基于Bert和自适应聚类的在线日志解析方法

卢家伟, 卢士达, 刘思思, 吴承荣

引用本文

卢家伟, 卢士达, 刘思思, 吴承荣. 基于Bert和自适应聚类的在线日志解析方法[J]. 计算机科学, 2024, 51(11): 65-72.

LU Jiawei, LU Shida, LIU Sisi, WU Chengrong. Online Log Parsing Method Based on Bert and Adaptive Clustering [J]. Computer Science, 2024, 51(11): 65-72.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230900161

基于 Bert 和自适应聚类的在线日志解析方法

卢家伟[1]　卢士达[2]　刘思思[2]　吴承荣[1]
1 复旦大学计算机科学技术学院　上海 200082
2 复旦大学网络信息安全审计与监控教育部工程研究中心　上海 200082
(jwlu22@m.fudan.edu.cn)

摘　要　日志解析是一种从原始日志文件中提取有效信息的技术,它可以用于系统故障诊断、性能分析、安全审计等领域。日志解析的主要挑战在于日志数据的非结构化、多样性和动态性。不同的系统和应用程序可能使用不同的日志格式,随着时间的推移,日志格式也会发生变化。文中提出一种能够自适应不同日志源和日志格式变化的在线日志解析方法 BertLP,它使用预训练语言模型 Bert,并结合自适应聚类算法对日志中的单词进行静动态识别,从而对日志进行分组生成日志模板。BertLP 方法不需要人工定义日志模板或正则表达式,也不需要对单词进行频率统计,而是通过学习日志消息的语义和结构特征,来自动识别日志字段和类型。在多个公开日志数据集上的对比实验显示,BertLP 方法在日志解析的准确率上比现有最佳方法提高了 6.1%,并且在日志解析任务上表现更好。

关键词:日志解析;Bert;自适应聚类;语义提取
中图分类号　TP181

Online Log Parsing Method Based on Bert and Adaptive Clustering

LU Jiawei[1],LU Shida[2],LIU Sisi[2] and WU Chengrong[1]
1 School of Computer Science and Technology,Fudan University,Shanghai 200082,China
2 Engineering Research Centre of Network Information Security Audit and Monitoring of Ministry of Education,Fudan University,Shanghai 200082,China

Abstract　Log parsing is a technique for extracting valid information from raw log files,which can be used in areas such as system troubleshooting,performance analysis and security auditing. The main challenge of log parsing is the unstructured,diversity and dynamics of log data. Different systems and applications may use different log formats,and log formats may change over time. Therefore,this paper proposes BertLP,an online log parsing method that can automatically adapt to different log sources and log format variations. It uses a pre-trained language model,Bert,combined with an adaptive clustering algorithm for static and dynamic recognition of words in logs to group logs to generate log templates. Instead of manually defining log templates or regular expressions and performing frequency counts on words,BertLP automatically identifies log fields and types by learning semantic and structural features of log message. Comparative experiments on public log datasets show BertLP improves log parsing accuracy by 6.1% compared with the best available method and performs better on log parsing tasks.

Keywords　Log parsing,Bert,Adaptive clustering,Semantic extraction

1　引言

日志是记录系统或者应用程序运行状态和行为的文本文件,它可以反映系统或者应用程序的内部结构和逻辑[1-2]。日志在云服务、web 服务以及系统分析中有着重要作用,它可以帮助开发者和运维人员监控系统或者应用程序的性能、安全、可靠性等,以及发现和定位故障、异常、错误等问题[3]。因此,日志解析是提高系统或者应用程序质量和效率的必要手段。

现有的日志解析方法主要是通过对日志进行分组,或者采用正则表达式、字符串匹配、频率统计分析等方法,来从日志中提取有用的信息,从而把非结构化的日志数据转变为结构化数据,进而生成日志模板[4]。日志模板是一种抽象化的日志表示,它可以将日志中的静态内容(如关键字、操作符等)和动态内容(如资源标识、状态枚举值、参数等)区分开来,并用通配符来表示动态内容。日志模板可以帮助理解日志的含义,以及进行日志分类、异常检测、故障预测等任务[5-6]。

到稿日期:2023-09-28　返修日期:2024-03-13
基金项目:复旦大学网络信息安全审计与监控教育部工程研究中心与国家电网上海数据中心合作项目(09B307-9003001-0014-1)
This work was supported by the Engineering Research Centre of Network Information Security Audit and Monitoring of Ministry of Education,Fudan University and State Grid Shanghai Data Centre's Cooperative Project(09B307-9003001-0014-1).
通信作者:吴承荣(cwu@fudan.edu.cn)

然而，现有的日志解析方法也存在一些局限性，比如：

1）日志中的单词如果只基于频率统计，可能会损失日志的上下文和结构信息。例如，同一单词在不同的位置可能有不同的含义和作用，而频率统计无法区分这些差异。

2）日志中的多个动态内容之间会存在依赖关系。例如，一个动态内容可能是另一个动态内容的子集或者超集，或者两个动态内容可能存在相同或者相反的含义，而现有的方法大都缺乏对动态内容的深入分析，无法捕捉这些关系。

3）随着日志的更新也会出现一些新词。例如，系统或者应用程序可能增加了一些新功能或者修复了一些错误，导致日志中出现了一些之前没有出现过的单词，而现有的方法无法适应这些变化。

这些局限性会导致生成的日志模板不准确，从而影响后续的日志分析任务。因此，本文提出了一种基于 Bert 和自适应 K-means 聚类的在线日志解析方法 BertLP(Bert Log Parsing)，它可以解决上述问题，并且提高日志模板的质量和效果。本文的主要贡献如下：

