doi:10.11887/j.cn.202205001 http://journal.nudt.edu.cn

高性能互连网络中端口阻塞故障预测方法*

徐佳庆,胡小豚,杨汉芝,王 强,张 磊,唐付桥
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:随着系统规模、芯片功耗和链路速率的提升,高性能互连网络的整体故障率也不断上升,传统运维方式将难以为继,给高性能计算系统整体可靠性和可用性带来了巨大挑战。针对网络端口阻塞这类严重网络故障,提出无监督算法的预测模型。该模型从历史信息中挖掘征兆性规律并形成新的特征向量,应用*K*-means聚类算法对特征向量进行学习归类。在预测时,结合端口当前状态,利用二次指数平滑算法对未来状态进行预测,将得到的新特征向量使用*K*-means算法预判是否会发生阻塞故障。利用拓扑结构信息,分别对叶交换机和根交换机构建预测子模型,进而提升预测的精确率。结果表明,该预测模型能保持在召回率为88.2%的前提下,达到65.2%的准确率,可为运维人员提供有效的辅助。

关键词:互连网络;故障预测;机器学习
中图分类号:TP391 **文献标志码**:A **文章编号**:1001-2486(2022)05-001-12

Prediction method of port blocking failure in high performance interconnection networks

XU Jiaqing, *HU Xiaotao*, *YANG Hanzhi*, *WANG Qiang*, *ZHANG Lei*, *TANG Fuqiao*
(College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China)

Abstract: With the increase of system scale, chip power consumption and link rate, the overall failure rate of high-performance interconnection networks will continue rising, and the traditional operation and maintenance methods will be difficult to sustain, which brings great challenges to the overall reliability and availability of HPC(high performance computing). An unsupervised algorithm prediction model for serious network failures such as network port blocking was proposed. In this model, the symptomatic rules were extracted from the history information of the switch port status register and a new feature vector was formed. The *K*-means clustering algorithm was used to learn and classify the feature vectors. In the prediction, the DES(double exponential smoothing) algorithm was used to predict the port state in the future through a combination of the current state of the port, and a new feature vector was obtained and *K*-means algorithm was used to predict whether the port blocking failure would occur. The topology information was used to build independent sub prediction models with ToR switch ports and Spine switch ports respectively, so as to further improve the accuracy of prediction. The experimental results show that the prediction model can maintain the recall rate of 88.2%, and reach the accuracy rate of 65.2%. It can provide effective early warning and guidance for the operation and maintenance personnel in the actual system.

Keywords: interconnection network; failure prediction; machine learning

高性能计算机是指具有极快运算速度、极大存储容量、极高通信带宽的一类计算机,主要应用于大科学、大工程以及产业升级等领域,对国家安全、经济和社会发展具有举足轻重的意义,是国家科技发展水平和综合国力的重要标志。[1]为满足科研和生产活动对更高计算能力的需求,高性能计算机的性能发展遵循着“千倍定律”,即每过十年超级计算机的性能会提升一千倍。截至2022年6月,这类计算机最高运算速度已达到每秒百亿亿次浮点运算[2]。

高性能互连网络[3-4](high performance interconnection networks)是高性能计算机中最重要的全局性基础设施,相当于高性能计算机的神经系统,是实现系统中各类结点高速协同并行计算的关键,直接影响着系统的性能。高性能互连网络主要由高性能网卡(high performance adapter)、高阶交换机(high-radix switch)和高速链路(high-speed link)构成。尽管单个互连构件的故障概率很低,但随着系统规模的扩大和链路速率的提升,高性能互连网络的整体故障率将会不

* **收稿日期**:2020-11-08
基金项目:国家重点研发计划资助项目(2018YFB0204300);并行与分布处理国防科技重点实验室基金资助项目(6142110180101)
作者简介:徐佳庆(1982—),男,浙江德清人,助理研究员,博士,E-mail:xujiaqing@nudt.edu.cn

断上升[5],给高性能互连网络的日常运维带来了巨大的挑战,传统的自动化运维将难以应对。因此,需要引入机器学习算法自动地从海量运维数据中不断地学习,不断地提炼并总结故障规律,加速互连故障的诊断与预测,从而提高运维效率,提升整个高性能计算(high performance computing, HPC)系统的可用性。

在高性能互连网络运维中,由链路质量恶化引发的网络端口阻塞是一类故障定位复杂、故障影响范围大的严重故障。一旦发生网络端口阻塞:轻则会导致网络中的丢包率增加,端对端延迟增加;重则会造成整个网络瘫痪,严重影响整个系统的可靠性。当发生了网络端口阻塞故障时,运维人员通常需要将系统中正在运行的作业挂起,利用测试程序确定当前系统中不可达的结点对,然后通过路径查询工具获取结点对之间的公共路径,依次对这些公共路径经过的网络端口进行端口复位,从而确定发生故障的端口位置,最后往往通过更换故障光纤或光模块的方法来有效消除网络端口阻塞故障。整个故障定位的过程较为复杂,耗时较长,严重影响整个系统的可用性。若运维工作人员能提前获取网络端口阻塞故障的预测结果,就可以从以下两个方面提升 HPC 系统整体的可用性:

1)资源分配。在作业提交前,管理人员主动将作业分配给所在链路更健康的结点区域,以防因为链路故障而导致作业运行失败。

2)故障规避。对于存在阻塞隐患的链路,管理人员可以主动采取误路由的方法,在不影响作业正常运行的情况下提前对该隐患端口进行排查和处理,从而确保系统中所有链路均健康可用。

智能运维(artificial intelligence for IT operations, AIOps)的概念由 Gartner 于 2016 年首先提出的,是将人工智能应用于运维领域,基于已有的运维数据(日志、监控信息、应用信息等),通过机器学习的方式设计故障预测模型寻找故障发生前的规律,推测出系统在未来一段时间的运行状况,在故障发生之前进行预判,帮助运维人员提前采取一些有效措施规避故障,提高系统的可靠性及稳定性[6-13]。

数据中心(data center, DC)系统或 HPC 系统通常由计算、存储、网络三大部分构成。近年来,许多研究人员分别开展了针对计算、存储、网络的故障预测研究,致力于提高系统的整体可用性。在计算故障预测方面,在文献[14-18]中,研究人员建立了计算结点故障预测模型,虽然取得不错的效果,但都是基于软件层面上对故障进行分析预测,对基础性硬件故障也无法提供实质性指导。在文献[19]中,孙勤以"天河一号"超级计算结点运行状态数据集为基础,采用改进的 ReMAE 算法结点状态数据进行故障预测,召回率要高出其他集成式数据流挖掘算法 37%~50%,提高了对即将故障状态的预测的准确率。刘睿涛[20]依托神威系统,采用基于带时间标签多序列的故障预测算法对结点 CPU 等部件故障进行了预测分析,该方法虽然准确率达到了 60%~99%,但召回率偏低,在实际运维过程中容易出现漏报的情况。在存储故障预测方面,文献[21-22]均采用了近似于分类和预测相结合的两步模型对大数据中心磁盘故障进行预测。其中分类模型通过对历史故障数据的分析获取磁盘状态认知,结合预测模型对未来数据进行判定来表征磁盘是否存在故障。该方法更多地利用了数据的特性来避免标准有监督学习中的混淆效应,有效地提高了磁盘状态甄别的准确率。在网络故障预测方面,文献[23]提出了基于日志文件的网络故障预测方法,通过提取日志文件中的隐含序列对故障发生时间进行预判。该种方法不仅可以对多种交换机进行预测,还能增量学习新的特征,但基于日志文件的预测方式只能间接地反馈网络状态,缺乏对网络物理组件本身状态的认知。

