



# 智能化网格电信系统的故障预测方法

蔡珩, 戈磊

(中国电信股份有限公司上海分公司, 上海 200042)

**摘要:** 尝试用基于深度学习的相关人工智能技术, 分析服务器集群上的进程和端口网络, 并对网络节点进行状态预测。具体地, 结合运维过程中的先验知识对网络节点的特征进行细致选择, 预测网络中各个进程和端口的异常(崩溃)状态。实验结果表明, 进程节点的运行信息(如 CPU 和内存使用率)、进程间的通信情况以及进程节点在整个网络中的结构特征对于判断该节点的状态具有一定的指导价值, 而这些特征在时间维度上的变化量同样反映了进程/端口的状态。

**关键词:** 故障预测; 深度学习; 二分类

**中图分类号:** TP391.1

**文献标识码:** A

**doi:** 10.11959/j.issn.1000-0801.2018118

## Intelligent fault prediction method of telecom system

CAI Heng, GE Lei

Shanghai Branch of China Telecom Co., Ltd., Shanghai 200042, China

**Abstract:** Some approaches based on deep learning would be used to analyze the process and port network on a server cluster. Specifically, the features of nodes were carefully selected in server cluster network, by combining the prior knowledge from actual operations, and the abnormal state of processes or ports on the cluster was predicted. According to the research, the running information such as loads of CPU and memory, communications between processes and the structural features in the process network was valuable in predicting the states of processes and ports; furthermore, the changes of features mentioned above in the time dimension reflected the states of processes or ports, too.

**Key words:** fault prediction, deep learning, binary classification

### 1 引言

随着信息技术的发展, 大规模服务器集群和网络设备的部署和使用日益广泛, 在集群上运行各种服务<sup>[1]</sup>逐渐成为大型企业的选择。同时, 对集群上各种资源的监视和管理也得到学术界和工业界的大量关注。

人们总是希望部署一个可靠稳定的服务器集群, 然而事与愿违, 大部分集群上还是会出现如进程崩溃<sup>[2]</sup>、端口流量过大、端口阻塞<sup>[3]</sup>等异常情况。人们往往采取人工查看服务器日志的方式, 通过各个进程和端口的状态判断哪些进程或端口可能会发生异常并采取相应的措施, 但是这种方式耗时耗力, 并且人工查看具有局限性和一定的误差。



本文将采用一些基于深度学习的方法,对服务器集群上的进程和端口网络进行分析,并对该网络上的节点即某个进程或端口的状态进行预测。

本文首先对服务器集群上的进程网络进行建模,而和被广泛关注的社交网络<sup>[4-8]</sup>不同的是,缺乏对该进程网络上节点的了解,比如一个进程何时会与周围的进程产生通信、为什么会产生通信以及本文的预测目标:一个进程是否会发生崩溃,因为进程间的通信通常依赖于某个进程的具体功能和实际服务的使用情况。即便如此,本文还是可以类比社交网络中节点的相关性质,对该进程网络中的节点做出如下分析。

- 将某个进程的占用 CPU、内存情况看作进程节点的“固有属性”。
- 进程之间的通信看作进程网络中的边,那么,节点在网络中的中心度<sup>[9-10]</sup>可以衡量节点的活跃程度以及与外界联系的紧密程度。
- 把进程崩溃视作一个进程的行为,那么进程崩溃这一现象可以类比为社交网络中信息的扩散<sup>[11-12]</sup>。

对于端口网络  $N(V, E)$ , 其中, 节点  $v \in V$  为单个端口,  $e \in E$  表示端口之间的通信。需要注意的是, 本文并不关心端口之间的通信(这体现在进程的通信中); 相比之下, 更关心一个端口的具体状态; 更具体地, 本文关注端口的出入流量, 因为一个正常的端口往往具有较为稳定合理的出入流量, 该属性能在一定程度上反映端口异常与否。

因此, 端口网络  $N$  中的某个节点  $v$  可以表示为  $\{(in_1, out_1, t_1), (in_2, out_2, t_2) \dots\}$ , 其中,  $(in_i, out_i, t_i)$  表示在  $t_i$  时刻,  $v$  具有入流量  $in_i$ , 出流量  $out_i$ 。

基于以上分析, 本文把一个进程或端口发生崩溃、阻塞等异常的现象定义为网络中一个节点的状态; 在给定的时间戳下, 网络中的节点可以被分为两类: 处于异常状态的节点和处于正常状态的节点。因此, 预测网络中节点的异常<sup>[13]</sup>可以

转化为针对网络节点的二分类问题<sup>[14]</sup>。本文仔细地选取节点的相关属性作为节点分类的特征, 用卷积神经网络 (convolutional neural network, CNN)<sup>[15]</sup>对该模型进行分类, 并得到了较为可信的结果。

