**摘要:**

在大规模的在线服务系统，为了提高服务质量，工程师需要收集各种监测数据并编写许多规则来触发警报。然而，警报的数量远远超过了随叫随到的工程师能够正确调查的数量。因此，在实践中，使用手动规则将警报分为几个优先级，待命工程师主要关注处理优先级最高的警报（即严重警报）。不幸的是，由于在线服务的复杂性和动态性，这种基于规则的方法会导致错过严重警报或浪费非严重警报的故障排除时间。在本文中，我们提出了AlertRank，一个用于识别严重警报的自动自适应框架。具体来说，AlertRank提取了一组强大且可解释的特征（文本和时间警报特征、用于监控指标的单变量和多变量异常特征），采用XGBoost排名算法识别所有传入警报中的严重警报，并使用新的方法获取标签，用于培训和测试。对一家全球顶级商业银行的数据集进行的实验表明，AlertRank是有效的，平均F1得分为0.89，优于所有基线。实践的反馈表明，AlertRank可以显著地节省随叫随到工程师的人工工作。

1. 介绍

大型复杂的在线服务系统，如Google、Amazon、Microsoft、大型商业银行等，由数千个分布式组件组成，支持大量并发用户[1]。为了维护高质量的服务和用户体验，这些公司使用监控系统从服务组件收集各种监控数据，例如，指标/关键性能指标（KPI）[2]、日志[1]和事件[3]。通常，工程师会设置许多规则来检查监控数据并触发警报[4]–[6]。例如，如果所选指标超过给定的阈值（例如，CPU利用率超过90%），则会生成一个警报，通知随叫随到的工程师。然后工程师检查这个警报，如果此警报严重，则会创建事件单，以立即启动深入调查和适当维修的流程[7]。

不幸的是，在现实世界中，这些复杂的服务系统不断地产生大量的警报，远远超过了资源受限的随叫随到工程师能够正确调查的[5]、[6]、[8]。例如，图1显示，一家大型商业银行C（中国建设银行）[8]每天都会生成数千个警报。因此，在实践中，使用手动规则将警报分为不同的优先级，例如P1 critical、P2 error和P3 warning。这些规则通常包含KPI的固定阈值（例如，CPU利用率超过90%为P2错误，超过80%则为P3警告）、日志匹配关键字（例如，“警告”、“错误”和“失败”等不同关键字表示不同的严重性）等。一般情况下，在人力有限的情况下，随叫随到工程师主要关注最优先的警报（以下简称严重警报），在处理其他警报之前，这些警报在C银行的日计数仍然可以达到数百个。

图表

描述已自动生成

但是，简单的手动规则无法充分捕捉影响警报优先级的复杂交互因素的模式。此外，工程师手动定义和维护规则是一项劳动密集型工作，因为1）警报类型很多；2）系统更改可能会添加新类型的警报；3）工程师可能有不同的优先级偏好[5]。因此，在实践中，上述基于规则的方法通常会导致严重警报丢失和修复时间延长，或者导致在非严重警报上浪费故障排除时间。例如，在bank C的xIV中使用的数据集中，识别严重警报的精度和召回率分别只有0.43和0.68。

为了显示漏掉的严重警报的损害，我们在图2中展示了一个真实的案例。10:14出现故障和相关的即时P2警报，这表明监控的事务响应时间增加到500毫秒（超过P2规则中指定的阈值）。但是，工程师正忙于处理P1警报，因此没有立即处理该警报。直到10点45分客户打电话投诉，才发现故障，浪费了31分钟的维修时间。经过详细调查，多个指标（如成功率、交易量）在繁忙时间都出现了一些异常，基于这些异常，工程师认为应该有P1警报，但是没有安装P1规则来捕捉如此复杂的症状。这个案例有力地说明了需要多个特性来确定适当的警报优先级。