本文针对网络端口阻塞故障,提出了一种无监督的分类算法 *K*-means 算法和时间序列算法二次指数平滑(double exponential smoothing, DES)算法相结合的预测模型,通过从交换机端口状态寄存器的历史信息中挖掘出征兆性规律并形成新的特征向量,应用 *K*-means 聚类算法对特征向量进行学习归类。在预测时,结合端口当前状态,利用 DES 算法对未来一段时间的端口状态进行预测,将得到的新特征向量使用 *K*-means 算法预判是否会出现端口阻塞故障。

1 相关背景

系统 A、系统 B 和系统 C 是三个处于运行状态的超级计算机,其在线运行时间、互连网络规模和链路速率分别如表 1 所示:其中系统 A 的服役时间最长;系统 B 的互连网络规模最大;系统 C 的部署时间最晚。

根据故障的不同性质,可将互连网络的故障分为软件故障和硬件故障,其中硬件故障又可分为交换机故障、网卡故障及链路故障三类。由于各系统部署时间不同,运维数据统计的时间跨度

表 1　系统参数对比

Tab. 1　Comparison of three supercomputers

系统	上线时间/a	链路速率	交换机数量	链路数量	网卡数量
A	>9	QDR	>220	>3 100	>300
B	>6	FDR	>200	>46 000	>1 800
C	>4	EDR	>310	>3 200	>11 300

也有所不同。系统 A 统计了 2015 年 12 月至 2018 年5 月的互连故障数据;系统 B 统计了2017 年1 月至 2018 年 5 月的互连故障数据;系统 C 则统计了 2017 年 1 月至 2018 年 6 月的互连故障数据。三个系统各类互连故障的比例情况如表 2 所示。硬件故障在三个系统中均占据互连故障的 90% 以上。其中网卡故障的比例较小,主要集中于交换机故障和链路故障。在系统 A 中,交换机故障达到了 81.05%,而链路故障仅为 10.53%,其原因是该系统使用了 QDR 光纤,随着系统服役时间的增长,电子元器件的老化导致了交换机故障增多;而系统 B 和系统 C 则使用了 FDR 光纤和 EDR 光模块,其链路故障的比例分别达到了 76.61% 和 61.94%。从中不难发现,随着系统规模的增大以及链路速率的提升,链路故障已成为互连网络中最主要的一类故障,给互连网络的维护带来了极大的挑战。

表 2　不同互连网络故障比例

Tab. 2　Failure ratio of different interconnection networks

系统	网络相关故障的比例/%		硬件故障比例/%		
	软件故障	硬件故障	网卡故障	交换机故障	链路故障
A	6.86	93.14	8.42	81.05	10.53
B	0.53	99.47	0.46	22.93	76.61
C	3.60	96.40	6.72	31.34	61.94

2　模型设计以及相关原理

2.1　模型概况

研究目标是预测系统在正常运行时在未来一段时间内某个交换机端口是否出现网络阻塞现象。对于发生在 T_h 时刻的故障,期望能在 T_s 至 T_e 这段时间内预测出故障即将发生。$\Delta\tau_a$ 是 T_e 距离 T_h 的间隔,为运维人员用来处理故障的最短时间。

在离线学习过程中,假设在 T_h 出现故障,对于$[T_s, T_e]$中的任何时刻 T_x,$[T_x - \Delta\tau_m, T_x]$中的消息序列被标记为征兆性规律。$T_y \in [T_s, T_h]$时,$[T_y - \Delta\tau_m, T_y]$中的消息序列被标记为非征兆性规律,如图 1 所示。希望通过捕捉交换机端口出现故障前的规律性变化来预测网络阻塞是否发生。为此,将使用机器学习方法建立一个基于交换机端口网络阻塞故障历史数据集的预测模型,然后使用该模型来预测全系统交换机端口是否在未来会发生网络阻塞故障。在预测模型的设计过程中需要面对以下技术挑战。

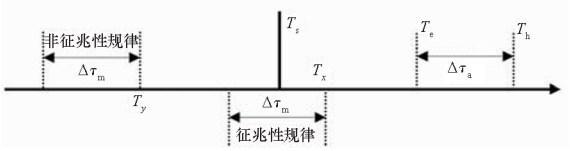


图 1　故障预测模型

Fig. 1　Model of failure prediction

1)数据不均衡。将以第 1 节中提及的系统 C 作为研究对象,该系统有近 18 000 个网络端口。而统计数据表明,该系统日均发生网络阻塞故障的端口不到 2 个,如表 3 所示。这种极度分布不均衡的数据集,让模型训练的难度大大增加。由于这种数据不平衡性的存在,预测模型大概率偏向于判断网络端口在未来一段时间处于健康状态。虽然可使用数据再平衡技术(如欠采样和过采样技术等)来解决这一挑战,但这些方法在提高召回率的同时也可能引入大量的误报,从而大大降低预测的准确性。

2)数据特征不明显。不同于基于以太网的交换机、服务器等物理组件提供的日志文件,它本身并不涉及设备本身的物理状态,而采集的数据反映的却是交换机本身底层的状态。这也导致正常样本和异常样本在原始数据表征上没有明显差异。

图 2 展示了预测模型的基本框架。整体分为离线训练和在线预测两大块。在离线训练中:第一步是从历史数据集中提取出有用特征向量序列作为新的训练集。第二步是利用训练集分别对聚类组件和预测组件进行学习训练。聚类组件通过学习故障端口阻塞故障发生前 2 d 和正常端口任

表3　网络阻塞故障示例

Tab. 3　Example of network blocking failures

故障时间	端口号	对端位置	错误信息
2019-03-13T12:44:00	280.26	389.01	sub_value:532,credit_used_vc0:532
2019-03-13T23:43:00	241.24	415.19	sub_value:532,credit_used_vc0:533
2019-03-14T09:10:00	270.27	403.00	sub_value:532,credit_used_vc0:534
2019-03-15T08:56:00	032.20	441.07	sub_value:541,credit_used_vc0:541
2019-03-15T10:19:00	279.27	404.04	sub_value:535,credit_used_vc0:535
2019-03-16T10:19:00	388.01	280.29	sub_value:533,credit_used_vc0:533
2019-03-16T13:53:00	449.11	042.19	sub_value:552,credit_used_vc0:552

意2 d的状态变化,找出网络阻塞这一过程在不同阶段状态的差异,并进行归类。预测组件则对网络阻塞端口从正常状态到故障发生前一刻的不同特征值进行学习训练,进而预测出在未来某个时间节点T_h网络端口状态。当离线模型训练完成后,在线预测组件结合某个端口当前的状态X_t,利用离线模型中的预测组件对时间节点T_h的状态进行预测,再依托聚类组件判断该端口的状态,判断是否会出现网络阻塞,给运维工作人员提供指导。

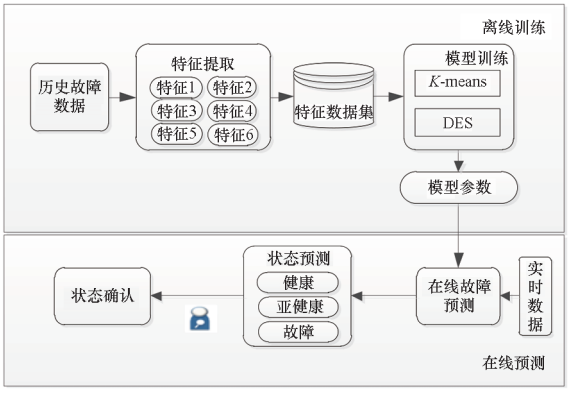


图2　整体模型构架

Fig. 2　Overall model framework

2.2　*K*-means的聚类算法

交换机端口发生网络阻塞故障是一个渐变的恶化过程,但该过程如何演变尚未彻底弄清,而无监督的聚类算法非常适合分析识别数据对象的内在关系,可以辅助科研人员揭露数据的真实变化情况。

