计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2018.20170804
55(3):524-536,2018

边缘计算应用:传感数据异常实时检测算法

张 琪 胡宇鹏 嵇 存 展 鹏 李学庆
(山东大学计算机科学与技术学院 济南 250101)
(d.steven@sdu.edu.cn)

Edge Computing Application: Real-Time Anomaly Detection Algorithm for Sensing Data

Zhang Qi, Hu Yupeng, Ji Cun, Zhan Peng, and Li Xueqing
(*School of Computer Science & Technology, Shandong University, Jinan* 250101)

Abstract With the rapid development of Internet of things (IoT), we have gradually entered into the IoE (Internet of everything) era. In face of the low quality of real-time gathering sensor data in IoT, this paper proposes a novel real-time anomaly detection algorithm based on edge computing for streaming sensor data. This algorithm firstly expresses the corresponding sensor data in the form of time series and establishes the distributed sensing data anomaly detection model based on edge computation. Secondly, this algorithm utilizes the continuity of single-source time series and the correlation between multi-source time series to detect anomaly data from streaming sensor data effectively and respectively. The corresponding anomaly detection result sets are also generated in the same process. Finally, the above two anomaly detection result sets would be effectively fused in a certain way so as to obtain more accurate detection result. In other words, this algorithm achieves a higher detection rate compared with other traditional methods. Extensive experiments on the real-world dataset of household heating data from the Jinan municipal steam heating system, which collects monitoring data from 3 084 apartments of 394 buildings, have been conducted to demonstrate the advantages of our algorithm.

Key words Internet of things; edge computing; time series data; correlation; outlier distance

摘 要 随着物联网技术的不断发展,已逐步进入“万物互联”的新时代.针对物联网中实时采集的传感数据总体质量低下的问题,提出基于边缘计算的传感数据异常实时检测算法.该算法首先对相应的传感数据以“时间序列”的形式进行表示,并建立基于边缘计算的分布式传感数据异常检测模型;其次利用单源时间序列自身的连续性以及多源时间序列之间的相关性,分别对实时传感数据中出现的数据异常进行有效检测,并分别形成相应的异常检测结果集;最后将上述2个异常检测结果集进行有效地融合处理,从而得到更加准确的异常数据检测结果.通过实验验证该算法的检测准确性和有效性,结果显示:该算法检测时间短并且异常检出率高.

收稿日期:2017-10-23;**修回日期**:2017-12-11
基金项目:国家重点研发计划项目(2016YFB1001100);山东省重点研发计划项目(2015GGX101009)
This work was supported by the National Key Research and Development Program of China (2016YFB1001100) and the Key Research and Development Program of Shandong Province (2015GGX101009).
通信作者:胡宇鹏(huyupeng@sdu.edu.cn)

关键词　物联网;边缘计算;时序数据;相关性;离群距离

中图法分类号　TP391

近年来,随着互联网、物联网、云计算等技术的不断发展与相互融合,我们已经逐步进入“万物互联、全面感知”的互联网新时代[1].近年来物联网的发展导致了大量的数据传感设备广泛应用于不同的领域,例如金融与能源行业、化工与石油行业、生物和医药行业等等.这些传感设备在改变人们的生活方式的同时也产生了海量的时序数据资源.以浙江省的电力网络为例,截止目前整个网络中已经部署了2000万台以上的智能电表设备,每台电表以1 Hz为单位采集“时序”用电信息,全省一年采集的用电数据量已经达到了拍字节规模(petabyte, PB)[2].

尽管传感设备的类型各有不同,但其获取的数据可以分为2类:1)实时状态数据.表示某个时刻运行状态,例如速度、功率等;2)累加数据.表示一定范围内的数据累加量,例如里程、热消耗等.通常情况下,传感器以一定的频率采集数据,并将数据发送至相应的数据接收端.数据接收端将收到一组或多组在时间上存在严格先后顺序的一组或多组观测值序列,即“时间序列数据”.这些时序数据精准地记录着某个具体参数的实时变化情况,并在一定的时间范围内反映该参数的发展趋势和变化规律.因此基于传感设备所采集的时间序列数据不仅为后续的数据趋势展现等数据可视化工作提供了重要的数据来源,同时也是后续数据挖掘工作(分类、聚类、关联、预测)的基础与前提.

以中国空间技术研究院的卫星总装集成测试(assembly, integration and test, AIT)为例,针对整颗卫星的AIT测试被分为11个不同的测试阶段,每个测试阶段平均需要高速采集5000多个卫星参数的“时序”状态数据.AIT测试人员,可以根据所获取到的卫星参数时序数据来判断卫星的某个具体部件的运转状态是否正常,而卫星设计师们则可以根据卫星在不同测试环节中的各个部件运转状态的汇总情况,得出相应的卫星整体“健康度”分析结论.而得出正确的“整星分析”结论的前提条件是传感器采集并传输的数据是可靠的.

但是在实际的数据采集场景中,传感设备在数据采集与传输的过程中,总是会出现一些异常,文献[3]针对实际的传感器数据集进行了数据异常检测的相关研究,研究结果表明:通过传感器获取高质量的数据是一件非常困难的事情,因为故障的出现次数、频率等都让人无法预料,并且相应的数据清洗与校准工作也并不容易.

根据上述情况的介绍,我们希望采取从时间序列数据分析的角度,对传感数据中出现的相应数据异常进行有效地识别,帮助后续的数据清洗工作对相应的数据异常进行有效的“平滑”操作,并保留传感数据中正常的数据波动情况,从而为后续的时间序列数据挖掘研究工作提供高质量的来源数据.

目前已有一些异常检测算法被提出.从早期的基于统计的检测算法[4]、基于距离的检测算法[5-6]、基于密度的检测算法[7]、基于神经网络的方法[8]、基于支持向量机的方法[9]以及聚类分析的方法[10]等.时序数据具有一些特殊的性质,异常检测算法应该考虑其特性.如Fu所言,在时序数据领域,异常检测的方法大部分都是基于模式识别与聚类进行异常检测[11].Vlachos等人[12]提出了准确周期检测的非参数方法并引进了时间序列新的周期距离策略.相似地,Keogh等人[13]通过检测当前数据与预制数据模型之间是否存在较大差异,从而实现时序数据的异常检测.Keogh等人[14]后来引入了一个基于检测数据象征性版本的特定的重新排序策略.除了离线方法以外,还存在一些在线检测算法.Chan等人[15]从多个时序数据生成了可理解的准确模型来进行异常检测.Wei等人[16]利用时间序列位图来进行异常检测.Fujimaki等人[17]设计了一个应用于大量遥感数据的新颖的异常检测系统,该系统主要利用相关向量的回归和数据的自回归进行异常检测.周大镯等人[18]在基于重要点(important point, IP)时间序列分割的基础上,提出了基于k近邻的局部异常检测算法.Cai等人[19]通过构建分布式递归计算策略以及k近邻快速选择策略提出了一种新的时间序列数据异常检测算法.

然而这些方法主要是针对单一传感来源数据进行相应的异常监测工作,而在传感网络中不同传感来源数据之间往往存在着大量“已知”的相关关系,具有相关关系的传感数据之间则肯定存在着一定的数据趋势变化规律,这些变化规律可以帮助我们对相应的数据异常进行有效的识别.

此外,需要特别强调的是:目前常见的传感数据异常检测处理方式是利用相对成熟的云计算模型[20]以及常见的大数据处理产品:Hadoop[21],Spark[22]

等,将各种数据采集设备所获取的数据直接传输到云计算中心进行数据存储,并利用云计算中心强大的计算能力完成相应的异常检测与数据清洗工作.这种方式也被称为:基于云计算的集中式大数据处理模式.根据思科、国际数据公司(International Data Corporation, IDC)等机构的相关研究表明,到2020年连接到网络的无线设备数量将达到500亿台,随着边缘设备数据量的增加,网络边缘设备产生的数据,受网络带宽与云数据中心计算能力的限制,已经无法像过去那样直接传输到数据中心进行相应的数据操作.因此需要对现有的集中式大数据处理模型进行相应的调整,将云计算模型的部分计算任务迁移到网络边缘设备上,在减缓网络带宽压力的同时,降低数据中心的计算负载.

