计算机研究与发展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330054
61(6): 1573−1584, 2024

搜索服务响应时间异常诊断

夏思博[1] 马明华[3] 金鹏翔[1] 崔丽月[1] 张圣林[1,2] 金 娃[1] 孙永谦[1] 裴 丹[3]
[1](南开大学软件学院 天津 300457)
[2](先进计算与关键软件(信创)海河实验室 天津 300450)
[3](清华大学计算机科学与技术系 北京 100084)
(xiasiboth@mail.nankai.edu.cn)

Response Time Anomaly Diagnosis for Search Service

Xia Sibo[1], Ma Minghua[3], Jin Pengxiang[1], Cui Liyue[1], Zhang Shenglin[1,2], Jin Wa[1], Sun Yongqian[1], and Pei Dan[3]
[1](*College of Software*, *Nankai University*, *Tianjin* 300457)
[2](*Haihe Laboratory of Information Technology Application Innovation*, *Tianjin* 300450)
[3](*Department of Computer Science and Technology*, *Tsinghua University*, *Beijing* 100084)

Abstract The timely response of network services is crucial to improving user experience. Taking search engine as a typical example of network services, service providers need to ensure that the search response time is within one second. In practice, the search response time can be affected by many service attributes, such as user browsers, ISPs, and page loading methods. To optimize effectively, service providers need to identify the rules that cause high search response time, which are combinations of the above attributes. However, existing work encounters three challenges. First, the amount of search logs is large. Second, the search logs are unevenly distributed. Third, the rules with high generality are needed. Therefore, we propose a framework called Miner (multi-dimensional extraction of rules). Miner takes advantage of self-paced sampling to overcome the first and second challenges. To address the third challenge, Miner employs Corels to generate rules with high generality and recall. Our experiments use search logs provided from two top-tier search engine companies in China. The results show that Miner outperforms the state-of-the-art methods in terms of generality and recall. Operators adopt rules generated by Miner and optimize the performance of the search engine.

Key words quality of networking service; self-paced sampling; search engine; search response time; data mining

摘 要 较低的网络服务响应时间对提升用户体验至关重要.以搜索引擎这一典型的网络服务场景为例,服务提供商应确保网络服务(搜索)响应时间在1 s以内.在实践中,服务响应时间会受到用户浏览器、运营商、页面加载方式等诸多服务属性的影响.为了进行针对性的优化,服务提供商需要找出使服务响应时间过长的规则,即一些属性的组合.然而现有研究工作遇到了3方面挑战:1)搜索日志数据量大;2)搜索日志数据分布不平衡;3)要求泛化度高的规则.因此设计了Miner(multi-dimensional extraction of rules),一种新型服务响应时间异常诊断框架.Miner使用自步采样机制应对第1个挑战和第2个挑战.针对第3个挑战,Miner使用Corels算法挖掘出泛化率高且召回率高的规则.使用2家国内顶级搜索引擎服务提供商的响应时间日志数据评估了Miner性能,结果显示Miner的泛化率和召回率均高于现有方法,并证明了Miner挖掘出的规则可被运维人员采纳并做针对性的优化.

收稿日期:2023-01-30;**修回日期**:2023-07-25
基金项目:国家自然科学基金青年科学基金项目(61902200, 62072264);天津市自然科学基金项目(21JCQNJC00180)
This work was supported by the National Natural Science Foundation of China for Yong Scientists (61902200, 62072264) and the Natural Science Foundation of Tianjin (21JCQNJC00180).
通信作者:张圣林(zhangsl@nankai.edu.cn)

关键词 网络服务质量；自步采样；搜索引擎；搜索响应时间；数据挖掘

中图法分类号 TP311

随着互联网服务的普及，网络服务质量管理越来越重要. 网络服务如搜索引擎、电子商务和社交网络的响应速度深刻影响用户体验，进而影响服务提供商的收入. 例如，研究表明 Amazon 每增加 0.1 s 的延迟就会使得收入降低 1%①，Bing 搜索响应结果慢 0.5 s 会造成收入降低 1.2%[1]. 因此，网络服务提供商会细致地记录用户每次请求到返回结果的响应时间，它表征了服务质量（quality of service, QoS）[2]. 本文以接入互联网的“门户”——搜索引擎为例，研究导致网络服务响应时间异常的原因.

服务响应时间在搜索引擎公司的记录形式是搜索服务响应日志，日志的内容为搜索响应时间以及与服务环境相关的特征. 这些特征包括用户地理位置、网络运营商 ISP、浏览器类型、搜索词条是否包含广告，以及图片个数等.

本文目标是挖掘响应时间过长的日志对应的规则. 规则是一些特征及其取值的组合，比如“运营商是移动且浏览器是 Chrome”就是一条规则，“不包含广告且采用同步页面加载方式”也是一条规则. 搜索引擎公司的运维工程师获取响应时间过长规则后即可展开针对性的调研和优化. 一般情况下，特征与响应时间的关系是比较复杂的. 在分析了大量服务响应日志后，本文发现仅考虑单一特征无法全面解释搜索响应时间过长的原因，而多维规则（2 维以上特征及其取值组合）才能准确描述响应时间过长的日志集.

在服务响应时间异常诊断方面，已进行了一系列的研究工作. 例如，数据库根因诊断工具 DBSherlock[3] 用经验性算法借助领域知识分析多维度特征的日志，但该方法需要领域知识的辅助，难以快速推广. 视频 QoS 诊断算法 HHH（hierarchical heavy hitter）[4-5] 使用层次化聚类算法诊断异常的服务响应时间，但这一算法需要针对不同数据集进行参数调整，无法灵活应对数据分布变化的情况. 搜索引擎服务响应时间异常诊断算法 FOCUS[6] 采用决策树算法挖掘响应时间日志的规律，然而该方法存在易过拟合和计算效率低下的缺点. 综上，已有相关方法主要面临着难以适应动态变化数据和高计算开销的问题. 搜索服务响应日志的异常诊断问题存在 3 点挑战：

1）搜索服务响应日志数据量大. 本文获取的数据集每天都有几十到上百万的搜索日志（经过 1‰的均匀采样处理后），因此需要效率很高的算法应对海量的搜索日志，快速挖掘出搜索响应时间过长的规则.

2）数据分布不平衡. 本文发现响应时间过长的搜索日志在全部数据中占比 30% 以下，而特定的特征组合对应的数据量是更加不平衡的，如运营商是联通并且搜索页面图片个数小于 10 的用户搜索响应时间过长，只占整体数据的 1% 以下，因此直接对数据进行分类算法处理以找出搜索响应时间最长的特征是困难的. 如应用决策树算法得出的规则组合并不能得到很好的召回率.

3）泛化度要求高. 根据搜索引擎的运维工程师在实际运维过程中提出的需要，算法挖掘出的规则需要能够描述一大类搜索响应时间过长的问题，而不是在很多条件组合下找到的局部问题，这样做针对性优化是比较困难和耗时的.

为解决这 3 个问题，本文提出搜索服务响应时间过长规则提取框架 Miner（multi-dimensional extraction of rules）. 首先使用自步采样方法逐渐聚焦于规则区分难度更大的样本，然后使用 Corels[7] 算法提取理论最佳的规则，最后进行反向筛选. 本文的贡献可总结为 3 个方面：

1）对搜索响应日志进行数据分布研究. 探索了各个特征维度与搜索响应时间的关系，包括图片数量、有无广告、页面加载方式、运营商、用户代理等.