K-means算法是一种经典的基于距离的聚类算法,采用距离作为相似性的评价指标,即认为两个对象的距离越近,其相似度就越大。而相似度是利用各聚类中对象的均值获得一个“中心对象”(引力中心)来进行计算。具体实现如下:首先从所有样本对象中选择出K个元素作为最开始的聚类目标$m_i(i=1,2,\cdots,K)$,结合式(1),计算数据集中每个样本到K个聚类目标的距离d_i,在找到样本的最小距离d_i后,将该样本归入与m_i相同的目标类中。

$$d(i,j)=\sqrt{(x_{i1}-x_{j1})^2+\cdots+(x_{in}-x_{jn})^2} \tag{1}$$

式中:$i=(x_{i1},x_{i2},\cdots,x_{in})$和$j=(x_{j1},x_{j2},\cdots,x_{jn})$是两个$n$维数据对象。遍历完所有对象后,利用式(2)重新计算m_i的值,作为新的聚类目标。

$$m'_K=\sum_{i=1}^{N}(x_i/N) \tag{2}$$

式中:m'_K为第K个聚类目标,N代表第K个簇中数据对象的个数。按照新的聚类目标将整个数据集中的对象重新归类。反复进行这个过程直至平方误差准则最小。定义平方误差准则如下:

$$E=\sum_{i=1}^{K}\sum_{p\in C_i}|p-m_i|^2 \tag{3}$$

式中:E表示所有对象的平方误差的总和,p代表训练集中的样本,m_i表示聚类目标C_i的平均值。

如上所述,利用*K*-means聚类算法可以用于多分类问题。可以把交换机端口状态的判断看作是分类问题。也就是说,如果只想评估端口是否可用,可以将其简化为一个分类问题。因此,*K*-means聚类算法可以在无监督的情景下实现端口状态的智能识别,并且该算法易于实现,时效性高,非常适合本文场景。

2.3　DES预测算法

如上所述,*K*-means算法提供了一种利用多个指标评估设备状态的有效方法,但不能独立进行预测。结合*K*-means算法的预测方法如图3所示,提取了交换机端口从$t-n$时刻到t时刻的不同特征的状态值,需要采用一种算法预测出$t+T$时刻的各个特征值,再结合*K*-means算法判断$t+$

T 时刻的设备故障状态。

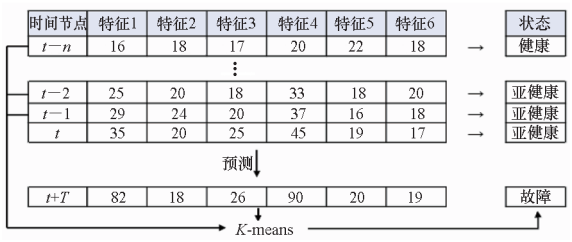


图 3　结合 *K*-means 的预测方法

Fig. 3　Prediction method combine with *K*-means

使用的特征数据是网络端口不同寄存器状态值在不同区间内的分布数量,它们的变化趋势近似于一条连续的曲线,短期趋势可以预测的。另一方面,当一个交换机端口接近故障时,其指标相比于正常情况下波动较大。因此,适合使用 DES 算法来完成这项工作。DES 算法是一种时间序列预测算法,主要用于短时预测,主要针对的是存在变化趋势但没有季节性规律的序列,比较适合本文的讨论场景。该算法是一种改进的指数平滑算法,能更灵敏地识别数据的变化,适用于变化较大的时间序列。DES 算法的主要特点是它对单个指数平滑结果进行指数平滑,如式(4)和式(5)所示。

$$S_t^{(1)} = ay_t + (1-a)S_{t-1}^{(1)} \tag{4}$$

$$S_t^{(2)} = aS_t^{(1)} + (1-a)S_{t-1}^{(2)} \tag{5}$$

其中:$S_t^{(1)}$ 代表 t 时刻的一次指数平滑值,$S_t^{(2)}$ 代表 t 时刻的二次指数平滑值,y_t 代表 t 时刻的实际值,$a \in (0,1)$ 为平滑系数。

$S_t^{(1)}$ 和 $S_t^{(2)}$ 在式(6)到式(8)中被用来计算在 $t+T$ 时刻的预测值 Y_{t+T}。

$$Y_{t+T} = a_t + b_t \cdot T \tag{6}$$

$$a_t = 2S_t^{(1)} - S_t^{(2)} \tag{7}$$

$$b_t = \frac{a}{1-a}(S_t^{(1)} - S_t^{(2)}) \tag{8}$$

选择 DES 算法结合 *K*-means 聚类算法来进行系统的网络阻塞故障预测。在这种方法中,先使用 DES 算法来预测每个特征在 $t+T$ 时刻的值 Y_{t+T},再利用 *K*-means 算法来判断 $t+T$ 时刻的系统状态。

3　实验与结果

3.1　实验系统介绍

系统 C 为本次实验的目标系统,该系统采用了 6 行 ×30 列的二维胖树(2D-Tree)拓扑结构,系统中共有 180 台叶交换机(ToR switch)和 132 台根交换机(Spine switch),网络拓扑如图 4 所示。叶交换机有 72 个网络端口,根交换机有 36 个网络端口。其中根交换机根据位置不同又可分为行根交换机(row spine switch, RSS)与列根交换机(column spine switch,CSS)。值得注意的是,在使用的数据集中,根交换机端口与叶交换机端口发生网络阻塞故障的比例为 1.6∶1,网络阻塞故障更多集中出现在根交换机端口上。

涉及的数据是基于自研高性能互连网络的带内管理机制对系统 C 进行采集的,采集时间从 2019 年 1 月持续到 2020 年 1 月。以全系统交换机的在用端口为对象,以 10 min 为采样间隔,对各端口的握手、重传、信用、流量等 12 个特征值进行收集。该系统所有交换机共有 17 712 个端口,每个

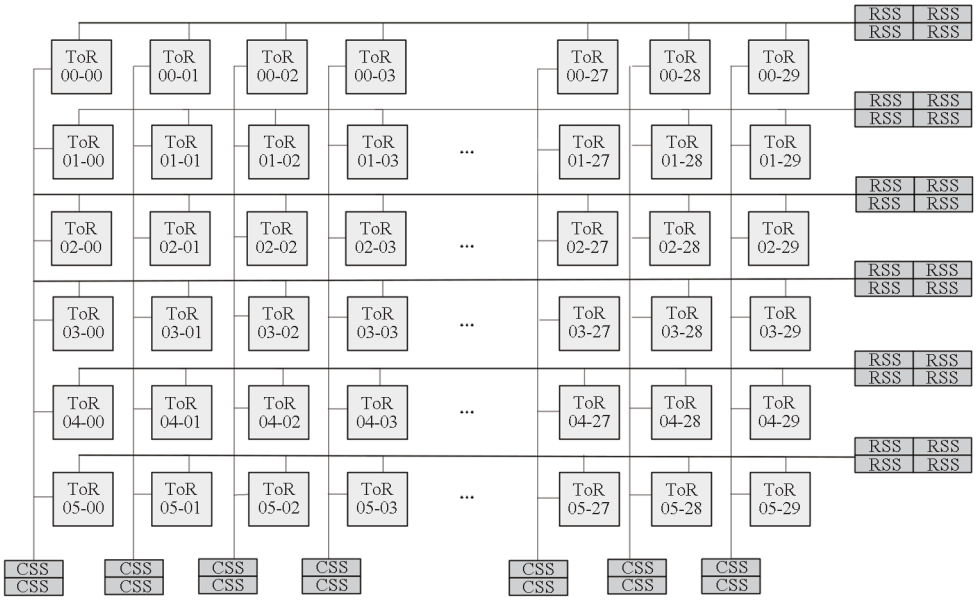


图 4　互连网络拓扑系统

Fig. 4　Interconnection networks topology system

原始样本时间跨度为1周。因此总的数据量有近93万条。表4展示了编号为100的交换机的24号端口在2019年10月12日上午部分时间段的原始数据详情。为了保证数据的时效性,运维人员对每一次网络阻塞故障发生的时间进行了准确记录。