根据目前基于云计算的集中式大数据处理模式所面临的限制和瓶颈,本文提出基于边缘计算的分布式传感数据异常检测模型.利用边缘计算的大数据处理思想,尽可能地将相应的数据在接近数据源的计算资源上进行相应地处理,在减轻网络传输带宽压力的同时,提高了数据处理的整体效率.在分布式传感数据异常检测模型的基础上,本文提出了基于时间序列的传感数据异常检测算法(anomaly detection of multi-source sensing data based on time series, ADMSD_TS).本算法将根据时间序列数据的离群距离测算以及时间序列之间的相关性对相应的传感数据异常进行检测.即通过对"单一"时间数据序列中离群点的识别以及对具有相关关系的"多源"时间序列之间的数据变化趋势为检测基础,高效识别出多源传感数据集中的相应数据异常.实验结果显示本算法性能良好,检测时间短并且异常检出率高.

1 问题定义

根据引言的介绍,本节将给出基于时间序列的数据异常检测问题的相关定义.

定义1. 传感器所采集并传输的多源传感数据,可以按照时间序列数据的形式,简化表示为

$$TS_m=\{S_1,S_2,\cdots,S_i,\cdots,S_m\},\tag{1}$$

$$S_i=\{s_1,s_2,\cdots,s_j,\cdots,s_n\},\tag{2}$$

其中,$1\leqslant i\leqslant m$,$1\leqslant j\leqslant n$,TS_m 表示多源传感数据的时间序列表示数据集合,m 表示集合内的数据个数.式(1)中 S_i 表示单一传感数据,在式(2)中 n 表示 S_i 的长度.其中 s_j 表示某个具体采集时刻的数据值,$s_j=(v_j,t_j)$,t_j 表示 s_j 的时间标签,v_j 表示时刻 t_j 的数据值,在时间序列中 t_j 是严格递增的.

根据上面单一传感数据的时间序列表示形式 S_i,我们将引入滑动窗口(slide window, SW)[23],用来存放 S_i 的部分数据,设SW的长度为 Len_{sw} 并忽略SW中时间序列数据的时间标签,我们给出SW中时间序列数据的离群距离的定义.

定义2. SW中的部分时间序列数据可以简化表示为

$$ST_n=\{v_1,v_2,\cdots,v_t,\cdots,v_n\}\ (1\leqslant t\leqslant n),\tag{3}$$

其中 $n=Len_{sw}$,而 $v_t(1\leqslant t\leqslant n)$ 与SW中的全部数据 ST_n 的离群距离 $dis^t_{outlier}$ 表示为

$$dis^t_{outlier}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(v_1-v_2)^2},\tag{4}$$

而当前值 v_t 与其离群距离 $dis^t_{outlier}$ 的比值可以表示当前值 v_t 的相对离群距离,记为

$$dis^t_r=\frac{dis^t_{outlier}}{v_t}.\tag{5}$$

定义3. 根据多源时间序列 $TS_m=\{S_1,S_2,\cdots,S_m\}$ 中已知的某种相关性,对 TS_m 进行必要的组合与转换,从而得到满足多元线性相关性的时间序列 TS'_k,并将其放入相关性参数集合 Ω_k 中,记为 $\Omega_k=\{S'_1,,S'_2,\cdots,S'_k\}$.在随后的相关性检测(data correlation detection, DCD)中,对 Ω_k 中的线性相关性进行数据异常检测.

根据定义3,我们将以同一滑动窗口SW中的 TS_m 的线性相关性为出发点,进行相应的 TS_m 相关性检测.而 TS_m 中可能并不存在相应的线性相关性或者存在着非线性相关性,因此 TS_m 首先需要转换成为具有线性相关性的多源时间序列 TS'_k,才能保证后续DCD检测的顺利进行.基于以上考虑,TS_m 的已知相关性可以被分为3种类型:

1) 基本相关

基本相关也被称为线性相关性,以热力系统为例,热功率 P 与轮轴转动速度 V 所组成的二元时间序列 $TS_2=\{P,V\}$,其中 $P=(p_1,p_2,\cdots,p_i,\cdots,p_n)$ 与 $V=(v_1,v_2,\cdots,v_i,\cdots,v_n)(1\leqslant i\leqslant n)$,如满足 $v_i\approx kp_i+b$,则说明 P,V 之间存在二元线性相关性,并将其放入关系参数集合 Ω_2 中,记为 $\Omega_2=\{P,V\}$.类似地,如三元时间序列数据 $TS_3=\{S_1,S_2,S_3\}$ 满足三元线性相关性.则将其放入参数集合 Ω_3 中,记为 $\Omega_3=\{S_1,S_2,S_3\}$.

2) 组合相关

组合相关是指已知 TS_m 中的单一时间序列 S_i 之间并不存在线性相关性,但是 S_i 之间进行相应地组合,则存在相应的线性相关性. 同样以热力系统为例,$TS_3=\{P,W,t\}$ 中的热功率 P 与瞬时温度观测值 W 以及采样时间 t 之间并不存在基本线性相关,但是根据热力学中的基本热功率定理,不难发现热功率 P 与单位温差 $\Delta W/\Delta t$ 之间必然线性相关. 即 $p_i\approx k\Delta W_i/\Delta t_i+b$. 而 ΔW_i 可以根据 W 获得,即 $\Delta W_i=W_i-W_{i-1}$,同理我们也可以根据 t 获得 Δt. 因此原始的 $TS_3=\{P,W,t\}$ 中的时间序列,可以进行相应地组合,转变成为二元线性模型,并将转化后的二元时间序列 $TS'_2=\{P,\Delta W/\Delta t\}$ 放入参数集合 Ω_2 中,记为 $\Omega_2=\{P,\Delta W/\Delta t\}$.

3) 转换相关

转换相关是指已知 TS_m 中时间序列之间不符合基本相关以及组合相关的要求,而是满足某种非线性相关模型(指数模型、对数模型、多项式模型等等). 同样以热力系统的相关定理为例:流量优化系数 m 与表征散热器传热系数 b 满足双曲线模型:$m=\dfrac{b}{1+b}$,飞轮机械能 E 与转动惯量 J 以及飞轮角速度 ω 满足多项式模型:$E=\dfrac{J\omega^2}{2}$ 等. 我们可以通过相应的数据变换方法,将非线性相关模型转换成为线性相关模型.

① 指数模型. $y=ae^{bx}$ 对等式两边进行取对数操作,转换为 $\ln y=\ln a+bx$,我们将原时间序列 Y 转换为新的时间序列 $\ln Y$,并在随后的 DCD 中检测 $\ln Y$ 与 X 的线性相关性.

② 双曲线模型. $y=\dfrac{1}{a+bx}$,等式两边进行取倒数操作转换为 $\dfrac{1}{y}=a+bx$,将原时间序列 Y 转换为新的时间序列 $\dfrac{1}{Y}$.

③ 多项式模型. $z=ax^2+by+c$,将原时间序列 X 进行乘方操作变为新的时间序列 X^2,然后在 DCD 中检测 Z 与 X^2,Y 的线性相关性.