图片包含 日程表

描述已自动生成

因此，迫切需要设计一种有效的基于机器学习的算法，充分利用多种特征对严重警报进行准确识别，帮助随叫随到工程师区分高优先级严重警报和非严重警报，以提高工程师的工作效率和服务质量。设计这种算法的挑战总结如下。

标注开销。由于每天都有数千个警报到达，手动标记每个警报的严重性是很乏味的。

各种各样的警报。在实际应用中，有许多类型的警报，例如应用程序、网络、内存和中间件。手动定义规则以确定每种警报的优先级将是一项耗费大量人力的工作。自动进近非常有用。

复杂和动态的在线服务系统。在这样一个复杂的系统中，警报优先级受多种因素的影响。此外，大规模的在线服务经常处于不断变化之中，如配置变更、软件升级等，这些都可能增加新的警报。因此，该方法应该是自适应的，以适应动态环境。

数据不平衡。一般来说，只有一小部分警报被认为是严重的。例如，在我们的实验数据集中，非严重警报和严重警报之间的比率约为50:1（xIV-A）。从不平衡数据中学习（例如二进制分类）是另一个挑战。

为了解决上述问题，我们提出了一个自动的自适应框架AlertRank来识别严重警报。AlertRank包括两个部分，离线培训和在线排名。在离线训练中，为了处理复杂的在线服务，AlertRank从历史警报和监控指标中提取一组可解释的特征来表征警报的严重性。历史警报中的解决方案记录被分组到极少数的集群中，每个集群都由工程师用严重性评分进行标记。然后自动为每个历史警报分配一个严重性评分（xIII-A2），避免手动标记。

AlertRank没有简单地对警报是否严重进行分类，而是将识别严重警报的问题描述为一个等级模型，它可以处理类别不平衡，并且对工程师来说非常友好。然后利用XGBoost排序来训练一个排名模型[9]。此外，利用增量训练管道使模型适应动态环境。在线排名过程中，当警报到达时，AlertRank从当前数据中提取特征，并应用经过训练的排名模型，根据输出的严重性得分对这些警报进行优先级排序，以便工程师能够根据AlertRank给出的指导方针对到达的警报进行调查。

本文的主要贡献如下：

* 据我们所知，本文首次提出了一种自动和自适应的方法来识别在线服务系统的严重警报。
* 我们从多个数据源设计了一整套功能，以处理复杂的在线服务。这些特征包括来自警报的纹理（主题[10]和熵[11]）和时间特征（频率、季节性[12]、计数、间隔时间[13]），以及度量[14]中的单变量和多变量异常特征[14]，这些特征是可解释的，可以进一步帮助工程师诊断事件。
* 与传统的分类模型不同，我们将识别严重警报的问题定义为一个排序模型，它可以处理类别不平衡，并指导工程师先修复哪个警报[15]。
* 我们新颖地提出使用历史记录单作为测试标签，并建议使用基于聚类的方法（只有聚类级别标签）自动为所有警报分配严重性评分标签，作为排名的训练标签。
* 在实际数据集上的实验表明AlertRank是有效的，F1平均得分为0.89。此外，我们的实际部署表明，AlertRank可以大大节省随叫随到工程师的手动工作

1. 背景
2. 报警管理系统

图3提供了由服务体系结构、监控系统和警报管理系统组成的IT操作生态系统的高层概要[3]。监控系统不断地从服务组件收集各种数据（例如kpi、日志和事件）。为了确保服务的可用性和稳定性，服务工程师手动定义了许多规则来检查监控数据并触发警报[5]。接收到警报后，警报管理系统通过一些常用技术（如警报聚合和关联）处理警报，以过滤重复警报和无效警报。对于剩下的警报，如I中所介绍的，待命工程师通常首先检查优先级较高的警报（P1）。如果警报严重且难以及时修复，则会创建事件通知单[7]。然而，如I所述，基于规则的严重性分类在实践中可能会失败。本文针对如何准确、自适应地识别严重警报这一关键问题进行了研究，以帮助工程师响应性地修复真正的严重警报，防止潜在风险。

