基于大语言模型的小样本日志异常检测

袁紫依[1,2], 张昊星[3], 张媛媛[3], 伍高飞[1,2], 张玉清[1,2,4]

[1]西安电子科技大学 广州研究院 广州 中国 510555
[2]中国科学院大学国家计算机网络入侵防范中心 北京 中国 101408
[3]中国信息通信研究院安全研究所 北京 中国 100191
[4]海南大学网络空间安全学院 海口 中国 570228

摘要 随着系统复杂性的增加, 日志规模也越发庞大, 人工对其分析已经变得不切实际。许多研究者提出了深度学习方法与日志异常检测相结合。然而, 这些方法也面临着一些挑战, 现有的基于深度学习的日志异常检测方法通常存在训练开销大、依赖于高质量训练数据以及需要定期重新训练等问题。最近, 大语言模型在许多领域如机器翻译、语言理解等领域展现出了强大的实力。因此本文将大语言模型与日志异常检测相结合, 通过利用大语言模型丰富的预训练知识, 提出了一种高效且无需微调的小样本场景下的日志异常检测方法。该方法首先采用分层次聚类, 从大量的正常日志中, 提取出一个小的、多样的、具有代表性的正常日志消息合集作为候选集, 可以反映出正常日志的广泛模式。同时采用基于解释的提示学习, 解释候选集中的每一条正常日志被判定为正常的原因, 增强模型对正常日志模式的理解。同时, 依据不同日志数据集的特征, 采用基于思维链的提示策略, 为不同的数据集构建了特定的提示模版。此外, 本文设计的提示模版在零样本场景下也能有效地进行日志异常检测。与现有日志异常检测方法相比, 该方法只需要极少量的训练数据, 就可以达到较高的准确度, 极大地减少了模型训练的开销, 且当日志进行大规模更新时, 也无需重新训练模型。为了评估该方法的性能, 使用两个公共数据集验证模型的有效性, 本文提出的方法在 BGL、Spirit 数据集上的 F1 分数分别为 81.54%和 96.55%, 且在两个数据集上的召回率分别为 95.00%和 97.77%, 本文提出的方法在 2 种数据集上都有着较高的召回率和 F1 值。实验表明, 只需要极少量训练数据的情况下, 本文提出的方法可以有效实现日志异常检测。

关键词 异常检测; 深度学习; 大语言模型; ChatGPT
中图法分类号 TP391.1 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2024.11.02

Few-Shot Log Anomaly Detection via Large Language Models

YUAN Ziyi[1,2], ZHANG Haoxing[3], ZHANG Yuanyuan[3], WU Gaofei[1,2], ZHANG Yuqing[1,2,4]

[1] Guangzhou Institute of Technology, Xidian University, Guangzhou 510555, China
[2] National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing 101408, China
[3] Security Research Institute of China Academy of Information and Communications Technology, Beijing 100191, China
[4] College of Cyberspace Security, Hainan University, Haikou 570228, China

Abstract With the increase of system complexity, the scale of log grows larger, making it impractical to analyze them manually. Some researchers have proposed deep learning methods combined with log anomaly detection. However, these methods face several challenges, existing log anomaly detection methods based on deep learning often have issues such as high training cost. Additionally, they rely heavily on high-quality training data and need to be retrained regularly. Recently, Large Language Models have shown promising results in various domains such as machine translation, language understanding and so on. In our work, we combine Large Language Models with log anomaly detection. By leveraging the rich pre-training knowledge of Large Language Models, we propose an efficient log anomaly detection method in few-shot scenarios without fine-tuning. The method employs hierarchical clustering to extract a small, diverse, and representative collection of normal log messages as a candidate set, which can reflect a wide range of normal log patterns. Additionally, we propose explanation-based prompt learning, which is used to explain each normal log in the candidate set, this method can enhance the model's understanding of normal log patterns. According to the characteristics of log datasets, a specific prompt template for different log datasets is constructed by using the chain of thought strategy. Therefore, the specific prompt template proposed in this paper can also effectively detect log anomalies in zero-shot scenarios. Compared with the existing log anomaly detection methods, the method only requires a very small amount of training data and can achieve

通讯作者: 张玉清, 博士, 教授, Email: zhangyq@nipc.org.cn。
本课题得到国家重点研发计划项目(No. 2023YFB3106400, No. 2023QY1202); 国家自然科学基金重点项目(No. U2336203, No. U1836210); 海南省重点研发计划项目(No. GHYF2022010); 北京市自然科学基金(No. 4242031)资助。
收稿日期: 2024-03-31; 修改日期: 2024-06-11; 定稿日期: 2024-09-05

high accuracy, which greatly reduces the cost of model training. When the log is updated on a large scale, there is no need to retrain the model. To evaluate the performance of the method, we use two public datasets to verify the effectiveness of the model. The F1 scores of the proposed method on BGL and Spirit datasets reach 81.54% and 96.55% respectively, and the recall scores on two datasets reach 95.00% and 97.77% respectively. The proposed method has high recall scores and F1 scores on two datasets. The results demonstrate that the proposed method is able to effectively achieve log anomaly detection with only a very small amount of training data.

Key words anomaly detection; deep learning; large language model; ChatGPT

1 引言

日志作为记录系统运行状态的重要数据, 能够反映系统的各种操作、事件和状态信息, 这些信息对安全分析和故障排除十分重要。随着计算机软硬件的飞速发展, 其运行产生的日志也急剧增加, 例如, 某通讯软件每小时可以产生超过 50GB 的日志[1], 从而导致传统的关键字匹配或者静态规则匹配方法的效率降低, 错误率增高。

近年来, 研究者们提出了基于机器学习的方法 [2], 包括支持向量机(Support Vector Machine, SVM)[3]、主成分分析(Principal Component Analysis, PCA)[4]。LogCluster[5]采用日志聚类的方式来进行日志异常检测, 首先将每个日志序列表示为特征向量, 通过计算新日志序列向量与已有簇群代表向量之间的相似度值来进行异常检测, 但不能捕获日志文本的语义信息, 容易导致高误报率。Log2Vec[6]则是基于异构图嵌入的日志异常检测方法, 将日志序列转换为低纬度的词嵌入后, 使用异常检测方法来进行聚类, 但该方法上下文检测能力较弱。

