，CoLog 在 Zookeeper 数据集上的召回率达到 99.94%。CoLog 检测日志文件中的点异常和集体异常的效能，主要得益于其通过不同的 Transformer 块从日志文件的各种模态中学习表示，以及捕捉这些模态之间交互作用的能力。这种方法使 CoLog 能够利用日志模态中的信息特征，包括日志的语义和序列特征。借助协作通道、多头印象注意力（MHIA）、MAL 和平衡层，CoLog 能更高效地检测点异常和集体异常。这种协作确保不会遗漏任何重要信息。值得注意的是，将 CoLog 扩展至更多模态十分简单。

表18 所有训练数据集上的突破性研究实验结果。

数据集	精确率	召回率	F1 分数	准确率
Casper	99.43	99.40	99.41	99.64
Jhuisi	99.82	99.48	99.65	99.70
Nssal	99.32	99.70	99.51	99.91
Honey7	99.77	98.90	99.33	99.77
Zookeeper	99.98	99.94	99.96	99.99
Hadoop	99.17	99.84	99.50	99.98
BlueGene/L	99.94	99.89	99.91	100

5.3.10 贡献编年史：贡献比较研究

对现有文献中各项贡献与我们研究进展的综合评估，凸显了显著的差异与协同效应。FastLogAD[77] 高效利用正常数据，并结合了一种创新的异常生成方法。相比之下，我们的方法 CoLog 通过跨多个模态对日志记录进行编码，进一步增强了这一点。pylogsentiment [38] 专注于利用情感分析（SA）进行日志异常检测，并解决类别不平衡问题。CoLog 则更进一步，将多模态情感分析（MSA）应用于日志异常检测，其性能优于最先进的方法。RAGLog[92]，是一种检索增强生成模型，主要强调将正常日志条目存储在向量数据库中。然而，我们的方法超越了单纯的存储，还能够学习异常特征。UMFLog [84] 采用了双模型

架构，将 BERT 用于语义特征提取，将变分自编码器（VAE）用于统计特征分析，同时处理长日志数据序列。CoLog 同样能够有效学习较长的日志记录序列，但通过引入协作方法来捕捉模态之间的交互作用，并利用 MAL 解决异质性问题，从而超越了 UMFLog。LogMS [59] 采用两步模型：首先使用基于多源信息融合的LSTM进行异常检测，随后使用 GRU网络进行概率标签估计。相比之下，CoLog 用于检测异常的统一框架提供了更加简洁且全面的解决方案。MDFULog [45] 解决日志数据中的噪声问题，并采用基于Informer的异常检测方法；而 CoLog 的模块，即协作 Transformer 和 MAL，则能够从根本上减轻噪声影响。最后，多特征融合（MFF）[44] 通过评估 HTTP 文本内容、状态码和频率特征，采用后期融合进行基于 MFF 的异常检测。然而，CoLog 的中期融合机制显著提升了性能指标。总体而言，CoLog 的贡献不仅体现了现有方法中的进展，还在一个连贯统一的框架中融合了它们的优势。因此，CoLog 的方法不仅具有多面性，能够应对日志异常检测所面临的广泛挑战，还展现出优于 FastLogAD、pylogsentiment、RAGLog、UMFLog、LogMS、MDFULog 和 MFF 各自贡献的性能指标。表 19 中的比较分析清晰地展示了 CoLog 如何处于创新前沿。CoLog 能够利用多种模态有效编码日志记录，通过 MAL 解决异质性问题，并在统一架构中展现出检测点异常和集体异常的卓越性能，这些都验证了 CoLog 作为日志异常检测领域一种创新方法的价值。

6 结论

本文介绍了 CoLog，这是一种利用多种日志模态进行协作编码和异常检测的新型框架。CoLog 采用监督学习方法，其在日志异常检测方面的成果凸显了其学术意义，为该领域确立了性能上限。在拥有充足带标签数据的情况下，CoLog 为评估无监督或半监督方法建立了“理论准确率”标准。因此，CoLog 可作为未来研究方向的参考基线，为开发更高效的模型提供比较依据和方向。此外，该方法通过实施 MAL 应对日志数据固有的异质性，确保在多个基准上的稳健性能。CoLog 能够在统一框架内检测点异常和集体异常，这使其有别于仅关注某一种异常的传统方法。这种全面的检测能力体现了 CoLog 的通用性和适应性，使其成为日志异常检测领域的一项重要进展。CoLog 的潜在应用十分广泛，从通过更准确地检测网络安全日志中的异常来加强安全措施，到通过实时识别系统异常来提高运营效率。此外，其稳健的性能

F1-score	Key Contributions
4.18	1) Efficient utilization of normal data. 2) Innovative anomaly generation method.
9.14	1) Implements sentiment analysis for log anomaly detection. 2) Addressing class imbalance.
9.00	1) A retrieval-augmented generation model that employs a vector database to store normal log entries.
9.56	1) Employs a dual-model architecture with BERT for semantic feature extraction and VAE for statistical feature analysis. 2) Handles long sequences of log data.
9.10	1) Employs a two-step model. The first step uses a multi-source information fusion-based LSTM to detect anomalies by utilizing semantic, sequential, and quantitative data. Following that, a probability label estimation-based GRU network is used.
7.00	1) Addresses noise in log data. 2) Informer-based anomaly detection.
3.10	1) Detects web scanning behavior by considering HTTP textual content, status code, and frequency features. 2) Employ late fusion MFF-based network to detect anomalies.
9.99	1) Encodes log records collaboratively according to various log modalities. 2) Employs MAL to address heterogeneity among modalities. 3) Outperforms state-of-the-art methods. 4) Detects point and collective abnormalities within a unified framework.

