

# 北京郵電大學

## 本科 毕业 设计（论文）



题目：基于腾讯定位数据的异常事件检测算法

姓 名 林文鼎  
学 院 信息与通信工程学院  
专 业 通信工程  
班 级 201421112  
学 号 2014210328  
班内序号 07  
指导教师 别红霞

2018 年 5 月

# 北京邮电大学

## 本科毕业设计（论文）任务书

学院	信息与通信工程学院	专业	通信工程	班级	2014211112
学生姓名	林文鼎	学号	2014210328	班内序号	07
指导教师姓名	别红霞	所在单位	信息与通信工程学院	职称	教授
设计(论文)题目	基于腾讯定位数据的异常事件检测算法				
	Anomalous Event Detection Based on Tencent Positioning Data				
题目分类	工程实践类 <input type="checkbox"/>	研究设计类 <input checked="" type="checkbox"/>	理论分析类 <input type="checkbox"/>		
题目来源	题目是否来源于科研项目	是 <input checked="" type="checkbox"/>	否 <input type="checkbox"/>		
<b>主要任务及目标:</b> 1. 通过研究腾讯定位数据，分析在异常事件出现时的数据异常性特征。 2. 研究基于位置服务数据的异常检测算法。 3. 研究基于上述位置服务数据形成的图像异常检测算法。 4. 实现相关的算法。					
<b>主要内容:</b> 通常在异常事件（如异常气象，交通管制等）出现时，定位数据较以往同时段的数值必然有异常的波动。本论文研究基于位置服务数据的异常事件检测算法，并确定出现异常的子区域位置。					
<b>主要输出包括:</b> 1. 分析数据并标出所要处理数据的异常。 2. 设计实现数据的异常检测算法 3. 设计实现基于图像的异常区域检测算法					
<b>主要参考文献:</b> [1] Patcha A, Park J M. An overview of anomaly detection techniques: Existing solutions and latest technological trends[J]. Computer Networks, 2007, 51(12):3448-3470.					
<b>进度安排:</b> 01/01-01/26 资料查阅，论文开题； 01/27-03/04 阅读综述论文，调研相关算法； 03/05-03/19 技术路线确定； 03/20-04/15 完成异常数据的标定以及实现异常数据的检测算法； 04/16-04/20 准备中期检查相关材料； 04/21-05/10 完成异常数据生成的图像检测算法并开发应用； 05/11-05/20 核对任务完成情况，毕设论文撰写； 05/21-06/04 汇总毕设相关材料，准备答辩。					
指导教师签字		日期	年 月 日		

本科生毕业设计（论文）答辩成绩评定标准														
答辩小组成绩评定	评价内容	具体要求	分值											
	选题	符合专业培养目标，符合社会实际、结合工程实际，难易适度，体现新颖性、综合性。	5	4	3.5	3	2							
	设计（论文）质量水平	全面完成任务书中规定的各项要求，文题相符，工作量饱满，写作规范，达到综合训练的要求，有理论成果和应用价值。	20	16	14	12	8							
	答辩准备	准备充分；有简洁、清晰、美观的演示文稿；准时到场。	5	4	3.5	3	2							
	内容陈述	语言表达简洁、流利、清楚、准确，思路清晰，重点突出，逻辑性强，概念清楚，论点正确；实验方法科学，分析归纳合理；结论严谨；表现出对毕业设计（论文）内容掌握透彻。	20	18	14	12	8							
	回答问题	回答问题准确、有深度、有理论根据、基本概念清晰。	10	8	7	6	4							
	答辩小组评分合计（满分 60 分）													
意见：														
答辩小组组长签字：_____ 年 月 日														
答辩小组成员：														
学院意见	最终成绩：百分制_____； 五分制_____													
	院长签章：_____ 学院盖章：_____ 年 月 日													
备注														

注：1. 毕业设计（论文）成绩由中期检查评分（满分 10 分）、指导教师评分/复议评分（满分 30 分）和答辩小组评分（满分 60 分）相加，得出百分制成绩，再按 100-90 分为“优”、89-80 分为“良”、79-70 分为“中”、69-60 分为“及格”、60 分以下为“不及格”的标准折合成五级分制成绩；