## 2 相关定义

### 2.1 进程网络

本文对服务器集群上的进程网络以及该网络中的节点、边和节点状态做出如下定义。

**定义 1** (进程标识符) 用一个进程的如下信息作为其标识符: 本地 IP 地址、本地主机名、本地进程组、进程描述以及进程号。换句话说, 上述 5 个字段可以唯一地确定一个进程。

如果两条日志记录中进程的标识符完全一致, 则认为是同一进程的记录。

**定义 2** (常驻进程) 把存在时长超过某一阈值  $\Delta T$  的进程定义为常驻进程。一方面, 本文关心那些运行时间较长的主要进程的状态; 另一方面, 服务器集群上存在临时启动的进程, 它们的进程号不断变化, 状态表现得极不稳定, 干扰对进程状态的判断。

常驻进程  $i$  体现在日志记录中的形式为: 存在进程  $i$  的两条记录, 时间戳分别为  $T_1$ 、 $T_2$  ( $T_1 < T_2$ ), 且  $T_2 - T_1 \geq \Delta T$ 。

**定义 3** (进程网络) 该进程网络可以用图  $G(V, E, T)$  来表示, 其中, 节点  $v \in V$  为某一具体进程, 用定义 1 中的 5 个字段来描述,  $T$  为当前网络的时间戳,  $e \in E$  表示进程间的联系, 即  $e(i, j)$  意味着进程  $i$  和  $j$  在时刻  $t$  存在通信, 边  $e$  的权重  $w$  为通信次数。

**定义 4** (进程的崩溃状态) 假设常驻进程  $i$  在进程日志中连续出现的时间戳分别为  $T_1$ 、 $T_2$  ( $T_1 < T_2$ ), 且  $T_1$  时刻进程  $i$  的标识符为  $a_1$ 、 $b_1$ 、 $c_1$ 、 $d_1$ 、 $e_1$ ,  $T_2$  时刻进程  $i$  的标识符为  $a_2$ 、 $b_2$ 、 $c_2$ 、 $d_2$ 、 $e_2$ , 其中,  $a$ 、 $b$ 、 $c$ 、 $d$ 、 $e$  分别对应本

地 IP 地址、本地主机名、本地进程组、进程描述以及进程号。

把常驻进程  $i$  在时刻  $T_2$  为崩溃状态定义为：与  $T_1$  时刻相比，除进程号外，进程  $i$  的其余 4 个标识符均未发生变化，即  $e_1 \neq e_2$ 。

换句话说，一个常驻进程在时刻  $T$  为崩溃状态意味着该进程此刻的进程号与其上一次在日志中出现的进程号不一致。

**定义 5** （远端进程）把在日志记录中本地 IP 地址或本地主机名为空的进程定义为远端进程。由于日志记录是由本地 probe（探针）对正在运行的进程进行遍历得到的，因此，日志记录中本地 IP 地址或主机名为空可以视作该进程不在这个服务器集群上，将这类进程称为远端进程。

## 2.2 端口网络

**定义 6** （端口标识符）用一个端口的如下信息作为其标识符：主机名、端口名。如果两条日志记录中主机名和端口名一致，认为这是同一端口的记录。

## 3 数据观察

在对进程和端口状态进行预测前，先从整体上对数据做一些基本的分析。

### 3.1 进程网络

#### 3.1.1 数据量

本进程网络数据集（上海电信 CSB 服务器集群的进程日志）由服务器 CSB 节点上的探针定时探测得到，本文选取了 2016 年 8 月 30 日 14—18 时共 2 858 063 条日志记录进行分析。其中，该时间段内的常驻进程共 973 个，进程崩溃次数为 25 次。在后面的二分类问题中，把在时刻  $T$  发生崩溃的进程定义为正样本，没有发生崩溃的进程定义为负样本。在此数据集上，正负样本比为 0.16%，是极为稀疏的。

#### 3.1.2 静态特征分布

对常驻进程  $i$  在时刻  $T$ ，定义以下特征：

- CPU 占用率；
- 内存使用量；
- 与之存在通信的进程数量（即进程网络中节点的度）；
- 与其他进程的通信总量（即进程网络中节点的边权之和）；
- 存在通信的远端进程数量；
- $i$  在进程网络  $G(V, E, T)$  中的中心度，包括 betweenness（中心度）和 closeness（紧密性）。

如图 1 所示，正负样本在 CPU 这一“固有属性”上存在一定的差异：对于负样本即正常进程，它们的 CPU 使用率集中在 20% 以下和 100% 以上；而正样本（异常进程）的 CPU 使用率分布较为分散。根据经验，这种分布是可以理解的，因为对于正在运行的进程，如果是计算密集型的，CPU 使用量会较高，否则一般不会太多地占用 CPU 资源。