自 DeepLog[7]被提出后, 基于日志序列的深度学习建模逐渐成为研究热点。DeepLog 采用长短期记忆神经网络模型, 将日志建模为自然语言序列, 通过学习正常执行中日志数, 并将偏离正常运行的行为标记为异常。Guo 等[8]提出了一个基于 BERT 模型的日志异常检测模型 LogBERT, 该模型通过学习正常日志序列的模式, 能够检测出偏离正常日志序列模式的异常。最近大语言模型, 如 ChatGPT[9], 在许多领域如机器翻译[10]、语言理解[11]等领域展现出了强大的实力。Qi 等[12]将 ChatGPT 与日志异常检测相结合, 首次提出了一种基于 ChatGPT 的日志异常检测框架 LogGPT, 通过利用 ChatGPT 的语言交互功能, 为日志异常检测设计了特定的提示策略, 该方法在零样本场景下进行检测, 即使不需要训练集也能够对异常日志进行检测, 极大地减少了模型训练的开销。

尽管现有方法在日志异常检测中取得了显著的性能, 但日志异常检测技术在实际场景中的应用仍然面临着挑战。首先, 现有方法大多依赖于大量高质量的训练数据, 如果数据不足或事缺乏代表性, 模型可能无法准确学习正常日志序列的模式。其次, 随着系统的更新和演变, 原有的模型可能无法使用新的日志格式和行为模式, 需要定期重新训练。为了应对这些挑战, 本文将 ChatGPT 与日志异常检测相结合, 提出了一种高效且无需微调的小样本场景下的日志异常检测方法, 采用分层次聚类的方法提取出一个小的、多样的、具有代表性的正常日志消息合集作为候选集, 并采用基于解释的提示学习, 为候选集中的每一条日志示例生成解释, 同时为异常检测构建了特定的提示策略。

本文的主要工作如下:

(1) 以大语言模型模型为基础, 提出了一种新的日志异常检测方法。该方法在小样本场景下进行检测, 无需对模型进行微调, 只需要极少量的训练数据, 就可以达到较高的精确度, 极大地减少了模型训练的开销。

(2) 设计了一种高效的候选采样算法以及示例选择算法。采用分层聚类算法从大量正常日志中提取出一个具有代表性的候选集, 同时引入了一种基于解释的提示学习策略。该策略能够分析并阐述候选集中每条日志被认定为正常的关键因素, 从而深化模型对日志内容的理解, 作出更准确的判断。

(3) 使用 BGL 和 Spirit 数据集两个公共数据集来评估模型的有效性, 所提出的方法在两个数据集上的 F1 分数分别为 81.54%和 96.36%, 且召回率分别为 95.00%和 96.74%。实验结果表明, 只需要极少量训练数据的情况下, 本文提出的方法可以有效实现日志异常检测。

2 相关工作

2.1 日志解析

日志解析的目的是将半结构化的日志消息转换为结构化的日志数据, 并提取出相应的参数, 如图 1 所示。一条日志语句主体通常由两部分组成: (1)模版, 通常是一段自然语言, 是用于描述系统事件的静态关键词, 这些关键词在日志语句的代码中明确写出; (2)参数, 例如对象 ID、执行时间、内存消耗等等, 表

明了系统状态关键变量的取值。日志解析是日志异常检测中的一个重要组成部分, 现有方法大致可以分为以下四种:

(1) 基于聚类的方法: 通过文本相似度或距离公式, 判断日志是否属于某一类模式, 并从每个聚类中提取日志模版。代表方法有 LKE[13]。

(2) 基于启发式的方法: 通过分析日志数据的统计特性和模式来提取通用模版, 该方法依赖于启发式的规则和模式识别来解析日志文件, 不需要事先定义日志的结构或格式。代表方法有 Drain[14]。

(3) 基于频繁模式挖掘的方法: 对日志事件中项的频率进行挖掘, 对频率进行统计, 将日志中频繁出现的常量项提取出来组成日志模版。代表方法有 LFA[15]。

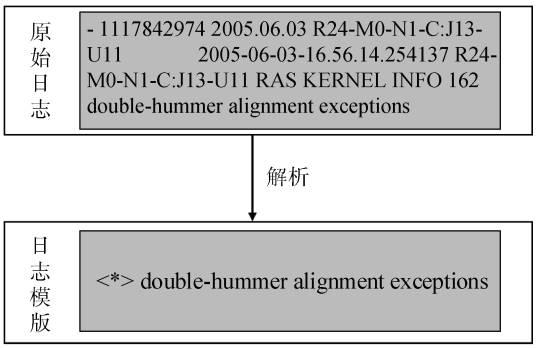

图 1 日志解析

Figure 1 Log parsing

2.2 日志异常检测

日志异常检测的工作过程通常是使用机器学习等方法对特征向量进行学习, 从而生成异常检测的模型, 通过模型来识别出异常日志。日志异常检测任务一般分为日志解析、特征表示和异常检测三个阶段。现有日志异常检测方法可以分为基于机器学习的日志异常检测和基于深度学习的日志异常检测。

(1) 基于机器学习的日志异常检测: 随着机器学习的发展, 研究者们将日志序列构建成时间序列, 再使用机器学习的方法来进行检测。例如, Xu 等[16]构建状态变量比率向量和消息计数向量, 再通过 PCA 来检测出异常日志。Ying 等[17]提出了一种基于 N-gram 和频繁模式挖掘的方法, 然后使用聚类和自训练的方法从历史日志中自动获取标记日志数据集, 使用平均加权技术对 KNN 算法进行了改进。LogCluster[5]采用事件计数向量来表示日志序列, 对日志序列进行聚类, 通过计算新日志序列向量与已聚类簇群代表向量之间的相似度来检测异常。

