基于深度学习的云计算系统异常检测方法

任　明[1],宋云奎[2]
(1. 中国银联股份有限公司,上海 201201;
2. 中国科学院软件研究所,北京 100190)

摘　要:在云计算服务中,异常检测是防止意外系统停机,并确保终端用户服务可靠性的关键技术。虽然操作控制台日志记录了云计算系统的运行时状态信息,但现有的云计算系统管理技术主要在出现问题后分析原因,而不能提前检测异常。因此,文中提出了一种基于深度学习的云计算系统异常检测方法。首先,提取日志模式,使用聚类将相似格式和内容的日志聚集为模式组;然后,将每个模式作为单词,将离散模式集作为文档,从而降低日志维度,得到低维度特征空间;最后,使用递归神经网络,即长短时记忆,处理训练过程中标记数据的“稀缺性”,捕获跨序列的依赖性,获得系统状态的鲁棒异常信号。使用 Web 系统日志进行实验的结果表明,与现有方法比较在检测复杂异常时具有更高的检测准确性。

关键词:异常检测;日志分析;文本挖掘;递归神经网络;云计算
中图分类号:TP301　　**文献标识码**:A　　**文章编号**:1673-629X(2019)05-0054-04
doi:10.3969/j.issn.1673-629X.2019.05.011

Anomaly Detection for Cloud Computing Systems Based on Deep Learning

REN Ming[1],SONG Yun-kui[2]
(1. China UnionPay Co.,Ltd.,Shanghai 201201,China;
2. Institute of Software,Chinese Academy of Sciences,Beijing 100190,China)

Abstract:In cloud computing services,anomaly detection is a key technology to avoid system outages and guarantee the service reliability of end users. Although operating consoles record the runtime status of cloud computing systems,existing management technologies locating the root causes of occurred faults cannot detect anomalies in advance. Therefore,we propose a deep learning-based anomaly detection approach for cloud computing systems. First,we cluster logs to groups according to the format and content of logs,and then extract execution patterns from the clusters. Second,we regard a pattern as a word and a pattern set as a set to lower the dimensionality of features in logs. Third,we use a recurrent neural network to address the sparsity of labelled data instances and the dependencies between series,and then define robust anomaly signatures and detect anomalies. We use the logs of a web system to validate the proposed approach,and the experimental results demonstrate that this approach has higher precision than existing ones in detecting complex anomalies.

Key words:anomaly detection;log analysis;text mining;recurrent neural network;cloud computing

0　引　言

随着计算技术的发展,大量在线服务和关键任务依赖于异构的云计算系统来完成,最小化这些系统的停机时间非常重要[1]。控制台日志记录了云计算系统的操作状态和事件,并且具有丰富的描述性信息。当前,日志分析的相关研究主要集中在系统异常检测与诊断领域,目的是快速检测出异常发生时的信号,并确定异常的根本原因[2]。云计算环境下,基于日志的异常检测的挑战是实现在可接受的性能条件下,处理分析大量系统特征[3]。另一个挑战是,需要面对异构计算系统以处理大量异构日志信息[4]。

为了解决以上两个问题,提出一种基于文本挖掘和深度学习的异构日志分析方法,从控制台日志中提取通用特征,建模为时序深度神经网络,以执行异常检测。首先从异构日志中学习日志格式,将相似的日志聚在一起,并提取日志集合的模式。然后,基于这些模

收稿日期:2018-06-07　　**修回日期**:2018-10-10　　**网络出版时间**:2018-12-21
基金项目:国家自然科学基金(61602454)
作者简介:任　明(1980-),男,工程师,研究方向为云计算智能运维。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20181221.1554.042.html

式提取随时间推移的顺序特征,以缩减特征维度。最后,将异常检测抽象化为序列分类问题,编码日志特征并映射到低维向量空间中,通过 LSTM(long short-term memory)[5]进行异常检测。