根据上述 TS_m 相关性类型的分析,我们不难发现:根据 $TS_m=\{S_1,S_2,\cdots,S_m\}$ 中已知的相关性,并对 TS_m 中的时间序列进行必要的组合操作或数值变化操作,从而得到新的多源时间序列 TS'_k,最后可以利用 TS'_k 表示 TS_m 的相关性. 即 $TS'_k=\{S'_1,S'_2,\cdots,S'_k\}$ 中存在多元线性模型 $f_{\text{linear}}(S'_i)=\boldsymbol{W}^{\mathrm{T}}S'_i+b$,$1\leqslant i\leqslant k-1$ 使得 $f_{\text{linear}}(S'_i)\approx S_k$,其中向量 $\boldsymbol{W}^{\mathrm{T}}$ 与 b 称为多元线性模型的系数. 不失一般性,我们可以将 $\boldsymbol{W}^{\mathrm{T}}$ 与 b 以向量 $\boldsymbol{W}=(\boldsymbol{W}^{\mathrm{T}};b)$ 的形式进行表示,并利用最小二乘法对 $\boldsymbol{W}$ 进行参数估计. 根据多元线性回归、矩阵计算等相应的先验知识,TS_m 中 m 的取值会对 $\boldsymbol{W}$ 参数估计的计算效率产生较大的影响,考虑到本文的研究目标,为多源传感流数据进行实时数据异常检测. 因此为了保证数据相关性检测的计算效率,当 TS_m 进行必要的组合与转换得到满足线性模型的时间序列 TS'_k 后,在不作其他约定的一般情况下,本文只考虑不超过五元线性模型的 TS'_k 相关性检测,即 $k\leqslant 5$.

2　传感数据异常实时检测算法

本文提出的 ADMSD_TS 算法将会从传感数据本身的时序连续性检测(temporal continuity detection, TCD)以及传感数据之间的相关性检测(data correlation detection, DCD)两个方面,对传感数据的异常进行相应地检测. 随后对这 2 种不同的检测结果进行相应地数据融合处理,完成最终的多源传感数据异常检测. 本节将介绍基于传感数据异常检测的边缘式数据处理架构以及 ADMSD_TS 算法的具体实现.

2.1　基于边缘计算的数据架构

在我们的前期研究工作中,我们提出了基于异构设备的数据提取模型,基于此模型从多个企业的物联网数据源中提取各种形式的异构数据,并将这些多源异构数据保存到我们构建的一体化的工业大数据平台(industrial big data platform, IBDP)中,IBDP 支持从数据提取、到数据处理、再到数据挖掘与可视化的智能大数据分析全过程[24].

根据引言部分的描述,我们不难发现:线性增长的集中式云计算能力已经无法满足数据量急剧增长的边缘数据的处理要求,此外将数据量持续增长的边缘数据集中到某个或某几个数据计算中心完成相应的数据计算任务,无论从技术上还是经济上来说,都将变得越来越不可行. 根据文献[25]中边缘计算相关内容的介绍以及系统架构的描述如图 1 所示,我们考虑将 IBDP 的部分数据计算任务进行相应地迁移,并在相应的数据源附近建立边缘层数据处理节点,"就近"完成相应的数据处理任务. 以本文所研究问题的具体场景为例,我们考虑在传感数据采集端附近,建立相应的边缘层数据节点,在接收传感数

据的同时,完成相关数据的异常检测任务.相应的系统整体架构如图 1 所示:

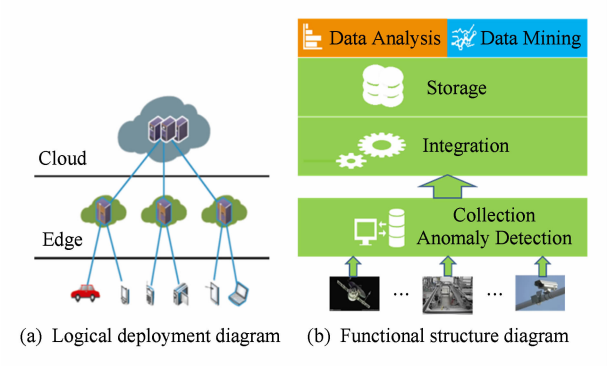

Fig. 1 System architecture diagram

图 1 系统整体架构图

根据图 1 的描述,我们将在数据采集层建立基于 Storm 的流数据处理结构,在进行数据接收的同时,对相应的传感数据进行异常检测.我们首先给出 ADMSD_TS 算法在 Storm 平台上的拓扑结构,如图 2 所示.

DataSpout 接收采集到的传感数据(data source),并将其发送至各个 TCDBolt,检查传感数据的时序连续性,如果通过 DataSpout 接收到的数据量大且参数类型繁多,还可以考虑通过数据划分模块(Partion)将数据进行相应的划分,并将划分后的数据传送至 TCDBolt 进行时序连续性检测.RelationSpout 将接收用户发送的传感数据不同参数之间的关系模型集(relation source),RelationSpout 会将相应的关系模型发送至 DCDBolt,用于 DCDBolt 检测传感数据之间的相关性,如果关系模型集相对庞大,则也可以考虑采用数据划分模块先将其划分并再次发送至相应的 DCDBolt.当 TCDBolt 完成时序相关性检测之后,多个 TCDBolt 将向对应的 DCDBolt 发送相应的传感数据,在 DCDBolt 进行数据相关性检查.与此同时,TCDBolt 也会将时序连续性的检查结果发送至 FusionBolt.等待 DCDBolt 中对应的相关性结果检测完毕以后,DCDBolt 也会将对应的相关性检测结果发送至 FusionBolt,完成最终的多源传感数据异常检测.用户也可以向 QuerySpout 发送相应的查询信息,QueryBolt 会接收到用户的查询信息,并按照用户的要求查询相应的数据异常情况并进行输出.需要说明的是,如果在硬件设备条件允许的情况下(大容量 SSD 磁盘阵列或者大容量的内存)我们可以对 DataSpout 接收到的传感数据进行复制,并分别向 TCDBolt 与 DCDBolt 进行发送,同时进行相应地相关性检测与连续性检测,并将检测结果发送至 FusionBolt 进行相应的数据融合并完成多源传感数据异常检测.有关性能优化的相关工作,将会在我们今后的研究中进行分析与讨论.

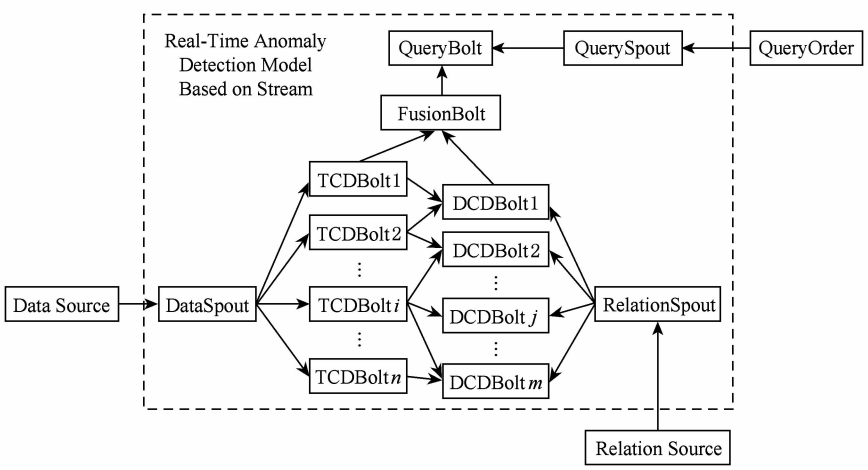

Fig. 2 Topology diagram of ADMSD_TS

图 2 ADMSD_TS 拓扑结构图

2.2 基于离群距离与序列相关性的异常检测

本节介绍基于时序数据的离群距离以及序列相关性的异常检测算法,该算法主要分为 3 步:1)对多元传感数据进行数据相关性检测(data correlation detection, DCD);2)对一元传感数据进行时序连续性检测(temporal continuity detection, TCD);3)对前 2 步的异常检测结果进行融合处理(fusion process, FP),获取最终的检测结果.

2.2.1 基于相对离群距离的数据异常检测

以时间序列形式表示的传感数据具有时间上的连续性和稳定性,根据定义 2 中离群距离与相对离群距离的相关定义,我们利用滑动窗口 SW 对单源

传感数据进行基于离群距离的数据异常检测.主要的检测步骤如下:首先设置滑动窗口 SW 的长度为 Len_{sw}、滑动窗口中子序列的最小长度阈值 ε_{size},子序列移动距离 len_{move} 以及相对离群距离阈值 ε_{dis}.其次利用式(7)(8)对进入滑动窗口的时间序列数据依次进行相对离群距离 dis_r^t 计算,当某个时序数据值 v_t 的相对离群距离超过阈值 ε_{dis} 且包含 v_t 的当前序列 ts 长度大于 ε_{size} 时,我们将选择以 v_t 为中心的子序列:$ts_{sub}=(v_i-len_{move}-1,v_i+len_{move})$ 进行再次计算 v_t 的相对离群距离 dis_r^t.以此方式进行循环操作,当出现 $dis_r^t>\varepsilon_{dis}$ 且其长度小于 ε_{size} 时,将传感数据在 t 时刻的数值记为异常数据,并将其放入 Ω_E 中.相应的算法实现如算法 1 所示.