(2) 基于深度学习的日志异常检测: 由于日志数据的内在复杂模式以及其具有长距离依赖等特点, 因此研究者们将深度学习应用于日志异常检测领域中。DeepLog[7]利用 2 层叠加的长短期记忆神经网络实现了对日志的异常检测。LogAnomaly[18]对日志模版语义表达进行了改进, 使用 Word2Vec[19]从日志模版中提取日志的语义和语法信息, 再将生成的模版向量输入到模型中进行训练, 最终用于预测推断。LogRobust[20]采用双向长短期记忆神经网络来实现日志异常检测, 该模型可以捕获日志序列中的上下文信息, 并且为重要程度不同的日志事件分配不同的权重, 降低了因日志解析错误而造成的影响。LogBERT[8]使用 BERT 模型来捕获整个日志序列的信息, 通过遮蔽日志键预测和超球体最小化体积两个字监督训练任务来学习正常日志序列的模式, 最终模型能够检测出偏离正常模式的异常。

3 模型介绍

本文提出一种基于大语言模型的日志异常检测方案, 如图 2 所示。首先, 对原始日志序列进行解析, 得到结构化的日志信息, 提取出对应的日志模版。其次, 设计了一种高效的分层聚类算法, 根据聚类算法从大量的正常日志数据中提取出一个最小比例的日志样本集, 以构建一个正常日志候选集合, 同时

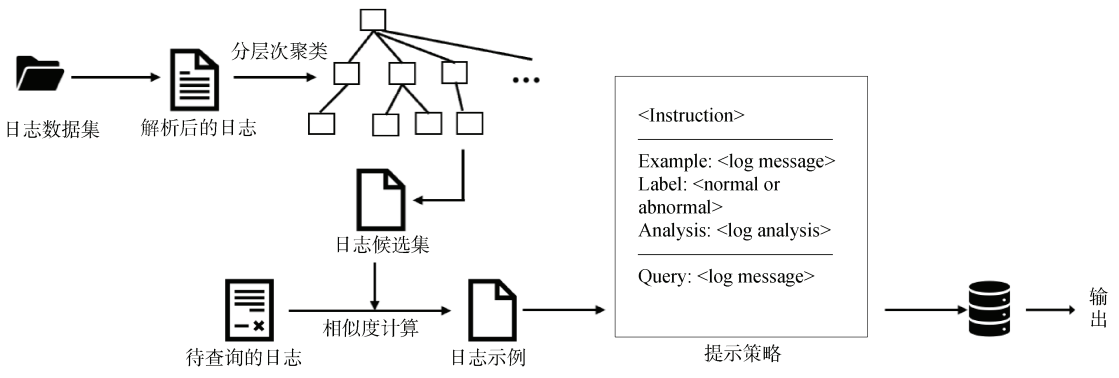

图 2 模型总框架

Figure 2 General framework of model

给出其中每一条日志被判定为正常日志的分析原因。当给定一个新的日志序列作为查询时, 模型从候选集合中选取相似度最高的 K_s 个示例, 组合成特殊格式的提示, 在特定提示策略的指导下生成应答。

3.1 日志解析

原始日志中通常包含大量重复的非结构化数据, 日志解析的目标则是将原始日志中的日志模版、动态变量和头信息等提取出来, 并采用结构化的格式进行表示。传统的日志解析方法依赖于手工编写的正则匹配来提取模版和关键参数, 但这需要大量人力, 以及需要定期维护模版更新。因此为了提高异常检测的效率和准确性, 本文采用先进的日志解析算法 Drain[14]从日志中提取相关数据。

Drain 是一种基于固定深度树的在线日志解析器, 通过 Drain 解析得到日志模版。原始的日志消息为“-1118767381 2005.06.14 R20-M0-N2-C:J13-U11 2005-06-14-09.43.01.478244 R20-M0-N2-C:J13-U11 RAS KERNEL FATAL program interrupt: illegal instruction.....0”。首先, 通过简单的正则表达式对其进行预处理, 得到结构化的日志序列, 其中每一条序列都包括“Label”“Content”“EventId”以及“EventTemplate”等信息, 之后再通过 Drain 提取到日志模版。

3.2 构建日志候选集

3.2.1 分层次聚类算法

在零样本场景下进行测试后发现, 模型的假阳率较高, 因此为了提升模型的准确率以及选取高质量的示例, 本文从大量的正常日志数据中提取出一个小的、多样的、有代表性的正常日志消息合集, 以实现候选集构建的最大多样性。首先对每个日志消息 token 化, 然后计算所有 token 的频率, 同时排除停用词, 以消除不相关的标记。对每个日志消息, 选择出现频率最高的前 K 个 token(K=5), 这些 token 构成了它们被归类到不同粗粒度簇的基础, 再利用日志消息的特殊格式进行细粒度的聚类。最后, 从每个细粒度簇中选择一条日志消息加入候选集合中。

3.3 日志示例选择

在日志异常检测过程中, 为了减少无关信息的干扰, 需要从日志候选集中选择 K_s 个示例来构建提示。这些示例应该表现出与查询的日志序列紧密相关, 模型可以从中学习查询的语义和模式。为了加强模型的准确性, 我们采用一种简单的聚类算法 kNN 来选择日志示例。对于每个被查询的日志序列, 我们计算它与所有候选日志消息 $sim(l,s_i)$, i 属于[1, K_s]之间的相似度。我们提出从基于 token 和特殊格式两个方面来度量日志消息之间的相似性。当给定一个日志消息 l 时, 我们提取 l 中 token 字符和其中的特殊字符, 以形成 l 的特征集合, 即 $F(l)$。基于此, 我们可以通过计算特征集合之间的 Jaccard 相似度计算 $sim(l_1,l_2)=\frac{|F(l_1)\cap F(l_2)|}{|F(l_1)\cup F(l_2)|}$。在计算所有的相似度之后, 我们选择候选集中相似度最高的 K_s 条日志示例。