接收于 Scientific Reports
数字对象标识符：10.1038/s41598-025-27693-4

在不同噪声条件下，结果展现了其在多种环境中的可靠性和有效性。

6.1 局限性

实时应用。本研究在批处理模式下评估 CoLog。实时异常检测通常还涉及其他约束，例如延迟和计算资源限制。考察并提升 CoLog 在实时场景中的性能，对于有效实现至关重要。

人类难以理解的日志条目。某些日志条目，尤其是来自操作系统的日志条目，可能难以为人所解读，因此首次解释和标注此类数据颇具挑战。

适应不断演变的日志结构。随着系统更新，日志会随时间推移而变化，因此需要持续调整并重新训练模型，以维持其有效性。

噪声鲁棒性能力。尽管 CoLog 对不同程度的噪声表现出鲁棒性，但在某些极端场景下或面对某些类型的噪声时，其性能仍可能受到显著影响。

6.2 未来工作

未来的工作可以探索将 CoLog 集成到实时监控系统中，以便对工业系统中检测到的异常进行主动异常管理和自动响应。此外，将 CoLog 应用于金融欺诈检测或健康监测等其他领域，也有助于进一步验证其有效性和适应能力。CoLog 假设数据集存在中度不平衡且日志模板稳定；这些假设通常适用于基准数据集，但在动态生产系统中可能并不成立。在极端不平衡场景（例如异常比例低于 1%）或模板快速演变的环境中，表征漂移可能导致性能下降。为缓解这一问题，可以将 CoLog 的平衡层与自适应重采样、在线校准或少样本微调相结合。我们还注意到，扩展持续学习也可能进一步提升模型的适应能力。未来的工作将探索这些方向，以确保模型在多样化的运行条件下具备鲁棒性。此外，在异常检测过程中，也可以采用其他形式的日志数据模态，包括定量信息。将 CoLog 扩展到其他模态并不复杂。研究能够更好地管理各种噪声类型的自适应机制，也可能是一个有益的方向。CoLog 展现出的出色性能和广泛适用性表明，它在推动多个行业的发展方面潜力巨大，尤其是在这些行业中，准确、及时的异常检测至关重要。通过解决当前日志分析的局限性，CoLog 为应对复杂数据挑战的更先进、更全面的解决方案铺平了道路。

致谢。 我们谨向Mahdi Asgharzadeh致以诚挚谢意，感谢他为 Alarmif 网站的开发作出的重要贡献。

声明

经费。不适用。

利益冲突/竞争性利益。作者声明，他们不存在可能影响本研究内容、分析或结论的利益冲突或竞争性利益。

发表同意。所有作者均已同意发表本研究，并确认认可其内容及在指定期刊上的传播。

数据可用性。 本研究期间生成和/或分析的数据集可在 loghub 存储库中获取，网址为 https://github.com/logpai/loghub；也可在 pylogsentiment 存储库中获取，网址为 https://github.com/studiawan/pylogsentiment。
材料可用性。本人工智能开发所使用的数据集、计算资源、算法和库可根据请求提供，但须遵守相关访问政策和许可协议。

代码可用性。 本文的实现代码已在 GitHub 上公开，地址为 https://github.com/NasirzadehMoh/CoLog，确保后续开发与协作的透明度和可复现性。

作者贡献。方法论、理论分析、算法、实现、实验和写作——Mohammad Nasirzadeh；指导、架构设计和修订——Jafar Tahmoresnezhad；理论分析和架构设计——Parviz Rashidi-Khazaee。

参考文献

[1] Samariya, D., Thakkar, A.：异常检测算法综述。《Annals of Data Science》10(3), 829–850（2023年）https://doi.org/10.1007/s40745-021-00362-9

[2]Zhao, X., Rodrigues, K., Luo, Y., Yuan, D., Stumm, M.：基于流重构原理的整个软件栈非侵入式性能分析。载于：第12届USENIX操作系统设计与实现研讨会（OSDI 16），第603–618页。USENIX协会，美国佐治亚州萨凡纳（2016年）。https://www.usenix.org/conference/osdi16/technical-sessions/presentation/zhao

[3] Yu, X., Joshi, P., Xu, J., Jin, G., Zhang, H., Jiang, G.：Cloudseer：通过交错日志监控云基础设施工作流。SIGARCH 计算机体系结构新闻 44(2), 489–502(2016年) https://doi.org/10.1145/2980024.2872407

[4] Lin, Q., Zhang, H., Lou, J.-G., Zhang, Y., Chen, X.：基于日志聚类的在线服务系统问题识别。在：第38届国际软件工程会议论文集。ICSE '16，第102–111页。

计算机协会，纽约州纽约市，美国（2016年）。https://doi.org/10.1145/2889160.2889232 。https://doi.org/10.1145/2889160.2889232[5] Roy, S., K¨onig, A.C., Dvorkin, I., Kumar, M.：Perfaugur：云服务性能异常的稳健诊断。在：2015年IEEE第31届数据工程国际会议，第1167–1178页（2015年）。https://doi.org/10.1109/ ICDE.2015.7113365[6] Oprea, A., Li, Z., Yen, T.-F., Chin, S.H., Alrwais, S.：通过挖掘大规模日志数据检测企业感染早期阶段。在：2015年第45届年度IEEE/I-FIP可靠系统与网络国际会议，第45–56页（2015年）。https://doi.org/10.1109/DSN.2015.14[7] Beschastnikh, I., Brun, Y., Ernst, M.D., Krishnamurthy, A.：利用csight从并发系统行为日志中推断模型。在：第36届国际软件工程会议论文集。ICSE 2014，第468–479页。计算机协会，纽约州纽约市，美国（2014年）。https://doi. org/10.1145/2568225.2568246 。https://doi.org/10.1145/2568225.2568246[8] Yen, T.-F., Oprea, A., Onarlioglu, K., Leetham, T., Robertson, W., Juels, A., Kirda, E.：Beehive：用于检测企业网络中可疑活动的大规模日志分析。在：第29届年度计算机安全应用会议论文集。ACSAC '13，第199–208页。计算机协会，纽约州纽约市，美国（2013年）。https://doi.org/10.1145/2523649. 2523670 。https://doi.org/10.1145/2523649.2523670[9] Cinque, M., Cotroneo, D., Pecchia, A.：用于分析软件故障的事件日志：一种基于规则的方法。IEEE软件工程汇刊39(6)，806–821（2013年）https://doi.org/10.1109/TSE.2012.67[10] Lou, J.-G., Fu, Q., Yang, S., Xu, Y., Li, J.：从控制台日志中挖掘不变量以进行系统问题检测。在：2010年USENIX年度技术会议（USENIX ATC 10）（2010年）[11] Lou, J.-G., Fu, Q., Yang, S., Li, J., Wu, B.：从交错轨迹中挖掘程序工作流。在：第16届ACM SIGKDD知识发现与数据挖掘国际会议论文集。KDD '10，第613–622页。计算机协会，纽约州纽约市，美国（2010年）。https://doi.org/10.1145/1835804.1835883 。https://doi.org/10.1145/1835804.1835883[12] Xu, W., Huang, L., Fox, A., Patterson, D., Jordan, M.I.：通过挖掘控制台日志检测大规模系统问题。在：ACM SIGOPS第22届操作系统原则研讨会论文集。SOSP '09，第117–132页。计算机协会，纽约州纽约市，美国（2009年）。https://doi.org/10.1145/1629575.1629587 。https://doi.org/10.1145/1629575.1629587[13] Xu, W., Huang, L., Fox, A., Patterson, D., Jordan, M.：在线系统问题