1　异常模式识别

控制台日志通常是由不同的应用程序或服务生成,因此日志具有异构性,表现为多样化和不均衡的单词分布。这就使得传统的文本挖掘方法(如主题建模)从云计算系统的控制台日志中提取有意义的特征面临挑战[6]。虽然控制台日志记录了云计算系统的健康状态信息,但现有的系统管理技术主要在出现问题后分析错误[7]。控制台日志通常是由应用程序源代码中定义的模板生成,因此具有预定义的格式[8]。此外,在应用程序运行过程中,日志常常是多余的。找到规则的格式来表示一组类似的日志有助于减少冗余,不丢失重要信息,并总结日志数据的含义。现有工作分析源代码以建立日志的常规格式,但该类方法只能在源代码应用,不能将不同编程语言和日志样式的应用程序的异构日志进行混合。文中提出了一种通用的日志模式学习方法,以捕获日志结构和每个日志字段的语义。模式定义为日志的语法结构,即具有相似格式和内容的日志可以归为一种模式。自动识别控制台日志语法结构的步骤包括:

(1)日志信息和时间戳标准化。

对日志数据进行标记以识别和检索每个日志记录的单词或短语的基本信息。但是,来自不同应用程序和系统的异构日志有不同的格式以及分隔符。如果没有特定的知识或人工检查,为所有异构日志数据集预先定义相同的分隔符会很不公平,因此,应该使用通用的分隔符,以避免符号间的干扰。文中将空格作为分隔符,用来分开除了数字之外的所有单词和特殊符号。异构日志可以有许多不同类型的时间戳格式,文中在日志中检测所有的时间戳并将其转换为标准格式(yyyyy/MM/DD HH:MM:SS)。

(2)日志聚类。

由于没有日志格式、用法和来源等方面的领域知识,理解和分析异构日志首先需要理解日志数据的语法结构。聚类算法基于数据内在属性和关系,对数据实例进行分类。因此,文中将聚类算法应用于异构日志,以获得数据的初始化“视图”。采用分层聚类生成异构日志的层次结构,提供了多粒度的数据视图,根据位置将日志从粗到细粒度组织成树结构。同时,数据索引和搜索是建立在分层树结构基础上,以达到提高效率的目的。文中使用的分层树结构利用 OPTIC[9]聚类方法。OPTIC 通过从一个特定的数据点向所有邻近的数据点扩展,从而搜索密集的数据区域,这些数据点在一个预定义的阈值下足够接近。聚类算法根据数据点排序生成层次化的聚类结构,将较为稀疏的数据区域内的密集数据区域作为聚类,形成较稀疏区域的子聚类。OPTIC 具有两个参数 eps 和 min-points,其中 eps 指定聚类的最大宽度,min-points 控制有效聚类需要包含的最小样本数量。

(3)模式识别与匹配。

在对日志数据进行聚类之后,生成异构日志的整体语法结构,但仍然需要在每个聚类中获得具体模式。由于在每个聚类中,日志记录具有相似的格式,在聚类中使用序列比对进行模式识别。模式识别首先在叶节点中完成,然后从叶子向后传播到根节点。在生成日志模式后,需要对输入的异构日志进行解析,文中将这些模式表示为正则表达式。任何输入日志都将与提取的日志模式,即正则表达式匹配,如果不能匹配,则生成异常值。

(4)特征表示。

使用提取的模式解析输入日志,将日志映射到一个模式。文中提取模式的集合,计算每个模式的频率,而不是简单搜集时间间隔 $(t, t+\Delta t)$ 的日志数据。选择合适的时间间隔 Δt,首先需要降级特征表示的稀疏性,同时使用较小的时间粒度来进行更精细的检测。借鉴 TF-IDF 思想,从日志中提取合适的特征。TF-IDF 在信息检索和文本挖掘中表示文档的特征。文中将每种模式作为一个词,而发生在时间阶段 $(t, t+\Delta t)$ 之间的众多模式作为一个文档。

$$\mathrm{TFIDF}(p_i, e_t, E) = \mathrm{TF}(p_i, e_t) \times \mathrm{IDF}(p_i, E) \tag{1}$$

$$\mathrm{TF}(p_i, e_t) = \begin{cases} 1 + \log(f(p_i, e_t)), f(p_i, e_t) > 0 \\ 0, f(p_i, e_t) \leqslant 0 \end{cases} \tag{2}$$

$$\mathrm{IDF}(p_i, E) = \log(|E| / (1 + |p_i \in E, f(p_i, e_t) \neq 0|)) \tag{3}$$