3.4 构建提示策略

3.4.1 基于思维链的提示策略

Wei 等[21]引入了“思维链”(Chain of Thought, CoT)概念, 即一系列中间推理步骤。在解决复杂问题时, CoT 可以模拟人类的思维过程, 从而增强大语言模型在挑战性任务重的性能。在没有给出明确定义的情况下, 正常日志和异常日志之间的界限是不明确的。我们经过初步的测试发现, 当使用如图 3 所示的简单提示进行测试时, 大语言模型会出现假阳率较高的情况, 因此我们构建了一种基于 CoT 的提示策略, 通过隐式和显式来引导大语言模型执行异常检测, 如图 4 所示。(1)隐式: 我们要求大语言模型为每一组日志序列给出一个分析原因, 以作为其将日志判定为正常/异常的依据。通过分析原因, 模型不太可能产生明显不合逻辑的答案。(2)显式: 我们在提示内容中明确给出异常日志和正常日志的定义, 以防止模型过度思考。例如, “如果序列包含纠错或状态更新措施, 将日志标记为正常”, 以及“如果序列包含有关系统、硬件、通信、网络的错误, 将其标记为异常”。同时我们要求模型在做出判断时, 重点关注日志语义内容, 通过语义分析来判断日志是否正常, 同时需要参考示例日志给出的解释内容。根据 BGL 和 Spirit 的不同特征, 设计了两种不同的复杂提示, 如图 4 所示, 针对 BGL 数据集, 我们设计了模版 1, 重点检测系统和硬件错误, 并区分正常的错误修正日志; 针对 Spirit 数据集, 我们设计了模版 2, 重点检测应用层面的异常和关键故障, 同时关注系统故障的显式警报。

简单提示

Your task is to analyze the entire log sequence (sorted by timestamp) and classify it into normal or abnormal. Output format: Only return back in a JSON format with the following keys: analysis, label('abnormal' or 'normal').

图 3 日志异常检测简单提示策略

Figure 3 Simple prompt of log-based anomaly detection

提示模版1

Your task is to analyze the entire log sequence (sorted by timestamp) and classify it into normal or abnormal. Do it with these steps:(1)Pay attention to the semantic content of each log.(2)Pay attention to log that apparently different from examples.(3)If the contents of sequence are similar closely to examples, analyze the log based on the analysis.(4)If it contains error corrective actions or status updates (such as 'CE' or 'check'),consider it as normal.(5)If it contains errors about system or hardware or communication or network(such as keywords 'like', 'interrupt', 'panic', 'failed', 'severed'),consider it as abnormal. Output format: Only return back in a JSON format with the following keys: analysis, label('abnormal' or 'normal').
Here are some normal log messages:
-Example Log:
-analysis:
Follow the above instructions to analyze the following log sequence, provide which step(s) that supported your conclusion.

提示模版2

Your task is to analyze the entire log sequence (sorted by timestamp) and classify it into normal or abnormal. Do it with these steps:(1)Pay attention to the semantic content of each log.(2)If the contents of sequence are similar closely to examples, analyze the log based on the analysis.(3)Pay close attention to explicit alerts related to hardware or system failures.(4)If it contains repeated specific kernel errors or critical failures or unrecoverable errors, consider it as abnormal. Output format: Only return back in a JSON format with the following keys: analysis, label('abnormal' or 'normal').Here are some normal log messages:
-Example Log:
-analysis:
Follow the above instructions to analyze the following log sequence, provide which step(s) that supported your conclusion.

图 4　日志异常检测复杂提示策略

Figure 4　Enhanced prompt of log-based anomaly detection

3.4.2　基于解释的提示学习

由于在零样本场景下进行测试后我们发现, 模型的假阳率(即错误地将正常日志标记为异常的情况)较高, 因此为了提高模型的准确性, 本文采取了一种基于深入理解正常日志行为模式的策略来更好地识别异常。具体来说, 我们定义了一系列的提示, 明确要求模型解释为什么特定的日志条目被视为正常日志, 并为日志候选集中的每一条日志示例生成解释。如图 5 所示。

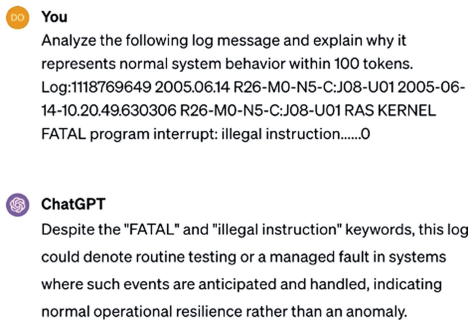

图 5　基于解释的提示学习

Figure 5　Explanation-based prompt learning

4　实验

4.1　数据集

如表 1 所示, 本文在 2 个公共数据集上评估所提出的方法。2 个数据集是指: BGL[22](Blue Gene/L,BGL)和 Spirit[23]。

表 1　2 个公共数据集统计信息

Table 1　Statistics information of two public datasets

数据集	条数	模版数量(2k)	异常条数	是否有标签
BGL	4713493	120	348460	是
Spirit	272298569	—	172816564	是

BGL: BGL 数据集是从部署在 Lawrence Livermore 国家实验室的 BlueGene/L 超级计算机收集到的 4747963 条日志数据集, 其中 348460 条日志(7.34%)被标记为异常。

Spirit: Spirit 数据集采集自 Sandia 国家实验室的 Linux 生产集群, 包含 512 个节点, 时间跨度为 23 天。其中包含了 272298969 条日志消息, 其中 172816564 条(63.47%)被标记为系统异常。

选择以上两个数据集的原因如下: 1)这些数据集通常用于日志异常检测方法的评估; 2)这些数据集包含可以用来计算评估指标的真实标签; 3)这些数据集包括可以用于将日志消息分组的日志标识符。对于数据集中的每一组日志消息, 如果一组日志序列中包含至少一个异常, 那么就将该日志序列标记为异常。

我们将训练集和测试集按照 6∶4 的比例进行分割。由于大语言模型的 API 请求速率受到限制, 因此我们采用随机抽样的方法从测试集中选择 2000 组日志序列, 同时进行手动检查, 确保测试集内异常日志的比例。

4.2 基准方法

实验将本文提出的方法与下面 4 种基线方法进行了对比:

(1) DeepLog[7]: 使用长短期记忆神经网络将日志序列建模为自然语言序列。

(2) LogAnomaly[18]: 应用双向长短期记忆神经网络和注意力机制来进行异常检测。

(3) LogBERT[8]: 基于 BERT 模型的日志异常检测模型 LogBERT, 该模型通过学习正常日志序列的模式, 能够检测出偏离正常日志序列模式的异常。

(4) LogGPT[12]: 一种基于 ChatGPT 的日志异常检测框架 LogGPT, 通过利用 ChatGPT 的语言交互功能, 为日志异常检测设计了特定的提示策略。