前文提到数据集存在样本不均衡的问题,如果训练集中正常数据集远大于故障数据集,则预测精度会很高,但也会使得实际部署时产生过多的漏报。因此剔除了大量正常数据集,重点关注故障数据集,使得正常数据和异常数据尽可能平衡。

表4 原始数据示例

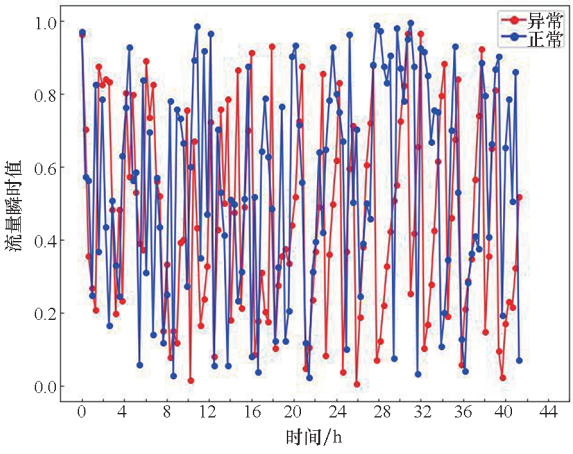
Tab. 4　Example of original data

时间	PORT	CREDIT_R	CREDIT_T	VC0_R	VC0_T	VC1_R	VC1_T	VC2_R	VC2_T	FLIT_R	FLIT_T	HANDUP	RETRY
09:06:43	H100_P24	3.76×10^9	1.21×10^9	3.54×10^9	2.58×10^9	1.46×10^9	3.52×10^9	1.09×10^9	4.19×10^9	2.26×10^9	1.67×10^9	5	f
09:16:45	H100_P24	0.97×10^9	0.22×10^9	1.11×10^9	3.58×10^9	3.25×10^9	0.41×10^9	1.76×10^9	0.78×10^9	2.50×10^9	2.58×10^9	5	f
09:26:48	H100_P24	2.50×10^9	0.32×10^9	3.28×10^9	0.28×10^9	0.36×10^9	4.79×10^9	2.42×10^9	1.66×10^9	2.85×10^9	3.45×10^9	5	f
09:36:50	H100_P24	4.03×10^9	4.20×10^9	8.39×10^9	1.25×10^9	1.44×10^9	1.10×10^9	3.09×10^9	2.54×10^9	3.17×10^9	4.26×10^9	5	f
09:46:53	H100_P24	1.29×10^9	8.99×10^9	2.69×10^9	2.16×10^9	2.54×10^9	1.72×10^9	3.76×10^9	3.42×10^9	3.48×10^9	0.78×10^9	5	f
09:56:55	H100_P24	3.34×10^9	1.90×10^9	2.40×10^9	3.08×10^9	3.67×10^9	2.43×10^9	0.13×10^9	0.04×10^9	3.87×10^9	1.59×10^9	5	f
10:06:57	H100_P24	0.79×10^9	2.89×10^9	2.42×10^9	3.99×10^9	0.52×10^9	2.97×10^9	0.86×10^9	0.91×10^9	4.17×10^9	2.25×10^9	5	f
10:17:00	H100_P24	2.84×10^9	3.89×10^9	4.16×10^9	0.61×10^9	1.66×10^9	3.59×10^9	1.56×10^9	1.71×10^9	0.18×10^9	2.92×10^9	5	f
10:27:03	H100_P24	0.63×10^9	0.59×10^9	1.71×10^9	1.52×10^9	2.39×10^9	4.27×10^9	2.39×10^9	2.53×10^9	0.40×10^9	3.69×10^9	5	f
10:37:05	H100_P24	2.72×10^9	1.59×10^9	3.57×10^9	2.45×10^9	2.98×10^9	6.81×10^9	3.31×10^9	3.34×10^9	0.70×10^9	0.15×10^9	5	f

3.2　特征挖掘

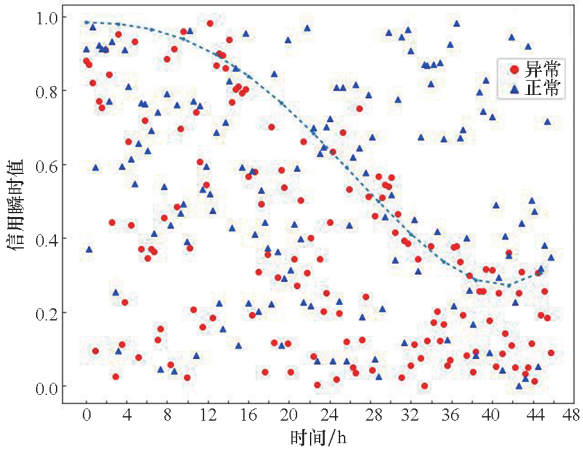
从上一节内容可知,本文采集了握手、重传、信用等12个特征数据,但是并不是每一个特征值在对正常样本和异常样本进行区分时都有所帮助,尤其是进行故障预测时。另外在构建机器学习的模型时,特征选择被证明是非常关键的,因此从特征数据中选择出稳定、可预测的特征值非常重要。现有的特征选择方法主要分为两大类,统计指标(如卡方、互信息等)和基于机器学习的方法(如随机森林等)。但对于信用、流量等特征值而言,由于数据本身存在时间敏感性和随机性的特点,传统的特征选择方法往往使得预测模型性能不佳。

通过分析发现,流量等特征在故障发生前和正常时候并无明显差异,这类相似的特征被定义为"无效特征",在构建特征向量时予以舍弃,如图5(a)中的流量(图示中的瞬时值均已归一化处理)。与"无效特征"形成对比的是类似于图5(b)中的信用值,被定义为"有效特征"。图5展示了交换机端口从正常状态到网络阻塞故障发生的那一刻,2 d内流量、数据链路层信用两个不同特征的变化趋势。从图5(a)中可以看出,直到网络阻塞发生的前一刻为止,异常端口的流量值变化趋势基本上和正常端口一样,处于一种随机波动的状态。而在图5(b)中,可以看到异常端口的数据链路层信用值随着时间在沿着虚线逐渐变小,而正常端口则依然保持在0~1区域内随机分布状态。流量、信用等特征值虽然都携带了与时



(a) 流量对比

(a) Comparison of flit



(b) 信用对比

(b) Comparison of credit

图5　不同特征对比

Fig. 5　Comparison of different features

间段高度相关的信息,并且每一个数值的大小都是真实正常的数据,但只有数据链路层信用等这类特征值在某一段时间内都保持在一个缩小的范围波动才能反映出交换机端口状态的变化。

另外,握手、重传等特征值往往只在故障发生前 0.5 h 内会出现异常,这种特征使得预测模型在时效性上效果很差,也予以舍弃。根据此种现象,最后选取了数据链路层信用、虚通道信用(分别对应表 2 中的 CREDIT 和 VC)等 6 个特征值作为“有效特征”。以数据链路层信用为例,任意某个时刻的值取[0,1]区间内的任何一个值都是合理的,但较长时间停留在一个较小的数值区域则说明该端口存在一定的网络阻塞风险。基于这个特点,通过将数据链路层信用等特征的值域均分地划分为 6 个区间(区间 1 为[0 ~ 0.17],区间 2 为(0.17 ~ 0.34],区间 3 为(0.34 ~ 0.51],区间 4 为(0.51 ~ 0.68],区间 5 为(0.68 ~ 0.85],区间 6 为(0.85 ~ 1]),统计某个端口在时间跨度为 100 个采样点的窗口内特征值在这 6 个区间的分布数量,将其作为一个新的特征值来衡量该端口当前时刻的健康状态。接下来再以 10 个采样点为滑动距离向前滑动,统计下 100 个采样点的分布数量作为下一时刻的状态信息,采样窗口滑动方式如图 6 所示。