通过挖掘控制台日志中的模式进行检测。在：2009年第九届IEEE数据挖掘国际会议，第588–597页（2009）。 https://doi.org/10.1109/ICDM.2009.19[14] Fu, Q., Lou, J.-G., Wang, Y., Li, J.：通过非结构化日志分析进行分布式系统中的执行异常检测。在：2009年第九届IEEE数据挖掘国际会议，第149–158页（2009）。 https://doi.org/10.1109/ICDM.2009.60[15] Yamanishi, K., Maruyama, Y.：用于网络故障监控的动态系统日志挖掘。在：第十一届ACM SIGKDD国际知识发现与数据挖掘会议论文集。KDD '05，第499–508页。计算机协会，纽约，纽约州，美国（2005）。 https://doi.org/10.1145/1081870.1081927。 https://doi.org/10.1145/1081870.1081927[16] Rouillard, J.P.：使用简单事件关联器（sec）进行实时日志文件分析。在：LISA，第4卷，第133–150页（2004）[17] Prewett, J.E.：使用Logsurfer分析集群日志文件。在：第四届Linux集群年度会议论文集，第1–12页（2003）。CiteSeer，美国宾夕法尼亚州州立学院[18] Hansen, S.E., Atkins, E.T.：使用Swatch实现自动化系统监控与通知。在：LISA，第93卷，第145–152页（1993）。美国加利福尼亚州蒙特雷[19] Memon, A.U., Cordy, J.R., Dean, T.：使用语法推断进行日志文件分类与异常分析[20] Cui, T., Ma, S., Chen, Z., Xiao, T., Tao, S., Liu, Y., Zhang, S., Lin, D., Liu, C., Cai, Y., Meng, W., Sun, Y., Pei, D.：LogEval：用于大语言模型日志分析的综合基准套件（2024）。 https://arxiv.org/abs/2407.01896[21] Du, Q., Zhao, L., Xu, J., Han, Y., Zhang, S.：采用多头缩放点积注意力机制的基于日志的异常检测。在：Strauss, C., Kotsis, G., Tjoa, A.M., Khalil, I.（编）：数据库与专家系统应用，第335–347页。施普林格，Cham（2021）。 https://doi.org/10.1007/978-3-030-86472-9 31[22] Le, V.-H., Zhang, H.：采用深度学习的基于日志的异常检测：我们究竟走了多远？在：第四十四届国际软件工程会议论文集。ICSE '22，第1356–1367页。计算机协会，纽约，纽约州，美国（2022）。 https://doi.org/10.1145/3510003.3510155。 https://doi.org/10.1145/3510003.3510155[23]Pithode, K., Patheja, P.S.：关于使用深度学习技术进行日志异常检测的研究。在：2022年应用人工智能

与计算国际会议（ICAAIC），第290–298页（2022）。
https://doi.org/10.1109/ICAAIC53929.2022.9793238

[24] Chen, Z., Liu, J., Gu, W., Su, Y., Lyu, M.R.：经验报告：用于异常检测的基于深度学习的系统日志分析（2022）。https://doi.org/10.48550/arXiv.2107.05908。
https://arxiv.org/abs/2107.05908

[25] Cheng, X., Luo, D., Chen, X., Liu, L., Zhao, D., Yan, R.：提升自我：利用自记忆的检索增强文本生成。载于：Oh, A., Naumann, T., Globerson, A., Saenko,K., Hardt, M., Levine, S.（编）：神经信息处理系统进展，第36卷，第43780–43799页。Curran Associates, Inc.，???（2023年）

[26] Huang, S.-C., Pareek,A., Jensen, M., Lungren, M.P., Yeung, S., Chaudhari, A.S.：医学图像分类的自监督学习：系统综述与实施指南。NPJ数字医学 **6**(1), 74 (2023)https://doi.org/10.1038/s41746-023-00811-0

[27] Jeong, J., Tian, K., Li, A., Hartung, S., Adithan, S., Behzadi, F., Calle, J., Osayande, D., Pohlen, M., Rajpurkar, P.：多模态图像-文本匹配可改进基于检索的胸部X线报告生成。载于：Oguz, I., Noble, J., Li, X., Styner, M., Bau mgartner, C., Rusu, M., Heinmann, T., Kontos, D., Land- man, B., Dawant, B. (编)《深度学习医学影像》。机器学习研究论文集，第227卷，第978–990页。PMLR，??? (2024)。 https://proceedings.mlr.press/v227/jeong24a.html[28] Le, M., Vyas, A., Shi, B., Karrer, B., Sari, L., Moritz, R., Williamson, M., Manohar, V., Adi, Y., Mahadeokar, J., Hsu, W.-N.：Voicebox：大规模文本引导的多语言通用语音生成。载于：Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (编)《神经信息处理系统进展》，第36卷，第14005–14034页。Curran Associates, Inc., ??? (2023)[29] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.：Swin Transformer：采用移位窗口的层次化视觉变换器。载于：IEEE/CVF国际计算机视觉会议 (ICCV) 论文集，第10012–10022页 (2021)。 https://doi.org/10.48550/arXiv.2103.14030[30] Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.：视频Swin Transformer。载于：IEEE/CVF计算机视觉与模式识别会议 (CVPR) 论文集，第3202–3211页 (2022)。
https://doi.org/10.48550/arXiv.2106.13230

[31]Saleki, S., Tahmoresnezhad, J.: Agry：一种结合预训练 EfficientNet 与卷积神经网络进行植物病害分类的精准农业综合框架。Multimedia Tools and Applications 83(24), 64813–64851（2024）