(5) ERINE[24]: 通过融合自回归网络和自编码网络, 由纯文本和大规模知识图谱组成的 4TB 语料库上训练的大规模知识增强模型。

4.3 评价指标

为了衡量本文提出的方法在日志异常检测中的有效性, 我们使用精确度(Precision)、召回率(Recall)、F1(F1-Score)作为评价指标。其计算公式如式(1)～式(3)所示:

$$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \tag{1}$$

$$\text{Recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \tag{2}$$

$$\text{F1}=\frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \tag{3}$$

其中, TP(True Positive, TP)表示模型正确预测出的异常日志序列的数量。FP(Fake Positive, FP)被模型错误识别为异常的正常日志序列的数量。FN(Fake Negative, FN)表示被模型判定为正常的异常日志序列数量, 即没有被检测出的异常日志序列的数量。TN(True Negative, TN)表示模型正确识别出的正常日志序列的数量。

4.4 实验设定

在本文的初步实验中, 底层的大语言模型是 ChatGPT, 并通过 OpenAI 提供的 gpt-3.5-turbo-0125 来进行实验。在对比试验中, 采用的是百度千帆大模型提供的 ERINE-3.5-8K 来进行测试。为了减少模型的不稳定性, 我们将温度设置为 0, 日志序列窗口大小设置为 10, 示例数量设置为 3。

4.5 对比实验

4.5.1 不同模型的性能对比

表 2～表 3 分别给出了几种方法在 2 个数据集上的精确度、召回率、F1 值。如表 2 和表 3 所示, 对于非大语言模型, DeepLog 在 BGL 和 Spirit 数据集上的 F1 分数分别为 68.29%和 77.30%, LogAnomaly 在 BGL 和 Spirit 数据集上的 F1 分数均为 67.86%, 这两种方法在日志异常检测中性能较差, 这是因为 DeepLog 和 LogAnomaly 都采用长短期记忆神经网络构建异常检测模型, 该模型通过按照顺序处理日志序列数据, 每一步的输出依赖于前一步的计算结果, 这种处理方式限制了模型同时捕获长距离依赖关系的能力, 也无法学习日志序列内部的全局连接特征, 此类方法在日志异常检测中存在一定的局限性。此外, DeeoLog 是基于日志模版索引来进行日志异常检测, 不能捕获到模版的语义信息, 因此效果不如 LogAnomaly。对于 LogBERT, 该方法直接将原始日志消息转换为特征向量, 可以避免日志解析错误带来的影响, 因此 LogBERT 的表现优于 DeepLog 和 LogAnomaly, 在 BGL 和 Spirit 数据集上取得了 86.52%和 95.71%的 F1 分数。

对于大语言模型, 在没有训练数据的情况下, LogGPT 虽然能够检测到所有异常, 但由于误报过多, 导致精确度极低。同时, 我们将 ERINE 作为底层大语言模型并使用与本文相同的方法来进行实验, ERINE 在少量训练数据下表现中等, 召回率较高,

表 2 BGL 数据集上的实验结果

Table 2 Experimental results on BGL datasets

	方法	训练集条数	精确度(%)	召回率(%)	F1(%)
非大语言模型	DeepLog	1729300	**87.50**	56.00	68.29
	LogAnomaly	1729300	84.96	56.50	67.86
	LogBERT	1729300	82.06	**91.50**	**86.52**
大语言模型	LogGPT	0	10.06	100.0	18.28
	ERINE	92	41.41	76.00	53.61
	本文	**92**	**71.42**	**95.00**	**81.54**

表 3 Spirit 数据集上的实验结果

Table 3 Experimental results on Spirit datasets

	方法	训练集条数	精确度(%)	召回率(%)	F1(%)
非大语言模型	DeepLog	1019960	63.00	100.00	77.30
	LogAnomaly	1019960	63.00	100.00	77.30
	LogBERT	1019960	97.76	93.73	**95.71**
大语言模型	LogGPT	0	63.06	100.0	77.34
	ERINE	113	83.53	81.34	82.42
	本文	**113**	**95.35**	**97.77**	**96.55**

意味着其能够检测出大多数的异常, 但精确度相对较低, 在 BGL 和 Spirit 数据集上分别取得了 41.41%和 83.53%的精确度, 因此导致 F1 分数值处于中间水平。而本文提出的方法在 BGL、Spirit 数据集上的 F1 分数分别为 81.54%和 96.55%, 且在两个数据集上的召回率分别为 95.00%和 97.77%, 所以本文提出的方法在 2 种数据集上都有着较高的召回率和 F1 值。可以看出, ERINE 表现不如本文实验, 我们猜测是 ERINE 在预训练阶段没有充分涉及与日志相关的语料, 在理解复杂日志数据和减少误判方面表现相对较差。其次, ChatGPT 在设计时可能更加注重设计模型的适应性, 能够更好地调整其语言模型以适应具体的应用场景, 比如日志异常检测。此外, 本文提出的方法利用到的训练集大小远远少于其他基准方法, 只需要极少量的训练数据, 就可以实现较高的性能, 本文提出的方法在 BGL 数据集上取得了最好的召回率, 在 Spirit 数据集上取得了最好的 F1 值。实验表明, 只需要极少量训练数据的情况下, 本文提出的方法可以有效实现日志异常检测。

4.5.2 不同样本数量下模型性能对比

采用提示模版 1 和提示模版 2 分别对 BGL 和 Spirit 数据集进行测试, 在不同的样本数量条件下对本文提出的模型性能进行对比, 结果如表 4 所示。

对于 BGL 数据集, 在样本数量为 1 时, 模型能够识别出大部分异常数据, 但由于精确度低, 许多正常日志被识别为异常日志, 导致 F1 分数值较低。当样本数量为 3 时, 模型性能显著提升, 特别是 F1 值与精确度, 召回率略有增加, 模型在异常日志识别方面更加准确。当样本数量增加到 5 时, 各方面性能都有所下降, 可能是由于样本数量增加导致模型复杂度增加。