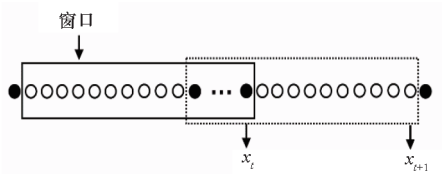


图 6　采样窗口滑动方式

Fig. 6　Sliding method of sliding window

图 7 则展示了某个端口的数据链路层信用依照图 6 所示规则,于网络拥塞故障发生前不同时刻在 6 个区间的数量分布情况。从图 7 中可以发现,区间 1 的数值变化可以最为完整地展现该端口从正常到出现故障整个过程的趋势。为此进一步研究了故障端口不同“有效特征”在区间 1 中的变化趋势,用来替代特征值本身的变化规律。

时间节点	区间1	区间2	区间3	区间4	区间5	区间6	状态
$t-n$	15	16	17	17	18	18	健康
⋮							
$t-k-2$	24	18	21	20	11	6	亚健康
$t-k-1$	26	23	23	21	5	2	亚健康
$t-k$	29	26	27	18	2	0	亚健康
⋮							
t	88	12	0	0	0	0	故障

图 7　区间分布

Fig. 7　Data distribution

通过分析发现,叶交换机端口和根交换机端口特征值在区间 1 上的数量分布变化趋势虽然类似,但它们在具体数值上存在较大差异。同样以数据链路层信用为例,图 8 展示了叶交换机和根交换机网络阻塞端口数据链路层信用在区间 1 中故障出现前 2 d 内的变化过程。

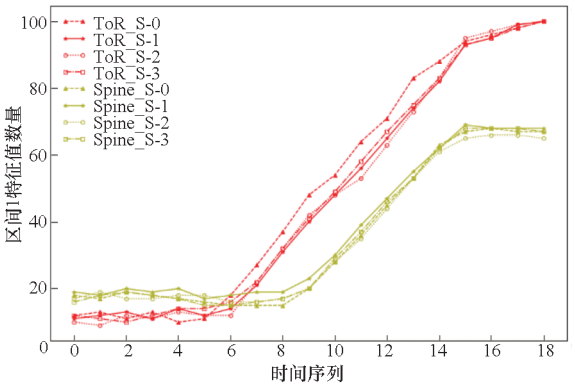


图 8　叶、根交换机故障端口区间 1 数值变化对比

Fig. 8　Numerical comparison of interval 1 in ToR and Spine switch

图 8 中 ToR_S 表示叶交换机,Spine_S 表示根交换机(下同),0 ~ 3 表示四个端口号。在曲线的前段部分,根交换机端口(黄色曲线)的特征值在区间 1 上的数量比叶交换机端口要多,但随着端口健康状态的恶化,可以看出在同一时间段上根交换机端口的特征值在区间 1 上的数量明显比叶交换机要少,并且更早地出现拐点。从这里可以看出,基于拓扑结构分别对叶交换机和根交换机的特征数据进行单独分析更为合理。为此,筛选了叶交换机和根交换机网络阻塞故障端口各 100 个。原始样本数据均是从故障未发生前某个时间点开始到故障发生时那一刻为止 2 d 内的数据。同时在对应时间段随机抽取了叶交换机和根交换机正常端口各 100 个。按照图 6 所示规则,分别计算出这些端口 6 个“有效特征”在区间 1 的分布情况,组合成新的特征向量来表示某个端口的当前状态。比如在 T 时刻,可以得到如式(9)所示新的特征向量。

$$\boldsymbol{X}_T = [20,15,17,14,17,16] \tag{9}$$

3.3　模型训练

3.3.1　K-means 聚类模型训练

从图 5(b)中可以得知,某个端口发生网络阻塞故障是一个渐变过程,从实际运维角度考虑,可将这个过程笼统地划分为健康、亚健康、故障三个阶段。采用无监督的 K-means 聚类算法对新得到的特征向量 $\boldsymbol{X}_i(i=0,1,\cdots,n)$ 进行研究分析。首

先要确定聚类算法的种数 K,使用轮廓系数法,具体过程如下:在保证正常端口和故障端口数量比例均衡的前提下,对交换机端口历史数据挖掘出新的特征向量集形成训练集。分别对三种情况进行了分析:第一种是叶交换机正常与网络阻塞端口各取100个进行分析;第二种是根交换机正常与网络阻塞端口各取100个进行分析;第三种是将前两种情形选取的400个端口混合分析。三种情况下同样对 K 值从2~10进行取值,得到聚类数 K 与轮廓系数 S 的关系,如图9所示。利用sklearn. clurster库中的 K-means 算法对上述三种情形进行聚类分析。可以看到,单独对叶交换机、根交换机端口的特征向量集进行聚类分析时,轮廓系数最大值都是在 $K=3$ 时取得,这表示 $K=3$ 为最佳聚类数。同时可以看到,将叶交换机和根交换机端口的特征向量集混合后,不仅没有进一步提升 $K=3$ 时的分类效果,反而使得 K 在不同取值时整体轮廓系数相近,聚类效果变差,这也再次证明了基于拓扑结构对不同层级的交换机端口分类研究分析的必要性。在选定了 $K=3$ 进行聚类分析后,还需对循环次数进行选取。统计了在不同循环次数训练后各个类别的数目,并重新用统计的样本数据进行预测。

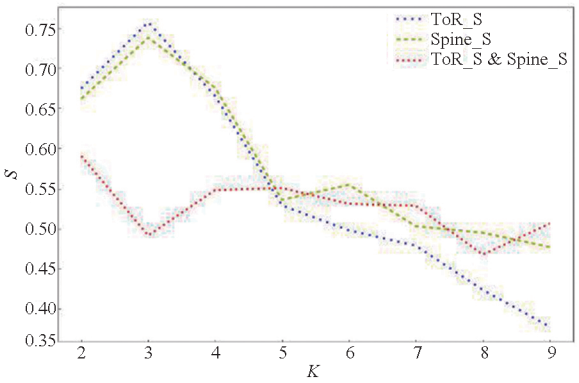
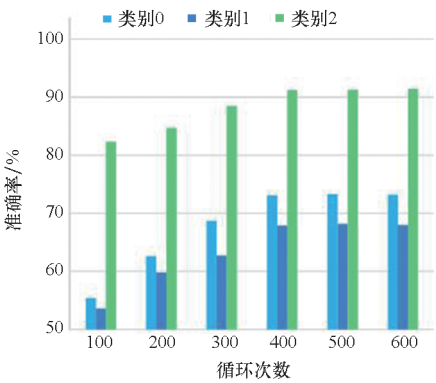