[32] Tahmoresnezhad, J., Hashemi, S.: 通过迁移特征学习实现视觉域适应。知识与信息系统 50(2), 585–605（2017）[33] Khorshidpour, Z., Tahmoresnezhad, J., Hashemi, S., Hamzeh, A.: 针对规避攻击的领域不变特征提取。国际机器学习与控制论期刊 9(12), 2093–2104（2018）[34] Khodadadi, M., Tahmoresnezhad, J.: HyMo：使用一种新颖的多模态混合模型进行智能合约漏洞检测（2023）。https://arxiv.org/abs/2304.13103[35] Almodovar, C., Sabrina, F., Karimi, S., Azad, S.: Logfit：使用微调语言模型进行日志异常检测。IEEE 网络与服务管理汇刊 21(2), 1715–1723 （ 2024 ） https://doi.org/10.1109/TNSM. 2024.3358730[36] Du, M., Li, F., Zheng, G., Srikumar, V.: DeepLog：通过深度学习从系统日志中进行异常检测与诊断。载于：2017 年 ACM SIGSAC 计算机与通信安全大会论文集。CCS '17，第 1285–1298 页。计算机协会，纽约，NY，美国（2017）。https://doi.org/10.1145/3133956.3134015 . https://doi.org/10.1145/3133956.3134015[37] Guo, H., Yang, J., Liu, J., Bai, J., Wang, B., Li, Z., Zheng, T., Zhang, B., Peng, J., Tian, Q.: LogFormer：用于日志异常检测的预训练与调优流程。AAAI 人工智能大会论文集 38(1), 135–143（2024）https://doi.org/10.1609/aaai.v38i1.27764[38] Studiawan, H., Sohel, F., Payne, C.: 基于深度学习的情感分析在操作系统日志异常检测中的应用。IEEE 可靠与安全计算汇刊 18(5), 2136–2148（2021）https://doi.org/10.1109/TDSC. 2020.3037903[39] Wang, S., Jiang, R., Wang, Z., Zhou, Y.: 基于深度学习的计算机网络异常检测与日志分析（2024）。https://doi.org/10.48550/arXiv.2407.05639 . https://arxiv.org/abs/2407.05639[40]Landauer, M., Onder, S., Skopik, F., Wurzenberger, M.: 日志数据异常检测深度学习综述。机器学习与应用 12, 100470（2023）https://doi.org/10.1016/j.mlwa.2023.100470[41] Li, H., Li, Y.: LogSpy：面向分布式系统的系统日志异常检测。载于：2020 年人工智能与计算机工程国际会议（ICAICE），第 347–352 页（2020）。https://doi.org/10.1109/ICAICE51518. 2020.00073[42]Wang, Q., Zhang, X., Wang, X., Cao, Z.: 日志序列异常检测

基于对比对抗训练和双特征提取的方法。Entropy **24**(1)(2022) https://doi.org/10.3390/e24010069[43] Qi, J., Luan, Z., Huang, S., Fung, C., Yang, H., Li, H., Zhu, D., Qian, D.: Logencoder：基于日志的异常对比表示学习。IEEE 网络与服务管理汇刊**20**(2), 1378–1391(2023) https://doi.org/10.1109/TNSM.2023.3239522[44] Zhao, Q., Liu, B., Wu, H., Zhang, Z., Hu, R., Hua, H., Zhang, C.: 一种用于在线 Web 日志网页扫描行为检测的多特征融合方法。载于：2023年第四届计算机与人工智能技术国际会议（CAIT），第246–252页（2023）。https://doi.org/10.1109/CAIT59945.2023.10469208[45] Li, M., Sun, M., Li, G., Han, D., Zhou, M.: Mdfulog：不稳定日志异常检测模型的多特征深度融合。Applied Sciences **13**(4)(2023) https://doi.org/10.3390/app13042237[46]Lu, S., Wei, X., Li, Y., Wang, L.: 使用卷积神经网络检测大数据系统日志中的异常。载于：2018年 IEEE 第16届可信、自主与安全计算国际会议、第16届普适智能与计算国际会议、第4届大数据智能与计算及网络空间科学与技术大会（DASC/PiCom/DataCom/CyberSciTech），第151–158页（2018）。 https://doi.org/10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00037[47] Wang, Z., Tian, J., Fang, H., Chen, L., Qin, J.: Lightlog：一种用于边缘端日志异常检测的轻量级时序卷积网络。Computer Networks **203**, 108616(2022) https://doi.org/10.1016/j.comnet.2021.108616[48] Zhang, L., Li, W., Zhang, Z., Lu, Q., Hou, C., Hu, P., Gui, T., Lu, S.: Logattn：基于自编码器注意力机制的无监督日志异常检测。载于：Qiu, H., Zhang, C., Fei, Z., Qiu, M., Kung, S.-Y.（编），《知识科学、工程与管理》，第222–235页。施普林格，Cham（2021）。 https://doi.org/10.1007/978-3-030-82153-1 19[49] Hashemi, S., M¨antyl¨a, M.: Onelog：迈向端到端软件日志异常检测。Automated Software Engineering **31**(2) (2024)https://doi.org/10.1007/s10515-024-00428-x[50] Xiao, C., Huang, J., Wu, W.: 使用混合深度学习模型检测集群系统中的异常。载于：Shen, H., Sang, Y.（编），《并行架构、算法与编程》，第393–404页。施普林格，新加坡（2020）。 https://doi.org/10.1007/978-981-15-2767-8 35

[51] Wang, P., Zhang, X., Cao, Z.: Log 异常 y 基于语义特征的检测

和主题特征。载于：Tari, Z., Li, K., Wu, H.（编）《并行处理算法与架构》，第407–427页。施普林格，新加坡（2024）。
https://doi.org/10.1007/978-981-97-0808-6 24[52] Farzad, A.：采用过采样的日志消息异常检测。《国际人工智能与应用杂志》（IJAIA）**11**（4）（2020）[53] Farzad, A., Gulliver, T.A.：使用 Auto-B/LSTM 和 Auto-GRU 进行日志消息异常检测与分类（2021）。https://doi.org/10.48550/arXiv.1911.08744。
https://arxiv.org/abs/1911.08744[54]Gu, S., Chu, Y., Zhang, W., Liu, P., Yin, Q., Li, Q.：结合双向切片GRU与GA注意力机制的系统日志异常检测研究。载于：2021年第4届人工智能与大数据国际会议（ICAIBD），第577–583页（2021）。
https://doi.org/10.1109/ICAIBD51990.2021.9459087[55] Li, X., Chen, P., Jing, L., He, Z., Yu, G.：Swisslog：一种稳健、统一的深度学习日志异常检测方法，可检测多种故障。载于：2020年IEEE第31届国际软件可靠性工程研讨会（ISSRE），第92–103页（2020）。https://doi.org/10.1109/ISSRE5003.2020.00018[56] Wang, Z., Chen, Z., Ni, J., Liu, H., Chen, H., Tang, J.：用于离散事件序列异常检测的多尺度单类循环神经网络。载于：第27届ACM SIGKDD知识发现与数据挖掘会议论文集。KDD '21，第3726–3734页。计算机协会，纽约，纽约州，美国（2021）。
https://doi.org/10.1145/3447548.3467125。
https://doi.org/10.1145/3447548.3467125[57]Yang, L., Chen, J., Wang, Z., Wang, W., Jiang, J., Dong, X., Zhang, W.：通过概率标签估计进行半监督的基于日志的异常检测。载于：2021年IEEE/ACM第43届国际软件工程会议（ICSE），第1448–1460页（2021）。https://doi.org/10.1109/ICSE43902.2021.00130[58] Zhang, D., Dai, D., Han, R., Zheng, M.：SentiLog：通过基于日志的情感分析检测并行文件系统中的异常。载于：第13届ACM存储与文件系统热点主题研讨会论文集。HotStorage '21，第86–93页。计算机协会，纽约，纽约州，美国（2021）。
https://doi.org/10.1145/3465332.3470873。
https://doi.org/10.1145/3465332.3470873[59] Yu, Z., Yang, S., Li, Z., Li, L., Luo, H., Yang, F.：LogMS：一种基于多源信息融合与概率标签估计的多阶段日志异常检测方法。《物理学前沿》**12**（2024）https://doi.org/10.3389/fphy.2024.1401857
[60] Zhang, C., Wang,X., Zhang, H., Zhang, J., Zhang, H., Liu, C., Han, P.：LayerLog：一种基于层次语义的日志序列异常检测方法。《应用软》