对于 Spirit 数据集, 在不同样本数量下模型整体性能稳定, 即使在样本数量较少的情况下, 模型性能也表现较好。

4.5.3 不同提示模版下模型性能对比

为了验证在不同提示模版下的模型的性能, 本文在零样本场景下分别采用简单提示和复杂提示进行测试, 实验结果如表 5 所示。

表 4 不同样本数量下的实验结果

Table 4 Experimental results with different shots

数据集	样本数量	精确度(%)	召回率(%)	F1(%)
BGL	1	43.10	87.50	57.75
	3	**71.42**	**95.00**	**81.54**
	5	61.12	87.00	71.90
Spirit	1	**95.49**	97.61	96.54
	3	95.35	**97.77**	96.55
	5	95.08	96.66	**95.86**

表 5 不同提示模版下的实验结果

Table 5 Experimental results with different templates

数据集	模版类型	FP	TN	TP	FN
BGL	简单提示	1767	33	200	0
	复杂提示	1343	457	186	14
Spirit	简单提示	682	58	1260	0
	复杂提示	137	603	1258	2

在简单提示下, 模型的假阳率较高, BGL 数据集中有 1800 组正常日志序列, 其中只有 33 组正常日志被正确识别, Spirit 数据集中有 740 组正常日志序列, 其中只有 58 组正常日志被正确识别。

在复杂提示下, 模型的性能显著提高。在 BGL 数据集中, 有 457 组正常日志被正确识别, 与简单提示相比, 假阳率有所降低。在 Spirit 数据集中, 采用复杂提示后的提升效果更加明显, 有 603 组正常日志被正确识别, 与简单提示相比, 正常日志正确识别个数增长近 10 倍。同时, 异常日志的识别率也保持在较高水平, 仅有 2 组异常日志被错误识别为正常日志。

4.5.4 不同模型时间开销对比

表 6 展示了不同方法在不同数据集上的时间消耗。可以看到, 非大语言模型中, LogAnomaly 在 BGL 数据集上的训练时间最长, LogBERT 训练时间最短。在 Spirit 数据集中, 所有非大语言模型方法的训练时

间都相对较短。除了训练时间的差异外, 所有非大语言模型的测试时间都较短。

表 6　不同方法在数据集上的时间开销

Table 6　Time consumption of different methods on datasets

数据集	方法	训练时长	预测时长(s)	推理时长(min)
BGL	DeepLog	27min	2	
	LogAnomaly	2h 3min	17	
	LogBERT	9min	4	
	简单提示			13
	复杂提示-1			22
	复杂提示-3			23
	复杂提示-5			25
Spirit	DeepLog	6min	2	
	LogAnomaly	13min	24	
	LogBERT	38min	3	
	简单提示			13
	复杂提示-1			21
	复杂提示-3			22
	复杂提示-5			23

对于大语言模型, 简单提示实验是采用简单提示在零样本场景下进行的, 复杂提示实验则是采用复杂提示在样本数量为 1、3、5 场景下进行的。可以看出, 大语言模型的推理时长随着提示复杂性的增加而增加, 但总体上两个数据集的推理时长差异不大。由于本文选择的测试日志序列数量为 2000, 且每组日志序列由 10 条日志组成, 平均一组日志序列的推理时长大约在 0.66 s 左右。相比之下, 非大语言模型在预测时间上表现出更高的效率, 例如 LogBERT 在 Spirit 数据集上的预测时间仅为 3 s。

尽管大语言模型在处理复杂提示时展现了强大的能力, 但其推理时长的增加可能会影响其在实时异常检测场景中的应用, 在实时监控系统中, 依赖大语言模型进行异常检测可能会导致检测延迟, 从而影响系统的即使反应能力。

4.6　消融实验

消融实验可以验证模型各个部分的有效性, 因此分别在 BGL 和 Spirit 日志数据集上进行消融实验, 验证小样本学习、解释学习对日志异常检测模型的有效性。其中实验一代表完整实验, 实验二代表去掉基于解释的提示学习, 实验三代表同时去掉基于解释的提示学习和小样本学习。表 4 和表 5 显示了在两个数据集上进行两项消融实验的实验结果, 可以得出以下结论。(1)当取消基于解释的提示学习时, 在不同的数据集上, 模型准确率分别下降了约 17%和 7%, F1 值分别下降了 15%和 3%。这是因为该方法通过构造具有指导性的提示来引导模型的学习过程, 增强了模型对数据的理解。(2)当同时取消基于解释的提示学习和小样本学习时, 在不同的数据集上, 模型的准确率分别下降了约 33%和 5%, F1 值下降了 27%和 2%。这一变化在 Spirit 数据集上表现的并不明显, 因为 Spirit 数据集中异常日志比例高, 当模型利用小样本学习和基于解释的提示学习对日志进行测试时, 由于实验提供的样本均为正常日志, 因此模型对异常日志进行判断时, 会受到正常样本日志的影响, 错误地将异常日志判定为正常日志。而在 BGL 数据集上, 观察到了模型性能显著下降, 这是由于尽管大语言模型具有丰富的预训练知识, 但在缺乏领域知识的前提下, 其处理大范围的日志数据的能力仍然有限, 尤其是在处理这类复杂的日志分析任务中, 模型需要理解日志数据的一般性特征, 还需要能够捕捉到领域特定的细微差别, 因此每一个部分对于模型的有效运行都是不可或缺的。

表 7　BGL 数据集上的消融实验

Table 7　Ablation experimental results on BGL datasets

方法	精确度(%)	召回率(%)	F1(%)
实验一	**71.42**	**95.00**	**81.54**
实验二	53.27	89.50	66.79
实验三	38.00	93.50	54.04

表 8　Spirit 数据集上的消融实验

Table 8　Ablation experimental results on Spirit datasets

方法	精确度(%)	召回率(%)	F1(%)
实验一	**95.35**	97.77	**96.55**
实验二	88.66	**99.36**	93.71
实验三	90.12	99.20	94.44

5　讨论

目前已经有许多成熟的基于深度学习和基于机器学习的日志异常检测方法, 并且取得了不错的性能。现代软件产品更新换代速度快, 使得其运行产生的日志模式持续更新, 例如引入新的日志消息格式, 或者改变特定操作产生的日志消息的频率。这种持续的变化会导致日志模式不断演进, 导致先前训练的模型面临着检测能力下降的问题。为了克服这一挑战, 通常需要定期对模型进行重新训练来适应新的日志模式, 但这需要消耗大量的训练成本。然而对于大语言模型来说, 由于其预训练数据庞大, 具备广泛的语言理解能力, 这使得模型能够快速适应新