2）提出 Miner 性能诊断框架. 针对数据量大和数据分布不平衡的挑战，Miner 使用自步采样算法抽取分类难度协调的样本. 针对要求规则泛化率高的挑战，使用 Corels 生成规则，并提出迭代式反向筛选提升规则的泛化率. 据了解，本文是将自步采样和规则生成算法结合的工作.

3）使用国内 2 家顶级搜索引擎的真实搜索响应日志数据来评估 Miner 和现有方法. 验证了 Miner 的泛化率和召回率显著高于其他规则生成算法.

1 相关工作

互联网服务运行过程中记录的日志具有重要的

① http://highscalability.com/blog/2009/7/25/latency-is-everywhere-and-it-costs-you-sales-how-to-crush-it.html

价值，日志可被粗略地分为非结构化日志和结构化日志. 一些性能诊断和故障定位方面的相关工作使用结构化多维度日志数据，即对于给定指标的异常，找到 1 个或多个特征和取值的组合，交给运维人员进行后续的调研分析. 本文主要对结构化的多维度日志数据进行研究.

为了找到与性能问题最相关的取值组合，需要考察特征取值和指标之间的关系，通常使用信息论中的相关概念进行衡量. 机器学习模型中，决策树和随机森林是依赖于信息论的方法. FOCUS[6] 和 PerfXplain[8] 主要利用决策树算法做性能诊断. 其中文献 [6] 和本文的场景一致，因为诊断搜索响应时间过长，其用每天的搜索响应日志结合响应时间正常标签和响应时间过长的标签训练一个决策树模型，然后综合归纳多天的决策树，得出规则用于解释响应时间过长. 该方法为提升计算效率，采用每天的数据做训练，并利用启发式的算法整合多天的结果，但是这样的整合过程不具有通用性，比如生成的规则可能是 2 个相反的条件，因此就不能进行整合. 该算法得出的结果将在第 5 节中进一步分析. 文献 [8] 是改进决策树的算法，决策树的基本思想是贪心地选择信息增益最高的分界条件进行拆分，这样不能保证生成规则是最优解. 而算法[7] 可用于 2 分类问题从数据中生成规则，并且其结果是可被证明的最优的规则生成算法. 然而，该算法的复杂度很高，不适用于海量的搜索响应日志. 对于枚举型特征的有效取值过多的情况，DeCaf[9] 训练随机森林拟合指标，并定义节点级别的分数计算，提取分数最高的节点对应的决策路径作为根因描述.

除了特征和指标之间的关系，特征与特征之间也存在可以挖掘的信息. HALO[10] 注意到特征内部可能含有潜在的层级关系，即某个特征可能是另一个特征的下级特征，特征取值确定后另一特征的取值也确定. HALO 算法使用两两间的条件熵构建属性层级图，随后使用随机游走、自顶向下搜索和尾部去除得到故障最集中的属性组合.

还有一些忽略特征的方法直接提取属性值，应用关联规则挖掘技术. 文献 [11] 在预先聚类好的簇间应用对比集挖掘算法 STUCCO. FDA[12] 首先使用 FP-Growth 算法挖掘属性值组合中的频繁项集，然后根据提升度衡量频繁项集与故障的关联程度，最后使用 2 个预设规则筛选频繁项集. 此类方法的缺陷是只能处理枚举型特征的数据，不能有效处理数值型特征的数据.

启发式算法和聚类算法 DBSherlock[3] 用于诊断数据库系统中的性能问题，即数据库管理员标定或通过异常检测得到一段时间的性能问题，该算法可以从多维的数据库指标日志中诊断出性能问题的具体表现. 诊断算法的核心思想是从标定的异常和相邻的正常区间找出指标数据的分界点，比如异常时段 CPU 利用率普遍高于 60%，综合多种指标就可以得出一组诊断规则组合. 这种启发式方法可以迁移到本文的场景的原因是问题的输入输出是类似的，异常时段的标记即搜索响应时间过长的日志. 但是，由于这种方法对指标的分界点过于粗糙，从而导致诊断规则的有效性不够好. HHH[4-5] 是一种层次化的聚类算法. 针对搜索响应时间过长的问题，文献 [5] 针对性地使用 HHH 聚类算法来诊断搜索响应时间的瓶颈，因为该算法可以使用高维度的层次化聚类来分辨出真实的瓶颈和其他部分. 然而，应用该算法需要对数据集进行参数调整，一方面调参是困难的，另一方面在线搜索响应时间的数据特征是变化的，用固定的参数分析动态变化的数据导致性能不稳定，因此 HHH 不适用于在线搜索响应时间的诊断.

受文献 [13] 启发，本文将从多维度搜索服务响应日志中抽取最有益于规则生成的样本. 相比于文献 [6, 8]，本文的规则生成算法具有可证明的最优的性质.

2 经验性研究

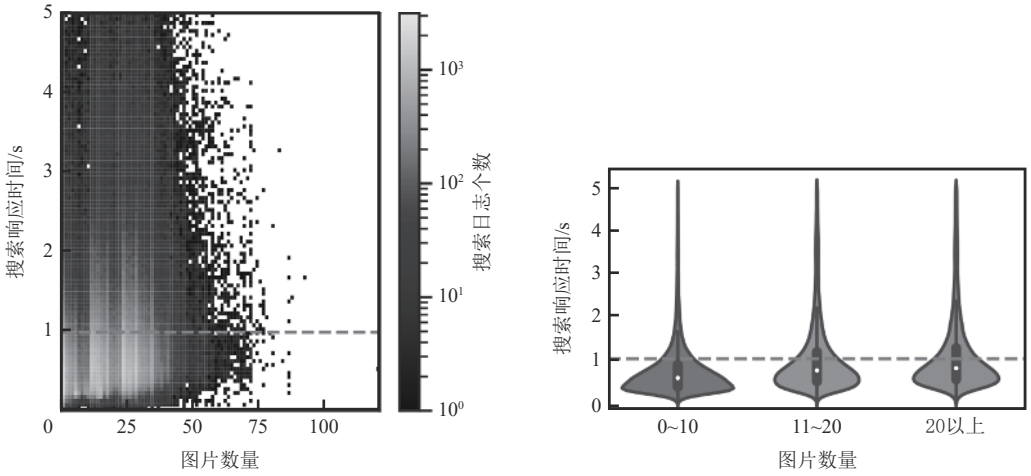
Miner 的输入是搜索响应时间日志，其格式如表 1 所示. 本文获取到中国 2 家顶级的搜索引擎公司的搜索响应时间日志数据（数据集 *A* 和数据集 *B*），具体介绍见第 4 节. 可以观察到，这 2 家不同公司的日志都包括有时间戳（*Timestamp*）、搜索响应时间（*SRT*）和影响搜索响应时间的特征，如图片数量（*#Image*）、浏览器类型（*UA*）、是否有广告（*Ad*）、运营商类型（*ISP*）、用户所在省份（*Province*）和页面加载方式（*PageType*），这些属性字段在不同搜索引擎公司的记录是相似的. 观察一段时间的搜索响应时间，它是一个连续的数值分布，因为搜索引擎公司普遍要求保证用户的搜索响应时间在 1 s 以内，超过 1 s 的搜索响应时间就被认为是过长的，具有潜在的性能问题，本文的主要目的就是诊断搜索响应时间过长的原因.