图示

描述已自动生成

1. 数据描述
2. 预警：预警数据作为我们研究的主要数据对象，具有多维属性。表1给出了一个具有几个主要属性的示例警报。“内容”指定警报的详细信息，“解决记录”详细记录警报的解决方式，通常由随叫随到的工程师编写或由系统自动生成。“解决记录”可用于标记每个警报的严重性评分，将在III-A2中介绍。我们将分别从“内容”和“时间”属性中提取警报文本和时间特征（III-B）。

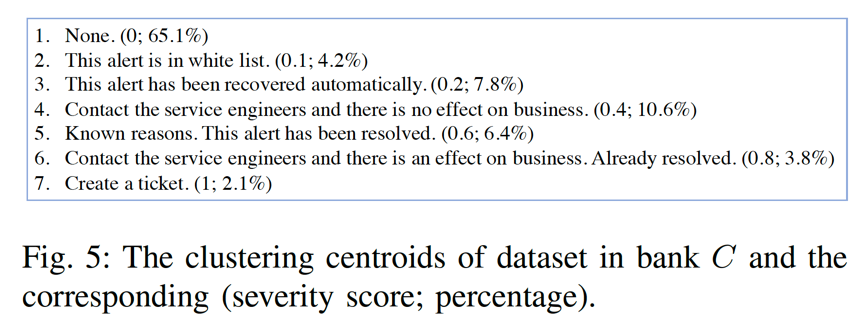
表格

描述已自动生成

1. 关键性能指标：KPI（指标）是服务管理中的另一个关键数据类型，它以固定的时间间隔（例如每分钟）连续收集，以测量服务器的健康状态（例如CPU利用率）或业务（例如，响应时间）[2]，[16]。KPI异常（例如突然峰值或下降）可能表示业务和服务器存在一些问题。除了警报数据外，我们还从重要业务KPI和服务器KPI中提取KPI异常特征，以表征系统的总体状态（III-C）。
2. ticket：一般来说，tickets的创建是根据警报或失败。在调查警报时，如果工程师发现很难及时修复警报，或者警报对业务有重大影响，他们会根据该警报创建一张ticket，以便跟进修复流程[7]、[17]、[18]。另一方面，如图2中的故障情况，在服务管理中是关键的，因为它们直接影响服务的稳定性。故障通常由用户报告或由维修工程师发现，而不是事先由任何警报报告。工程师会开出罚单并迅速修复故障。通常，在诊断故障并确定其根本原因时，工程师可能会发现一些被错误忽略的早期警报（例如，图2中的警报）。因此，我们可以通过是否与ticket相关联来确定警报是否严重。
3. 设计
4. 概述
5. *AlertRank概述*：图4说明了AlertRank的体系结构。在离线学习模块中，受多特征融合思想的启发，我们根据上下文中的领域知识，从历史警报（纹理和时间特征）和kpi（单变量和多变量异常特征）中提取一组可解释特征。然后利用流行的XGBoost排序算法构造了一个排序模型[9]。用于训练的故障严重程度得分由表一中的解决记录获得，将在III-A2中详细介绍。在在线排名中，将从输入的警报和相应的kpi中提取的特征直接输入到经过训练的排名模型中。该模型根据输出严重性评分对传入警报进行排序，以便随工程师可以首先检查严重性得分较高的警报。严重性得分超过特定阈值的警报被视为严重警报，并根据训练集上的性能选择阈值。此外，为了使我们的模型能够适应动态的IT环境，我们构建了一个增量的训练管道，在这个管道中我们的模型被周期性地使用最新的数据进行训练，以便能够及时正确地捕捉到变化。
6. *自动标记：*正如I所讨论的，标记开销对于我们的问题是一个很大的挑战。在AlertRank中，我们新颖地利用历史记录单和解析记录来获取标签，而无需人工操作。