的日志格式和内容。

随着大语言模型的出现, 如何将大语言模型运用于日志异常检测领域, 是我们重点思考的问题。传统的日志异常检测方法往往需要大量标注的数据来训练模型, 而大语言模型可以利用预训练知识, 仅需少量标注样本或甚至无需标注样本就能够进行有效的异常检测, 显著减轻了数据标注的工作量和成本。基于我们的研究可以发现, 通过利用大语言模型丰富的预训练知识, 只是利用极少量的样本作为训练数据, 就可以有效实现日志异常检测。在训练数据方面, 与其他模型相比, 本文提出的方法只需要 113 条正常的Spirit日志作为训练数据, 就达到了97.77%的F1分数, 而LogBERT则需要1019960条数据, 最终获得了95.71%的F1分数。

为了更好地进行故障预警以及协助运维人员处理告警事件, 不应只停留于识别当前日志是否出现异常, 也应该注重检测结果的可解释性。在过去的日志检测方法中, 其决策过程是不可解释的, 因此很难为模型检测到的异常提供更多的解释, 这给运维人员及时识别和预防异常带来了挑战。大语言模型通常具有强大的自然语言理解能力, 这使得模型不仅能够检测出异常日志, 还能以自然语言的形式提供问题分析、解决建议等, 进一步提升了模型的可解释性。在本文的实验中, 我们要求模型对日志序列进行分析并提供相应的解释, 基于模型给出的分析来判断是否存在异常。同时, 要求模型指出是根据哪些具体步骤支撑其最终结论。

由于大语言模型在预训练阶段接触到了来自多个领域的文本, 具备跨领域的知识理解能力, 这使得同一个模型可以被应用于不同领域的日志异常检测任务, 而无需针对某个领域单独训练模型。由于标注了异常信息的日志数据集相对较少, 这导致大多数日志数据集难以用于进行广泛的测试验证。同时, 我们考虑到这种方法主要依赖于 ChatGPT 的语言理解能力, 通过理解日志本身的语义内容来判断系统行为是否异常。这意味着, 如果日志的表达方式比较隐晦或需要特定领域知识才能理解日志含义, 如HDFS(Hadoop Distributed File System, HDFS)的日志, 就不适用于该方法。因此, 我们选择了 BGL 和 Spirit 两个日志数据集进行实验, 这些数据集不仅具有充分的异常标注信息, 而且其日志文本在语义上相对容易被模型理解。

像 ChatGPT 这样的大语言模型是闭源且部署在云端的, 我们需要通过 API 进行访问就可以利用模型的强大功能, 但这也意味着用户需要依赖于云服务器和 API 的稳定性, 这在一定程度上限制了模型的灵活性和可定制性。本文探讨的方法不仅局限于ChatGPT, 同样适用于其他大语言模型, 如 T5[25]和OPT[26]等, 但不同大语言模型之间性能存在差异, 选择合适的大语言模型对于提升日志异常检测十分关键。同时这些模型可以在本地环境中进行部署和使用, 允许用户更深入地理解和定制模型, 提供了更大的灵活性和适应性, 特别是在本地部署和互联网访问受限的环境中。

大语言模型与非大语言模型都扮演着至关重要的角色, 这些模型在许多领域上都展现出了巨大的潜力, 每种类型的模型都有其独特的优势和局限性, 如表 9。在决定采用大语言模型或是非大语言模型来解决特定的自然语言处理任务时, 理解模型的这些关键差异及其实际应用影响至关重要, 这不仅需要评估模型的技术性能和资源消耗, 还需要考虑到模型的可维护性以及如何与现有系统集成等。

表 9 大语言模型与非大语言模型在日志异常检测领域的优缺点对比

Table 9 Comparison of advantages and disadvantages of LLM and non-LLM in log anomaly detection

	大语言模型	非大语言模型
优点	泛化能力强, 可以快速适应新的日志格式和内容	资源效率高, 对计算资源和存储的需求更低
	对数据依赖性弱, 在小样本场景也能实现准确的异常检测	针对性强, 可以针对特定类型的日志进行定制化设计
	高度的理解能力, 减轻了数据标注的工作量和成本	实时性能好, 推理速度快, 适用于实时异常检测
	成本效益高, 需要大量的计算资源和存储空间	对数据依赖性强, 需要大量的标注数据
缺点	构建有效提示的难度高, 模型过分依赖精心设计的提示来实现最佳性能	需要定期重新训练来适应新的日志格式和内容
	推理速度慢, 在需要实时异常检测的场景下可能不够理想	泛化能力弱, 对未见过的日志检测准确率低

6 结束语

本文将大语言模型与日志异常检测相结合, 提出了一种基于小样本场景下的日志异常检测方案。该方法首先采用分层次聚类算法从大量的正常日志中提取出一个小的、具有代表性的正常日志候选集, 可以反映出正常日志的广泛模式, 同时通过提示学习, 使模型深入理解正常日志的模式。该方法仅需要极少量的训练数据, 就可以达到较高的精确度, 极大地减少了模型训练的开销。我们相信, 我们的研究结果和发现可以为 ChatGPT 在日志异常检测领域中的应用提供有价值的参考。未来的工作我们将从以下几个方面进行研究:

(1) 增强模型的泛化能力。进一步研究如何通过更先进的算法和技术, 使得模型能够更好地适应日志格式和模式的多样性, 从而提升模型在未见过的日志上的泛化能力。

(2) 实时异常检测的优化。探索如何利用大语言模型进行实时的日志异常检测, 确保在高吞吐量日志数据量的场景下, 也能够实现高效准确的异常检测。

(3) 可解释性和透明度的提升。虽然本文中已经通过提示学习增强了模型对日志数据的理解程度, 但进一步提供模型决策过程的可解释性仍然是值得探索的。继续探索如何生成更加直观、用户友好的解释, 帮助运维人员快速理解异常日志的成因。

参考文献

[1] Alrashdi I, Alqazzaz A, Aloufi E, et al. AD-IoT: Anomaly Detection of IoT Cyberattacks in Smart City Using Machine Learning[C]. *2019 IEEE 9th Annual Computing and Communication Workshop and Conference*, 2019: 305-310.