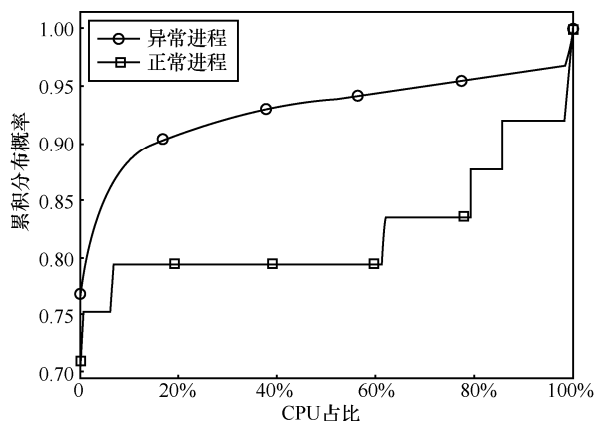


图 1 CPU 占比的整体分布

图 2 为存在通信的进程数量这一非结构特征的整体分布。可以看到，存在通信的进程数量在正负样本之间没有显著区别（进程网络中节点的度大多为 1）；进程节点的带权重的度以及远端通信进程数量这两个特征的结果也和图 2 类似。也就是说，正负样本在这 3 个非结构特征上没有明显差异。

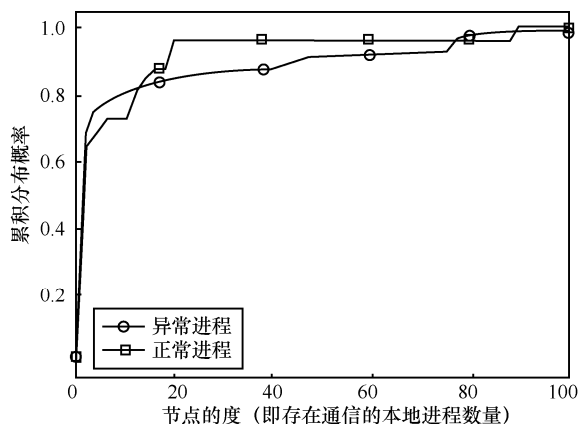


图2 存在通信的进程数量的整体分布

对该进程网络中的结构特征的分布情况进行考察。再次回到本文的目标：对于给定的进程，其是否为崩溃状态。对于较大规模的服务器集群，其上运行着大量进程，不难想象，一个进程关联的其他进程越多，该进程就越重要，其对服务器的负载就越重，崩溃的可能性就越大。

因此，选择进程网络图上的结构特征来衡量一个进程的重要性或者核心程度，希望通过进程节点的中心度<sup>[9-10]</sup>来帮助对进程状态进行分析。图3给出了进程节点的中心度的分布情况。其中，横坐标为中心度的十进对数，纵坐标为累积分布概率。

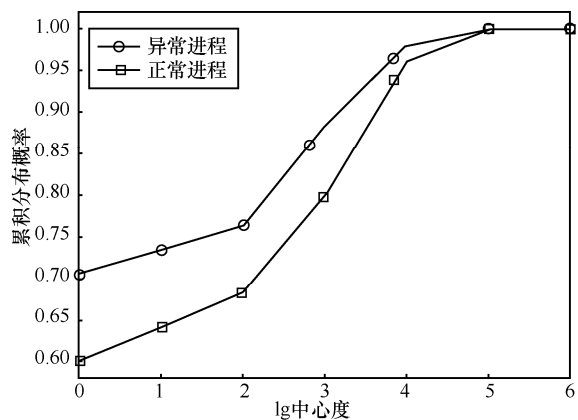


图3 中心度的整体分布

可以看到，相对于正常进程，正样本（异常进程）的中心度偏小，但整体分布是相似的，紧密性也是类似的结果。但这并不意味着“中心度高的重要进程更容易崩溃”这一假设不正确，因

为往往一个中心进程崩溃会导致与其存在通信的其他进程或者该进程的子进程产生异常，而日志记录是在某个特定的时间点通过探针生成的，因此很有可能在生成日志记录的时候，以该中心进程为核心的进程组都进行了重启，因此正样本的中心度分布和样本的整体分布没有较大区别。

值得注意的是，中心度和紧密性具有一定的相关性，二者作为特征训练分类器的效果也许会有意想不到的效果，在实验中也将看到这一点。

### 3.1.3 时间间隔分布

前文对进程的静态特征做了整体分析，但是还应当注意到，进程的崩溃是一个过程，时间维度上的特征也许会较好地反映进程的状态。

本文考虑进程从正常状态到崩溃状态的时间间隔，体现在日志记录中即同一进程的进程号两条不同的连续日志记录的时间戳的差。如果进程一直保持着正常状态（没有崩溃），则倾向于认为其在日志记录中出现的时间戳应当是比较稳定的，即不会突然在一段时间内没有日志记录。而对于发生崩溃的进程，由于其重启等因素，可能会有较长时间间隔没有日志记录的现象。