**二进制标签**。本文所要解决的问题是识别严重警报。因此，我们需要给每个警报一个二进制标签（严重/非严重），以供评估（IV-B）。如II-B3所述，根据是否与罚单相关，每个警报可被标记为严重或非严重。

**连续标签**。为了训练排名模型，我们还根据每个警报的resolution record为每个警报标记一个特定的严重性评分（介于0到1之间），这比简单的二进制标签更全面。这是因为每个警报的实际严重程度在所有严重/非严重警报中都有所不同。记录由系统自动生成。一般来说，有几种不同类型的resolution record，它们表示不同的严重程度。我们采用TF-IDF矢量化[11]和k-means [19]对解决记录进行聚类，k值由轮廓系数（k=7）[20]。聚类质心如图5所示，经验丰富的工程师为每个聚类给出一个严重性评分。这样，每个警报都可以根据其resolution record自动分配严重性分数，从而节省大量的人工工作。此外，在我们对长达18个月的数据集的研究中，我们观察到除非有新的工程师参与，否则resolution record的模式通常保持不变。

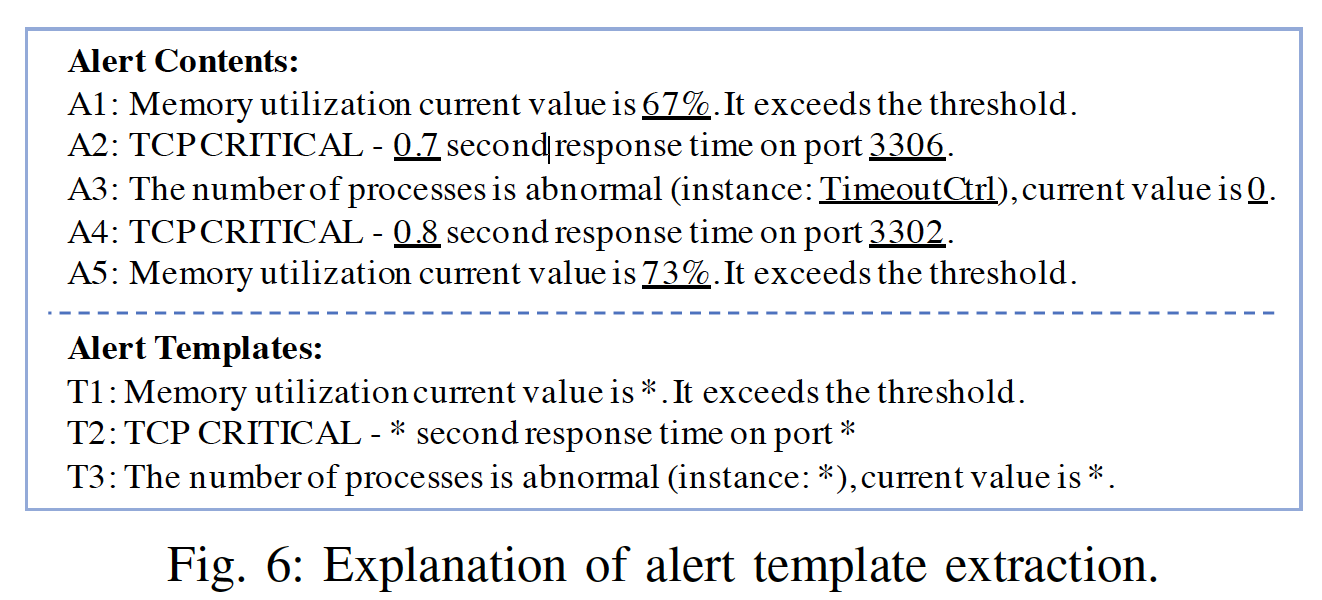


1. 报警特征抽取

**1） 预警预处理：**在提取预警特征之前，我们首先对预警数据进行如下预处理。

**标记化**。警报的文本描述（表一中的“内容”）通常结合了单词和符号（例如标点符号）。标记化过滤掉这些符号并将剩余的文本划分为标记。我们进一步删除了停止词，那些频繁出现的词，如“to”、“are”和“is”，因为它们在识别严重警报时没有用处。