[2] Cao Q M, Qiao Y R, Lyu Z. Machine Learning to Detect Anomalies in Web Log Analysis[C]. *2017 3rd IEEE International Conference on Computer and Communications*, 2017: 519-523.

[3] Liang Y, Zhang Y Y, Xiong H, et al. Failure Prediction in IBM BlueGene/L Event Logs[C]. *Seventh IEEE International Conference on Data Mining*, 2007: 583-588.

[4] Xu W, Huang L, Fox A, et al. Detecting Large-Scale System Problems by Mining Console Logs[C]. *The ACM SIGOPS 22nd symposium on Operating systems principles*, 2009: 117-132.

[5] Vaarandi R, Pihelgas M. LogCluster - a Data Clustering and Pattern Mining Algorithm for Event Logs[C]. *2015 11th International Conference on Network and Service Management*, 2015: 1-7.

[6] Liu F C, Wen Y, Zhang D X, et al. Log2vec: A Heterogeneous Graph Embedding Based Approach for Detecting Cyber Threats within Enterprise[C]. *The 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019: 1777-1794.

[7] Du M, Li F F, Zheng G N, et al. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning[C]. *The 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017: 1285-1298.

[8] Guo H X, Yuan S H, Wu X T. LogBERT: Log Anomaly Detection *via* BERT[C]. *2021 International Joint Conference on Neural Networks*, 2021: 1-8.

[9] Achiam J, Adler S, Agarwal S, et al. Gpt-4 technical report[EB/OL]. 2023: ArXiv Preprint ArXiv:2303.08774.

[10] Jiao W X, Wang W X, Huang J T, et al. Is ChatGPT a Good Translator? yes with GPT-4 as the Engine[EB/OL]. 2023: 2301.08745.https://arxiv.org/abs/2301.08745v4.

[11] Frieder S, Pinchetti L, Griffiths R R, et al. Mathematical capabilities of chatgpt[J]. *Advances in Neural Information Processing Systems*, 2024, 36.

[12] Qi J X, Huang S H, Luan Z Z, et al. LogGPT: Exploring ChatGPT for Log-Based Anomaly Detection[EB/OL]. 2023: 2309.01189. https://arxiv.org/abs/2309.01189v1.

[13] Fu Q, Lou J G, Wang Y, et al. Execution Anomaly Detection in Distributed Systems through Unstructured Log Analysis[C]. *2009 Ninth IEEE International Conference on Data Mining*, 2009: 149-158.

[14] He P J, Zhu J M, Zheng Z B, et al. Drain: An Online Log Parsing Approach with Fixed Depth Tree[C]. *2017 IEEE International Conference on Web Services*, 2017: 33-40.

[15] Nagappan M, Vouk M A. Abstracting Log Lines to Log Event Types for Mining Software System Logs[C]. *2010 7th IEEE Working Conference on Mining Software Repositories*, 2010: 114-117.

[16] Han S B, Wu Q H, Zhang H, et al. Log-Based Anomaly Detection with Robust Feature Extraction and Online Learning[J]. *IEEE Transactions on Information Forensics and Security*, 2021, 16: 2300-2311.

[17] Ying S, Wang B M, Wang L, et al. An Improved KNN-Based Efficient Log Anomaly Detection Method with Automatically Labeled Samples[J]. *ACM Transactions on Knowledge Discovery from Data*, 2021, 15(3): 1-22.

[18] Meng W, Liu Y, Zhu Y, et al. Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs[C]. *IJCAI*. 2019, 19(7): 4739-4745.

[19] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space[EB/OL]. 2013: 1301.3781.https://arxiv.org/abs/1301.3781v3.

[20] Zhang X, Xu Y, Lin Q W, et al. Robust Log-Based Anomaly Detection on Unstable Log Data[C]. *The 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019: 807-817.

[21] Wei J, Wang X Z, Schuurmans D, et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models[EB/OL]. 2022: 2201.11903.https://arxiv.org/abs/2201.11903v6.

[22] Oliner A, Stearley J. What Supercomputers Say: A Study of Five System Logs[C]. *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2007: 575-584.

[23] Stearley J, Oliner A J. Bad Words: Finding Faults in Spirit's Syslogs[C]. *2008 Eighth IEEE International Symposium on Cluster Computing and the Grid*, 2008: 765-770.

[24] Wang S H, Sun Y, Xiang Y, et al. ERNIE 3.0 Titan: Exploring Larger-Scale Knowledge Enhanced Pre-Training for Language Understanding and Generation[EB/OL]. 2021: 2107.02137.https://arxiv.org/abs/2112.12731v1.

[25] Colin R, Noam S, Adam R, et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer[J]. *JOURNAL OF MACHINE LEARNING RESEARCH*, 2020, 21: 1-67.

[26] Zhang S, Roller S, Goyal N, et al. OPT: Open Pre-Trained Transformer Language Models[EB/OL]. 2022: 2205.01068.https://arxiv.org/abs/2205.01068v4.

袁紫依 于 2022 年在成都信息工程大学信息安全专业获得学士学位。现在西安电子科技大学电子信息专业攻读硕士学位。主要研究领域为网络安全和异常检测。Email: yuanzy@stu.xidian.edu.cn

张媛媛 于 2009 年在中国人民大学获得硕士学位。现任职中国信息通信研究院, 高级工程师。主要研究领域为网络与数据安全方向。Email: zhangyuanyuan2@caict.ac.cn

张玉清 于 2000 年在西安电子科技大学获得博士学位。现任中国科学院大学教授, 博士生导师, CCF 会员。研究领域为网络与信息系统安全。Email: zhangyq@nipc.org.cn

张昊星 于 2013 年在中国科学院大学获得硕士学位。现任职中国信息通信研究院, 高级工程师。主要研究领域为数据安全、信息安全以及互联网安全监管治理。E-mail: zhanghaoxing@caict.ac.cn

伍高飞 于 2015 年在西安电子科技大学获得博士学位。现任西安电子科技大学副教授, 硕士生导师, CCF 会员。主要研究领域为网络与信息系统安全、AI 安全、密码学方向。Email: wugf@nipc.org.cn