具体地，定义进程*i*的一个时间间隔 $\Delta T = T_2 - T_1$ ，其中， $T_1$ 、 $T_2$ 为*i*的连续两条日志记录的时间戳（ $T_2 > T_1$ ）；定义 $\Delta T = T_2 - T_1$ 为正样本的时间间隔，如果恰好在 $T_2$ 时刻进程号发生变化。

图4为正负样本的时间间隔的整体分布。其中，横坐标为进程距离上次出现在探针日志中的时间间隔（分钟），纵坐标为累积分布概率。

再一次地，可以发现在时间间隔这一特征上没有显著区别。由于探针每3 min采样一次，因此时间戳的差集中在3、6或9等数值上。

除了时间间隔外，对于进程*i*的两条日志记录 $T_1$ 、 $T_2$ ，本文在分类问题中还会考虑 $T_1$ 、 $T_2$ 上的特征差，即进程的特征随时间的变化量。后面将看到，这一考虑是十分有效的。

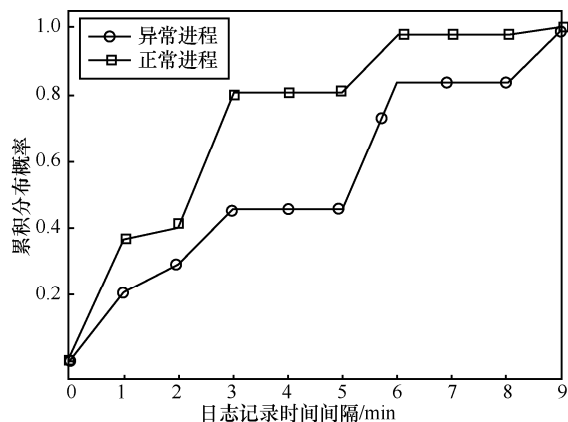


图4 时间间隔的整体分布

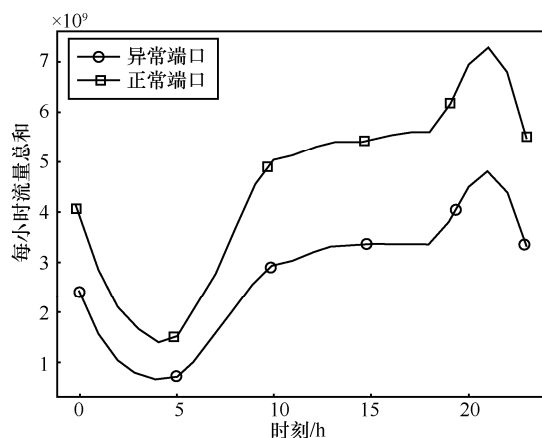


图5 前一天平均入口流量分布

## 3.2 端口网络

### 3.2.1 数据量

端口数据集共包含上海电信 IDC（互联网数据中心）2017年4月份的端口流量监控记录，日活跃端口数量均值为6 175.5，活跃端口的每日数据为每5 s一条流量记录以及当日告警（出现异常）的次数。在后面的二分类问题中，把当日告警次数非零的端口定义为正样本，无告警记录的端口记为负样本。在此数据集上，正负样本比为0.77%，正样本也是很稀疏的。

### 3.2.2 端口特征

对具体某个端口，定义以下特征：

- 第 $k$ 天前的各小时出入口总流量；
- 第 $k$ 天前出入口流量的整体方差；
- 第 $k$ 天前的出入口整体平均流量；
- 第 $k$ 天前的告警次数。

其中， $k$ 的值选取了1、2、3。

图5为端口在前一天各个小时中的平均入口流量分布。可以看到异常端口和正常端口的差距是较为明显的：相对于正常端口，异常端口各个小时的平均流量有明显的下降，可以猜测发生了端口阻塞，导致异常端口在同时段内流量较少。

同样地，如图6所示，异常端口和正常端口在前3天的出口流量方差这一指标上也是区别显著：相对于正常端口，异常端口的出口流量在一天的方差很大（很可能是由于异常端口时断时

续地进行工作，出入流量的方差相比正常端口自然要大一些）。图7是端口在前3天出现告警次数的平均值。可以看到，如果当天端口出现异常情况，那么在前3天这个端口也很有可能已经出现了异常情况。