算法 1. 时间序列连续性检测 TCD.

输入:时间序列数据 TS、滑动窗口 SW 长度 Len_{sw}、子序列移动距离 Len_{move}、滑动窗口的最小长度阈值 ε_{size} 以及相对离群距离阈值 ε_{dis};

输出:异常参数集合 Ω_{ab}.

① $\Omega_{ab}=\varnothing$; /* 初始化参数集合 */
② Hashmap $mapForTCD=new\ HashMap()$;
/* 建立新的 Haspmap 用于存储异常参数 */
③ $qTS=InitQueue(Len_{sw})$;
/* 初始化异常数据队列 */
④ $listSW=InitList(Len_{sw})$;
/* 初始化滑动窗口列表 */
⑤ while $TS.length()>Len_{sw}$
⑥ $calcSWDis(SW,TS,Len_{sw})$; /* 从 TS 中输出 Len_{sw} 进入 SW,并计算 $dis_{outlier}^t$ */
⑦ for each v_i $value$ in SW
⑧ if $dis_{outlier}^t/v_i>\varepsilon_{dis}$ && $Len>\varepsilon_{size}$ /* 如果 v_i 的相对离群距离大于 ε_{dis} 且当前序列长度大于 ε_{size} */
⑨ $ts=(v_i-len_{move}-1,v_i+len_{move})$;
⑩ $ts_{sub}=\{ts,v_i,len_{move}\}$;
⑪ $qTS.enQueue(ts_{sub})$; /* 将包含的子序列 ts_{sub} 放入队列 qTS 中 */
⑫ end if
⑬ end for
⑭ while $qTS.length()\neq 0$ /* 从队列 qTS 中选取 ts_{sub} 再次进行判断 */
⑮ $ts_{sub}=qTS.deQueue()$;
⑯ if $calcValueDis(t_{sub})>\varepsilon_{dis}$ && $t_{sub}.length<\varepsilon_{size}$ /* 如果 v_t 在 ts_{sub} 中的 ε_{size} 依然大于 ε_{dis} 且 ts_{sub} 的长度已经小于 ε_{size} */
⑰ $\Omega_{ab}=\Omega_{ab}\cup ts_{sub}$ /* 将 ts_{sub} 放入异常集合 Ω_{ab} */
⑱ else /* 否则再次缩小 ts_{sub} 的移动长度 len_{move},建立新的 ts_{sub} */
⑲ $ts_{sub}.len_{move}=ts_{sub}.len_{move}/2$;
⑳ $ts=(ts_{sub}.value-len_{move}-1,ts_{sub}.value+len_{move})$;
㉑ $ts_{sub}.ts=ts$;
㉒ $ts_{sub}.len_{move}=len_{move}$;
㉓ $qTS.enQueue(ts_{sub})$;
㉔ end if
㉕ end while
㉖ end while
㉗ if $\Omega_{ab}\neq\varnothing$ /* 循环结束取出 Ω_{ab} 中的异常存入 Hashmap 中 */
㉘ for each ts_{sub} in Ω_{ab}
㉙ $mapForTCD.put(abID,ts_{sub})$;
/* 将异常参数存入 Hashmap 中 */
㉚ end for
㉛ end if
㉜ Return $mapForTCD$. /* 输出保存异常的 Hashmap,算法结束 */

算法 1 主要利用传感数据自身的时序连续性并通过计算相对离群距离,对传感数据中可能出现的异常进行检测.本算法可以检测单源传感数据的数据异常情况,检测情况如图 3 所示,通过 TCD 算法,可以检测出该传感数据在框 1、框 2、框 3 中的数据异常情况.

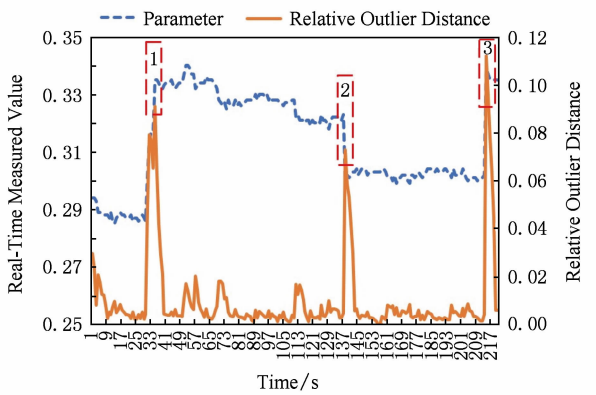

Fig. 3　Data abnormal detection for TCD

图 3　TCD 数据异常检测

算法 1 只考虑了单源传感数据内在的时序连续性,而忽视了多源传感数据之间的相关性.因此只利用 TCD 进行传感数据的异常检测,可能存在部分数据异常无法有效发现的问题.

2.2.2 基于参数相关性的数据异常检测

传感器获取的传感数据之间往往具有一定的相关性. 我们可以利用传感数据之间的相关性来判断某个传感数据在一定的时间范围内是否出现了异常.

根据定义 1,多源传感数据集合的时间序列表示:$TS_m=\{S_1,S_2,\cdots,S_m\}$,当多源传感数据 TS_m 进入滑动窗口 SW 时,我们将选取 SW 中滑动窗口长度 Len_{sw} 的离散数据观测值,组成多源传感数据集合 $TS'_m=\{S'_1,S'_2,\cdots,S'_m\}$,其中 $S'_i(1\leqslant i\leqslant m)$ 中包含 Len_{sw} 长度的传感数据观测值,即 $S'_i=\{v_{i1},v_{i2},\cdots,v_{it}\}$,$1\leqslant t\leqslant Len_{sw}$.

根据定义 3,我们将 TS'_m 中满足已知相关性的部分时间序列集合 TS'_{sub} 进行必要的组合与转换操作,使其成为具有线性相关性的多源时间序列 TS'_k,并将其分别放入不同的参数集合 Ω_k 中,随后在 DCD 操作中分别验证 Ω_k 中的传感数据实际观测值之间是否满足相应的线性相关性约束.

下面我们以定义 3 中的基本相关为例(组合相关、转换相关的处理流程与基本相关类似),说明 DCD 的异常检测流程.

例如在热力系统中,温度恒定时,热功率 P 与轮轴转动速度 V 是线性相关的. 根据 Pearson 线性相关系数计算公式,可以求出 P 与 V 的线性相关系数:

$$r=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{i=1}^{n}(y_i-\bar{y})^2}},\tag{6}$$

其中,x,y 是 2 个不同的参数观测值,$\bar{x},\bar{y}$ 是参数 x,y 的平均值.

另外,利用最小二乘法来求解线性逼近函数 $y=kx+b$:

$$k=\frac{\sum_{i=1}^{n}x_iy_i-n\bar{x}\bar{y}}{\sum_{i=1}^{n}x_i^2-n\bar{x}^2},\tag{7}$$

$$b=\bar{y}-k\bar{x},\tag{8}$$

其中,$x,y,\bar{x},\bar{y}$ 的含义与式(6)相同.

根据定义 3,可以将热功率传感数据与流速传感数据以时间序列的形式表示为 S_P,S_V. 利用式(6)验证 2 个参数之间的线性相关性,并利用式(7)与式(8)来获取热功率 P 与轮轴转动速度 V 的时间序列集合 $TS_2=\{S_P,S_V\}$ 的二元线性模型的系数 k 与 b. 最后将 TS_m 放入参数集合 Ω_2 中,随后在 DCD 检测中验证 TS_2 中时序数据实测值之间是否满足相应的线性相关性约束.