其中,p_i 为模式;e_t 为时间周期模式集合;E 为全部时间监测集合;$f(p_i, e_t)$ 为模式 p_i 在时间周期 e_t 中出现的频率;$|E|$ 为时间周期的数量;$|p_i \in E, f(p_i, e_t) \neq 0|$ 为出现模式 p_i 的时间周期的数量。

2　异常检测

云计算系统异常检测问题定义为:给定云计算系统组件为 K,控制台日志集合为 $L(K)$,推断在时间窗口 W 内发生异常的概率为 $P(W)$。输入是长度为 L 的历史特征序列:$(x_{t-L+1}, \cdots, x_{t-1}, x_t)$;目标是二元向量 d_t:在 t 时刻的检测期内发出报警以及不在检测期内未发出警报取值为“P”,否则取值为“N”。学习模型输出报警概率,如果这种概率超过预先定义的阈值,

就会发出报警。文中将该问题抽象为基于监督学习的二元分类问题,其中监督学习模型的训练数据包括从各种云计算系统的控制台日志中提取的特征和系统管理员提供的异常标签,检测任务的主要目标是在异常发生之前发出警报。将异常检测过程分为预测期和感染期两个阶段。预测期是在异常发生之前预定义的时间段,从报警时间到异常开始时间,即定义的时间窗口 W,在此期间给管理员发出警报,以提供足够的时间来采取行动。感染期是在异常发生之后预先定义的时间,从组件发生异常时会持续到问题被修复,恢复到正常状态。

将历史特征向量序列作为输入,使用检测模型对当前特征向量进行分析,输出即将发生异常的概率。如果概率超过了预先定义的阈值,那么在不久之后就会发出预警信号。早期的警报信号通常很弱,因此很难使用简单的模型来捕获。计算机系统具有时序动态性,随后的状态取决于长期的历史趋势。传统的监督学习方法,如逻辑回归、SVM 和基于树的分类器,仅仅将输入序列视为独立的特性,不能捕获输入之间的时间依赖关系。文中应用递归神经网络(recurrent neural network,RNN)以生成和标记序列。网络的内部状态不仅依赖于当前的输入,而且还依赖于以前的系统状态。然而,传统的 RNN 不能存储很长时间以前的输入信息,削弱了其建模输入序列长期结构的能力。LSTM 使用 RNN 架构,并改进存储和访问信息,通过引入内存单元[10]来存储以前的时间步骤的信息,以解决长期依赖关系的问题。由于计算系统异常检测存在很强的时间依赖性,文中基于 LSTM 网络以建模计算机系统的动态性。

在递归神经网络的检测架构中,输入特征向量序列 $x=(x_{t-L+1},\cdots,x_{t-1},x_t)$ 的序列长度为 L^2,传递给由多个循环加权连接构成的隐藏层,以计算隐藏向量序列 $h=(h_{t-L+1},\cdots,h_t)$,输出向量序列 $y=(y_{t-L+1},\cdots,y_t)$。输出向量 y_t 可以用来参数化目标 d_t 的概率分布 $\Pr(d_t \mid y_t)$。在 LSTM 内存单元的结构中,内存单元的访问由“输入”、“输出”和“遗忘”门组成。存储在内存单元中的信息可以在 LSTM 中获得比传统 RNN 更长的时间,这使得模型更加关注于上下文。问题形式化为二分类问题,即目标 d_t 是二元向量以表示两类数据。检测网络的输出 y_t 是一个二进制向量,以表示系统状态,文中使用其估计二项分布 $\Pr(d_t \mid y_t)$,可以通过输出层的 Softmax 函数来参数化:

$$\Pr(d_t=k \mid y_t)=\frac{e^{y_t^k}}{\sum_{i=1}^{K}e^{y_t^k}} \tag{4}$$

对于目标函数,文中使用二进制的交叉熵代价函数训练:

$$C=-\sum_{i=1}^{K}w_k\times[d_t^i\log(y_t^i)+(1-d_t^i)\log(1-y_t^i)] \tag{5}$$

其中,$K=2$ 是类的数量;目标 d_t^k 解码为 1 或 0;w_k 是 k 类的权重。

3 实 验

3.1 实验设置

实验数据集来自某在线交易系统中 Web 服务器集群的日志记录。每个集群由多个组件组成,包含各种类型的应用程序。当系统管理员发现问题时,会记录该系统异常,分散在整个监测期间。首先将历史日志的时间序列离散化,其中每个时间段($\Delta t=10$ 分钟)的日志作为一个文档。将数据集划分为时间顺序的训练集和测试集,其中训练集包括前 3/5 时间,其余 2/5 时间用于测试。设置参数 eps=0.14 和 min_points=8,设定提取模式的正则表达式模式,然后对每个日志匹配模式。构建基于模式的 TF-IDF 特征向量,利用特征向量来检测系统异常。在训练阶段,所有检测期的实例都被认为是正常的,而丢弃出现异常的实例。

文中将深度学习方法(即 LSTM)与监督学习方法,包括逻辑回归、支持向量机和随机森林进行实验对比。逻辑回归线性搜索惩罚参数,并报告最佳性能;SVM 使用线性内核的 LIBSVM 包[11];随机森林设置树的数目和随机抽取变量的数量来构建树[12]。文中使用 LSTM,首先建立相对较小的 LSTM 网络,每个层有 2 个隐藏层和 24 个隐藏单元[13]。将所有的权重参数均匀初始化在范围[-0.1,0.1],同时初始化 LSTM 遗忘门,设置偏差值为 1.2。然后,用批量大小为 3 的预参数自适应更新,使用小批量随机梯度下降对网络进行训练,基础学习速率为 10,衰变因子 0.9。对每个模型进行 15 个周期的训练,并将其与衰减因子 0.9 相乘,每增加一个新时期,将基础学习速率乘以 8。由于数据集高度不平衡,损失函数设为 0.95。考虑以下三个性能评价指标:

(1)AUC:等于精确率和召回率乘积,计算(精确率,召回率)对,可以得到 AUC 曲线,曲线下方的面积越大,表明准确率和召回率越高,方法效果越好。

(2)检测间隔:在检测期间,模型最早能够正确报警的时间,警报和异常的起始时间之间的时间差越大,系统管理员能够诊断和防止即将出现的异常的可能性就越大。

(3)检测频率:在检测期间,所检测并报告为警报的时间比例,更高频率报警,确定检测为异常的概率越高。

3.2　实验结果

图 1 给出了不同序列长度下精确率和召回率曲线下的面积。实验结果表明,LSTM 优于传统的 logistic 回归、SVM 和随机森林。同时,序列长度从 1 增加到 2 对异常检测效果有很大提高,但更长的序列长度并不能大幅度提升模型。因为传统模型假定输入特性序列之间是独立的,不考虑序列中各点之间的依赖关系,而 LSTM 能够捕获特征时间序列之间的时间依赖关系。

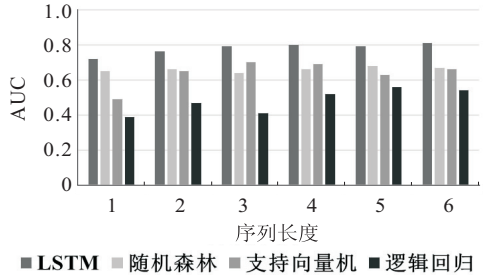

图 1　多种异常检测方法效果比较

图 2 给出了随着序列长度的增加,LSTM 检测效果的变化。实验结果表明,较长的序列并不能提高异常检测的性能,这是由于早期预警信号可能与之前的几个邻近阶段相关。平均可检测间隔的大小表明,早期的警报信号发生在异常开始接近的时间。