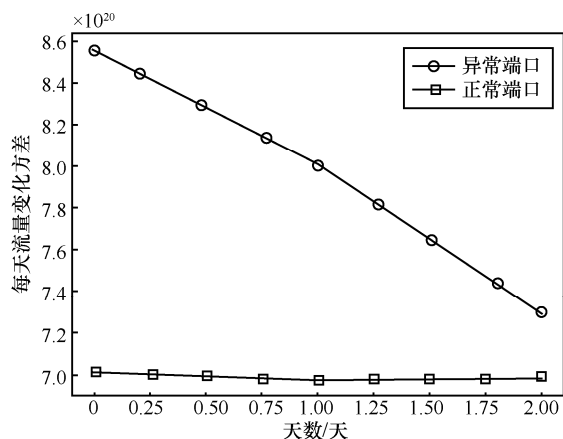


图6 前3天出口流量方差

值得注意的是，图5和图6的结果是在2017年4月份的数据集上得出的，不同时间段内异常端口的表现可能不一致，如异常端口可能会出现出入口流量方差较小的情况（和图6恰恰相反），这有可能是因为异常端口被阻塞，流量在一天的大部分时间内恒定地处于较低水平，表现出方差较小的情况。但经过大量数据的测试，发现异常端口和正常端口在出入口流量方差和小时总量这两个指标上总是有较为显著的区别。

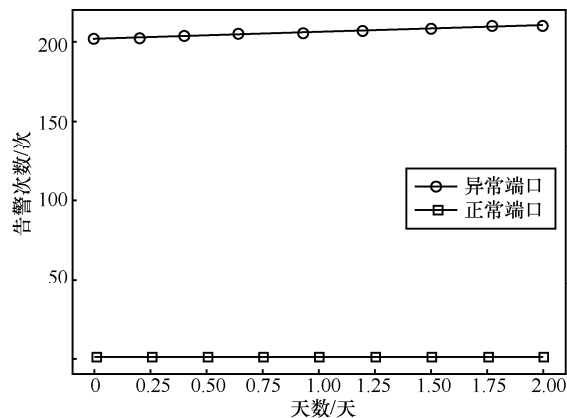


图7 前3天告警次数分布

## 4 实验方法

将用分类问题的思路判断给定进程或端口的状态。进一步地,由于状态只有异常和正常两种,因此目标简化为二分类<sup>[14-18]</sup>问题:给定一个进程/端口的相关描述,输出该进程/端口所处的状态(异常与否)。

### 4.1 训练方法

#### 4.1.1 进程网络

将第3.1节中提及的非结构化和结构化特征组合起来作为一个进程的基本特征向量。除此之外,还考虑基本特征随时间的变化量:给定进程 $i$ ,时间戳 $T_0$ 和其截至 $T_0$ 时刻所出现的日志记录的时间戳序列 $T_i, i \geq 0$ ,满足:

$$T_{k+1} < T_k, k \geq 0 \quad (1)$$

定义进程 $i$ 在 $T_0$ 时刻的 $k$ 阶特征变化量为:

$$\Delta f_k = g\{f(T_0) - f(T_k), T_0 - T_k\} \quad (2)$$

其中,  $f(T)$  为  $T$  时刻进程  $i$  的基本特征向量,  $g(v, t)$  为  $\text{time\_decay}^{[19]}$  函数, 定义为:

$$g(v, t) = vte^{1-t} \quad (3)$$

即对向量  $v$  乘时间间隔系数  $te^{1-t}$ 。

将上述  $k$  ( $k$  为参数) 阶特征变化量和基本特征向量拼接起来, 得到一个进程的完整特征, 作为样本数据的输入。

根据样本的输入特征, 在训练集上训练得到 CNN, 对测试集中的每个进程或输出判断结果, 用

准确率、召回率和 F1 得分来衡量分类器的优劣。

其中, 由于正负样本比过小(正样本过少), 本文采用过采样(over-sampling)<sup>[20]</sup>的方法生成训练集, 用交叉验证(cross-validation)<sup>[21-22]</sup>的方式对分类器进行训练, 然后对正样本极少的原始数据集进行测试。

#### 4.1.2 端口网络

类似地, 对于端口的异常预测这一问题, 人工提取第3.2节中提及的各个特征作为输入, 通过训练集得到一个 CNN 分类器, 对测试集中的每个进程或输出判断结果, 用准确率、召回率和 F1 得分来衡量分类器的优劣。

同样, 由于正负样本比过小(正样本过少), 本文采用部分随机采样(partly-random sampling)的方法对负样本进行采样, 以平衡正负样本比生成训练集, 然后用交叉验证的方式对分类器进行训练, 对正样本极少的原始数据集进行测试。