1）提出了一种基于 Bert 的日志模板生成模型，它可以利用日志的上下文和结构信息来学习单词的语义表示，并根据单词之间的依赖关系来生成更准确和更具有语义表达力的单词向量。

2）提出了一种基于自适应 K-means 聚类的静动态簇分类方法，它可以对单词向量根据语义相似或者语义距离进行自适应聚类，形成静态簇和动态簇；通过计算单词向量和各个簇中心向量的距离来判断单词的静动态类别，解决动态内容之间的依赖关系和新词问题，使模型适应日志的变化。

3）在多个公开的数据集上进行了实验，结果表明本文提出的方法在日志解析方面均显著优于现有的方法。

2 相关工作

2.1 日志解析

日志解析是将日志从半结构化数据转换为结构化数据的过程，以便后续的分析和挖掘。在以往的有关日志解析的研究中，主流的方法有两种：一种是通过频繁项挖掘等方法，根据单词出现的频率识别出日志消息中的常量，进而提取出日志模板；另一种则是通过聚类的方法，根据聚类之后的类别数来确定日志模板。

基于频繁项挖掘的方法通常是从日志中提取出最常出现的词组，并根据这些词组来生成日志模板，这样可以快速地处理不同长度的日志。SLCT[6]是最早针对日志解析的频繁项挖掘方法，该方法把日志消息中频繁出现的单词以及与其相关联的高频单词视为常量，以此来生成日志模板。但这种方法对不同长度的词组处理起来较为困难[7]。为了解决这一问题，Vaarandi 等[8]提出了 LogCluster，该方法允许通过聚类算法来处理不同长度的单词组合，与 SLCT 相比在处理日志消息方面更灵活，效果更好。而 Logram[9]不同于以往的频繁项挖掘方法，它使用 n-gram 字典来提取日志中最常出现的词组，并根据这些词组来生成日志模板。

基于聚类的方法通常使用相似度度量来将相似的日志归为一类，并从每一类中提取出日志模板。SHISO[10]是最早在线的日志解析方法。该方法使用一个树形结构来进行日志解析，树中的每个节点都与一个日志组和一个事件模板相关联。SHISO 在遍历树的过程中通过比较每个日志组中的日志消息和事件模板来找到最合适的日志组。但 SHISO 对路径爆炸很敏感，因此其效率往往不理想。LenMa[11]与 SHISO 相似，LenMa 将每条日志消息记录为一个长度向量，向量的每一维为对应要素的字符数。该方法通过比较不同日志消息的长度向量来进行日志解析。而为了加速解析过程，Spell[12]设置了前缀树和反向索引的日志组数据结构，基于最长公共子序列和编辑距离来处理不同长度的日志消息。

除了频繁项挖掘和聚类的方法以外，近几年也涌现了许多其他的用来进行日志解析的方法。He 等[13]提出了一种基于固定深度搜索树的在线日志解析方法 Drain，树中每个节点都事先嵌入一定的启发式规则，根据日志长度和单词的位置将日志消息划分到不同的日志组，从而生成日志模板。该深度解析树也可以随着在线解析的日志数据流不断更新模板。ULP[14]是一种基于正则表达式的解析方法，该方法首先对日志进行分组，再根据字符串匹配和局部频率分析识别出各组日志中的不变量，进而生成日志模板。

目前的日志解析方法大多较为关注如何找出日志消息中的常量部分，进而对分组后的日志生成日志模板。后续的分析也主要针对日志模板中包含的常量部分展开，而较少考虑到除去常量外的参数部分。而且较多的方法考虑到了日志中单词出现的频率，日志除了常量部分的单词具有语义外，参数部分也具有语义，比如用户 id、资源标识、状态枚举值等等。如何更好地结合常量和参数的语义关系进行静动态内容的识别，并充分利用参数中的语义信息，仍然是日志解析中的一大挑战。

2.2 Bert 模型

Bert 模型是由 Google 在 2018 年提出的[15]，模型结构如图 1 所示。它利用了 Transformer 的编码器结构，通过在大规模的无标注文本上进行 MLM(Masked Language Model)和 NSP(Next Sentence Prediction)的预训练任务，学习双向的语义表示，然后 Bert 模型可以在特定的任务上进行微调，实现快速和高效的迁移学习。

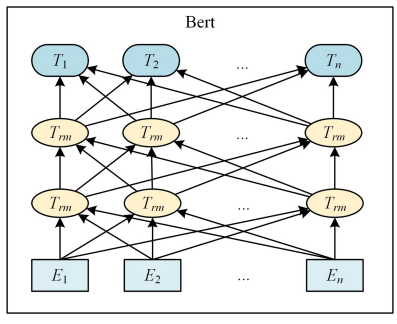

图 1 Bert 模型结构图

Fig. 1 Bert model structure diagram

近年来，国内外涌现出许多用于进行文本处理的语言模型[16]，如 GPT-3，Jurassic，Cerebras-GPT 等。一部分语言模型由于参数量较大，会占用过多的计算资源，造成过度的内存和时间消耗，例如 GPT-3 的 1 750 亿个参数、Jurassic 的1 780亿

个参数等；而另一部分语言模型由于其结构特点，并不能很好地对文本进行处理。例如 GPT-3 是单向结构，只能考虑文本的左侧或者右侧的上下文；而 Cerebras-GPT 使用了更多更复杂的预训练任务，会增加语言模型的复杂度和训练时间。