Computing 132, 109860（2023）
https://doi.org/10.1016/j.asoc.2022.109860[61] Fu, Y., Liang, K., Xu, J.：Mlog：一种使用语义表示的基于 Mogrifier LSTM 的日志异常检测方法。《IEEE服务计算汇刊》16(5)，3537–3549（2023）https://doi.org/10.1109/TSC.2023.3289488[62] Yu, D., Hou, X., Li, C., Lv, Q., Wang, Y., Li, N.：使用基于注意力机制的双向长短期记忆网络检测非结构化日志中的异常。载于：2021年第7届IEEE网络智能与数字内容国际会议（IC-NIDC），第403–407页（2021）。
https://doi.org/10.1109/IC-NIDC54101.2021.9660476[63] Zhang, X., Xu, Y., Lin, Q., Qiao, B., Zhang, H., Dang, Y., Xie, C., Yang, X., Cheng, Q., Li, Z., Chen, J., He, X., Yao, R., Lou, J.-G., Chintalapati, M., Shen, F., Zhang, D.：不稳定日志数据上的鲁棒基于日志的异常检测。载于：2019年第27届ACM欧洲软件工程会议与软件工程基础研讨会联合会议论文集。ESEC/FSE 2019，第807–817页。计算机协会，美国纽约州纽约市（2019）。https://doi.org/10.1145/3338906.3338931。
https://doi.org/10.1145/3338906.3338931[64] Yang, R., Qu, D., Gao, Y., Qian, Y., Tang, Y.：nlsalog：用于安全管理中日志序列异常检测的框架。《IEEE Access》7，181152–181164（2019）https://doi.org/10.1109/ACCESS.2019.2953981[65] Niu, W., Liao, X., Huang, S., Li, Y., Zhang, X., Li, B.：用于基于日志的异常检测的鲁棒宽度与深度学习框架。《应用软计算》153，111314（2024）
https://doi.org/10.1016/j.asoc.2024.111314[66] Yang, H., Lin, F., Chai, Y., Qie, K., Lin, W., Wang, Y., Zhang, C., Guo, M.：一种基于自注意力机制和双向门控循环单元模型的日志异常检测算法。载于：Jia, Y., Zhang, W., Fu, Y., Wang, J.（编）：2023年中国智能系统大会论文集，第877–888页。施普林格，新加坡（2023）。
https://doi.org/10.1007/978-981-99-6847-3 76[67] Niu, W., Li, Z., He, Z., Wang, A., Li, B., Zhang, X.：FSMFLog：结合完整语义信息与多特征融合发现异常日志。《IEEE物联网期刊》11(3)，4442–4453（2024）
https://doi.org/10.1109/JIOT.2023.3300690[68]Otomo, K., Kobayashi, S., Fukuda, K., Esaki, H.：基于潜变量的网络系统日志异常检测。《IEICE信息与系统汇刊》102(9)，1644–1652（2019）
https://doi.org/10.1587/transinf.2018OFP0007[69] Qian, Y., Ying, S., Wang, B.：通过变分自编码器检测分布式系统中的异常。载于：2020年IEEE系统、人类与控制论国际会议（SMC），第2822–2829页（2020）。https://doi.org/10.1109/

SMC42975.2020.9283078

[70] Wadekar, A., Gupta, T., Vijan, R., Kazi, F.：用于日志文件系统无监督异常检测的混合 CAE-VAE。载于：2019年第10届国际计算、通信与网络技术会议（ICCCNT），第1–7页（2019）。https://doi.org/10.1109/ICCCNT45670.2019.8944863[71] Wang, X., Cao, Q., Wang, Q., Cao, Z., Zhang, X., Wang, P.：基于对比学习和多尺度质量的鲁棒日志异常检测。《超级计算杂志》78(16)，17491–17512（2022）https://doi.org/10.1007/s11227-022-04508-1[72]Catillo, M., Pecchia, A., Villano, U.：AutoLog：通过系统日志深度自编码进行异常检测。《应用中的专家系统》191，116263（2022）https://doi.org/10.1016/j.eswa.2021.116263[73] Han, X., Yuan, S.：通过领域自适应进行无监督跨系统日志异常检测。载于：第30届ACM信息与知识管理国际会议论文集。CIKM '21，第3068–3072页。计算机协会，美国纽约州纽约市（2021）。https://doi.org/10.1145/3459637.3482209。https://doi.org/10.1145/3459637.3482209[74] Xia, B., Bai, Y., Yin, J., Li, Y., Xu, J.：LogGAN：一种使用排列事件建模进行异常检测的日志级生成对抗网络。《信息系统前沿》23，285–298（2021）https://doi.org/10.1007/s10796-020-10026-3[75] Zhao, Z., Niu, W., Zhang, X., Zhang, R., Yu, Z., Huang, C.：Trine：在一个生成对抗网络中使用三个Transformer编码器进行Syslog异常检测。《Applied Intelligence》52(8)，8810–8819（2022）https://doi.org/10.1007/s10489-021-02863-9[76] Yamanaka, Y., Takahashi, T., Minami, T., Nakajima, Y.：LogELECTRA：面向非结构化日志的自监督异常检测（2024）。https://doi.org/10.48550/arXiv.2402.10397。https://arxiv.org/abs/2402.10397[77]Lin, Y., Deng, H., Li, X.：FastLogAD：通过掩码引导的伪异常生成与判别进行日志异常检测（2024）。https://doi.org/10.48550/arXiv.2404.08750。https://arxiv.org/abs/2404.08750 [78] Qi, J., Luan, Z., Huang, S., Wang, Y., Fung, C., Yang, H., Qian, D.：Adanomaly：采用对抗学习的系统日志自适应异常检测。载于：NOMS 2022-2022 IEEE/IFIP网络运营与管理研讨会，第1–5页（2022）。https://doi.org/10.1109/NOMS54207.2022.9789917[79] Guo, H., Yuan, S., Wu, X.：LogBERT：通过BERT进行日志异常检测。载于：2021年国际神经网络联合会议（IJCNN），第1–8页（2021）。