我们还是以热力系统传感数据 S_P,S_V 为例,如果 (S_P,S_V) 满足参数 Ω_2 的线性约束条件,根据式(8),表达式 $\frac{S_{vt}-b}{S_{pt}}$ 的取值必然在 $(k-\varepsilon,k+\varepsilon)$ 的范围内,如果表达式的取值超出了的约束范围,则认为传感数据 S_P,S_V 出现异常. 经过验证,如果 TS_2 满足 Ω_2 的线性相关约束,则将 TS_2 合并到正确参数集合 Ω_R 中,否则将其放入错误参数集合 Ω_E 中. 当全部的相关性验证完毕之后,我们将输出异常参数集合 Ω_{ab}($\Omega_{ab}=\Omega_E-\Omega_R$). 例如错误参数集合 $\Omega_E=\{S_1,S_2,S_3\}$,正确参数集合 $\Omega_R=\{S_2,S_3,S_4\}$,那么得到的异常参数集合 $\Omega_{ab}=\{S_1\}$. DCD 算法如算法 2 所示.

算法 2. 基于数据相关性的异常检测 DCD.

输入:参数集合 Ω_k 列表、基于 SW 的传感数据集合 TS;

输出:异常参数集合 Ω_{ab}.

① $\Omega_R=\Omega_E=\Omega_{ab}=\varnothing$; /* 初始化参数集合 */

② Hashmap $mapForDCD=new\ HashMap()$;

/* 建立新的 Haspmap 用于存储异常参数 */

③ $qPare=InitQueue(\Omega_k)$;

/* 初始化参数集合队列 */

④ $listTS=InitList(TS)$;

/* 初始化传感数据集合列表 */

⑤ while $qPare.length()\neq 0$

⑥ $item=qPare.Dequeue()$;

⑦ for each parameter p_i in $item$

⑧ $st_{p_i}=listTS.get(p_i)$; /* 根据 p 取出相应的传感时序数据 */

⑨ end for

⑩ $corr=corrDetc(item,st_{p_i},st_{p_j}\cdots)$;

/* 验证相关时序数据是否满足参数集合的约束 */

⑪ if $corr$ is true /* 满足相关性约束 */

⑫ $\Omega_R=\Omega_R\cup item$; /* 将相关时序数据表示的参数并入 Ω_R */

⑬ else $\Omega_E=\Omega_E\cup item$;

/* 否则将参数并入 Ω_E */

⑭ end if

⑮ end while

⑯ $\Omega_{ab}=\Omega_E-\Omega_R$; /* 得到异常参数集合 */

⑰ if $\Omega_{ab}\neq\varnothing$

⑱　for each parameter ab_i in Ω_{ab}

⑲　　$mapForDCD.put(abID,ab_i)$; /* 将异常参数存入 Hashmap 中 */

⑳　end for

㉑ end if

㉒ Return $mapForDCD$. /* 输出保存异常的 Hashmap,算法结束 */

算法 2 主要利用传感数据之间的相关性,对传感数据中可能出现的异常进行检测.但该算法只考虑了传感数据之间的相关性,而忽视了传感数据自身的时序连续性.因此只利用 DCD 进行传感数据的异常检测,可能存在 2 个问题:

1) 如果参数集合 Ω_k 中元素较少,很难利用传感数据的相关性对异常的传感数据进行准确的定位.假设参数集合 Ω_2 中只存在一组线性相关序列 $TS_2=\{S_1,S_2\}$.经过 DCD 的相关性检测,我们发现传感数据(S_1,S_2)中存在数据异常情况.如图 4 中框 1、框 2、框 4 所示.根据图 4,我们只知道(S_1,S_2)中存在数据异常情况,但是到底是 S_1 存在异常、还是 S_2 存在异常、还是两者全部存在数据异常则无法进行更加准确的异常数据定位.

2) 如果参数集合 Ω_k 中所有参数同时发生异常,则相应数据异常可能无法被 DCD 成功检测.根据图 4 中框 3 所示,当 $TS_2=\{S_P,S_V\}$ 中 S_P 与 S_V 同时出现数据异常,且出现异常的数据也满足相应的二元线性模型的约束,则相应的数据异常在 DCD 中将无法被成功检测.

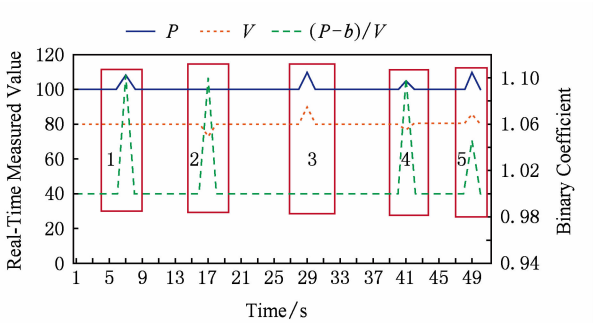

Fig. 4　Data abnormal detection for DCD

图 4　DCD 数据异常检测示意图

2.2.3　ADMSD_TS 算法实现

通过 2.2.1 节、2.2.2 节的介绍,我们不难发现 DCD 与 TCD 均能对传感数据的异常进行相应地检测,但是这 2 个方法自身都存在着一些缺陷,可能导致相应的异常检测结果出现偏差.因此在前 2 步检测结果的基础上,再次对 DCD 与 TCD 的异常检测结果进行有效地数据融合处理(fusion process,FP),从而得出更加准确的异常检测结果.融合处理算法如算法 3 所示.

算法 3. 异常检测结果的融合操作 Fusion Process.

输入:异常 ID 列表 $listForAB$,$mapForTCD$,$mapForDCD$,DCD 算法中的 Ω_R;

输出:异常结果集 Ω_{result}.

① $\Omega_{result}=\varnothing$; /* 初始化异常结果集 */

② $\Omega_{del}=\Omega_{add}=\varnothing$; /* 初始化 2 个临时集合用于异常数据的整合 */

③ Hashmap $mapForResult=new\ HashMap()$; /* 建立新的 Haspmap 用于存储异常结果 */

④ for each $abID$ in $listForAB$

⑤　$\Omega_m=mapForDCD.get(abID)$;

⑥　$\Omega_c=mapForTCD.get(abID)$;

⑦　if $\Omega_m\neq\varnothing$ && $\Omega_c\neq\varnothing$

⑧　　for each ab_m in Ω_m /* 解决 DCD 中的问题 1 */

⑨　　if $ab_m\notin\Omega_c$

⑩　　$traverse(ab_m,\Omega_m)$; /* 寻找 $ab_i\notin\{\Omega_k-ab_m\}$ */

⑪　　$\Omega_{del}=\Omega_{del}\cup ab_i$;

⑫　　end if

⑬　end for

⑭　for each $ab_c\in\Omega_c$ && $ab_c\in\Omega_R$ /* 利用 TCD 的数据异常,解决 DCD 的问题 2 */

⑮　　$traverse(ab_c,\Omega_R)$; /* 寻找 $ab_i\in\Omega_k-ab_c$ */

⑯　　$\Omega_{add}=\Omega_{add}\cup ab_i$;

⑰　end for

⑱　$\Omega_{result}=\Omega_c\cup(\Omega_m-\Omega_{del})\cup\Omega_{add}$;

⑲　if $\Omega_{result}\neq\varnothing$ /* 循环结束取出 Ω_{result} 中的异常存入 Hashmap 中 */

⑳　　for each ab_i in Ω_{result}

㉑　　$mapForResult.put(abID,ab_i)$; /* 将异常结果存入 Hashmap 中 */

㉒　　end for

㉓　end if

㉔ end if

㉕ end for

㉖ Return $mapForResult$. /* 输出保存异常的 Hashmap,算法结束 */

融合处理算法将前 2 步 DCD 与 TCD 所获取的

传感数据异常结果集进行了相应地融合,其基本步骤如下:

算法行①~③完成相应数据变量的初始化操作.