**警报分析。**警报的描述是由监控系统生成的半结构化文本，包含两类信息：描述抽象警报事件的常量字符串和记录某些系统变量（如IP地址、KPI值、文件路径、实例名称等）的参数。一种常见的做法是解析警报，它提取常量字符串以形成警报模板。例如，在图6中，警报A1和A4可以分别适合于模板T1和T2。其余部分是变量（带下划线的文本）。请注意，忽略一些KPI值（例如67%和73%）并不重要，因为它们反映在基于规则的严重性（表I中的“严重性”）。



分析方法已经在日志数据中得到了很好的研究，但还没有应用到警报中。我们采用FT-tree[21]，最先进的日志解析方法之一，将每个警报与特定警报模板相匹配。“正确的”警报模板通常是警报中频繁出现的单词的组合，基于这一关键观察，FT-tree动态地维护了一个包含这些频繁单词的树结构，该结构隐式定义了警报模板集。除了高精度之外，FT-tree自然具有增量可追溯性，因为这一隐式定义的警报模板集随着新类型警报的到来而动态和增量地演化（例如，由于上述软件升级）[21]。

**2) 文本特征：**

经过预警预处理后，将半结构化预警转化为规范化模板。然后我们从警报模板中提取一些文本特征。主题。直观地说，每个警报模板都可以看作是描述一个或多个IT操作主题的文档。我们可以应用主题模型[22]来提取隐藏的语义特征。文献中提出了许多主题模型，例如潜在Dirichlet分配（LDA）[23]。传统的主题模型，如LDA，通常可以很好地处理由富文本组成的文档，而我们的警告文本通常非常短。因此，我们采用了**Biterm Topic Model（BTM）**[10]，这是一种专门针对短文本设计的算法，在短文本中表现出比LDA更好的性能。BTM显式地建模单词共现模式（即比特项），而不是文档，以增强主题学习[10]。根据给定的主题，发现每个主题对应的隐藏主题数。我们的问题中的主题数量是根据连贯性得分（n个主题=14个）来选择的，这是一个衡量抽取主题质量的指标[24]。图7展示了从我们的实验数据集中提取主题和相应关键字的一些示例。例如，我们可以推断T#1和T#2分别与oracle数据库和syslog相关。BTM可以输出警报属于每个主题的概率。图8（a）显示了警报主题与严重程度得分之间的关系，我们可以观察到属于不同主题的警报表示不同的严重程度。考虑到警报可能具有混合主题，我们利用输出概率作为主题特征。

文本

描述已自动生成

**熵。**考虑到警报是一堆单词，不同的单词在识别严重警报时具有不同的重要性（熵）。例如，单词“timeout”比“port”信息更丰富。因此，除了主题特征外，我们还考虑了每个警报模板的熵。IDF（Inverse Document Frequency，逆文档频率）在文本挖掘中被广泛应用于衡量词的重要性，它在降低频繁词的权重的同时提高了稀有词的权重[11]。对于每个字，IDF值计算为图片包含 文本

描述已自动生成，其中N为警报总数，Nw为包含w的警报数量。直观地看，如果一个词频繁出现在历史警报中，那么它的辨别力远远低于只出现在少数警报中的单词[1]。根据训练数据计算出的每个单词的IDF，我们可以将每个警报的熵计算为图片包含 文本