https://doi.org/10.1109/IJCNN52387.2021.9534113

[80] Nedelkoski, S., Bogatinovski, J., Acker, A., Cardoso, J., Kao, O.：基于自注意力分类的非结构化日志异常检测。载于：2020年IEEE国际数据挖掘会议（ICDM），第1196–1201页（2020）。https://doi.org/10.1109/ICDM50108.2020.00148[81] Zang, R., Guo, H., Yang, J., Liu, J., Li, Z., Zheng, T., Shi, X., Zheng, L., Zhang, B.: MLAD：用于多系统日志异常检测的统一模型（2024）。https://doi.org/10.48550/arXiv.2401.07655。 https://arxiv.org/abs/2401.07655[82] Wittkopp, T., Acker, A., Nedelkoski, S., Bogatinovski, J., Scheinert, D., Fan, W., Kao, O.：A2Log：基于注意力增强的日志异常检测（2021）。https://doi.org/10.48550/arXiv.2109.09537。 https://arxiv.org/abs/2109.09537[83] Tan, X., Han, N., Lu, S., Chen, W., Wang, D.：Semlog：一种基于语义的方法，用于检测大数据系统日志中的异常。载于：2023年IEEE第29届并行与分布式系统国际会议（ICPADS），第1199–1206页（2023）。https://doi.org/10.1109/ICPADS60453.2023.00174[84] He, S., Deng, T., Chen, B., Sherratt, R.S., Wang, J.：一种基于多特征的无监督日志异常检测方法。Computers, Materials & Continua**76**（1）（2023）https://doi.org/10.32604/cmc.2023.037392[85] Han, C., Guan, B., Li, T., Kang, D., Qin, J., Wu, Y.：基于匹配网络的少样本日志异常检测。IEEE 网络与服务管理汇刊**21**（3），2909–2925（2024）https://doi.org/10.1109/TNSM.2024.3363626[86] Zhang, L., Jia, T., Jia, M., Li, Y., Yang, Y., Wu, Z.：面向分布式数据库的多变量日志异常检测。载于：第30届ACM SIGKDD知识发现与数据挖掘会议论文集。KDD ’24，第4256–4267页。计算机协会，纽约，美国（2024）。https://doi.org/10.1145/3637528.3671725。https://doi.org/10.1145/3637528.3671725[87] Ede, T.v., Aghakhani, H., Spahn, N., Bortolameotti, R., Cova, M., Continella, A., Steen, M.v., Peter, A., Kruegel, C., Vigna, G.：Deepcase：安全事件的半监督上下文分析。载于：2022年IEEE安全与隐私研讨会（SP），第522–539页（2022）。https://doi.org/10.1109/SP46214.2022.9833671[88]Li, X., Niu, W., Zhang, X., Zhang, R., Yu, Z., Li, Z.：基于双向长短期记忆注意力机制的语义和时间特征提升日志异常检测性能。载于：2021年第二届电子、通信与信息技术国际会议（CECIT），第661–666页（2021）。https://doi.org/10.1109/CECIT53797.2021.00121

[89]Guo, Y., Wen, Y., Jiang, C., Lian, Y., Wan, Y.：利用多头注意力检测日志异常（LAMA）（2021）。https://doi.org/10.48550/arXiv. 2101.02392 。https://arxiv.org/abs/2101.02392[90] Wan, Y., Liu, Y., Wang, D., Wen, Y.：Glad-paw：通过位置感知加权图注意力网络进行基于图的日志异常检测。载于：Karlapalem, K., Cheng, H., Ramakrishnan, N., Agrawal, R.K., Reddy, P.K., Srivastava, J., Chakraborty, T.（编），知识发现与数据挖掘进展，第66–77页。施普林格，Cham（2021）。https://doi.org/10.1007/978-3-030-75762-5 6[91] Decker, L., Leite, D., Viola, F., Bonacorsi, D.：应用于计算中心预测性维护异常检测的演化粒分类器比较。载于：2020年IEEE演化与自适应智能系统会议（EAIS），第1–8页（2020）。https://doi.org/10.1109/EAIS48028.2020.9122779[92] Pan, J., Liang, W.S., Yidi, Y.：Raglog：使用检索增强生成进行日志异常检测。载于：2024年IEEE世界公共安全技术论坛（WFPST），第169–174页（2024）。https://doi.org/10.1109/WFPST58552. 2024.00034[93]Liu, Y., Tao, S., Meng, W., Wang, J., Ma, W., Chen, Y., Zhao, Y., Yang, H., Jiang, Y.：利用大语言模型和提示策略进行可解释的在线日志分析。载于：第32届IEEE/ACM程序理解国际会议论文集。ICPC '24，第35–46页。计算机协会，纽约，美国（2024）。https://doi.org/10.1145/ 3643916.3644408 。https://doi.org/10.1145/3643916.3644408[94]Liu, Z., Zhou, B., Chu, D., Sun, Y., Meng, L.：不确定缺失模态下基于模态转换的多模态情感分析。Information Fusion 101，101973（2024）https://doi.org/10.1016/j.inffus.2023.101973[95] Deng, Y., Li, Y., Xian, S., Li, L., Qiu, H.：Mual：通过跨模态注意力和差异损失增强多模态情感分析。《国际多媒体信息检索期刊》13（3），31（2024）https://doi.org/10.1007/ s13735-024-00340-w[96] Ahuja, G., Alaei, A., Pal, U.：一种针对含有文本信息图像的新型多模态情感分析方法。Multimedia Tools and Applications，第1–30页（2024）https://doi.org/10.1007/s11042-024-19999-8 [97] Wang, D., Guo, X., Tian, Y., Liu, J., He, L., Luo, X.：TETFN：用于多模态情感分析的文本增强型Transformer融合网络。Pattern Recognition 136，109259（2023）https://doi.org/10.1016/j.patcog.2022.109259[98] Li, Z., Guo, Q., Feng, C., Deng, L., Zhang, Q., Zhang, J., Wang, F., Sun, Q.：基于交互式Transformer和软映射的多模态情感分析。《无线通信与移动计算》2022（1），6243347（2022）