算法行④~⑥获取同一个 $abID$ 的 DCD 检测异常集 Ω_m 以及 TCD 检测异常集 Ω_c.

算法行⑦~⑬利用 Ω_c 对 Ω_m 中具有相关性传感数据集合进行精确定位,并根据 Ω_c 中的电源传感数据异常,剔除 Ω_m 中不存在异常的传感数据.将不存在异常的传感数据保存在 Ω_{del} 中.

算法行⑭~⑰利用 Ω_c 对 DCD 检测中的符合相关关系的正确数据集 Ω_{R} 进行再次筛选,将筛选出符合正确的相关关系但是存在数据异常的传感数据,即 DCD 未监测到的传感数据异常,并将新筛选出的异常数据保存在在 Ω_{add} 中.

算法行⑱利用 Ω_{del} 与 Ω_{add} 将 Ω_c 与 Ω_m 的结果进行相应地合并操作,合并操作将完成以上 2 个数据检测方法,异常数据结果的有效融合.数据集的整合,不仅可以补充 TCD 检测与 DCD 检测未能发现的数据异常,同时也剔除了不存在异常的相应数据.

算法行⑲~㉖最后将融合后的异常数据集保存在结果数组中,本算法结束.

通过算法 1~3 的介绍,本文提出的 ADMSD_TS 算法实现如算法 4 所示.

算法 4. 基于时间序列的多源传感数据异常检测算法 ADMSD_TS.

输入:时间序列数据 TS、滑动窗口 SW 长度 Len_{sw}、子序列移动距离 len_{move}、滑动窗口的最小长度阈值 $\varepsilon_{\text{size}}$ 以及相对离群距离阈值 ε_{dis};

输出:异常结果集 Ω_{result}.

① $\Omega_{\text{result}}=\varnothing$; /* 初始化异常结果集 */

② Hashmap $mapForTCD=new\ HashMap()$;

/* 建立新的 Haspmap 存储 TCD 的异常集合 */

③ Hashmap $mapForDCD=new\ HashMap()$;

/* 建立新的 Haspmap 存储 DCD 的异常集合 */

④ Hashmap $mapForResult=new\ HashMap()$;

/* 建立新的 Haspmap 存储融合后的异常结果 */

⑤ while $TS.length()>Len_{\text{sw}}$

⑥ $mapForTCD=TCD(TS_{\text{len}})$;

/* TCD 检测 */

⑦ $mapForDCD=DCD(TS_{\text{len}})$;

/* DCD 检测 */

⑧ $mapForResult=Fusion(mapForTCD, mapForDCD)$; /* 异常数据融合处理 */

⑨ end while

⑩ Return $mapForResult$. /* 输出异常数据集,算法结束 */

2.3 ADMSD_TS 算法性能分析

通过对 ADMSD_TS 算法的详细介绍,不难发现本算法主要分为时序连续性检测 (TCD)、数据相关性检测(DCD)以及融合处理(fusion process, FP)三个主要步骤.假设多源时间序列数据 $TS_m=\{S_1, S_2, \cdots, S_m\}$ 中每个 $S_i(1\leqslant i\leqslant m)$ 的长度为 n, TS_m 按照时间的先后顺序进入滑动窗口 SW,SW 中的一个多源时间序列集可以表示为 $TS_m^{\text{len}}=\{S_1^{\text{len}}, S_2^{\text{len}}, \cdots, S_m^{\text{len}}\}$.

1) TCD 复杂度分析. TCD 的计算时间与滑动窗口 SW 长度 Len_{sw} 以及子序列移动距离 len_{move} 有关.由于最小长度阈值 $\varepsilon_{\text{size}}$、相对离群距离阈值 ε_{dis} 的限制,子序列的平均移动次数 k 将为某个固定常数, TS_m 中的 m 一般也为某个固定常数. TCD 的计算复杂度为 $O(k\times m\times Len_{\text{sw}}\times len_{\text{move}})$,考虑到 k 与 m 为固定常数,而 $Len_{\text{sw}}\ll n$ 且 $Len_{\text{move}}\ll n$,TCD 的计算复杂度在最坏情况下不超过 $O(n^2)$.

2) DCD 复杂度分析. DCD 的计算时间与滑动窗口 SW 长度 Len_{sw} 以及参数集合 Ω_k 有关.首先需要根据不同的参数集合 Ω_k,求出 TS_m^{len} 中部分多源时间序列 TS_k^{len} 的相应线性模型参数向量.根据定义 3 中的限制条件,DCD 算法只考虑不超过五元线性模型的 $TS_k^{f'}$ 相关性检测,即 $k\leqslant 5$.因此将在常数时间 C 内完成相应的参数向量计算.假设全体参数模型的数量为 sum_k,而每个参数模型的相关性检测的时间复杂度为 $O(Len_{\text{sw}})$,DCD 的计算复杂度为 $O(sum_k\times Len_{\text{sw}})$,考虑 $Len_{\text{sw}}\ll n$ 且 $sum_k\ll n$,DCD 的计算复杂度在最坏情况下不超过 $O(n^2)$.

3) FP 复杂度分析. FP 主要对 TCD 与 DCD 的异常检测结果,进行相应地优化操作,补充 TCD 与 DCD 未能发现的数据异常,同时也剔除了不存在异常的相应数据.根据算法 3 的详细流程,FP 的计算复杂度为 $O(n^2)$

综上所述,ADMSD_TS 算法对 SW 中的多源时间序列 TS_m^{len} 的计算时间复杂度在最坏情况下不超过 $O(n^2)$,当 SW 的个数为 m 时,ADMSD_TS 算法的整体时间复杂度不超过 $O(mn^2)$.

3 实验研究

3.1 实验环境

硬件环境:浪潮英信 NF5270M4 服务器、至强 E5V4 处理器、16 GB 内存、2TB 硬盘.

软件环境:JRE1.7.0_13,ZooKeeper-3.4.6,Storm0.9.1.

操作系统:CentOS6.5.

数据集:济南市政供暖系统中394栋楼宇的16 909个住户,预处理后的数据集为该规模下的住户供暖情况数据集(data of Jinan municipal steam heating system, JMSHSD).

在数据收集过程中,每个楼宇的发送器每个小时聚合了该楼宇中每个房间的数据,然后将这些数据传送给数据接收器.当一个接收器接收的数据达到阈值,或者接收器等待时间达到阈值.接收器一次将当前其接收的数据全部传输到我们预设的DataSpout端口,并将相关数据分发至TCDBolt开始进行时序连续性检测.当TCDBolt完成时序相关性检测之后,多个TCDBolt将向对应的DCDBolt发送相应的传感数据,在DCDBolt进行数据相关性检查.与此同时,TCDBolt也会将时序连续性的检查结果发送至FusionBolt.等待DCDBolt中对应的相关性结果检测完毕以后,DCDBolt也会将对应的相关性检测结果发送至FusionBolt,进行相应的异常数据集融合操作并完成最终的多源数据异常检测的最终结果.

数据根据处理方式的不同,其相应处理平均时间对比如表1所示,其中云计算的接收数据规模较大,带宽压力较高,其网络传输时间明显比边缘计算要长.由于云中心节点的计算能力相对较强,与边缘计算相比,其DCD所消耗的时间较短.在TCD与FP步骤中,由于受到数据规模及带宽的压力,云计算的处理时间相对较高.因此通过综合的比较与分析,边缘计算的异常检测性能更好.

Table 1　Comparison of Time-Consuming in Each Step

表1　各阶段耗费时间对比

Operation	Time-Consuming in Edge Computing/ms	Time-Consuming in Cloud Computing/ms
Receive	249.1	312.30
Save	165.2	290.00
DCD	0.7	0.65
TCD	2.2	2.80
FP	0.6	0.90

3.2　异常检测结果分析

本节我们对传感数据异常检测的实验结果分析主要分为2个步骤:

1)利用本文提出的异常检测算法(ADMSD_TS),对数据集JMSHSD中的部分传感数据(累计热量、热功率、累计流量、流速以及温差)进行相应的异常检测,并利用数据检测结果对ADMSD_TS算法及其内部的TCD,DCD操作检测的结果进行详细的分析.