相较于这些语言模型，Bert 模型的大小适中，参数量只有 3.5 亿，且预训练任务较少，既不会占用过多的计算资源，也不会损失过多的表达能力。Bert 模型也是第一个基于双向的自注意力机制的语言表示模型，能够充分利用上下文信息捕捉词语之间的复杂关系，同时可以实现并行化处理，提高运行效率。Bert 在多个自然语言处理任务上取得了最佳性能，包括阅读理解、自然语言推理、命名实体识别、情感分析等；同时，其在日志分析领域也被广泛应用。LogBERT[17] 和 LAnoBERT[18] 是两种基于 Bert 的日志异常检测框架，前者是基于预训练和微调进行日志异常检测，而后者是基于预训练的 MLM 任务进行日志异常检测。

3 BertLP 模型

3.1 模型概述

由于原始的日志具有格式多样化和非结构化的特点，如果在对原始日志进行解析的过程中不能很好地利用其中的语义关系进行模板提取，在之后的分析过程中就会丢失日志中的很多关键信息，也会对后续的异常检测、故障预测等造成一定的影响。因此，确保日志解析的模板提取准确性和语义挖掘深度对日志分析来说至关重要。

本文提出了一种基于 Bert 和自适应聚类的在线日志解析方法 BertLP，如图 2 所示，主要研究如何通过预训练和自适应聚类区分日志中的静态和动态内容，进而对日志进行分组，生成模板，实现对大规模、高维、动态变化的日志的有效解析。

该方法主要包括 3 个模块：离线预训练模块、自适应聚类模块和在线解析模块。离线预训练模块利用 Bert 模型对大量无标签的日志数据进行自监督学习，提取日志数据的语义特征；自适应聚类模块首先使用预训练得到的 Bert 模型对日志中每个单词生成相应的单词向量，然后通过自适应 *K*-means 聚类算法对这些单词向量进行聚类，形成静态内容簇和动态内容簇；在线解析模块对日志数据流中的每条日志中的每个单词进行动态内容和静态内容的识别，将每条日志中的动态内容用通配符代替，进而转换为一个模板和一组参数，最后使用哈希表统计每组的日志数量和参数分布，从而实现对不同类型、不同规模、不同分布的日志数据的灵活分组。

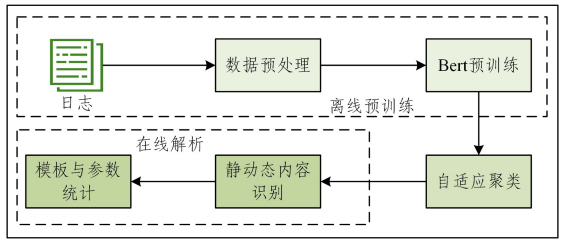

图 2 BertLP 模型框架

Fig. 2 BertLP model framework

3.2 离线预训练模块

该模块的目的是利用大量无标签的日志数据，通过 Bert 模型进行自监督学习，从而提取日志数据的语义特征。首先需要将原始日志拆分为训练集和测试集，本文将原始日志数据集的 80% 作为训练集，用于对 Bert 模型进行预训练；20% 作为测试集，用于验证模型的效果。

一个符合 syslog 规范的样板日志数据如表 1 所列，本文将原始日志中以空格分割的每一部分称为一个单词或者一个 Token。其中，Seq id 为每条日志事件所属的流程序列号，Event id 为日志事件编号，Timestamp 为时间戳，Level 为日志级别，Event 为日志事件内容。表 1 中每一行为一条日志事件，每一段日志流包含多条日志事件。本文将这些日志通过 Embedding 编码为向量形式进行后续的模型训练，编码形式由 Token Embeddings，Segment Embeddings 和 Position Embeddings 组成，编码结构如图 3 所示。

表 1 日志片段

Table 1 Log snippets

Seq id	Event id	Timestamp	Level	Event
0	1117838976	2021-06-03-12:35:34	INFO	User login from 192.168.1.8
0	1117838977	2021-06-03-12:36:12	INFO	Connection successful with server 192.168.2.1
0	1117838978	2021-06-03-12:36:51	ERROR	Connection failed with server 192.168.2.2 due to timeout
0	1117838979	2021-06-03-12:37:29	INFO	User login from 192.168.1.9
0	1117838980	2021-06-03-12:38:07	INFO	Connection successful with server 192.168.2.3
1	1117838981	2021-06-03-12:38:46	ERROR	Connection failed with server 192.168.2.4 due toserver error
1	1117838982	2021-06-03-12:39:24	INFO	User login from 192.168.1.10
1	1117838983	2021-06-03-12:40:03	INFO	Connection successful with server 192.168.2.5
1	1117838984	2021-06-03-12:40:41	ERROR	Connection failed with server 192.168.2.6 due to server error
1	1117838985	2021-06-03-12:41:20	INFO	User login from 192.168.1.11

Token Embeddings：Token 嵌入层的作用是将输入的原始日志文本转换为固定维度的向量表示形式，借助两个特殊的 Token（[CLS]和[SEP]）将输入切分为一个一个的文本段。对于事件内容，本文也在预处理中将其拆分为一个一个的单词。例如选取表 1 中 Seq id 为 0 的部分，其输入的 Token Embeddings 为“‘[CLS]’‘0’‘1117838976’‘2021-06-03-12:35:34’‘INFO’‘User’‘login’‘from’‘192.168.1.8’‘[SEP]’‘0’‘1117838977’‘2021-06-03-12:36:12’…‘[SEP]’”形式的一个 256 维向量。