为了研究不同特征对搜索响应时间的直接影响，本文用数据集 *A* 的数据进行了经验性研究. 首先本文对特征和搜索响应时间的分布做关联分析，如图 1 所示. 图 1(a) 是热度图，每个点对应的数值表示出现

Table 1 Examples of Search Response Time Logs

表 1 搜索响应时间日志示例

Timestamp	*#Image*	*UA*	*Ad*	*ISP*	*Province*	*PageType*	*SRT*/ms
1 411 315 200	0	Chrome	无广告	CRTC	Heilongjiang	sync	2011.14
1 411 315 200	13	Chrome	无广告	CHINANET	Guangdong	async	686.0
1 411 315 200	24	Chrome	无广告	UNICOM	Hunan	async	787.0
1 411 315 200	4	Safari	无广告	OTHER	Guangdong	async	811.0
1 411 315 200	25	Safari	有广告	UNICOM	Jiangsu	async	2203.0



(a) 搜索响应时间随图片数量变化的热度图 (b) 搜索响应时间随图片数量变化的提琴图

Fig. 1 Correlation between image numbers and search response time

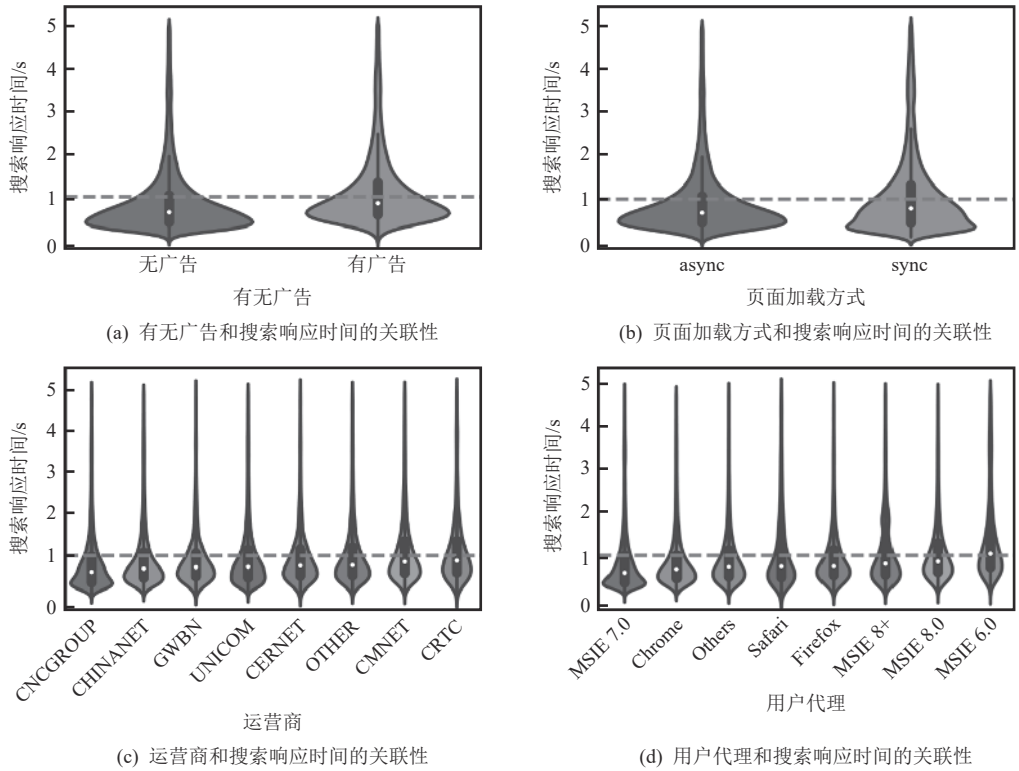
图 1 图片数量和搜索响应时间的关联性

这样的搜索日志个数, 本文在搜索响应时间为 1 s 处用虚线标记了性能问题的分界线. 从图 1(a)观察到图片数量和搜索响应时间的关系是比较复杂的, 并非直观感觉的图片数量多的情况下其搜索响应时间就会长, 因为图片数量越多, 服务器处理、网络传输或浏览器渲染的时间都可能变长, 而本文发现图片多的搜索词条更有可能是热门词条, 它们在服务器中有充足的缓存和针对性的优化, 因而搜索响应时间不会明显变长. 本文的另一个观察是不同的图片数量范围对应的搜索响应时间变化是有明显的分界线的, 如图片数量 0~10, 11~20, 20 以上, 该原因据了解是因为搜索引擎公司单屏幕显示的图片大多是有这样的缓存分区. 于是图 1(b)是把图片数量按照观察到的分界线做的提琴图, 可以看出 3 个分区的搜索响应时间分布是平衡的, 从 3 分位数来看, 图片数量越多, 搜索响应时间趋向于更长, 但是仅图片维度的指标反映不是很明显.

图 2 类似地将枚举型特征(在有限集合中选取的特征)与搜索响应时间的关系制成提琴图. 可以从单维度的特征中得出和 PerfXplain[8] 类似的结论, 单维度的特征并不能很好地解释搜索响应时间过长的问题, 而需要把多维度特征结合起来. 然而从多维度特征中挖掘规则需要对其效果和效率做权衡, 本文希望设计的方法能有更好的效果并能快速得出分析结果.

3 方法介绍

为了解决搜索响应时间过长诊断的问题, 本文提出 Miner 诊断框架. 针对搜索响应日志数据量大的挑战, 需要在所有响应日志中进行采样. 而响应时间过长诊断问题面临数据分布不平衡的挑战, 因此采样时需要关注数据中分布较少但是对响应时长有影响的响应日志.Miner 使用自步采样方法应对这 2 个挑战, 自步采样方法将整体数据划分为响应时间正常、响应时间过长 2 个部分. 自步采样方法在每轮迭代时选取一个特征子集, 并根据选取的特征子集尝试对数据分类, 得到对应的规则难度分布, 然后按比例抽取不同难度的样本, 组成采样结果. 采样完成后,



(a) 有无广告和搜索响应时间的关联性 (b) 页面加载方式和搜索响应时间的关联性

(c) 运营商和搜索响应时间的关联性 (d) 用户代理和搜索响应时间的关联性

Fig. 2 Correlation between categorical attribute dimension and search response time

图 2 分类属性维度和搜索响应时间的关联性

使用 Corels 方法生成描述规则. 描述规则的形式是“如果 [条件], 则 [断言 1], 否则 [断言 2]”, 我们从描述规则中提取断言响应时间过长的条件作为 Miner 框架的结果输出. Miner 框架并非黑盒预测模型, 它的输出是人类可读的描述性规则, 因为本框架目的是为运维工程师提供潜在的优化方向, 但如何具体展开优化并不在本文讨论范围内. Miner 诊断框架可总结为 3 个步骤, 分别是自步采样、规则生成和反向筛选, 如图 3 所示. 特征选择过程在自步采样和规则生成过程中完成, 这样的特征选择在模型训练中进行可有效地提升模型的效果[14]. 下面详细介绍这 3 个步骤.