https://doi.org/10.1155/2022/6243347

[99] Hu, G., Lin, T.-E., Zhao, Y., Lu, G., Wu, Y., Li, Y.: UniMSE：迈向统一的多模态情感分析与情绪识别（2022）。https://doi. org/10.48550/arXiv.2211.11256 . https://arxiv.org/abs/2211.11256[100] Wang, Z., Wan, Z., Wan, X.: Transmodality：一种利用Transformer进行多模态情感分析的端到端融合方法。载于：2020年Web会议论文集。WWW '20，第2514–2520页。计算机协会，纽约，NY，美国（2020）。https://doi.org/10.1145/3366423.3380000. https://doi.org/10.1145/3366423.3380000[101] Delbrouck, J.-B., Tits, N., Brousmiche, M., Dupont, S.: 一种用于情绪识别和情感分析的基于Transformer的联合编码方法。载于：第二届多模态语言大挑战及研讨会（Challenge-HML）。计算语言学协会，???（2020）。https://doi.org/10.18653/v1/ 2020.challengehml-1.1 . http://dx.doi.org/10.18653/v1/2020.challengehml-1.1[102] Zadeh, A., Chen, M., Poria, S., Cambria, E., Morency, L.-P.: 用于多模态情感分析的张量融合网络（2017）。https://doi.org/10.48550/ arXiv.1707.07250 . https://arxiv.org/abs/1707.07250[103] Chen, M., Wang, S., Liang, P.P., Baltruˇ saitis, T., Zadeh, A., Morency, L.-P.: 结合词级融合与强化学习的多模态情感分析。载于：第19届ACM国际多模态交互会议论文集。ICMI '17，第163–171页。计算机协会，纽约，NY，美国（2017）。https://doi.org/10.1145/3136755.3136801 . https://doi. org/10.1145/3136755.3136801[104] Yu, Z., Yu, J., Cui, Y., Tao, D., Tian, Q.: 用于视觉问答的深度模块化协同注意力网络。载于：IEEE/CVF计算机视觉与模式识别会议论文集（CVPR）（2019）[105] Li, Z., Xiang, Z., Gong, W., Wang, H.: 使用堆叠式时间卷积网络进行集体异常与点异常检测的统一模型。《应用智能》52(3)，3118–3131（2022）https://doi.org/10.1007/s10489-021-02559-0 [106] Liu, H., Zhang, H.-s., Tang, Y., Yao, Y.: 一种用于检测列车车轴温度传感器点异常和子序列异常数据的统一方法。《IEEE传感器杂志》23(20)，24772–24786（2023）https://doi.org/10.1109/JSEN.2023. 3307623[107] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: 注意力就是一切。《神经信息处理系统进展》30（2017）

[108]Chalapathy, R., Chawla, S.: 异常检测深度学习：综述（2019）。https://doi.org/10.48550/arXiv.1901.03407. https://arxiv.org/abs/ 1901.03407 [109] Liang, Y., Zhang, Y., Xiong, H., Sahoo, R.: IBM BlueGene/L事件日志中的故障预测。载于：第七届 IEEE 国际数据挖掘会议（ICDM 2007），第583–588页（2007）。https://doi.org/10.1109/ICDM.2007.46[110] Chen, M., Zheng, A.X., Lloyd, J., Jordan, M.I., Brewer, E.: 使用决策树进行故障诊断。载于：国际自主计算会议，2004年。会议论文集，第36–43页（2004）。https://doi.org/10.1109/ICAC.2004. 1301345[111] He, S., Zhu, J., He, P., Lyu, M.R.: 经验报告：用于异常检测的系统日志分析。载于：2016年 IEEE 第27届国际软件可靠性工程研讨会（ISSRE），第207–218页（2016）。https://doi.org/10.1109/ ISSRE.2016.21[112] Reimers, N., Gurevych, I.: Sentence-BERT：使用孪生 BERT 网络生成句子嵌入（2019）。https://doi.org/10.48550/arXiv.1908.10084 . https:// arxiv.org/abs/1908.10084 [113] Zhang, L., Jia, T., Tan, X., Huang, X., Jia, M., Liu, H., Wu, Z., Li, Y.: E-log：面向数据库的细粒度弹性日志异常检测与诊断。《IEEE 服务计算汇刊》（2025）[114] Qiu, K., Zhang, Y., Feng, Y., Chen, F.: Loganomex：一种基于ELECTRA-DP和门控双线性神经网络的无监督日志异常检测方法。《网络与系统管理杂志》33(2)，1–29（2025）[115]Qiu, K., Yan, M., Luo, T., Chen, F.: Fedaware：一种基于分形收缩自编码器的分布式物联网入侵检测方法。《沙特国王大学计算机与信息科学杂志》37(7)，1–21（2025）[116] Kingma, D.P., Ba, J.: Adam：一种随机优化方法（2017）。https://doi.org/10.48550/arXiv.1412.6980 . https://arxiv.org/abs/1412.6980[117] Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., Gao, J.: 大语言模型：综述（2024）。https://doi.org/10.48550/ arXiv.2402.06196 . https://arxiv.org/abs/2402.06196 [118] Al-Zoghby, A.M., Al-Awadly, E.M.K., Ebada, A.I., Awad, W.A.: 多模态机器学习概述。《ACM 亚洲及低资源语言信息处理汇刊》24(1)，1–20（2025）[119] Liang, P.P., Zadeh, A., Morency, L.-P.: 多模态机器学习基础与趋势：原则、挑战与开放问题。《ACM 计算

《Surveys》56(10)，1–42（2024）

[120] Qiu, K., Zhang, Y., Zhao, J., Zhang, S., Wang, Q., Chen, F.：基于联合链式交互注意力机制的多模态情感分析方法。Electronics**13**（10），1922（2024）[121] Su, Y., Tan, Y., An, S., Xing, M., Feng, Z.：语义驱动的双重一致性学习用于弱监督视频异常检测。Pattern Recognition**157**，110898（2025）https://doi.org/10.1016/j.patcog.2024.110898[122] Su, Y., Li, J., An, S., Xing, M., Feng, Z.：采用局部到全局专家混合的联邦弱监督视频异常检测。Information Fusion**123**，103256（2025）https://doi.org/10.1016/j.inffus.2025.103256[123] Studiawan, H., Sohel, F., Payne, C.：支持取证分析的自动日志解析器。载于：第16届澳大利亚数字取证会议（2018）[124] He, P., Zhu, J., Zheng, Z., Lyu, M.R.：Drain：一种采用固定深度树的在线日志解析方法。载于：2017 IEEE国际Web服务会议（ICWS），第33–40页（2017）。https://doi.org/10.1109/ICWS.2017.13[125] Mikolov, T., Chen, K., Corrado, G., Dean, J.：向量空间中的词表示高效估计（2013）。https://doi.org/10.48550/arXiv.1301.3781。https://arxiv.org/abs/1301.3781[126] Nguyen, H.M., Cooper, E.W., Kamei, K.：面向不平衡数据分类的边界过采样。International Journal of Knowledge Engineering and Soft Data Paradigms**3**（1），4–21（2011）[127]He, H., Bai, Y., Garcia, E.A., Li, S.：ADASYN：用于不平衡学习的自适应合成采样方法。载于：2008 IEEE国际神经网络联合会议（IEEE世界计算智能大会），第1322–1328页（2008）。https://doi.org/10.1109/IJCNN.2008.4633969[128] Mani, I., Zhang, I.：针对不平衡数据分布的k近邻方法：涉及信息抽取的案例研究。载于：不平衡数据集学习研讨会论文集，第126卷，第1–7页（2003）。ICML 美国[129] Ivan, T.：编辑最近邻规则的实验。I.E.E.E. TRANS. SYST. MAN CYBERN.；美国；1976；第6卷；第6期；第448–452页；参考文献3篇（1976）[130] Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J., Napolitano, A.：缓解类别不平衡的混合方法RusBoost。IEEE系统、人类与控制论汇刊：A辑：系统与人类**40**（1），185–197（2010）https://doi.org/10.1109/TSMCA.2009.2029559