Segment Embeddings：Segment 嵌入层的作用是区分输入的句子是否是语义相似的。例如选取表 1 中 Seq id 为 0 的 5 条日志，把 0 赋值给第一条日志的各个 Token，把 1 赋值给第二条日志的各个 Token，依此类推，其输入的 Segment Embeddings 为“000000000111…”形式的一个 256 维向量。

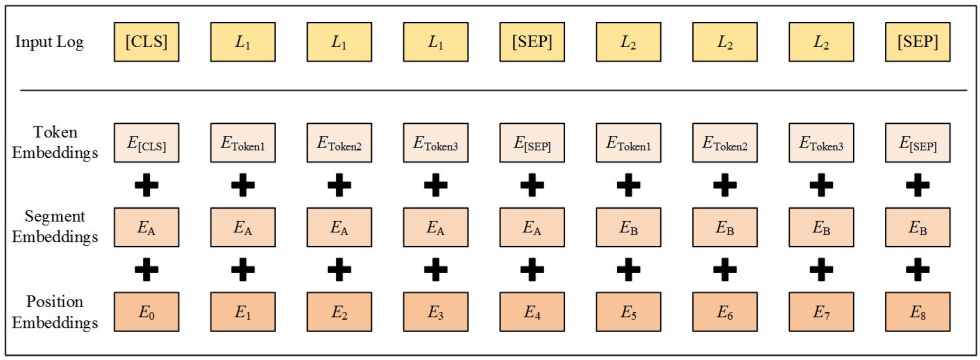

图 3 Embedding 结构

Fig. 3 Embedding structure

Position Embeddings：Position 嵌入层的作用是表示每个 Token 在序列中的位置信息。例如表 1 中 Seq id 为 0 的日志序列中有 4 条日志的 Level 是一样的，但是它们应该用不同的向量表示，在这里分别为位置 0、位置 1、位置 3 和位置 4。

3 个嵌入层最终都是(1，n，256)的向量表示形式，最后将 3 个嵌入层的向量按元素相加，即为 Bert 编码层的输入。离线预训练模块的整体架构图如图 4 所示。

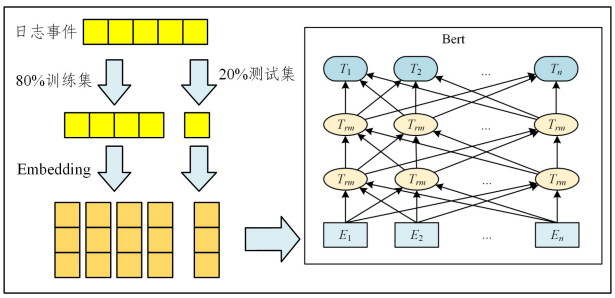

图 4 离线预训练模块结构

Fig. 4 Structure of offline pre-training module

Bert 模型使用了两种预训练任务：掩码语言模型(Masked Language Model，MLM)和下一句预测(Next Sentence Prediction，NSP)。MLM 任务是在输入序列中随机遮挡一些单词，然后让模型预测被遮挡的单词；NSP 任务是给定两个句子，让模型判断它们是否是连续的。

BertLP 的离线预训练模型针对识别单词静动态类别制定了预训练任务 MWM(Masked Word Model)，通过输入的 3 个嵌入层特征向量的总和，来学习日志中每个单词的上下文语义信息，基于双向 Transformer 训练一个深度神经网络模型，同时 MWM 可以捕获到日志中单词与单词之间的长期依赖关系。

针对 MWM 预训练任务，该模型使用了 Bert 模型的预训练任务 MLM 中传统的 80%-10%-10%的掩码策略，将 80%的单词直接替换为[Mask]，10%的单词替换为任意的单词，剩下的 10%的单词保持不变。表 2 列出了表 1 中 Seq id 为 0 的日志事件可能的掩码情况，其中黑色加粗部分为预训练任务 MWM 中被随机掩码的部分。

表 2 Mask 单词的掩码样例

Table 2 Sample cases of Mask word

Seq id	Event id	Timestamp	Level	Event
0	1117838976	2021-06-03-12:35:34	INFO	User login from 192.168.1.8
0	1117838977	2021-06-03-12:36:12	**ERROR**	Connection successful with server 192.168.2.1
0	**1117838978**	2021-06-03-12:36:51	ERROR	Connection failed with server 192.168.2.2 due to timeout
0	1117838979	**2021-06-03-12:37:29**	INFO	User login from 192.168.1.9
0	1117838980	2021-06-03-12:38:07	[**Mask**]	Connection successful with[**Mask**] 192.168.2.3

从表 2 可以看出，第一个事件没有被 Mask，而第二个事件的 Level 被替换为了第三个事件的 Level，第三个事件的 Event id 和第四个事件的 Timestamp 保持不变，而第五个事件的 Level 和 Event 中一个单词均被替换为了[Mask]。

离线预训练模块可以充分提取原始日志事件中各个单词之间的语义特征，便于后续进行聚类分析与在线解析。

3.3 自适应聚类模块

自适应聚类模块首先使用预训练得到的 Bert 模型对每条日志中每个单词生成相应的单词向量，然后通过自适应 K-means 聚类算法对这些单词向量进行聚类，形成静态内容簇和动态内容簇，并计算每个簇的聚类中心。自适应聚类模块的整体架构如图 5 所示。

首先，该模块用预训练得到的 Bert 模型对日志中的每个单词生成特征向量，通过自适应的K-means聚类算法对这些特征向量进行聚类。K-means 算法是一种无监督学习算法，它的目标是将数据点分为 K 个簇，使得每个簇内的数据点之间的相似度最大，而不同簇之间的相似度最小。