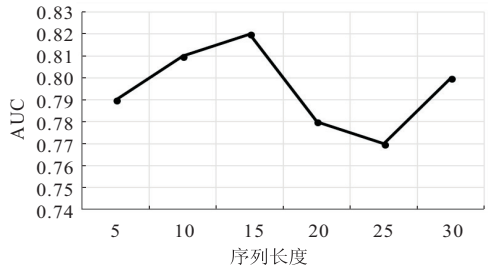

图 2　不同序列长度的检测效果变化

图 3 给出了随着训练时间的增加,训练的学习曲线和检验性能的变化。实验结果表明,测试的性能在训练的第 20 个阶段达到了顶峰,因此,需要监测训练性能的变化,及时停止训练,以避免模型过度拟合。

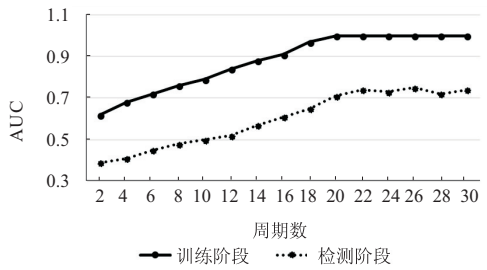

图 3　异常检测性能变化

4　结束语

提出了一种云计算系统的日志驱动的异常检测方法,从流式日志数据中自动提取系统状态特征,发现历史数据中的日志语法结构,并通过 LSTM 方法实现早期的异常检测。下一步将研究在线深度学习机制,使用不断增加的异常事件来动态更新模型。

参考文献:

[1] 王力群,黄必栋. 基于日志分析平台的监控系统的设计与实现[J]. 计算机应用与软件,2017,34(12):158-162.

[2] YUAN Ding, MAI Haohui, XIONG Weiwei, et al. Sherlog: error diagnosis by connecting clues from runtime logs[J]. ACM SIGARCH Computer Architecture News, 2010, 38(1):143-154.

[3] KIM C H, RHEE J, ZHANG Hui, et al. Introperf: transparent context-sensitive multilayer performance inference using system stack traces[J]. ACM SIGMETRICS Performance Evaluation Review, 2014, 42(1):235-247.

[4] XU Wei, HUANG Ling, FOX A, et al. Detecting large-scale system problems by mining console logs[C]//Proceedings of 22nd symposium on operating systems principles. Big Sky, Montana, USA: ACM, 2009:117-132.

[5] GERS F A, SCHRAUDOLPH N N, SCHMIDHUBER J. Learning precise timing with lstm recurrent networks[J]. Journal of Machine Learning Research, 2003, 3(1): 115-143.

[6] KIMURA T, ISHIBASHI K, MORI T, et al. Spatio-temporal factorization of log data for understanding network events[C]//Proceedings of conference on computer communications. Toronto, Canada: IEEE, 2014:610-618.

[7] SIPOS R, FRADKIN D, MOERCHEN F, et al. Log-based predictive maintenance[C]//Proceedings of 20th SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: ACM, 2014:1867-1876.

[8] VAARANDI R. A data clustering algorithm for mining patterns from event logs[C]//Proceedings of IEEE workshop on IP operations and management. Kansas City, MO, USA: IEEE, 2013:118-126.

[9] ANKERST M, BREUNIG M M, KRIEGEL H P, et al. Optics: ordering points to identify the clustering structure[C]//Proceedings of SIGMOD international conference on management of data. New York, NY, USA: ACM, 1999:49-60.

[10] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural Computation, 1997, 9(8):1735-1780.

[11] CHANGC C, LIN C J. Libsvm: a library for support vector machines[J]. ACM Transactions on Intelligent Systems and Technology, 2011, 2(3):27-36.

[12] 王梓杰,周新志,宁　芊. 基于 PCA 和随机森林的故障趋势预测方法研究[J]. 计算机测量与控制,2018,26(2):21-23.

[13] SUTSKEVER I, VINYALS O, LE Q V. Sequence to sequence learning with neural networks[C]//Proceedings of neural information processing systems. New York, NY, USA: ACM, 2014:3104-3112.