3.1 自步采样

本节介绍自步采样方法, 以及如何在多维度搜索响应日志中应用自步采样方法. 自步采样是一种应对不平衡数据集的采样方法, 不平衡数据集根据数据的类别标签可划分为多数集和少数集. 本质上, 自步采样是一种欠采样方法, 即从多数集中抽取与少数集等量的数据用于后续的模型训练, 在本文的场景则为规则生成. 相比已有的欠采样方法, 自步采样考虑了用规则准确预测一条数据样本的难度. 根据第 2 节的经验性研究, 仅使用单维度的属性判断准确率较低, 这表明存在部分数据样本具有较大的分类难度. 因此采样时需要考虑样本的分类难度, 在迭代中逐渐给难度更大的样本更大的比例. 这样可以有效地应对搜索响应日志数据集中存在的不平衡、噪声等问题[13].

本文所用的自步采样分为 5 个环节, 分别是数值型特征离散化、最优特征子集选择、单特征判断生成、规则难度计算、分桶采样与合并. 首先对数值型

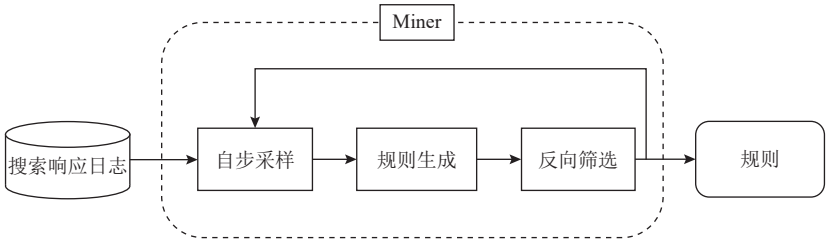


Fig. 3 The diagnostic framework of long search response time

图 3 搜索响应时间过长的诊断框架

特征离散化；然后根据熵最小原则，在所有特征中选取一个对应判断方法，即如何只根据这个特征对搜索响应日志进行分类；之后以此应用特征子集中的每一个特征及其判断方法对数据的多数集进行分类，计算分类误差的总和作为样本的规则难度；最后从多数集按比例抽取难度不同的样本，与少数集合并成为采样结果作为下一步规则生成模型的输入.

3.1.1 数值型特征离散化

搜索响应日志的特征包括枚举型和数值型. 为了统一处理，需要将数值型特征转化为枚举型特征，即数值型特征的离散化.

转化的依据是熵最小原则，对于输入数据的某一个数值型特征，在其值域上寻找使得熵最小的分隔点，按照分隔点将值域分成 2 个区域后，递归一次寻找子区域内熵最小的分隔点，2 个相邻分隔点形成的区间作为一个单位，落在此区间内的数值转化为同一个枚举型取值. 对所有数值型特征都进行离散化，并依据此方法对特征进行处理，不依赖于人工注入的领域知识.

3.1.2 最优特征子集选择

经过 3.1.1 节的处理，此时数据特征均为枚举型特征. 最优特征子集选择的目的是选取分类能力最强的一部分特征，这些特征比其他特征能更好地预测一条搜索日志是否会出现响应时间过长的问题. 本文使用带权熵作为选取标准，对于某一个特征，若其下有 n 个取值，则带权熵计算公式为：

$$e=\sum_{i=1}^{n}w_ie_i=\sum_{i=1}^{n}\left(\frac{\#\,i}{\#\,total}\sum_{j\in\{0,1\}}p_{i,j}\ln\,p_{i,j}\right).$$

Miner 选取带权熵最小的 m 个特征，作为最优特征子集.

3.1.3 单特征判断生成

单特征判断生成环节解决的是如何根据某一特征预测数据样本是否会出现响应时间过长的问题. 每个特征下都有一组取值，为了得到单特征判断，对每个取值计算响应时间过长数据的占比，与原始数据中响应时间过长占比进行比较. 如果某一取值的占比大于原始占比，就预测当前特征为此取值的数据为响应时间过长.

3.1.4 规则难度计算

规则难度是指用规则准确刻画某一条样本的难度，如果使用单特征即可对某一类样本准确预测是否响应时间过长，说明此类样本比较容易预测，即规则难度较低；反之如果需要使用多维度特征组合才能预测是否响应时间过长，说明规则难度较大.

定义规则难度为判断误差的总和，所用判断方法的函数 *judge* 由 3.1.3 节得出，在第 p 轮迭代，多数集中第 j 个样本的规则难度计算为：

$$hard(j)=\sum_{i=1}^{p}\sum_{f=f_{i,1}}^{f_{i,m}}judge_f(j).$$

3.1.5 分桶采样与合并

根据文献 [13]，数据样本可以分为平凡样本、边界样本和噪声样本. 不平衡数据集中大部分数据样本都属于平凡样本，已经使用足够多的平凡样本训练模型后，继续抽取平凡样本能提供的新信息极少；边界样本是对分类器训练和规则生成最有益的样本，应该加大平凡样本和边界样本的权重；而过多使用噪声样本进行训练则会降低模型的泛化率.

自步采样的整体目标是从多数集中抽取合适的样本，与等量的少数集混合后投入后续的模型训练，从而应对数据集的分布不平衡问题. 在每轮迭代中，经过 3.1.4 节得到了多数集的规则难度，本节按比例抽取不同难度的多数集样本.

进行多轮迭代后，Miner 生成的规则已经可以较好地描述响应时间过长的数据样本，此时平凡样本能带来的增益已经极少. 因此，在迭代中应根据当前迭代的轮数控制平凡样本的比例，定义自步采样系数 $\alpha=\tan\frac{p\pi}{2P}$（$p$ 为当前迭代数，P 为迭代总轮数）. 其中 α 控制采样时高难度样本的占比，其随着迭代轮数的增加而快速增大. 但是当 α 很大时，还是会抽取合适比例的平凡样本，这可以防止规则生成算法陷入局部问题的规则生成，提高规则泛化率.

对得到的规则难度进行分桶. 根据规则难度的值域被平均分为 k 份. 多数集中的每个样本根据规则难度投入相应的桶中，根据 α 和桶的平均规则难度在每个桶中抽取响应的样本. 记多数集中采样数量为 *size*，则第 l 个桶应采集样本数量计算为：

$$num_l=size\times\frac{\overline{hard[l]}+\alpha}{\sum_l(\overline{hard[l]}+\alpha)},$$

其中，*size* 表示要从多数集采样的总样本数；*num* 是指从每个桶采样的数量，每个桶采样数 *num* 的和即为 *size*；$\overline{hard[l]}$ 表示第 l 个桶中样本的平均 *hard* 值.

3.1.6 自步采样算法

自步采样算法的伪代码如算法 1 所示. 算法的输入是类别不平衡的原始数据集 D、使用特征数 m、规则难度分桶数目 k、迭代次数 P 以及采样数量 *size*. 在参数选择上，$m\geqslant k$，否则会存在无效的分桶；采样数

量 *size* 与计算成本、数据特征总维度、数据总量、数据精度、数据分布不平衡情况等有关，根据实践经验表明，采样数量要求最少为 1 000，否则会影响规则生成算法的效果，从而影响搜索服务响应时间异常诊断的准确性. 在其他的搜索服务场景下，可以根据业务专家的先验知识确定采样数量. 具体的运行参数设置在 4.3.2 节列出. 算法迭代 P 轮，每轮中根据当前所用样本计算一个离散化方法，在第 1 轮则根据全部原始数据计算. 之后选取最优特征子集、计算单特征判断方法、计算规则难度，每轮迭代更新自步采样系数 α，并计算相应的不同难度样本的采样比例. 迭代结束后，在少数集中抽取相同数目的样本作为输出.