在日志中，作为静态内容的单词总是成组出现，作为动态内容的单词总是单一出现，如表 1 中 Seq id 为 0 的日志事件所示，对于第二个日志事件和第三个日志事件，“Connection”和“server”总是成组出现，而“successful”“failed”总是单一出现。因此本文认为，静态内容的单词会具有相似或相关的语义信息和上下文信息，它们的输出向量也比较接近，通过聚类可以较好地区分这两类单词向量，后续实验也证实了这一点。

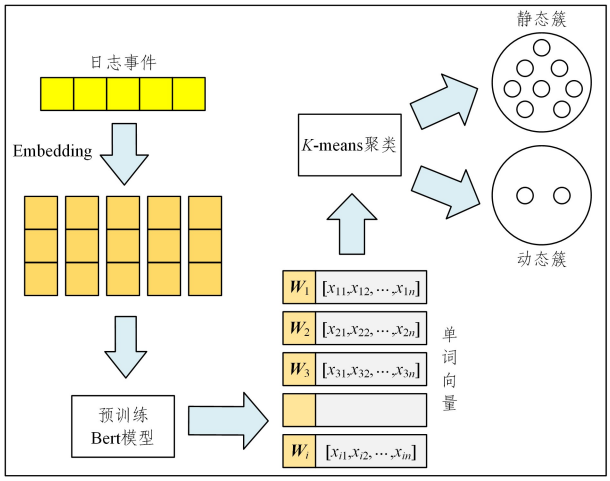

图 5 自适应聚类模块结构

Fig. 5 Structure of adaptive clustering module

本文选取的自适应策略是 K 值从 2 逐一递增，自适应的最佳 K 值通过 SSE(误差平方和)来选取。SSE 是一种评价聚类效果的指标，表示每个数据点与其所属簇中心之间距离的平方和，SSE 越小，表示聚类效果越好。SSE 的定义如下：

$$SSE=\sum_{i=1}^{n}\sum_{j=1}^{m}\| x^{(i)}-y^{(j)} \|_2^2 \tag{1}$$

当 SSE 下降幅度小于某个阈值时，即达到了最佳的聚类效果，停止增加 K 值，基于此时的最佳 K 值生成含有不同单词数量的单词簇。

对于生成的若干单词簇，计算每个簇内单词与其所属簇的中心距离的平均值作为簇的距离特征。簇内单词与其所属簇的中心距离较小的簇是静态簇，而簇内单词与其所属簇的中心距离较大的簇是动态簇。因此，如果一个簇的距离特征低于平均值的一定比例(如 0.5)，则将其划分为静态簇；如果一个簇的距离特征高于平均值的一定比例(如 1.5)，则将其划分为动态簇。之后根据静态簇和动态簇来生成日志模板。

自适应聚类模块最后会得到静态簇和动态簇，在之后的在线解析中会根据日志中单词与静动态簇中心向量之间的距离远近，来判断该单词是日志中的静态内容还是动态内容，从而进行分组，生成日志事件模板。

3.4 在线解析模块

在线解析模块对日志事件流中每条日志中的每个单词进行静动态内容识别，将其中的动态内容用通配符代替，生成一个临时模板和一组参数，并将其以“模板-参数”键值对的形式存入哈希表中。若之后在解析时遇到相同的临时模板，则更新相应键值对中的参数部分，最后使用哈希表统计每组的日志数量和参数分布，从而实现对不同类型、不同规模、不同分布的日志数据的灵活分组。在线解析模块的整体架构如图 6 所示。

首先，对于每条新到来的日志，使用预训练 Bert 模型对其进行编码，并得到日志中每个单词对应的向量表示。对于每个单词向量，计算其与 K 个簇中心向量之间的距离，并将其归属到距离最近的簇中，根据该簇的标记，将该单词视为静态内容或动态内容。

然后，将同一条日志中的所有静态内容连接起来，作为该日志的模板。将动态内容用特殊的占位符〈*〉替换，作为该日志的参数部分。比如，一条日志“1117838976 2021-06-03-12:35:34 INFO User login from 192.168.1.8”可以生成一个模板〈*〉〈*〉 INFO User login from 〈*〉和一组参数[‘1117838976’,‘2021-06-03-12:35:34’,‘192.168.1.8’]。

为了更好地对日志进行分组，本文使用哈希表来存储不同事件模板及其对应的日志列表。哈希表的键是事件模板，值是一个数组，数组中每一个元素是一个日志的参数部分。比如，表 1 中 Seq id 为 0 的第一条日志和第四条日志就具有共同的事件模板，在哈希表中的键值对可以是〈*〉〈*〉 INFO User login from 〈*〉:[{1117838976,2021-06-03-12:35:34,192.168.1.8},{1117838979,2021-06-03-12:37:29,192.168.1.9},…]。

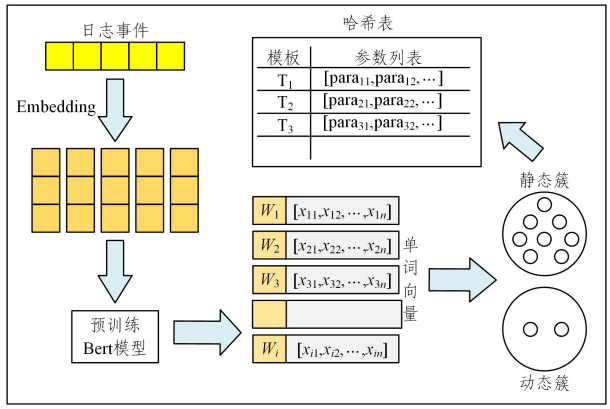

图 6 在线解析模块结构

Fig. 6 Structure of online parsing module

对于每条新到来的日志，先用上述格式表示出来，然后根据其模板在哈希表中查找是否已经存在相同的模板。如果存在，则将该日志的参数部分添加到对应的数组中；如果不存在，则在哈希表中创建一个新的键值对，并将该日志的参数部分作为数组的第一个元素。这样可以将具有相同事件模板的日志归为一组，表 3 所列即为表 1 中的所有日志最后形成的哈希表示例。