选取数据集JMSHSD中部分传感数据(累计热量、热功率、累计流量、流速以及温差)共计100 000条,相关传感数据的参数描述如表2所示.传感数据部分实际观测值如表3所示.根据相应的热力学原理对表3的部分传感数据进行简单的分析,不难发现P_i和$\Delta W_i/\Delta t_i$以及s_i和$\Delta v_i/\Delta t_i$可能具有线性相关性,随后我们利用相关系数的计算公式对上述传感数据进行了相应的计算.计算结果显示:P和$\Delta W/\Delta t$的相关系数为0.880,S和$\Delta V/\Delta t$的相关系数为0.926.因此它们都满足线性相关的约束.

Table 2　Sensor Data Parameter Description

表2　传感数据参数描述

Parameter	Description
$roomID$	Number of room
W	Accumulated value of heat
W_i	Real time value of heat
ΔW_i	$\Delta W_i=W_i-W_{i-1}$
P	Thermal power
P_i	Real time value of thermal power
V	Total flow
V_i	Real time value of total flow
ΔV_i	$\Delta V_i=V_i-V_{i-1}$
s	Accumulated value of velocity
s_i	Real time value of velocity
T_i	Real time difference of temperature

Table 3　Sensor Data Correlation Description

表3　传感数据相关性描述

$roomID$	$\Delta W_i/\Delta t_i$	P_i	$\Delta V_i/\Delta t_i$	s_i	T_i
1	0.911	0.910	0.911	0.902	0.88
5	2.170	2.180	2.170	2.149	0.88
25	0.900	0.900	0.900	0.904	0.87
27	1.821	1.830	1.821	1.804	0.85
32	0.010	0.010	0.010	0.010	1.05
43	1.869	1.870	1.869	1.868	0.87
63	1.817	1.800	1.817	1.820	0.86

利用本文提出的 ADMSD_TS 算法进行异常数据检测,检测结果如表 4 所示. 根据检测结果:在 100 000 条传感数据中,总共有 560 条 P,W 或 PW 异常数据,452 条 V,s 或 Vs 异常数据以及 213 条 T 异常数据. 异常数据总数可以表示为 AD_{sum},成功检测出的异常数据可以表示为 AD_{cor},则异常数据的检测精度 AD_{pre} 的计算为

$$AD_{\text{pre}}=\frac{AD_{\text{cor}}}{AD_{\text{sum}}}. \tag{9}$$

DCD 只能发现 560 条异常数据中的 430 条($AD_{\text{pre}}=0.77$)以及 452 条中的 382 条异常数据,而且对于单一序列 T,DCD 则无法发现其中的异常数据. 而 TCD 能发现 560 条异常数据中的 476 条、452 条异常数据中的 354 条,并能够对其发现的异常数据进行精确的定位. 而本文提出的 ADMSD_TS 算法能够对 DCD 以及 TCD 的数据检测结果进行相应的数据融合操作,从而有效地避免了上述 2 个方法所存在的缺陷,因此 ADMSD_TS 算法的检测结果要明显好于前面的 2 个相对单一的异常数据检测方法.

Table 4 Anomaly Detection Results

表 4 异常检测结果

Detected Abnormal Parameters	DCD	TCD	ADMSD_TS	All Abnormal Parameters
W		74	74	83
P		129	129	157
WP		273	273	320
$W/P/WP$	430		67	
V		28	28	35
s		63	63	71
Vs		263	263	346
$V/s/Vs$	382		74	
T		177	177	213

2) 基于数据集 JMSHSD 中的全部传感数据,分别利用本文提出的异常检测算法(ADMSD_TS)与基准方法(AD_IP[18],AD_KNN[19])进行传感数据异常检测,并对实验结果进行比较与分析.

我们选取数据集 JMSHSD 中近一年的全部传感数据共计 2 016 983 条,并将全部数据以月为单位分别利用 ADMSD_TS,AD_IDP 以及 AD_KNN 进行数据异常检测. 随后计算异常数据检测精度 AD_{pre} 的平均值,最后得到的异常数据检测结果如图 5 所示:

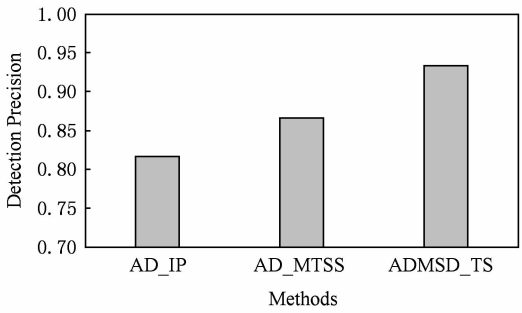

Fig. 5 Comparison of abnormal detection precision

图 5 异常数据检测精度对比

根据图 5 所示的异常数据检测精度比较,不难发现 ADMSD_TS 算法的检测结果要明显好于 AD_IDP 与 AD_KNN. 以上 2 个对比方法虽然分别采用基于时间序列重要点分割以及基于快速选择策略的 k-近邻搜索去寻找相应的异常数据,但是上述方法并没有很好地利用多源时间序列之间"广泛"存在的相关关系对时间序列数据的变化趋势进行准确地判断,从而无法对多源相关数据异常进行有效地识别. 因此上述方法的异常数据检测精度与 ADMSD_TS 算法相比,具有相对明显的差距.

4 结 论

本文提出了一种新的基于离群距离与序列相关性的异常检测算法,该算法采用边缘计算的处理模型,通过对参数之间的相关性与参数自身的时序连续性对相应的传感数据进行检测. 通过在济南市政供暖数据集上的进行的算法验证,本算法具有处理速度快、异常检出率高的特性. 未来我们还将继续优化该算法的边缘计算模型,并希望能将该检测算法推广到更广泛的实时数据应用场景中.

参 考 文 献

[1] Shi Weisong, Sun Hui, Cao Jie, et al. Edge computing—An emerging computing model for the Internet of everything era [J]. Journal of Computer Research and Development, 2017, 54(5): 907-924 (in Chinese)
(施巍松, 孙辉, 曹杰, 等. 边缘计算:万物互联时代新型计算模型[J]. 计算机研究与发展, 2017, 54(5): 907-924)

[2] Wang Xiaqing, Fang Zicheng, Wang Peng, et al. A distributed multi-level composite index for KNN processing on long time series [C] //Proc of the 21st Int Conf on Database Systems for Advanced Applications. Berlin: Springer, 2017: 215-230

[3] Sharma A B, Chen Haifeng, Ding Min, et al. Fault detection and localization in distributed systems using invariant relationships [C] //Proc of the 43th Int Conf on Dependable Systems and Networks (DSN). Piscataway, NJ: IEEE, 2013: 1-8

[4] Barnett V, Lewis T. Outliers in Statistical Data [M]. New York: Wiley, 1994: 20-29

[5] Knox E M, Ng R T. Algorithms for mining distancebased outliers in large datasets [C] //Proc of the 24th Int Conf on Very Large Data Bases. San Francisco: Morgan Kaufmann, 1998: 392-403

[6] Knorr E M, Ng R T. Finding intensional knowledge of distance-based outliers [J]. Journal of Very Large Data Bases, 1999, 99(12): 211-222

[7] Ramaswamy S, Rastogi R, Shim K. Efficient algorithms for mining outliers from large data sets [C] //Proc of the 29th Int Conf on Management of Data. New York: ACM, 2000: 427-438

[8] Markou M, Singh S. Novelty detection: A review—part 2: Neural network based approaches [J]. Signal Processing, 2003, 83(12): 2499-2521

[9] Mourão-Miranda J, Hardoon D R, Hahn T, et al. Patient classification as an outlier detection problem: An application of the one-class support vector machine [J]. Neuroimage, 2011, 58(3): 793-804

[10] Wang J S, Chiang J C. A cluster validity measure with outlier detection for support vector clustering [J]. IEEE Trans on Systems, Man, and Cybernetics, Part B: Cybernetics, 2008, 38(1): 78-89