图9　轮廓系数

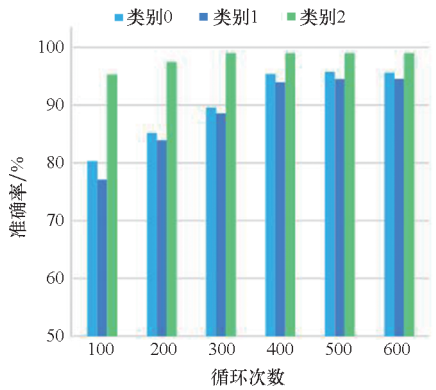
Fig. 9　Silhouette coefficient

不同循环次数下分类的准确率如图10所示。从图10中均可以发现,随着训练的循环次数增加,分类的准确率都会提高。图10(a)是将叶交换机和根交换机端口的特征向量集混合后聚类的效果,准确率较低。其中类别0准确率最高为73.3%,类别1为68.2%,类别2为91.4%。而从图10(b)和(c)中可以看出,当按照网络拓扑结构单独对叶交换机和根交换机端口特征向量集进行聚类分析时,三种类别分类的整体准确率均有明显提升。图10(b)中类别0的准确率由80.3%提升至95.8%,类别1由77.1%提升至94.6%,类别2由95.3%提升至99.1%。图10(c)中类别0的准确率由83.5%提升至95.2%,类别1由79.3%提升至93.9%,类别2由92.4%提升至99.5%。另外,从图10中可以发现,不管是否按照拓扑结构分类进行聚类分析,分类准确率在循环500次时就取得了最好分类效果,循环次数继续增加对分类效果几乎没有影响。考虑模型的计算成本,选取500作为 K-means 聚类模型的循环训练次数。



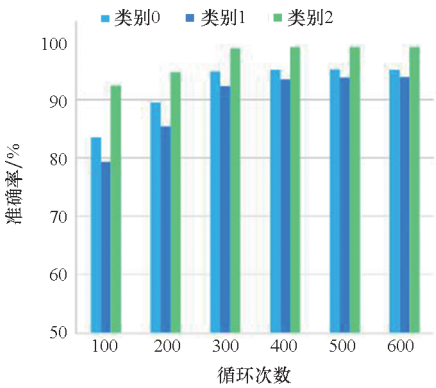
(a) 叶交换机和根交换机混合分类

(a) ToR_S & Spine_S hybrid clustering



(b) 叶交换机分类

(b) ToR_S clustering



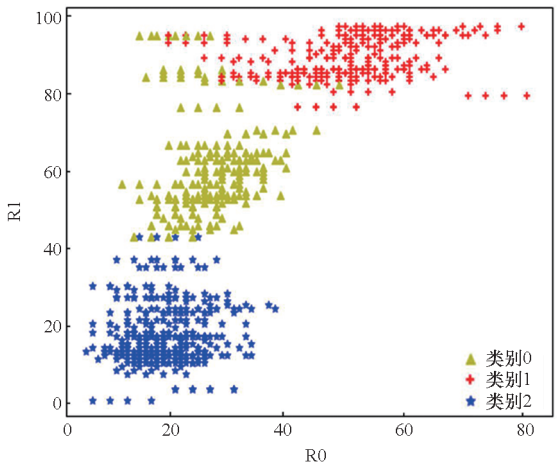
(c) 根交换机分类

(c) Spine_S clustering

图10　不同循环次数下分类的准确率

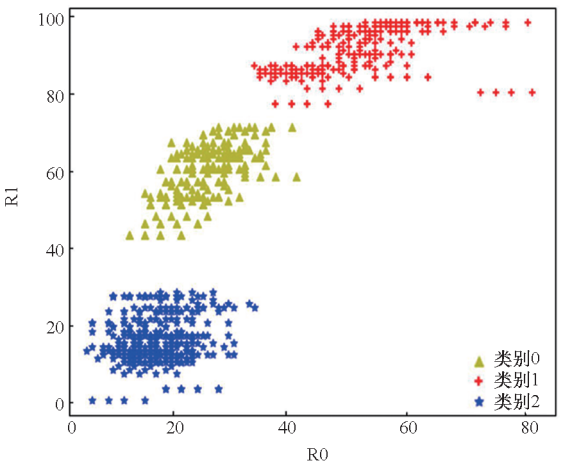
Fig. 10　Classification accuracy under different cycles

图 11 展示了循环次数为 500 时的聚类效果,图示的横坐标和纵坐标分别为新特征向量 $\boldsymbol{X}_T$ 的第一个特征值与最后一个特征值。图 11(a)为将叶交换机和根交换机端口特征向量集混合后聚类的效果,可以明显看到类别 0 与类别 1、类别 2 都有重叠区域。此时类别 0 和 1 的准确率分别约为 73% 和 68%,类别 2 的准确率约为 91%。图 11(b)为叶交换机端口单独聚类的效果,其中类别 0 有 841 个,类别 1 有 383 个,类别 2 有 2 576 个,每一类的预测准确率均上升到 94% 以上,其中类别 2 的准确率最高,达到了 99%。按照同样的方法,单独对根交换机端口特征向量集进行聚类时,每一类的准确率达到了 93% 以上,类别 2 同样高达 99%。值得注意的是,结合图 8 曲线变化趋势和 3 种类别的数量分布,可以确定基于运维角度划分的健康、亚健康、故障 3 种情形分别对应类别 2、类别 0、类别 1。



(a) 叶交换机和根交换机混合聚类效果

(a) ToR_S & Spine_S hybrid clustering results



(b) 叶交换机聚类效果

(b) ToR_S clustering results

图 11　聚类效果($K=3$)

Fig. 11　Clustering results($K=3$)

3.3.2　DES 预测模型训练

在这个环节,采用 DES 算法分别对叶交换机、根交换机的 6 个“有效特征”在区间 1 上的变化趋势进行学习。由于时间序列在故障出现前具有明显的变动倾向,为了使预测模型灵敏度更高,该模型的平滑系数 a 应取较大值。选取 0.5、0.6、0.7、0.8 和 0.9 共 5 个 a 值,分别计算各个预测值和它们的标准误差,选取使得标准误差最小的 a 值作为预测模型的参数。

表 5 展示了叶交换机网络阻塞端口数据链路层信用值分布数的平均值在不同的平滑系数 a 下不同时刻的预测值。预测值与真实值的标准误差见表 6。结合表 5 和表 6,发现叶交换机端口的数据链路层信用在平滑系数为 0.9 时标准误差最小。因此在对叶交换机端口数据链路层信用建立预测模型时,选择平滑系数为 0.9。对于根交换机端口以及在不按拓扑分类的对比实验中,采用相同的方法求得模型的平滑系数分别为 0.8 和 0.9。对于虚通道信用等其他 5 个特征值,也采用相同的处理方式对其趋势进行学习、预测。作为对比,还对不区分叶交换机和根交换机端口的情况下,将所有交换机端口特征向量集统一进行学习训练。图 12 展示了基于网络拓扑结构不分类和分类两种情景下,故障端口实际值和预测值的差异。同样以数据链路层信用为例,不做分类处理时,如图 12(a)所示。图 12(b)和图 12(c)为

表 5　不同平滑系数下的预测值

Tab. 5　Predicted values for different smoothing coefficients

时间序列	真实值	预测值				
		$a=0.5$	$a=0.6$	$a=0.7$	$a=0.8$	$a=0.9$
1	18	18	18	18	18	18
2	17	18	18	18	18	18
3	18	17	17	17	16	16
4	19	18	18	18	18	19
5	23	19	19	20	20	20
6	30	23	24	25	26	26
7	39	31	33	34	35	36
8	47	42	44	46	47	47
9	55	52	54	55	55	55
10	62	61	62	63	63	63
11	69	69	70	70	69	69
12	68	76	76	76	76	76
13	68	75	74	72	70	69
14	68	73	71	70	69	68

表 6　不同平滑系数下的标准误差

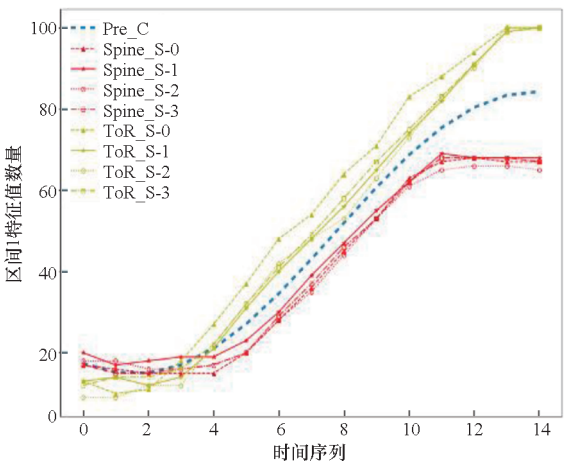
Tab. 6　Standard deviation for different smoothing coefficients