描述已自动生成，其中#w是该警报中的字数。图8（b）显示了具有高熵的警报更可能是严重的。

图形用户界面, 图示

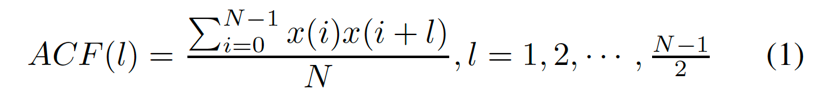
描述已自动生成

**3） 时间特征：**除了文本特征外，还应考虑由警报时间派生的时间特征。

**频率。**一些警报（如CPU过载）频繁发生，而其他警报（如设备停机）则很少发生。图8（c）显示了频率和严重程度评分之间的关系。一般来说，频率较低的警报更可能严重。因此，每个警报的频率可以指示确定警报是否严重。

**季节性。**根据我们的观察，有些警报是准周期性的。例如，每晚运行批处理任务可能会导致CPU和磁盘警报。直觉上，季节性警报的信息量不如不定期警报。对于一个警报a，我们得到一个时间序列C（a）={c1（a）；c2（a）······ch（a）}，其中ck（a）是警报a在第k个时间点出现的次数，h是时间段的个数。这里的时间间隔设置为15分钟。显然，如果a是季节性警报，那么C（a）就是一个周期性的时间序列。

在我们的方法中，我们采用最流行的时间序列周期性检测技术之一的自相关函数（ACF）来表征警报的周期性[25]，[26]。具体地说，给定一个长度为N、滞后时间为l的时间序列x，我们得到:



很明显，如果时间序列是周期性的且长度为T，则自相关在某些滞后点（即T；2T；3T）变得很高；因此，较大的ACF（l）意味着更强的季节性。我们使用ACF（l）的最大值作为最终的季节性特征。从图8（d）可以看出，季节性值越大，严重程度得分越低。

**警报计数**。我们将警报计数定义为在一个时间窗口内（例如，当前警报前30分钟）发生的警报数量[13]。直观地说，工程师需要更多地关注短时间内突发的警报（图8（e））。此外，我们还考虑了在这个时间窗口内不同严重级别的警报的数量。

**故障间隔时间。**间隔时间定义为警报与警报进行之间的时间间隔[13]。显然，如果在一个系统长时间（长时间的时间）没有发出警报的情况下突然发生警报，那么警报可能很严重，需要更多的关注（图8（f））。

**4） 其他特性：**除了上面描述的这些精心设计的特性之外，我们还直接从警报数据的属性中采用了一些简单的特性（表一）。

* *基于规则的严重性。*图8显示了严重程度与真实等级之间的关系。很明显，P1和P2警报的严重性评分都较高，因此只关注P1警报而忽略P2警报是不合理的。但是，我们仍然可以利用原始严重性作为一个特性。
* *警报时间。*显然，警报时间对警报的重要性有影响。例如，如图8（h）所示，在繁忙时间（9:00-11:00和14:00-16:00）发生的警报比晚上发生的警报更严重。因此，我们采用三个特征来表征警报的发生时间，即是否在繁忙时间、白天或晚上、工作日或周末。
* *类型。*我们还注意到警报类型对其严重性级别有影响。图8（i）显示了几种不同类型警报（应用程序、操作系统、网络、内存、中间件和其他）下的平均严重性得分。很明显，应用程序警报往往具有更高的严重性得分，这可能是因为这些警报与服务可靠性和用户体验更密切相关。

1. KPI特征抽取

此外，由于kpi可以描述服务器和应用程序的运行状况[27]，因此我们引入kpi来更好地捕捉警报的严重性。请注意，尽管业务KPI和服务器KPI可以直接触发警报，但仅凭这些警报无法准确描述系统的运行状况，因为它们是根据简单的阈值规则生成的。因此，在本研究中，我们尝试设计一个更精确和通用的方法来捕捉KPI异常。特别是，我们认为应该注意那些与重要业务KPI或服务器KPI异常同时发生的警报。