[11] Fu T-C. A review on time series data mining [J]. Engineering Applications of Artificial Intelligence, 2011, 24(1): 164-181

[12] Vlachos M, Philip S Y, Castelli V. On periodicity detection and structural periodic similarity [C] //Proc of the SIAM Int Conf on Data Mining. Society for Industrial and Applied Mathematics. Philadelphia: SIAM, 2005: 449-460

[13] Keogh E, Lonardi S, Chiu B Y. Finding surprising patterns in a time series database in linear time and space [C] //Proc of the 8th Int Conf on Knowledge Discovery and Data Mining. New York: ACM, 2002: 550-556

[14] Keogh E, Lin J, Lee S-H, et al. Finding the most unusual time series subsequence: Algorithms and applications [J]. Knowledge and Information Systems, 2007, 11(1): 1-27

[15] Chan P K, Mahoney M V. Modeling multiple time series for anomaly detection [C] //Proc of the 5th IEEE Int Conf on Data Mining. Piscataway, NJ: IEEE, 2005: 90-97

[16] Wei Li, Kumar N, et al. Assumption-free anomaly detection in time series [C] //Proc of the 17th Int Conf on Scientific and Statistical Database Management. Piscataway, NJ: IEEE, 2005: 237-242

[17] Fujimaki R, Yairi T, Machida K. An anomaly detection method for spacecraft using relevance vector learning [C] // Proc of the Advances in Knowledge Discovery and Data Mining. Berlin: Springer, 2005: 785-790

[18] Zhou Dazhuo, Liu Yuefen, Ma Wenxiu. Time series anomaly detection [J]. Journal of Computer Engineering and Applications, 2008, 44(35): 145-147 (in chinese)
(周大镯, 刘月芬, 马文秀. 时间序列异常检测[J]. 计算机工程与应用, 2008, 44(35): 145-147)

[19] Cai Lianfang, Thornhill N F, Kuenzel S, et al. Real-time detection of power system disturbances based on *k*-nearest neighbor analysis [J]. IEEE Access, 2017, 5(3): 5631-5639

[20] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing [J]. Communications of the ACM, 2010, 53(4): 50-58

[21] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system [C] //Proc of the 26th Symp on Mass Storage Systems and Technologies (MSST). Piscataway, NJ: IEEE, 2010: 1-10

[22] Zaharia M, Chowdhury M, Franklin M J, et al. Spark: Cluster computing with working sets [J]. HotCloud, 2010, 15(1): 10-17

[23] Moon Y S, Whang K Y, Loh W K. Duality-based subsequence matching in time-series databases [C] //Proc of the 17th Int Conf on Data Engineering. Piscataway, NJ: IEEE, 2001: 263-272

[24] Ji Cun, Shao Qingshi, Sun Jiao, et al. Device data ingestion for industrial big data platforms with a case study [J]. Sensors, 2016, 16(3): Article No. 279

[25] Fang Wei. Paradigm shift from cloud computing to fog computing and edge computing [J]. Journal of Nanjing University of Information Science & Technology, 2016, 8(5): 404-414 (in Chinese)
(方巍. 从云计算到雾计算的范式转变[J]. 南京信息工程大学学报, 2016, 8(5): 404-414)

Zhang Qi, born in 1982. PhD candidate at Shandong University. His main research interests include pattern recognition, data mining, machine learning.

Hu Yupeng, born in 1983. PhD candidate at Shandong University. His main research interests include big data management, big data analytics, big data business intelligence, service computing and collaborative computing.

Ji Cun, born in 1989. PhD. His main research interests include services computing, and services system for manufacturing.

Zhan Peng, born in 1988. PhD candidate at Shandong University. His main research interests include data mining, machine learning, deep learning.

Li Xueqing, born in 1964. Professor and PhD supervisor. His main research interests include artificial intelligence, service computing and collaborative computing.

2016 年《计算机研究与发展》高被引论文 TOP10

排名	论文信息
1	刘峤, 李杨, 段宏, 刘瑶, 秦志光. **知识图谱构建技术综述**[J]. 计算机研究与发展, 2016, 53(3): 582-600 Liu Qiao, Li Yang, Duan Hong, Liu Yao, Qin Zhiguang. Knowledge Graph Construction Techniques [J]. Journal of Computer Research and Development, 2016, 53(3): 582-600
2	刘知远, 孙茂松, 林衍凯, 谢若冰. **知识表示学习研究进展**[J]. 计算机研究与发展, 2016, 53(2): 247-261 Liu Zhiyuan, Sun Maosong, Lin Yankai, Xie Ruobing. Knowledge Representation Learning: A Review [J]. Journal of Computer Research and Development, 2016, 53(2): 247-261
3	张蕾, 章毅. **大数据分析的无限深度神经网络方法**[J]. 计算机研究与发展, 2016, 53(1): 68-79 Zhang Lei, Zhang Yi. Big Data Analysis by Infinite Deep Neural Networks [J]. Journal of Computer Research and Development, 2016, 53(1): 68-79
4	王兴伟, 李婕, 谭振华, 马连博, 李福亮, 黄敏. **面向"互联网+"的网络技术发展现状与未来趋势**[J]. 计算机研究与发展, 2016, 53(4): 729-741 Wang Xingwei, Li Jie, Tan Zhenhua, Ma Lianbo, Li Fuliang, Huang Min. The State of the Art and Future Tendency of "Internet+" Oriented Network Technology [J]. Journal of Computer Research and Development, 2016, 53(4): 729-741
5	孟小峰, 杜治娟. **大数据融合研究: 问题与挑战**[J]. 计算机研究与发展, 2016, 53(2): 231-246 Meng Xiaofeng and Du Zhijuan. Research on the Big Data Fusion: Issues and Challenges [J]. Journal of Computer Research and Development, 2016, 53(2): 231-246
6	甘丽新, 万常选, 刘德喜, 钟青, 江腾蛟. **基于句法语义特征的中文实体关系抽取**[J]. 计算机研究与发展, 2016, 53(2): 284-302 Gan Lixin, Wan Changxuan, Liu Dexi, Zhong Qing, Jiang Tengjiao. Chinese Named Entity Relation Extraction Based on Syntactic and Semantic Features [J]. Journal of Computer Research and Development, 2016, 53(2): 284-302
7	单言虎, 张彰, 黄凯奇. **人的视觉行为识别研究回顾、现状及展望**[J]. 计算机研究与发展, 2016, 53(1): 93-112 Shan Yanhu, Zhang Zhang, Huang Kaiqi. Visual Human Action Recognition: History, Status and Prospects [J]. Journal of Computer Research and Development, 2016, 53(1): 93-112
8	庄严, 李国良, 冯建华. **知识库实体对齐技术综述**[J]. 计算机研究与发展, 2016, 53(1): 165-192 Zhuang Yan, Li Guoliang, Feng Jianhua. A Survey on Entity Alignment of Knowledge Base [J]. Journal of Computer Research and Development, 2016, 53(1): 165-192
9	付志耀, 高岭, 孙骞, 李洋, 高妮. **基于粗糙集的漏洞属性约简及严重性评估**[J]. 计算机研究与发展, 2016, 53(5): 1009-1017 Fu Zhiyao, Gao Ling, Sun Qian, Li Yang, Gao Ni. Evaluation of Vulnerability Severity Based on Rough Sets and Attributes Reduction [J]. Journal of Computer Research and Development, 2016, 53(5): 1009-1017
10	曹珍富, 董晓蕾, 周俊, 沈佳辰, 宁建廷, 巩俊卿. **大数据安全与隐私保护研究进展**[J]. 计算机研究与发展, 2016, 53(10): 2137-2151 Cao Zhenfu, Dong Xiaolei, Zhou Jun, Shen Jiachen, Ning Jianting, Gong Junqing. Research Advances on Big Data Security and Privacy Preserving [J]. Journal of Computer Research and Development, 2016, 53(10): 2137-2151

数据来源:CSCD,中国知网;统计日期:2017 年 11 月 27 日