算法 1. 自步采样算法.

输入：原始数据集 D，最优特征子集大小 m，难度分桶数目 k，采样数量 *size*，迭代次数 P；

输出：采样的日志 S.

① *major_set*←D 中响应时间正常的日志；
② *minor_set*←D 中响应时间过长的日志；
③ *hard*←长度与 *major_set* 相同的空数组；
④ for i = 1 to P
⑤ 离散化数值型特征；
⑥ *features*←带权熵最小的 m 个特征；
⑦ for f in *features*
⑧ *judge*←根据特征 f 生成的判断方法；
⑨ 根据 *judge* 更新 *major_set* 的 *hard* 值；
⑩ end for
⑪ *bin*[1,2, ⋯, k]←将 *major_set* 根据 *hard* 的值域分桶；
⑫ h[1,2, ⋯, k]←*bin* 数组对应桶内平均 *hard* 值；
⑬ $\alpha = \tan\frac{i\pi}{2P}$；
⑭ S←∅；
⑮ for l = 1 to k
⑯ $S.add$(抽取 $size \times \frac{\overline{hard[l]}+\alpha}{\sum_l(\overline{hard[l]}+\alpha)}$ 个来自第 l 个桶的样本)；
⑰ end for
⑱ end for
⑲ $S.add$(抽取 *size* 个来自 *minor_set* 的样本)；
⑳ 输出 S.

3.2 规则生成

采样完成后，Miner 用 Corels[7] 提取一条规则. Corels 是一个可证明的最佳规则列表学习器，该机制在可解释性的基础上寻找最优的逻辑模型，而其缺点在于复杂度和输入的数据量与维度是非线性的关系，因此本方法要提升规则生成的效率，就用到了自步采样. 根据 3.1 节的自步采样算法，随着迭代的进行，继续抽取平凡样本能提供的新信息是有限的，因此需要控制平凡样本的比例，增大规则难度较大的边界样本的比例，使得选取的更难分类样本生成的规则变得更加精准. Corels 推翻了越复杂和越不透明的模型具有越优秀的性能这一认知，旨在实现一个具有最优性能的、简单可解释的学习系统. 为解决困难的离散优化问题，Corels 通过自定义的方式，采用界限技术和有效的数据结构——字典树，针对实际问题目标，可以快速生成最佳的规则列表并提供一个其最优解的最优性证明. 在该算法的分支定界过程中，Corels 维护每个不完整规则列表可以达到一个正则化风险函数的最小值的下限. 总之，Corels 为可解释的决策规则提供了最优性保证. 关于 Corels 相比传统规则提取算法的优势，本文将在 4.4.3 节使用消融实验进行说明.

当 Corels 无法提取出有效的响应时间过长规则时，将会是 Miner 框架的第 1 个停止条件.

3.3 反向筛选

得到生成的规则后，在原始数据中去除被规则判断是过长的搜索响应时间日志，因为这部分数据不能为规则生成模型提供新的搜索响应时间过长相关特征，保留这些数据不利于提高模型的泛化率.

所有数据中，规则判断为搜索响应时间正常的数据作为下一轮的初始数据，重新投入算法的训练. 规则筛选的对象是产生规则的数据，因此此过程称为反向筛选. 当筛选后的数据是空集时，表明此过程已经提取所有日志中的响应时间过长规则，这是 Miner 框架的第 2 个停止条件.

3.4 总算法

为了准确描述 Miner 框架，在算法 2 中列出 Miner 的伪代码. Miner 的主要输入是原始数据集 D，反向筛选的最大轮次数 *max_iter*，以及自步采样相关的参数. 在每轮迭代中，使用自步采样提取具有表现力的数据样本，接下来应用 Corels 方法生成规则，反向筛选掉符合新规则的数据. 算法停止的条件是达到最大轮次数，或新生成的规则是空集，或筛选后数据为空. 每轮迭代生成的规则汇总成为 Miner 的输出.

算法 2. Miner 算法.

输入：原始数据集 D，最优特征子集大小 m，难度分桶数目 k，采样数量 *size*，自步采样迭代次数 P，反

向筛选最大轮次数 *max_iter*;

输出: 搜索响应时间过长的描述规则 *rules*.

① *rules*←∅;
② for *i* = 1 to *max_iter*
③ S_i←*SelfPacedSample*(*D*, *m*, *k*, *size*, *P*);
④ *new_rules*←*Corels*(S_i);
⑤ if *new_rules* ==∅
⑥ break;
⑦ end if
⑧ *D* = *filter*(*new_rules*, *D*);
⑨ if *D* ==∅
⑩ break;
⑪ end if
⑫ *rules.add*(*new_rules*);
⑬ end for
⑭ 输出 *rules*.

4 实验与分析

4.1 数据集

本文的实验采用中国 2 家搜索引擎公司真实的搜索响应时间日志数据, 分别记为数据集 *A* 和数据集 *B*, 这 2 个搜索引擎公司所占市场份额在中国排名前列. 数据集 *A* 含有 56 天大约 2 000 万条的日志, 该数据集是从文献 [6] 中获取的, 包含 6 个特征; 数据集 *B* 含有 3 天 74 万条的日志, 包含 7 个特征.

本文相信这 2 个搜索引擎公司提供的搜索响应时间性能问题是有足够代表性的, 在实验影响因素中也会进一步分析.

4.2 对比实验

1) DBSherlock[3] 是一种启发式算法, 其核心思想是从标定的异常和相邻的正常区间找出指标数据的分界点, 综合多种指标就可以得出一组诊断规则组合. 虽然该算法用于数据库的故障诊断, 但可以迁移到本文场景.

2) HHH[4-5] 是一种聚类算法, 其使用高维度的层次化聚类分辨出真实的瓶颈和其他部分. 该算法可直接应用在搜索响应时间诊断上.

3) FOCUS[6] 主要利用决策树算法, 使用响应时间过长的标签训练决策树模型, 合并路径上的决策条件作为响应时间过长的规则. 使用启发式方法整合多天数据, 也是本文直接相关的搜索响应时间过长诊断的工作.

4) DeCaf[9] 是一个基于随机森林的模型. 它为构成随机森林的决策树中每个节点定义一个分数, 用来衡量节点和故障的关系. 对所有节点的分数排序, 取分数最高的前数个节点, 其对应规则作为响应时间过长的描述.

5) FDA[12] 使用关联规则挖掘算法 FP-Growth 找出频繁出现的特征组合, 使用提升度衡量特征组合与故障的关系, 最后根据 2 条预设方法筛选得到的特征组合作为模型输出.