总之，通常会测量多个kpi来监视一个服务，而工程师手动检查所有kpi是非常耗时的。在我们的方法中，我们选择了一些与服务可用性和用户体验密切相关的具有代表性的业务KPI（响应时间、成功率、事务量、处理时间）和服务器KPI（CPU利用率、I/O等待、内存利用率、负载、网络包数、进程数、磁盘I/O），并采用多KPI异常检测技术，准确测量业务和服务器的整体状态。图9（a）给出了一个服务的三个重要业务kpi的示例，粉色带表示异常。

图示

描述已自动生成

我们使用了最先进的基于LSTM的多变量时间序列异常检测算法[14]，[28]。与传统的递归神经网络（RNN）相比，LSTM具有更好的保持长期依赖记忆的能力，因为它使用加权自循环来积累或忘记受上下文约束的过去信息[14]。因此，它可以学习过去和现在的数据值之间的关系，并在各种序列数据中表现出显著的性能[29]。图9（b）展示了我们方法中的LSTM结构。输入向量是时间窗，其中xi是表示时间i的每个重要KPI值的M维向量（例如，图9（a）中m＝3）。目标是预测向量xt+1，预测误差可以用来表征异常程度。在我们的模型中，我们使用整体预测误差和每个维度（单变量和多变量）的误差作为特征。在图8（j）和（k）中，我们分别绘制了业务KPI和服务器KPI的严重性得分与归一化总体预测误差之间的关系。我们可以观察到，预测误差越大，严重程度得分越高。

表格

描述已自动生成

**特征工程概要。**总之，表II总结了AlertRank中总共采用的40个特性。我们在仔细的数据分析和与经验丰富的工程师讨论的基础上设计这些特征，因此这些特征与丰富的领域知识相关联。如果AlertRank应用于其他场景，可能需要在我们的模型中引入一些新特性，但AlertRank的核心思想和管道是通用的。基于我们的特征工程研究，我们发现警报严重程度与各种特征之间的关系非常复杂（图8）。这也解释了为什么简单的基于规则的策略无法准确识别严重警报。例如，图2中的警报被划分为P2，但是KPI异常特征和警报时间会增加该警报的严重性。每种特征的有效性将在IV-C2中演示。

1. 排序模型

AlertRank提出了一个机器学习框架，在数据融合的基础上融合各种特征，并学习一个排序模型来识别严重警报。我们新颖地将此问题描述为一个排序模型，而不是二进制分类，原因如下。首先，分类（严重/非严重）给出的二分法结果容易出现假阳性或假阴性。特别是，漏报会使一些潜在的故障无法检测，从而导致服务可用性降低。然而，排名模型可以根据严重程度评分对警报进行优先级排序，并指导工程师先修复哪个警报，这样更方便用户使用。第二，排名模型能有效地处理班级失衡问题[15]。排名模型的有效性将在IV-C3中介绍。

为了训练排序模型，如III-A2中所介绍的，每个警报都会根据其解决记录自动标记为严重性分数。有三种常见的排名方法，即点式、成对式和列表式[30]。在我们的问题中，训练数据自然是逐点的（每个警报都有一个严重性评分）。另外，其他两种排序方法（如LambdaMART[31]）只提供了警报的相对顺序，不能为每个警报输出特定的严重程度分数。因此，我们采用基于回归树的XGBoost逐点排序算法[9]。XGBoost是一种基于梯度提升树的模型，被数据科学家广泛使用，并在许多问题上实现了最先进的性能[9]。

在实际应用中，警报管理系统连续接收多个流式警报，工程师倾向于定期（例如，每15分钟）对警报进行批量调查。因此，在线测试过程中，使用经过训练的排序模型，根据每15分钟的输出严重性得分对这些传入警报进行优先级排序。如果一个警报的预测严重性得分超过一个阈值，则认为该警报是严重的，并根据其在训练集上的最佳性能正确地选择了该阈值。一般来说，工程师首先检查严重性评分较高的警报。此外，正如我们在I中提到的，为了适应动态的在线服务，排名模型会定期使用最新的训练数据进行增量训练。