### 4.2 标签的提取

对于进程网络, 按照定义4给出的方式, 人工从原始数据集中提取进程标签(正负样本)。即首先过滤日志得到常驻进程, 对每个常驻进程, 判断其上一条时间戳的日志记录的进程号是否与当前时间戳的进程号一致。若不一致, 则认为在当前时间戳该进程发生崩溃, 采集为正样本, 否则为负样本。

而端口网络数据集中含有标签信息, 直接提取即可。

## 5 实验结果

### 5.1 进程网络

本数据集(上海电信 CSB 服务器集群的进程日志)时间跨度为2016年8月30日14—18时, 共2 858 063条日志记录。其中, 常驻进程973个, 进程崩溃次数为25次, 正负样本比为0.16%。

本文随机地对正负样本进行分割, 使训练集和测试集的大小一致。由于正负样本比过低,

本文采用正样本过采样和交叉验证的方式进行训练, 每次训练和测试过程重复 10 次, 结果取平均值。

首先考虑具体的某个基本特征对分类效果的影响。

图 8 给出了不同特征对分类结果的影响的比较。横向地和没有剔除任何特征的分类器相比, 可以发现, CPU、内存使用量以及远端进程数量这 3 个特征对区分正负样本的作用是显著的, 剔除其中任何一个都会使 F1 得分有明显的下降。这个结果和特征的分布具有一致性, 因为正负样本的 CPU 占用率和内存使用量的分布有着明显的区别; 而对于剔除节点的度、边权和以及图的结构化特征中心度和紧密性, 可以发现随着正负样本比的降低, F1 得分反而在增加。

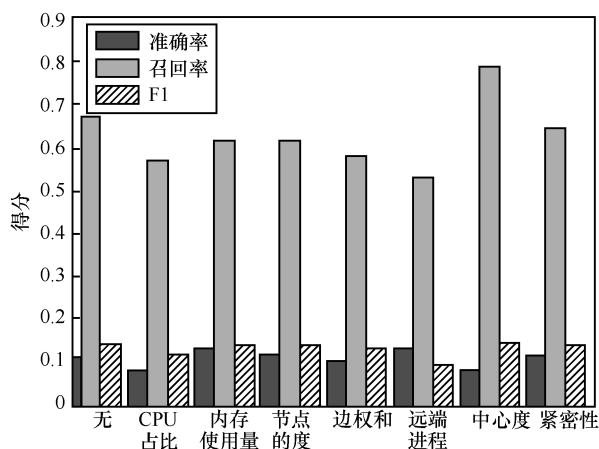
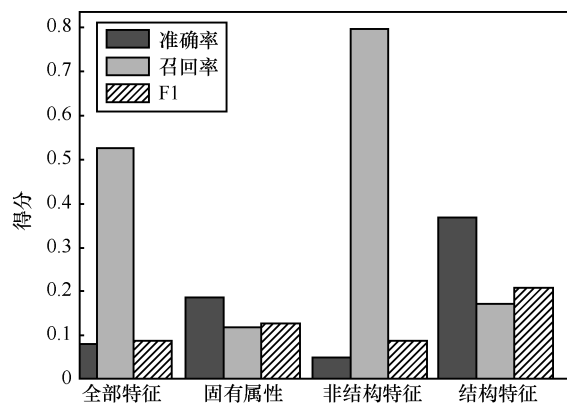


图 8 单个特征对结果的影响

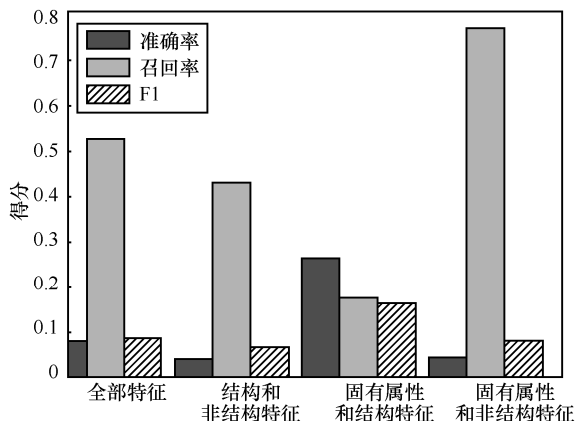
进一步地, 为了考虑不同特征之间的相关性的影响, 把上述的特征分为 3 类: 进程的运行信息, 即 CPU 和内存使用情况; 进程的通信情况, 体现为进程网络中节点的度 (或带权重的边权和) 以及存在通信的远端进程数量; 进程网络中节点的结构特征, 即中心度和紧密性。

图 9 展示了不同特征对分类结果的影响。可以看到, 第二类特征即进程的通信情况, 极大地提高了分类器的召回率, 但是准确率很低, 这是因为有大量的负样本在该特征上和正样本具