4.3 实验设置

4.3.1 度量方式

本文对效果的度量指标有泛化率 *Generality* 和召回率 *Recall*, *Generality*=规则包含的日志行数/日志总行数, *Recall*=规则包含的搜索响应时间过长日志/所有搜索响应时间过长的日志, 被多条规则包含的日志仍只算作 1 行. 在此本文并没有对比这些方法的精确率 *Precision*, 因为文献 [6] 中也探讨了规则诊断并不能精确地分类出全部的搜索响应时间过长的日志, 因为很多日志是由于随机原因导致变慢的, 如搜索的词条未被缓存或者网络状况的抖动, 但这些并不是本文重点关注来优化的随机事件. 本文对效率进行度量, 即 Miner 方法在不同数据量和不同特征维数下的运行时间. 其中每组实验运行 10 次取平均值.

4.3.2 参数设置

1) DBSherlock 模型的超参数. 分区数设置为 100, 归一化差值阈值设置为 0.12, 异常距离乘数设置为 5, 随机采样比例设置为 0.01, 迭代次数设置为 100.

2) FOCUS 模型的超参数. 终止阈值设置为 0.01.

3) DeCaf 模型的超参数. 分割所需最小样本数设置为 0.01, 特征采样率设置为 0.6, 决策树个数设置为 50.

4) FDA 模型的超参数. 最小支持度设置为 0.03 (数据集 *A*) 和 0.001 (数据集 *B*), 提升度阈值设置为 1.1, 支持度阈值设置为 1.1. Miner 框架使用的参数如表 2 所示.

Table 2 Parameters Setting of Miner

表 2 Miner 参数设置

超参数	说明	取值
m	最优特征子集大小	5
k	难度分桶数目	5
size	采样数量	1 500
P	自步采样迭代次数	5
max_iter	反向筛选最大轮次数	5

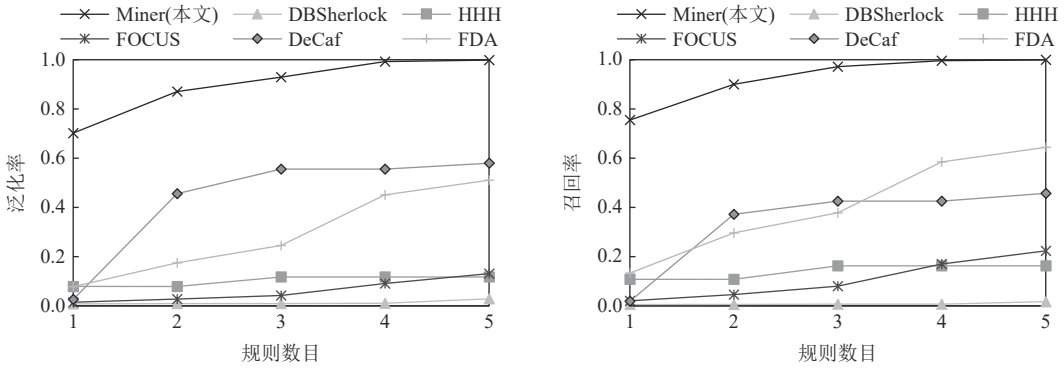
4.3.3 实验影响因素

实验的影响因素主要是实验对象和实验数据. 在本文的实验中, 虽然使用了 2 个顶级搜索引擎公司的数据, 但其仍无法代表所有搜索引擎数据. 之后将使用更多的数据集, 可能不局限于搜索响应时间, 如 Web 响应时间都是可以用本文的方法进行诊断的. 针对性能问题的场景, 本文不具体研究每种性能问题是否有可行的解决方案, 而是提供给运维人员诊断报告. 对尚未发现的运维人员根据经验发现的性能问题, 则不在本文诊断出的范围内. 实验中的构造影响因素主要包括实验的指标计算方式和随机因素. 对于指标计算方式, 本文使用了已有工作常用的召回率和泛化率, 后续将引入更多的指标来度量本文方法的效果. 对于实验的随机因素, 本文通过重复多次实验来减少其影响.

4.4 结果与分析

4.4.1 Miner 生成的诊断规则的效果

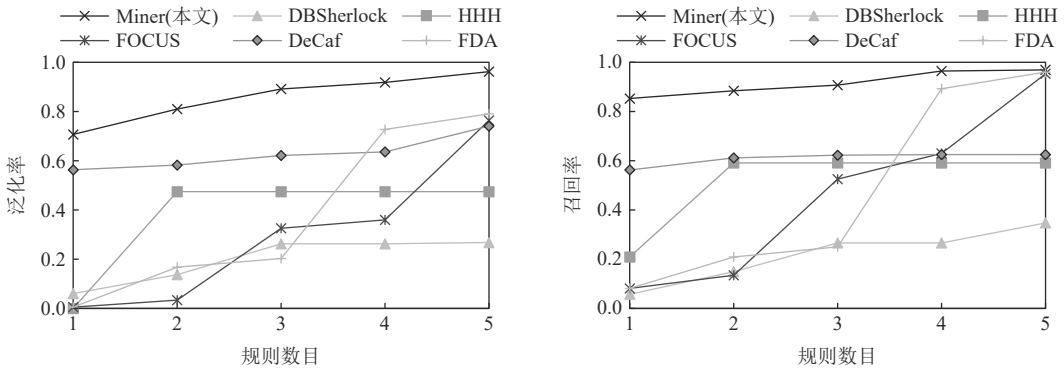
2 个数据集的效果比较如图 4 和图 5 所示, 其中图 4(a)和图 5(a)是诊断规则的泛化率, 图 4(b)和图 5(b)是召回率, 挖掘出的规则效果按照降序排列, 可以观察到 Miner 的效果都显著高于其他的对比方法, 如 DBSherlock, HHH, DeCaf. 在这 2 个数据集上, 5 种对比算法效果都不理想, 挖掘出的规则组合不具有普遍性, 因此泛化率和召回率都很低.



(a) 数据集*A*诊断规则的泛化率 (b) 数据集*A*诊断规则的召回率

Fig. 4 Comparison of diagnostic effectiveness in dataset *A*

图 4 数据集 *A* 中的诊断效果比较



(a) 数据集*B*诊断规则的泛化率 (b) 数据集*B*诊断规则的召回率

Fig. 5 Comparison of diagnostic effectiveness in dataset *B*

图 5 数据集 *B* 中的诊断效果比较

Miner 的一个特点是生成的第 1 条规则就具有很高的泛化率和召回率, 2 个数据集的实验都验证了这一点, 这说明结合自步采样和规则生成算法能很好地应对多维度搜索响应日志的分布不平衡问题; 此外, 以数据集 *A* 为例, 当数据不能用一条规则充分刻画时, Miner 的迭代式反向筛选可以在迭代过程中充分提高规则集的整体效果.

4.4.2 Miner 生成的诊断规则的效率

为了考察 Miner 算法生成的诊断规则的效率, 本文组织了实验考察数据量和特征维度数对诊断时间的影响. 实验平台软硬件环境如表 3 所示.

关于数据量对运行时长的影响, 图 6 展示了 Miner

Table 3 Experiment Platform

表 3 实验平台

软硬件	配置
CPU	E5-2 650, 2.20 GHz
内存	DDR4, 128 GB, 2 400 MHz
GPU	Matrox Electronics Systems Ltd. G200eR2
操作系统	Ubuntu 16.04.7

的运行时长与数据量变化的关系. 此实验的数据是从数据集 A 中抽取的, 从 100 万条搜索响应日志开始, 以 100 万条为单位增加, 直到 1 000 万条日志为止. 实验反映 Miner 耗时随着数据量的增加而大致呈线性增长, 平均每 100 万条日志耗时 70.37 s, 可以满足实际环境下对大量数据进行诊断的时间要求. 实验的另一个结论是运行时间主要集中在前几轮迭代, 因为反向筛选减少了后几轮迭代运行的处理规模.