表 3 哈希表示例

Table 3 Example of hash table

模板	参数
〈*〉〈*〉 INFO User login from 〈*〉	[{‘1117838976’,‘2021-06-03-12:35:34’,‘192.168.1.8’},{‘1117838979’,‘2021-06-03-12:37:29’,‘192.168.1.9’},{‘1117838982’,‘2021-06-03-12:39:24’,‘192.168.1.10’},{‘1117838985’,‘2021-06-03-12:41:20’,‘192.168.1.11’}]
〈*〉〈*〉 INFO Connection successful with server 〈*〉	[{‘1117838977’,‘2021-06-03-12:36:12’,‘192.168.2.1’},{‘1117838980’,‘2021-06-03-12:38:07’,‘192.168.2.3’},{‘1117838983’,‘2021-06-03-12:40:03’,‘192.168.2.5’}]
〈*〉〈*〉 ERROR Connection failed with server 〈*〉 due to 〈*〉	[{‘1117838978’,‘2021-06-03-12:36:51’,‘192.168.2.2’,‘timeout’},{‘1117838981’,‘2021-06-03-12:38:46’,‘192.168.2.4’,‘server error’},{‘1117838984’,‘2021-06-03-12:40:41’,‘192.168.2.6’,‘server error’}]

最后，遍历哈希表中的所有键值对，将每个键值对视为一组具有相同模板的日志，并统计每个组中的日志数量和参数分布，这样就完成了日志分组和模板生成的过程。

4 实验

4.1 实验数据

为了验证本文提出的方法能够高效准确地进行日志解析，在本节中进行实验验证。本节实验使用 LogPai 基准[19-20]的 10 个日志数据集来评估 BertLP 方法在进行日志解析时的准确性。这些数据集由一组日志文件组成，由各种系统生成，包括 Apache，BGL，Hadoop 等，如表 4 所列，它们在日志解析效果评估中被广泛使用。

表 4 LogPai 日志数据集

Table 4 LogPai log dataset

日志数据集	描述	大小
Apache	Apache 服务器访问和错误日志	4.90 MB
BGL	BlueGene/L 超级计算机日志	708.76 MB
Hadoop	Hadoop 分布式计算工作日志	48.61 MB
HDFS	Hadoop 分布式文件系统日志	1.47 GB
HPC	高性能集群日志	32.00 MB
Openstack	Openstack 软件日志	58.61 MB
Proxifier	Proxifier 软件日志	2.42 MB
Spark	Spark 工作日志	2.75 GB
ThunderBird	ThunderBird 超级计算机日志	29.60 GB
Zookeeper	Zookeeper 服务日志	9.95 MB

日志语法和内容存在明显差异的日志属于不同系统的日志。例如由两个团队分别开发的软件所生成的日志，且没有就日志的生成规范进行约定。如果在某个开发项目中，开发组对生成的日志格式和风格、术语有所约定，那么尽管是不同开发人员所写的程序生成的不同日志文件，也属于相同系统的日志。在实验中，针对不同系统的日志，将会分别独立地进行离线预训练，以适应不同系统的日志特点。

本文还比较了 BertLP 与 4 种现有的日志解析工具，分别是 Spell[12]，Drain[13]，Logram[9] 和 ULP[14]。本文选择这些工具是因为在先前的研究[14,21-22]中表明，这些工具与其他的日志解析工具相比有最高的准确性和效率，所以本文也使用其公开访问的源代码的最新版，在上述的日志数据集上进行相同的实验，与本文提出的 BertLP 方法进行对比实验。

4.2 实验结果及分析

对于日志是否被成功解析主要从以下几个方面来分析：1）日志中所有静态内容都被识别出来，并与 LogPai 团队提供的基准结果一致；2）日志中所有动态内容都被通配符〈*〉代替，并与 LogPai 团队提供的基准结果一致；3）每条日志对应的模板中，静动态内容的位置与 LogPai 团队提供的基准结果一致；4）每条日志对应的模板中，没用多余的静动态内容。表 5 所列为基于上述规则的日志匹配与日志不匹配的示例。本文计算日志成功匹配的数量与总日志数目的比值，并将其作为日志解析工具进行日志解析任务的准确率。

表 5 日志匹配示例

Table 5 Example of log matching

基准结果	实际解析结果	匹配与否	说明
〈*〉〈*〉 INFO Connection successful with server 〈*〉	〈*〉 INFO Connection successful with server 〈*〉	1	静动态内容均被识别出来，位置一致，匹配成功
〈*〉〈*〉 ERROR Connection failed with server 〈*〉 due to 〈*〉	〈*〉 ERROR Connection failed with server 〈*〉 due to timeout	0	有一个动态内容未被识别出来，匹配不成功

图 7 和表 6 给出了本文所提出的 BertLP 方法与其他日志解析方法的准确率结果。表 6 中加粗部分为某一日志数据集下最佳的日志解析方法，与本文所评估的所有其他方法相比，BertLP 在解析绝大部分日志数据集方面具有最高的准确率。