平滑系数	0.5	0.6	0.7	0.8	0.9
标准误差	17.39	14.46	12.30	10.85	10.00

基于拓扑分类后叶交换机、根交换机两种交换机端口分类学习预测的结果。其中虚曲线表示预测值，实线反映了不同网络阻塞叶交换机或根交换机端口的数据链路层信用在区间 1 分布的真实值，可以看出两者在整体上非常接近，标准误差均小于 15。可以明显看到预测值和实际值存在明显偏差，标准误差超过 40，这使得 *K*-means 聚类算法给系统状态的判断带来很大的偏差。

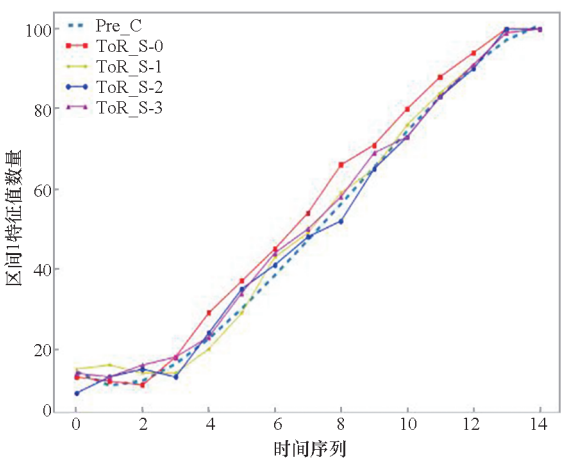
3.3.3　长短期记忆网络算法预测模型训练

与此同时，还尝试了采用长短期记忆（long short-term memory，LSTM）网络算法直接针对握手、重传、信用等特征值进行时序预测。下面以重传为例进行说明，选取网络端口故障发生前 2 d 内的数据（包括故障发生时）。在训练前，对单端口的原始数据进行累加、归一化等处理。



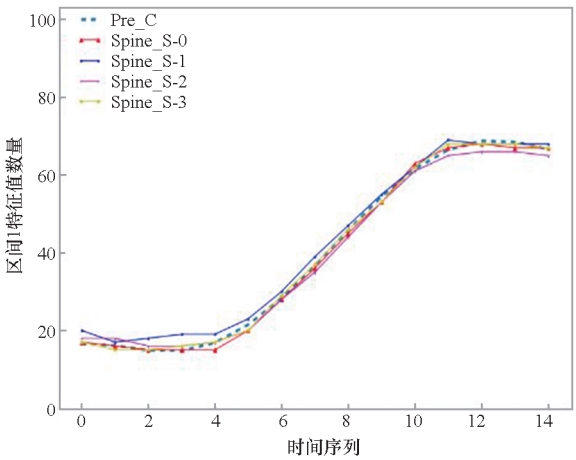
（a）混合预测效果

（a）Mixed prediction effect



（b）叶交换机预测效果

（b）ToR_S prediction effect



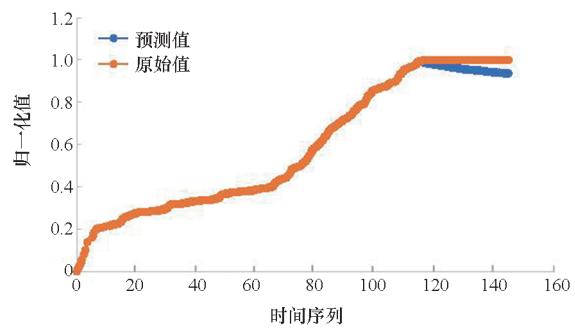
（c）根交换机预测效果

（c）Spine_S prediction effect

图 12　实际值与预测值对比

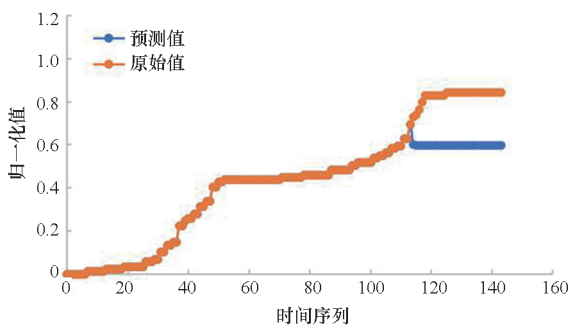
Fig. 12　Comparison of actual and predicted values

在学习率为 0.000 2、隐藏节点为 10、时间步为 20、Batch-size 为 60、迭代次数为 200、输入和输出节点数均为 1 等条件下，选取 85% 的原始数据作为训练集，余下 15% 作为验证集，预测准确率不到 20%。图 13 展示了任意选取的 4 个网络端口预测结果。经对比，基于 LSTM 算法模型预测效果明显差于 DES 模型，因此 LSTM 算法在本文场景中并不适用。



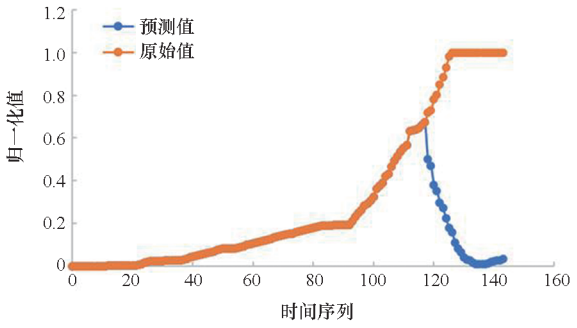
（a）端口 a

（a）Port a



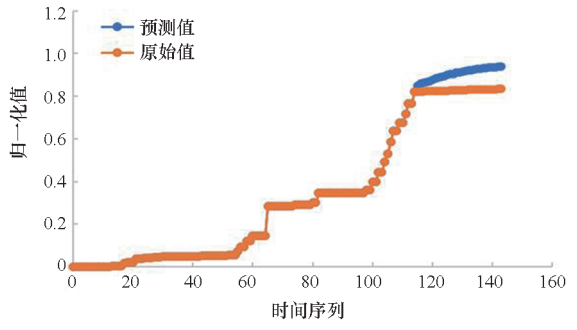
（b）端口 b

（b）Port b



(c) 端口 c

(c) Port c



(d) 端口 d

(d) Port d

图 13　实际值与基于 LSTM 算法的预测值对比

Fig. 13　Comparison of actual and predicted values by LSTM algorithm

3.3.4　在线预测

对于本文涉及的运维场景,运维工作者的期望预测结果是在有高召回率的前提下尽可能保持较高的精确率。这样既可以将系统中潜在的问题尽可能地暴露出来,又不至于消耗过多的运维人力资源。在本节中,将基于交换机端口当前的状态,结合前文训练的 *K*-means 聚类模型和 DES 算法,对未来一周的网络端口状态进行预测分析。*K*-means 聚类模型对预测得到的 $\boldsymbol{X}_i$ 状态归类为类别 0(亚健康)或类别 1(故障)时,都被统一划为网络阻塞故障进行统计分析。*TP* 表示正确预测出将来会网络阻塞故障的端口数,*FN* 表示将有网络阻塞故障漏报为正常的端口数,*FP* 为将正常情况误报为网络阻塞故障的端口数,*TN* 为正确预测将来正常的端口数。同样,分别对是否基于网络拓扑结构分类两种情形进行比较,具体结果如表 7 所示。