搜索响应日志的特征维度数量也会影响 Miner 的运行时长. 因此组织了在数据集 A 的 500 万条搜索日志上不同特征维度对应运行时间的实验, 对数据集 A 的总特征分别取数量为 3, 4, 5, 6 个特征的子集, 作为实验的输入, 其结果如图 7 所示. 从运行时长的 1 分位和 3 分位数来看, 更多的特征维度对应更长的运行时间, 但是具体的某些特征维度在运行过程中需要更多的处理时间, 导致在总特征维度数较少(3 维或 4 维)时运行时长比较接近, 这和 Miner 在自步采样中的维度处理方法有关.

图 6~7 实验结果表明 Miner 框架可以应对实际生产环境中产生的大量搜索响应日志. Miner 用自步采样提升了效率, 因此能满足实际计算效率的需求.

4.4.3 Miner 框架各步骤的作用

Miner 框架由自步采样、规则生成和反向筛选 3 个步骤组成, 为了衡量各个步骤的效果, 本文组织了消融实验, 对 Miner 中的 3 个步骤分别替换或去除, 使用数据集 B 完成实验. 实验结果如图 8 所示.

为了考察第 1 个步骤即自步采样的作用, 使用分层采样替换自步采样. 在每轮迭代中, 按照多数集和少数集的比例抽取相应数量的样本, 其余部分不作改动. 结果显示简单的分层采样得出的规则: 小于 3 条规则数目的召回率和泛化率都较低, 经过 3 轮反向筛选后才提高到可用的水平; 而去除反向筛选算法得到的规则具有良好的泛化率和召回率.

为了考察第 2 个步骤即 Corels 生成规则的作用, 使用 CART 决策树替换 Corels 规则生成算法. CART

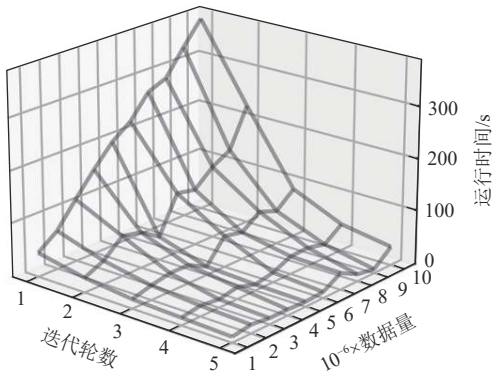


Fig. 6 Relationship of runtime and data sizes on Miner

图 6 Miner 运行时长与数据量的关系

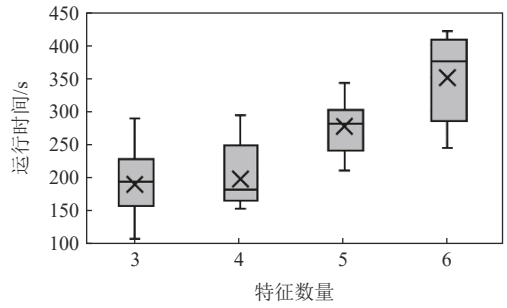
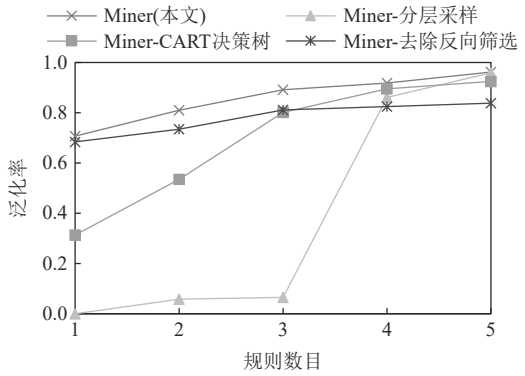
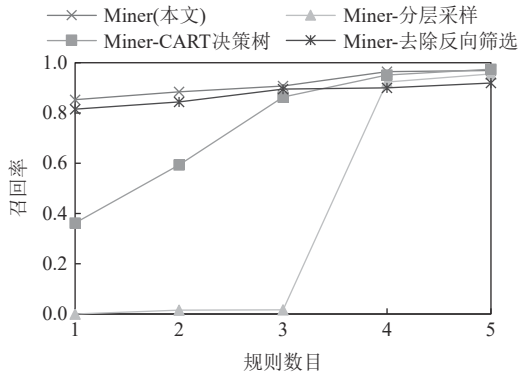


Fig. 7 Relationship of runtime and attribute dimensions on Miner

图 7 Miner 运行时长与特征维数的关系



(a) 消融实验下的泛化率



(b) 消融实验下的召回率

Fig. 8 Contribution of each step of Miner

图 8 Miner 各步骤的作用

决策树的泛化率和召回率曲线较为平滑，随着迭代轮数的增加，相对稳定地增加，即没有出现分层采样实验在规则数目为 3 条和规则数目为 4 之间的突变. 但是 CART 决策树在规则数目为 3 之前的表现不够理想，大致需要 3 条规则，才能在泛化率和召回率上和原始 Miner 的规则数目为 1 时相当.

对于第 3 个步骤即反向筛选在 Miner 中的作用，去除反向筛选过程，并在每轮迭代起始时重置一个与时间有关的随机数种子. 因为反向筛选是每轮迭代的最后一个步骤，因此数目为 1 时的表现与没有改动的 Miner 大致相当. 实验表明规则数目增加时，重置随机数种子也能对规则的泛化率和召回率有所提升，加入反向筛选（即原始 Miner）的迭代过程对规则的提升更大一些. 特别是反向筛选对泛化率的提升随迭代轮数的增多，相比重置随机数种子的优势逐渐增大.

4.4.4 实例分析

以数据集 *A* 为例，本文用 Miner 生成搜索响应时间的规则包括：

1）‘CHINANET’ ^ ‘not *#Image* 1~9’；

2）‘not *#Image* 1~9’ ^ ‘not others’；

3）‘无广告’ ^ ‘not MSIE 7.0’ ^ ‘not *#Image* 0~1’；

4）‘async’ ^ ‘*#Image* 0~1’；

5）‘not GWBN’ ^ ‘not Yunnan’ ^ ‘not Shanxi’.

可见图片个数和其他条件的组合是导致搜索响应时间慢的主要原因，几乎出现在每个搜索响应时间过长的规则中，这些规则可以被运维工程师所接受. 本文比较 FOCUS 得出最显著的影响条件是图片个数为 5~9. 在实际环境中，运维工程师采用 base64 编码，可压缩图片大小便于传输，在这种优化之后搜索响应时间过长的占比从 30% 下降至大约 20%. 因此本文能够使用得到的规则来指导相应的优化.

5 讨　论

Miner 框架存在一些限制，需要一个预设的阈值，但并不能自动从数据中发掘，而阈值的设定可根据专家经验或其他算法得出. Miner 基于 Corels 算法，处理的数据对象是结构化搜索日志，其特征维度的含义是可被理解的，由此得出的输出规则可用于后续的人工分析和优化. 自由文本、图像、语音等非结构化数据并不适用于 Miner. 此外，时间戳信息并不能作为 Miner 的特征维度来处理，因此 Miner 并不能挖掘出搜索响应时长在一段时间内的变化规律.