有相同的值, 在只有第二类特征的情况下, 分类器倾向于认为大部分样本都是正样本。因而崩溃预测在没有第二类特征的分类器上达到了最好的效果。



(a) 单组特征对分类的影响



(b) 组合特征对分类的影响

图 9 单组特征和组合特征对分类的影响

而第一类和第三类特征都可以在一定程度上反映进程的状态。这是因为正如在图 9 中, 正负样本的 CPU 和内存使用情况的分布不同, 而第三类特征即进程节点的中心度, 尽管分布相似, 但二者具有紧密的相关性, 结合在一起考虑便可以作为崩溃检测的指标之一。

此外, 图 10 给出了特征变化量对分类结果的影响的比较, 其中  $k=i$  表示将  $1, 2, \dots, i$  阶特征变化量均拼接加入进程的特征中。可以明显地看到, 特征变化量对分类结果有正相关的影响, 即考虑特征变化量越充分 ( $k$  越大), 分类结果越好。

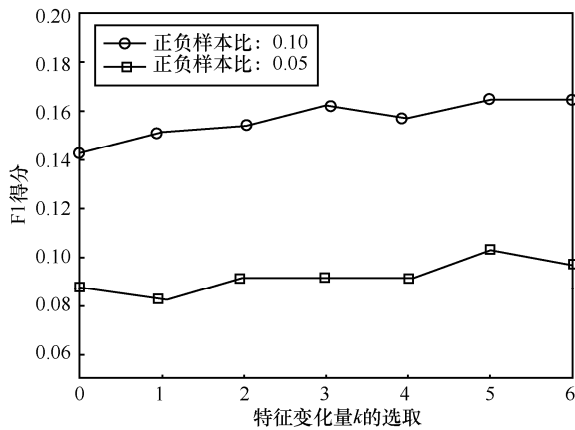


图 10 不同特征变化量对应的 F1 得分

这说明进程的崩溃不是突然的, 一个进程在发生崩溃的前后, 其 CPU、内存使用情况以及和其他进程的通信等属性往往会有突出的变化: 比如在 PC (个人计算机) 上, 往往一个进程占用内存过大会出现崩溃, 崩溃前内存使用量增加的趋势则反映了其崩溃的可能性。

实验结果表明, 将时间信息加入进程的特征进行训练, 得到了效果更好的分类器以对进程的崩溃进行检测。

## 5.2 端口网络

本文选取了上海电信 IDC 的 2017 年 4 月份的端口流量监控记录, 日活跃端口数量均值为 6 175.5。

本文将 4 月 1—13 日的数据作为训练集训练分类器, 用 4 月 14—20 日的数据作为测试集。

如表 1 所示, 训练得到的分类器对测试集的预测是较为准确的, 平均 F1 得分可以达到 0.8 左右。

表 1 端口网络实验结果

| 测试日期     | 正负样本比  | 准确率   | 召回率   | F1 得分 |
|----------|--------|-------|-------|-------|
| 4 月 14 日 | 0.850% | 0.941 | 0.640 | 0.762 |
| 4 月 15 日 | 0.882% | 0.973 | 0.655 | 0.783 |
| 4 月 16 日 | 0.751% | 0.953 | 0.891 | 0.921 |
| 4 月 17 日 | 0.669% | 0.971 | 0.805 | 0.880 |
| 4 月 18 日 | 0.703% | 0.723 | 0.791 | 0.756 |
| 4 月 19 日 | 1.42%  | 0.927 | 0.432 | 0.589 |
| 4 月 20 日 | 0.866% | 0.951 | 0.735 | 0.830 |

## 6 结束语

本文针对上海电信的进程网络和网络端口, 分析了网络中节点的有关属性, 选取节点的特征进行训练以对节点的异常状态进行预测。

通过仔细地选取特征和相应的预测结果, 可以得到如下结论。

- 由于数据集上正样本的稀疏性, 训练数据的正负样本比对训练结果有显著的影响。
- 相比于进程的通信情况, 进程节点的结构特征以及进程运行信息 (如 CPU 占用率、内存使用量等) 对于判断一个进程是否会崩溃更具参考价值。
- 从时间的维度看, 进程的运行和通信信息的变化量更能反映该进程的状态。
- 从时间的维度看, 端口过去 3 天的告警次数和出入流量方差更能区分其是否异常。

针对进程故障预测这一问题, 未来还可以从以下两方面着手考虑。

- 类比社交网络中的信息扩散<sup>[11-12]</sup>, 可以把进程崩溃看作进程的一种行为, 通过对进程节点之间的影响力进行建模, 可以预测进程的崩溃情况。同样的, 对于网络端口来说, 对网络上的边进行建模 (如端口的通信), 也可以将端口出现异常视作在端口网络中扩散的一种行为。
- 除了人工地选取进程特征进行训练, 还可以运用图表示<sup>[23]</sup>的方法对进程和端口网络进行建模, 用 embedding 的结果作为特征训练分类器。