当不按拓扑结构对叶交换机端口和根交换机端口加以区分时,在 17 712 个网络端口中,正确预测出网络阻塞端口为 11 个,正常端口为 17 654 个,网络阻塞故障端口漏报 6 个,将正常端口误报为网络阻塞的有 41 个。这种情况下,虽然整体的

表 7　预测结果对比

Tab. 7　Comparison of prediction results

分类	评价指标			
	TP	*FN*	*FP*	*TN*
不按拓扑分类	11	6	41	17 654
按拓扑分类	15	2	25	17 670

准确率达到了 99.7%,但精确率仅为 21.2%,召回率为 64.7%。若按拓扑结构分别对叶交换机和根交换机进行预测,则正确预测出网络阻塞的端口数上升为 15,提高了约 36.4%;漏报的网络阻塞故障端口数减少为 2,比例降低了 66.7%;将正常端口误报为网络阻塞的端口数减少为 25,降低了 39%。此时整体召回率提高到 88.2%,精确率为 37.5%,准确率为 99.8%。对于将正常端口误报为网络阻塞端口的情况,根据运维人员积累的运维经验,对这部分端口的实时状态进行甄别后,可以排除接近 68% 的误报端口,可将分类的情形下的准确率提升至 65.2% 左右。

4　结论

为了提高高性能计算机系统的持续可用性,本文提出了一种将无监督的分类算法(*K*-means 算法)和时间序列算法(DES 算法)相结合的预测模型,用于预测交换机端口在不久的将来是否会出现故障。该模型通过从交换机端口底层寄存器的历史状态信息中挖掘出征兆性规律并形成新的特征向量,应用 *K*-means 聚类算法对特征向量进行学习归类。在预测时,结合端口当前状态,利用 DES 算法对未来一段时间进行预测,将得到的新特征向量使用 *K*-means 算法预判是否即将出现网络阻塞。通过聚类算法解决了专家系统对网络阻塞故障的模糊状态划分,能对交换机端口当前状态给出明确分类。同时在运维专家系统的辅助下,结合不同类型交换机在互连网络拓扑结构中的差异性,分别针对叶交换机和根交换机独立构建子模型,使得该模型能保持在 88.2% 的召回率前提下,准确率达到 65.2%。因此,在网络阻塞故障发生之前,运维工作人员可以主动将存在隐患的端口隔离,提前进行处理,增强高速互连网络的持续可用性。对于故障预测期间发生的且不属于之前故障预测结果中的端口,网络监控软件可通过容错路由的方式对这些故障端口进行屏蔽,并通知运维工作人员及时对故障端口进行修复。

鉴于高速互连网络持续可用性影响着高性能

计算机的整体性能,故障预测将在高性能互连网络的日常维护中发挥越来越重要的作用,本文方法是朝这个方向迈出的重要一步。在后续工作中,针对数据采集情况和大规模的网络端口数量,如何加强对不同时刻网络端口的健康状态的归类以及各特征值的时序预测,仍需做进一步的探究。一是选择更优的算法进行组合,包括分类和时序预测两个环节;二是对现有的 *K*-means 聚类和 DES 时序预测算法进行参数调优,从而进一步提升故障预测的准确率。

参考文献(References)

[1] Anon. ORNL′s frontier first to break the exaflop ceiling[EB/OL]. [2022 - 09 - 05]. https://www.top500.org/.

[2] WRIGHT M. The opportunities and challenges of exascale computing [R]. Washington: US Department of Energy, 2010.

[3] DUATO J, YALAMANCHILI S, NI L. Interconnection networks: an engineering approach [M]. San Francisco: Morgan Kaufmann Publishers, 2002.

[4] DALLY W J, TOWLES B. Principles and practices of interconnection networks [M]. San Francisco: Morgan Kaufmann Publishers, 2004.

[5] DOMKE J, HOEFLER T, MATSUOKA S. Fail-in-place network design: interaction between topology, routing algorithm and failures[C]//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2014: 597 - 608.

[6] Gartner. Market guide for AIOps platforms[EB/OL]. (2021 - 04 - 06)[2022 - 04 - 30]. https://www.gartner.com/en/documents/4000217.

[7] MASOOD A, HASHMI A. AIOps: predictive analytics & machine learning in operations[M]//Cognitive Computing Recipes. Berkeley: Apress, 2019: 359 - 382.

[8] ANDENMATTEN M. AIOps—artificial intelligence für IT-operations[J]. HMD Praxis Der Wirtschaftsinformatik, 2019, 56(2): 332 - 344.

[9] FRONZA I, SILLITTI A, SUCCI G, et al. Failure prediction based on log files using random indexing and support vector machines[J]. Journal of Systems and Software, 2013, 86(1): 2 - 11.

[10] PITAKRAT T, OKANOVI Ć D, VAN HOORN A, et al. Hora: architecture-aware online failure prediction [J]. Journal of Systems and Software, 2018, 137: 669 - 685.

[11] XU C, WANG G, LIU X G, et al. Health status assessment and failure prediction for hard drives with recurrent neural networks [J]. IEEE Transactions on Computers, 2016, 65(11): 3502 - 3508.

[12] CHENG Y W, ZHU H P, WU J, et al. Machine health monitoring using adaptive kernel spectral clustering and deep long short-term memory recurrent neural networks[J]. IEEE Transactions on Industrial Informatics, 2019, 15(2): 987 - 997.

[13] DAS A, MUELLER F, SIEGEL C, et al. Desh: deep learning for system health prediction of lead times to failure in HPC[C]//Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing, 2018: 40 - 51.

[14] LIN Q W, HSIEH K, DANG Y N, et al. Predicting node failure in cloud service systems[C]//Proceedings of the 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2018: 480 - 490.

[15] CANFORA G, DE LUCIA A, PENTA M D, et al. Defect prediction as a multiobjective optimization problem [J]. Software Testing, Verification and Reliability, 2015, 25(4): 426 - 459.

[16] D′AMBROS M, LANZA M, ROBBES R. Evaluating defect prediction approaches: a benchmark and an extensive comparison [J]. Empirical Software Engineering, 2012, 17(4/5): 531 - 577.

[17] GAO J C, WANG H Y, SHEN H Y. Task failure prediction in cloud data centers using deep learning [J]. IEEE Transactions on Services Computing, 2022, 15(3): 1411 - 1422.

[18] ISLAM T, MANIVANNAN D. Predicting application failure in cloud: a machine learning approach[C]//Proceedings of IEEE International Conference on Cognitive Computing, 2017: 24 - 31.

[19] 孙勤. 基于在线机器学习的高性能计算机故障预测技术研究[D]. 长沙: 国防科技大学, 2017.
SUN Q. Research on failure prediction of supercomputers based on online machine learning[D]. Changsha: National University of Defense Technology, 2017. (in Chinese)

[20] 刘睿涛. 超级计算机故障分析、建模与预测技术研究[D]. 郑州: 战略支援部队信息工程大学, 2018.
LIU R T. Research on failure analysis, modeling and prediction for supercomputers[D]. Zhengzhou: Information Engineering University, 2018. (in Chinese)

[21] WANG Y, MA E W M, CHOW T W S, et al. A two-step parametric method for failure prediction in hard disk drives[J]. IEEE Transactions on Industrial Informatics, 2014, 10(1): 419 - 430.

[22] XU Y, SUI K, YAO R, et al. Improving service availability of cloud systems by predicting disk error[C]//Proceedings of the 2018 USENIX Conference on Usenix Annual Technical Conference, 2018: 481 - 493.

[23] ZHANG S L, LIU Y, MENG W B, et al. PreFix: switch failure prediction in datacenter networks[C]//Proceedings of Abstracts of the ACM International Conference on Measurement and Modeling of Computer Systems, 2018: 64 - 66.