6 总　结

本文提出 Miner 框架进行搜索响应时间过长诊断. 通过对搜索响应日志的经验性研究，表明仅使用单维度特征难以准确定位响应时间过长的问题，因此需要多维度的描述性规则；部分数据样本存在较大的分类难度，需要额外考虑. 本文提出使用自步采样，逐步给予难度更高的数据样本更大的采样比例，正确指引了规则生成算法的重点，此后 Miner 框架进行反向筛选，以提高输出规则集合的泛化率，能更全面地覆盖搜索响应时间过长的问题. 在国内 2 家顶级搜索引擎数据集上，Miner 生成的规则取得最优泛化率与召回率，这些规则可应用于搜索引擎的优化. 相信 Miner 这种不需要结合领域知识、快速有效的规则挖掘方法可迁移到其他的性能问题诊断场景，这也是未来工作的探索方向.

作者贡献声明：夏思博完成主要实验并撰写论文；马明华提出了算法思路和实验方案；金鹏翔、崔丽月修改了论文；金娃完成部分实验；张圣林、孙永谦、裴丹对论文提出指导意见.

参　考　文　献

[1] Schurman E, Brutlag J. The user and business impact of server delays, additional bytes, and http chunking in web search[C/OL]//Proc of the Velocity Web Performance and Operations Conf. Sebastopol, CA: O'Reilly Media, 2009[2023-07-19].https://www.researchgate.net/publication/280113406

[2] Sundaresan S, Magharei N, Feamster N, et al. Web performance bottlenecks in broadband access networks[C]//Proc of the 2013 ACM SIGMETRICS/Int Conf on Measurement and Modeling of Computer Systems. New York: ACM, 2013: 383−384

[3] Yoon D Y, Niu N, Mozafari B. DBSherlock: A performance diagnostic tool for transactional databases[C]//Proc of the 2016 Int Conf on Management of Data. New York: ACM, 2016: 1599−1614

[4] Jiang Junchen, Sekar V, Stoica I, et al. Shedding light on the structure of internet video quality problems in the wild[C]//Proc of the 9th ACM Conf on Emerging Networking Experiments and Technologies. New York: ACM, 2013: 357−368

[5] Liu Dapeng, Zhao Youjian, Pei Dan, et al. Narrowing down the debugging space of slow search response time[C/OL]//Proc of the 34th Int Performance Computing and Communications Conf. Piscataway, NJ: IEEE, 2015[2023-07-19].https://ieeexplore.ieee.org/abstract/document/7410318

[6] Liu Dapeng, Zhao Youjian, Sui Kaixin, et al. FOCUS: Shedding light on the high search response time in the wild[C]//Proc of the 35th Annual IEEE Int Conf on Computer Communications. Piscataway, NJ: IEEE, 2016: 1−9

[7] Angelino E, Larus-Stone N, Alabi D, et al. Learning certifiably optimal rule lists for categorical data[J]. arXiv preprint, arXiv: 1704.01701, 2018

[8] Khoussainova N, Balazinska M, Balazinska M. PerfXplain: Debugging mapreduce job performance[J]. arXiv preprint, arXiv: 1203.6400, 2012

[9] Bansal C, Renganathan S, Asudani A, et al. DeCaf: Diagnosing and triaging performance issues in large-scale cloud services[C]//Proc of the 42nd ACM/IEEE Int Conf on Software Engineering: Software Engineering in Practice. New York: ACM, 2020: 201−210

[10] Zhang Xu, Du Chao, Li Yifan, et al. HALO: Hierarchy-aware fault localization for cloud systems[C]//Proc of the 27th ACM SIGKDD Conf on Knowledge Discovery & Data Mining. New York: ACM, 2021: 3948−3958

[11] Castelluccio M, Sansone C, Verdoliva L, et al. Automatically analyzing groups of crashes for finding correlations[C]//Proc of the 11th Joint Meeting on Foundations of Software Engineering. New York: ACM, 2017: 717−726

[12] Lin F, Muzumdar K, Laptev N P, et al. Fast dimensional analysis for root cause investigation in a large-scale service environment[J/OL]. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 2020[2023-07-19].https://dl.acm.org/doi/abs/10.1145/3392149

[13] Liu Zhining, Cao Wei, Gao Zhifeng, et al. Self-paced ensemble for highly imbalanced massive data classification[C]//Proc of the 36th IEEE Int Conf on Data Engineering. Piscataway, NJ: IEEE, 2020: 841−852

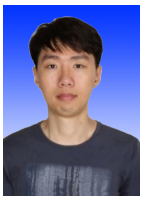
[14] Cheng Li, Wang Yijie, Liu Xinwang, et al. Outlier detection ensemble with embedded feature selection[C]//Proc of the 34th AAAI Conf on Artificial Intelligence. Palo Alto, CA: AAAI, 2020: 3503−3512

Xia Sibo, born in 2000. Master candidate. His main research interests include knowledge graph, and failure detection and diagnosis.

夏思博, 2000 年生. 硕士研究生. 主要研究方向为知识图谱、故障检测与诊断.

Ma Minghua, born in 1993. PhD. Researcher at Microsoft Research Lab Asia. His main research interests include cloud intelligence and AIOps.

马明华, 1993 年生. 博士. 微软亚洲研究院研究员. 主要研究方向为云智能、智能运维.



Jin Pengxiang, born in 1998. Master. His main research interests include failure diagnosis and root cause analysis.

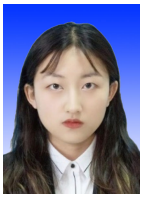
金鹏翔, 1998 年生. 硕士. 主要研究方向为故障诊断、根因分析.

Cui Liyue, born in 1997. Master. Her main research interests include anomaly detection, failure diagnosis, and machine learning.

崔丽月, 1997 年生. 硕士. 主要研究方向为异常检测、故障诊断、机器学习.

Zhang Shenglin, born in 1989. PhD, associate professor. Member of CCF, IEEE, and ACM. His main research interests include failure detection, and diagnosis and prediction in data center networks.

张圣林, 1989 年生. 博士, 副教授. CCF 会员、IEEE 会员、ACM 会员. 主要研究方向为数据中心网络中的故障检测、诊断与预测.



Jin Wa, born in 2001. Bachelor. Her main research interests include failure diagnosis and root cause analysis.

金　娃, 2001 年生. 学士. 主要研究方向为故障诊断、根因分析.

Sun Yongqian, born in 1988. PhD, associate professor. Member of CCF, IEEE, and ACM. His main research interest includes anomaly detection, root cause analysis, and failure diagnosis in service management.

孙永谦, 1988 年生. 博士, 副教授. CCF 会员、IEEE 会员、ACM 会员. 主要研究方向为服务管理中的异常检测、根因分析、故障诊断.

Pei Dan, born in 1973. PhD, associate professor. Member of CCF. Senior member of IEEE and ACM. His main research interest includes network and service management.

裴　丹, 1973 年生. 博士, 副教授. CCF 会员、IEEE 高级会员、ACM 高级会员. 主要研究方向为网络和服务管理.