[131] Hido, S., Kashima, H., Takahashi, Y.：面向不平衡数据的近似平衡装袋。《统计分析与数据挖掘》：ASA数据科学期刊 2（5-6），412–426（2009）https://doi.org/10.1002/sam.10061[132] Zhu, J., He, S., He, P., Liu, J., Lyu, M.R.：Loghub：用于人工智能驱动的日志分析的大型系统日志数据集集合。载于：2023年IEEE第34届国际软件可靠性工程研讨会（ISSRE），第355–366页（2023）。https://doi.org/10.1109/ISSRE59848.2023.00071[133] Marty, R., Chuvakin, A., Tricaud, S.：蜜网项目取证挑战赛2010之挑战5——日志谜案（2010）。https://www.honeynet.org/challenges/forensic-challenge-5-log-mysteries/[134] Garfinkel, S.：nps-2009-casper-rw：来自可启动USB的ext3文件系统。检索日期：2017年8月21日（2009）[135] Casey, E., Richard III, G.G.：DFRWS 2009年取证挑战赛。检索日期：2017年8月21日（2009）[136] Arcas, G., Gonzales, H., Cheng, J.：蜜网项目取证挑战赛2011之挑战7——受入侵服务器的取证分析。检索日期：2017年8月21日（2011）[137] Lin, Q., Zhang, H., Lou, J.-G., Zhang, Y., Chen, X.：基于日志聚类的在线服务系统问题识别。载于：第38届国际软件工程会议伴随论文集。ICSE '16，第102–111页。计算机协会，纽约州纽约市，美国（2016）。https://doi.org/10.1145/2889160.2889232 。https://doi.org/10.1145/2889160.2889232[138] Oliner, A., Stearley, J.：超级计算机揭示的信息：五种系统日志研究。载于：第37届IEEE/IFIP年度国际可靠系统与网络会议（DSN'07），第575–584页（2007）。https://doi.org/10.1109/DSN.2007.103[139] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.：PyTorch：一种命令式风格的高性能深度学习库。载于：Wallach, H., Larochelle, H., Beygelzimer, A., Alch´e-Buc, F., Fox, E., Garnett, R.（编）：神经信息处理系统进展，第32卷。Curran Associates, Inc.，???（2019）[140] Lemaˆıtre, G., Nogueira, F., Aridas, C.K.：Imbalanced-learn：用于应对机器学习中不平衡数据集困境的Python工具箱。《机器学习研究杂志》18（17），1–5（2017）[141] Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J.E., Stoica, I.：Tune：一种

分布式模型选择与训练研究平台（2018）。https://doi.org/10.48550/arXiv.1807.05118 。https://arxiv.org/abs/1807.05118[142] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., 等：Scikit-learn：Python中的机器学习。《机器学习研究杂志》12，2825–2830（2011）[143] Bodik, P., Goldszmidt, M., Fox, A., Woodard, D.B., Andersen, H.：数据中心指纹识别：性能危机的自动分类。载于：第五届欧洲计算机系统会议论文集。EuroSys '10，第111–124页。计算机协会，纽约，NY，美国（2010）。https://doi.org/10.1145/1755913.1755926 。https://doi.org/10.1145/1755913.1755926[144] Liu, F.T., Ting, K.M., Zhou, Z.-H.：孤立森林。载于：2008年第八届IEEE国际数据挖掘会议，第413–422页（2008）。https://doi.org/10. 1109/ICDM.2008.17[145] Arnold, C., Biedebach, L., K¨upfer, A., Neunhoeffer, M.：超参数在机器学习模型中的作用及其调优方法。《政治学研究与方法》12(4)，841–848（2024）https://doi.org/10.1017/psrm. 2023.61[146] Yang, L., Shami, A.：机器学习算法的超参数优化：理论与实践。《神经计算》415，295–316（2020）https://doi. org/10.1016/j.neucom.2020.07.061[147] Li, Z., Kamnitsas, K., Glocker,B.：分析神经网络在类别不平衡情况下进行图像分割时的过拟合。IEEE医学影像汇刊40(3)，1065–1077（2021）https://doi.org/10.1109/TMI.2020.3046692[148]Bichri, H., Chergui, A., Hain, M.：研究训练集/测试集划分比例对预训练模型在自定义数据集上性能的影响。《国际高级计算机科学与应用杂志》15(2)（2024）[149] Chen, W., Yang, K., Yu, Z., Shi, Y., Chen, C.P.：不平衡学习综述：最新研究、应用与未来方向。《人工智能评论》57(6)，137（2024）https://doi.org/10.1007/s10462-024-10759-6[150] Johnson, J.M., Khoshgoftaar, T.M.：类别不平衡下的深度学习综述。《大数据杂志》6(1)，1–54（2019）https://doi.org/10.1186/s40537-019-0192-5[151] Hart, P.：压缩最近邻规则（通信）。IEEE信息论汇刊14(3)，515–516（1968）

[152] Laurikkala, J.：通过平衡类别分布改进对难分类小类别的识别。载于：Quaglini, S., Barahona, P., Andreassen, S.（编）《医学人工智能》，第63–66页。施普林格，柏林，海德堡（2001）。https://doi.org/10.1007/3-540-48229-6 9

[153] Kubat、M.、Matwin、S. 等：应对训练集类别不平衡的困境：单侧选择。载于：ICML，第 97 卷，第 179 页（1997 年）。CiteSeer