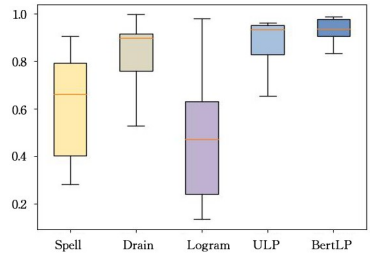

图 7 BertLP 与其他方法的准确率比较（盒图）

Fig. 7 Accuracy comparison between BertLP and other methods (box plot)

表 6 BertLP 与其他解析方法的准确率比较

Table 6 Accuracy comparison between BertLP and other parsing methods

Name	Spell	Drain	Logram	ULP	BertLP
Apache	0.283	0.697	0.267	0.687	**0.834**
BGL	0.717	0.825	0.497	0.912	**0.931**
Hadoop	0.395	0.545	0.215	0.653	**0.881**
HDFS	0.817	**0.997**	0.981	0.961	0.983
HPC	0.661	0.914	0.887	**0.945**	0.937
Openstack	0.392	0.528	0.273	0.782	**0.870**
Proxifier	0.413	0.913	0.518	0.933	**0.959**
Spark	0.872	0.920	0.209	0.954	**0.987**
ThunderBird	0.771	0.898	0.135	0.957	**0.975**
Zookeeper	0.906	0.962	0.742	0.949	**0.979**
Average	0.623	0.820	0.472	0.873	**0.934**

此外，由表 6 可以看出，本文提出的 BertLP 方法的平均准确率为 93.4%，其次是 ULP，平均准确率为 87.3%。为了衡量这些不同日志解析方法的效果差异性，本文采用非参数效应量 Cliff's delta 来进行度量。Cliff's delta 是一种量化两组数据差异程度的统计指标[23]，也可以作为相应假设检验的有效分析，定义如下：

$$Cliff'delta=\frac{1}{num_x \cdot num_y}\sum_i\sum_j sign(x_i-y_j) \tag{2}$$

当该效应量取值位于[0.147，0.33)时，说明两组数据的差异较小；当该效应量取值位于[0.33，0.474)时，说明两组数据的差异程度为中等水平；当该效应量取值位于[0.474，1)时，说明两组数据的差异较大。表 7 所列为本文提出的

BertLP 方法与其他日志解析方法之间的 Cliff'delta 值，可以看出，BertLP 与 Spell 和 Logram 的差异程度较大，而与 Drain 和 ULP 的差异程度为中等水平。

表 7 BertLP 与其他日志解析方法的 Cliff'delta 值

Table 7 Cliff'delta values of BertLP and other log parsing methods

Method	Cliff'delta
Spell	0.90
Drain	0.44
Logram	0.78
ULP	0.34

相较于现有的日志解析方法而言，比如 Spell，Drain，Logram 等，大多数关注于针对单条日志的长度或者遵循字符串匹配的规则来对日志进行分组，进而完成日志的解析，但忽视了日志中静态内容和动态内容的变化规律。相较于这些日志解析方法，ULP 采用了基于频率统计的方法来区分静动态内容，但也有一定的局限性。假设日志中的一些资源标识或者状态枚举值总是反复出现，就会增大这类变量出现的概率，从而被误判为静态内容。图 8 给出了本文所选取的 Apache 日志数据集中的日志片段节选。该片段中的资源标识（文件名）/etc/httpd/conf/workers2.properties 反复出现，这部分内容的频率在该片段中超过了 0.5，按照 ULP 方法，其会被识别为静态内容，但其实这部分是动态内容。

```
[Sun Dec 04 04:51:09 2005] [notice] jk2_init() Found child 6726 in scoreboard slot 8
[Sun Dec 04 04:51:09 2005] [notice] jk2_init() Found child 6728 in scoreboard slot 6
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
[Sun Dec 04 04:51:14 2005] [notice] workerEnv.init() ok /etc/httpd/conf/workers2.properties
```

图 8 Apache 日志数据集中资源标识举例

Fig. 8 Example of resource identification in Apache log dataset

本文提出的 BertLP 方法采用 Bert 模型来提取日志中单词的语义关系从而改进频率统计，即使某一部分资源标识或者状态枚举值的出现频率增大，但最后均会转换为一个单词向量。通过判断该单词向量到静态簇和动态簇的距离来决定这个单词是静态内容还是动态内容，从而避免由频率增大导致的静动态内容误判问题。

结束语 本文提出了一种基于 Bert 和自适应聚类的在线日志解析方法 BertLP，该方法共分为 3 个模块：离线预训练模块、自适应聚类模块和在线解析模块。离线预训练模块利用 Bert 来提取日志数据的语义特征；自适应聚类模块首先通过预训练得到的 Bert 模型为日志中每个单词生成单词向量，进而通过自适应 K-means 算法形成静态簇和动态簇；在线解析模块对日志数据流中的每条日志进行静动态内容识别，通过哈希表对日志进行分组，并统计模板和参数。与最新的日志解析方法相比，本文提出的 BertLP 方法在绝大部分日志数据集上具有更高的准确率，并且与其他最新的日志解析方法具有显著差异。未来的研究方向主要为：1）考虑到更多不同类型不同结构的日志，寻找最优的解决方案；2）研究基于用户或者基于参数的日志解析方法。

参 考 文 献

[1] YU S，CHEN N，WU Y，et al. Self-supervised log parsing using semantic contribution difference[J]. Journal of Systems and Software，2023，200：111646.