1. 评估
2. 数据集
3. 度量

根据输入的严重性等级，在15分钟内对输入的严重性进行排序。对于每个间隔，如果警报的严重性得分超过阈值，则将其视为严重警报。在训练集上，我们适当地选择阈值能体现性能最大化。因此，计算precision/recall/F1-score作为评估值。Precision可以体现已识别的严重警报的百分比。召回衡量正确识别的严重警报的百分比。准确度和召回率的平均值是F1。我们将所有区间的平均度量作为最终度量。此外，为了评估AlertRank（RQ3）的排名能力，我们还计算了另一个指标，precision@k，即top-k预警的准确率[15]。

1. 结果评估

多维特征起作用吗？？

AlertRank包含两种类型的特征：警报特征和KPI特征。在这个问题中，我们评估了每种类型的有效性，结果如表5所示。当使用这两种类型的特征时，我们的模型可以获得0.89的平均F1分数，精确度和召回率都很高。单是警报功能，F1的平均分数从0.89下降到0.76。但是，如果只使用KPI特性，F1的平均分数会从0.89急剧下降到0.36。这表明我们的模型从多个数据源中提取的集成特征中获益，并取得了最佳的整体效果。此外，结果还表明，预警特征比KPI特征更为强大。

1. 实践经验
2. 相关工作

A、现有工作

尽管人们在警报管理方面付出了大量努力，包括聚合[34]、相关性[35]、[36]和聚类[37]，但识别严重警报的有效方法仍然难以捉摸。[5] 提出了一种基于警报阈值之间线性关系的简单警报排序策略。然而，两个KPI之间线性关系的假设在现实中并不总是成立的。[6] 提出从历史数据中学习一种基于规则的策略来区分真实警报和不可操作警报。这两种方法都只考虑kpi警报并修改阈值策略。然而，在实际应用中，存在着各种各样的警报，如网络、数据库和日志。此外，警报的严重程度受多种因素的影响，但这些基于规则的方法无法捕捉这些因素之间的复杂关系。

识别严重警报也是入侵检测系统（IDS）[8],[38],[39]和软件工程领域的一个相关研究课题[32]，[40]。具体地说，IDS根据安全领域知识（例如源IP和目的IP）以及攻击路径对警报进行优先级排序，这与我们的场景不同。识别严重的错误报告是软件工程中的一个关键问题。这方面的大多数方法都是从错误报告中提取一些文本特征（例如频率和情感）[32]，[40]，然后应用机器学习技术来确定缺陷的严重程度。在我们的场景中，警报严重性比bug复杂得多，因为由于复杂和高度动态的在线服务环境，我们需要考虑各种因素。因此，这些方法表现不佳，这在IV-C1中得到了证明。

B、局限

本文的一个局限性是，由于篇幅的限制，我们没有提供太多关于参数选择的细节（例如，topic的数目、resolution record clusters的数目等）。研究不同参数对算法性能的影响可以作为我们今后的工作。此外，更合理、更通用的标记方法、结合特征选择技术和更自适应地选择决策阈值也是我们未来的工作。

1. 结论

大规模的在线服务系统每天都会产生大量的警报。对于工程师来说，在没有任何指导原则的情况下手动调查每个警报是非常耗时和容易出错的。在本文中，我们新颖地提出了一个基于排名的框架AlertRank，它能够自动、自适应地识别严重警报。AlertRank的一个关键组件是一组强大的、可解释的特性，用于描述警报的严重性。这些特征包括来自警报的内容和时间特征，以及来自监测指标的单变量和多变量异常特征。我们使用全球顶级银行的真实数据对AlertRank进行了评估，结果表明AlertRank是有效的，F1平均得分为0.89，大大节省了工程师研究警报的工作量。