## 参考文献:

- [1] DAMANI O P, CHUNG P E, HUANG Y, et al. ONE-IP: techniques for hosting a service on a cluster of machines[J]. Computer Networks and ISDN Systems, 1997, 29(8-13): 1019-1027.
- [2] MOOLENBROEK D V C, APPUSWAMY R, TANENBAUM

- A S. Integrated system and process crash recovery in the loris storage stack[C]//Networking, Architecture and Storage(NAS), 2012 IEEE 7th International Conference, October 14-17, 2012, Seoul, Korea (South). Piscataway: IEEE Press, 2012: 1-10.
- [3] MOHAMED E, ABDEL-WAHAB H, SALAMA I. Multicast address management in the internet: a study of the port blocking problem[R]. 1999.
- [4] NEWMAN M, BARABASI A L, WATTS D J. The structure and dynamics of networks[M]. Princeton: Princeton University Press, 2006: 419-421.
- [5] GRANOVETTER M. The strength of weak ties[J]. American Journal of Sociology, 1973, 78(6): 1360-1380.
- [6] ONNELA J P, SARAMAKI J, HYVONEN J, et al. Structure and tie strengths in mobile communication networks[J]. The National Academy of Sciences, 2007, 104(18): 7332-7336.
- [7] CHORMANSKI K, MATUSZAK M, MIEKISZ J. Scale-free graph with preferential attachment and evolving internal vertex structure[J]. Journal of Statistical Physics, 2013, 151(6): 1175-1183.
- [8] MILGRAM S. The small world problem[J]. Psychology Today, 1967, 2(1): 185-195.
- [9] FREEMAN L. A set of measures of centrality based on betweenness[J]. Sociometry, 1977, 40(1): 35-41.
- [10] NEWMAN M. Networks: an introduction[M]. Oxford: Oxford University Press, 2010.
- [11] GOMEZ-RODRIGUEZ M, LESKOVEC J, KRAUSE A. Inferring networks of diffusion and influence[J]. ACM Transactions on Knowledge Discovery from Data, 2010, 5(4): 1-37.
- [12] RODRIGUEZ M G, BALDUZZI D, SCHOLKOPF B. Uncovering the temporal dynamics of diffusion networks[C]//The 28th International Conference on Machine Learning (ICML), June 28-July 2, 2011, Bellevue, Washington, USA. [S.l.:s.n.], 2011: 561-568.
- [13] LIBEN-NOWELL D, KLEINBERG J. The link prediction problem for social networks[J]. Journal of the Association for Information Science & Technology, 2007, 58(7): 1019-1031.
- [14] LAST M. Kernel methods for pattern analysis[M]. Beijing: China Machine Press, 2005.
- [15] KIM I. Convolutional neural networks for sentence classification[J]. arXiv: 1408.5882, 2014.
- [16] CONKLIN J D. Applied logistic regression[J]. Technometrics, 2013, 44(1): 81-82.
- [17] BRANDES U. A faster algorithm for betweenness centrality[J]. Journal of Mathematical Sociology, 2001, 25(2): 163-177.
- [18] SABIDUSSI G. The centrality index of a graph[J]. Psychometrika, 1966, 31(4): 581-603.
- [19] JAPKOWICZ N. The class imbalance problem: significance and strategies[C]//The 2000 International Conference on Artificial Intelligence(IC-AI'2000), June 26-29, 2000, Las Vegas, USA. [S.l.:s.n.], 2000.
- [20] GEISSER S. Predictive inference: an introduction [M]. New York: Chapman and Hall, 1993.
- [21] KOHAVI R. A study of cross-validation and bootstrap for accuracy estimation and model selection[C]//The Fourteenth International Joint Conference on Artificial Intelligence, August 20-25, 1995, Montreal, Quebec, Canada. New York: ACM Press, 1995: 1137-1143.
- [22] LEIKE, A. Demonstration of the exponential decay law using beer froth[J]. European Journal of Physics, 2002, 23(1): 21.
- [23] MOHAR B. A linear time algorithm for embedding graphs in an arbitrary surface[J]. SIAM Journal on Discrete Mathematics, 2006, 12(1): 6-26.

## [作者简介]



蔡珩 (1976-), 女, 中国电信股份有限公司上海分公司工程师, 主要研究方向为 IT 智慧运营、利用大数据技术提升系统运维的智能化。



戈磊 (1973-), 男, 中国电信股份有限公司上海分公司企业信息化部高级项目经理, 主要研究方向为云计算、开源架构、大数据分析、Devops 运营、流程生命周期管控等。