[2] ZHOU R，HAMDAQA M，CAI H，et al. Mobilogleak：A preliminary study on data leakage caused by poor logging practices[C]// 2020 IEEE 27th International Conference on Software Analysis，Evolution and Reengineering（SANER）. IEEE，2020：577-581.

[3] AMAR A，RIGBY P C. Mining historical test logs to predict bugs and localize faults in the test logs[C]// 2019 IEEE/ACM 41st International Conference on Software Engineering（ICSE）. IEEE，2019：140-151.

[4] EL-MASRI D，PETRILLO F，GUÉHÉNEUC Y G，et al. A systematic literature review on automated log abstraction techniques[J]. Information and Software Technology，2020，122：106276.

[5] CHEN R，ZHANG S，LI D，et al. Logtransfer：Cross-system log anomaly detection for software systems with transfer learning[C]// 2020 IEEE 31st International Symposium on Software Reliability Engineering（ISSRE）. IEEE，2020：37-47.

[6] HE S，HE P，CHEN Z，et al. A survey on automated log analysis for reliability engineering[J]. ACM Computing Surveys（CSUR），2021，54（6）：1-37.

[7] VAARANDI R. A data clustering algorithm for mining patterns from event logs[C]// Proceedings of the 3rd IEEE Workshop on IP Operations & Management（IPOM 2003）（IEEE Cat. No. 03EX764）. IEEE，2003：119-126.

[8] 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（CNSM）. IEEE，2015：1-7.

[9] DAI H，LI H，CHEN C S，et al. Logram：Efficient Log Parsing Usingn-Gram Dictionaries[J]. IEEE Transactions on Software Engineering，2020，48（3）：879-892.

[10] MIZUTANI M. Incremental mining of system log format[C]// 2013 IEEE International Conference on Services Computing. IEEE，2013：595-602.

[11] SHIMA K. Length matters：Clustering system log messages using length of words[J]. arXiv：1611.03213，2016.

[12] DU M，LI F. Spell：Online streaming parsing of large unstructured system logs[J]. IEEE Transactions on Knowledge and Data Engineering，2018，31（11）：2213-2227.

[13] HE P，ZHU J，ZHENG Z，et al. Drain：An online log parsing approach with fixed depth tree[C]// 2017 IEEE International Conference on Web Services（ICWS）. IEEE，2017：33-40.

[14] SEDKI I，HAMOU-LHADJ A，AIT-MOHAMED O，et al. An Effective Approach for Parsing Large Log Files[C]// 2022 IEEE International Conference on Software Maintenance and Evolution（ICSME）. IEEE，2022：1-12.

[15] DEVLIN J，CHANG M W，LEE K，et al. Bert：Pre-training of deep bidirectional transformers for language understanding[J]. arXiv：1810.04805，2018.

[16] STROBELT H，HOOVER B，SATYANARAYAN A，et al. LMdiff：A visual diff tool to compare language models[J]. ar-

Xiv:2111.01582,2021.
[17] GUO H,YUAN S,WU X. Logbert:Log anomaly detection via bert[C]// 2021 International Joint Conference on Neural Networks(IJCNN). IEEE,2021:1-8.
[18] LEE Y,KIM J,KANG P. Lanobert:System log anomaly detection based on bert masked language model[J]. Applied Soft Computing,2023,146:110689.
[19] 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(DSN'07). IEEE,2007:575-584.
[20] ZHU J,HE S,HE P,et al. Loghub:A large collection of system log datasets for ai-driven log analytics[C]// 2023 IEEE 34th International Symposium on Software Reliability Engineering(ISSRE). IEEE,2023:355-366.
[21] ZHANG T,QIU H,CASTELLANO G,et al. System Log Parsing:A Survey[J]. IEEE Transactions on Knowledge and Data Engineering,2022,35(8):8596-8614.
[22] LANDAUER M,ONDER S,SKOPIK F,et al. Deep learning for anomaly detection in log data:A survey[J]. Machine Learning with Applications,2023,12:100470.
[23] MACBETH G,RAZUMIEJCZYK E,LEDESMA R D. Cliff's Delta Calculator:A non-parametric effect size program for two groups of observations[J]. Universitas Psychologica,2011,10(2):545-555.

LU Jiawei, born in 2000, postgraduate. His main research interests include machine learning and cyberspace security.

WU Chengrong, born in 1971, Ph.D, associate professor, master's supervisor, 2004 Shanghai Youth IT Top Ten new talent, is a member of CCF (No. 23842M). His main research interest is cyberspace security.

(责任编辑:何杨)

首支 CCF 教学基金评审结果公示 | CCF—睿芯教学基金

CCF—睿芯教学基金作为首个教学领域 CCF 产学合作基金,是由 CCF 与中科睿芯集团携手共同成立,旨在通过基金支持和项目引导,推动计算机实验实践教学的改革和创新,培养更多在 RISC—V 领域具备创新意识和实践能力的优秀计算机人才。本基金受到了学者们的广泛关注与支持,收到了来自北京交通大学、浙江大学等多所高校学者的申请。经综合评审,共 3 个项目获得资助。以下是本年度 CCF—睿芯教学基金入选学者名单(排名不分先后,按照课题发布顺序排序)。

单位	课题类型	课题名称
北京交通大学	揭榜挂帅课题	面向计算机 101 计划的硬件实验体系及一体化实验平台构建与实践
浙江大学	一般开放性课题	计算机组成与设计
山东师范大学	一般开放性课题	基于 RISC-V 架构的计算机组成原理实验教学设计

据 CCF 微信公众号