2. 此表原件一式三份，一份存入学生档案，一份装订到毕业论文中，一份交教务处存入档案馆。

北 京 邮 电 大 学

本科毕业设计（论文）诚信声明

本人声明所呈交的毕业设计（论文），题目《社交网络多媒体信息可信度评估》是本人在指导教师的指导下，独立进行研究工作所取得的成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：\_\_\_\_\_ 日期：\_\_\_\_\_

# 基于腾讯定位数据的异常事件检测算法

## 摘要

通过终端设备采集的定位数据的大小反映了这个区域的人口密度特征，而通过长时记录所得到的时序定位数据更能反映密度的变化特征。人口密度的变化特征通常在时间上符合一个规律，而在某些时刻由于例如自然灾害的原因会导致该地的定位数据异常，检出这种异常能够使有关部门做好应急响应。本文基于腾讯定位数据，对确定区域的定位数据进行异常检测，研究相关的算法，并尝试确定出现异常的子区域位置。

针对以上目的，本文对给定的定位数据进行了分析及预处理，采用了基于差分、小波、极大似然估计、局部异常因子的几种曲线异常检测算法对诸如台风过境的区域全天性异常进行了检测，然后使用了非线性自回归网络基于过往时刻数据对当前时刻定位值进行曲线预测，检测了局部时刻的异常值，最后运用了帧间差分法，针对检测出的异常出现时刻使用了不同大小的滑动窗计算区域定位数据变化平均值来确定异常发生的核芯子区域位置。

在基于给定的腾讯定位数据基础上，仿真结果表明：使用了曲线异常检测与预测的方法可检出区域性的整天性时段与局部性时刻异常，而通过相邻帧间差分法与滑动窗相结合能够将造成异常的主要区域检出。

**关键词** 定位数据 异常检测 曲线分析

# Anomalous Event Detection Based on Tencent Positioning Data

## ABSTRACT

The size of the positioning data collected by the terminal equipment reflects the population density characteristics of this area, and the time-series positioning data obtained by the long-term recording can better reflect the population density variation characteristics of the area. The characteristics of changes in population density usually conform to a rule in time. At some moments, for example, due to natural disasters, the location data of the area may be abnormal. The detection of such anomaly can enable the relevant departments to make an emergency response. In this paper, based on Tencent positioning data, we try to detect the anomaly of positioning data in a certain area, study relevant algorithms, and search the location of an abnormal sub-area in there.

For the above purposes, this paper analyzes and preprocesses the given positioning data, and adopts several curve anomaly detection algorithms based on difference, wavelet, maximum likelihood estimation, and local outlier factors for regional whole day anomaly such as typhoon crossing. Then a non-linear autoregressive network was used to curve the current local time value based on past time data to detect the hour anomaly. Finally, the inter-frame difference method was used to detect the occurrence area of the anomaly. Different sliding window sizes are used to calculate the average regional positioning data change values to determine the location of the core sub-area where the anomaly occurs.

Based on the given Tencent positioning data, the simulation results show that using the method of curve anomaly detection and prediction can detect regional all-day and local-time anomalies, and through adjacent inter-frame difference method and sliding The combination of windows can detect the main area causing the anomaly.

**KEY WORDS** positioning data anomaly detection curve analysis

# 目 录

<b>第一章 绪论</b>	1
1.1 课题背景	1
1.2 异常检测研究现状	2
1.2.1 基本概念与挑战	2
1.2.2 异常检测算法分类	3
1.3 论文主要工作	4
1.4 论文章节安排	5
<b>第二章 异常检测算法基础</b>	6
2.1 时序曲线离群点检测	6
2.1.1 极大似然估计	6
2.1.2 离散序列小波变换	7
2.1.3 最近邻及邻域算法	8
2.2 时序曲线预测	11
2.2.1 神经网络	11
<b>第三章 定位数据</b>	12
3.1 腾讯定位数据的形式	12
3.2 数据分析及应用	12
3.2.1 数据的小时变化规律	13
3.2.2 数据的日变化规律	15
3.2.3 数据的总时空特征	15
3.3 数据预处理及异常分析策略	16
3.3.1 数据预处理	16
3.3.2 曲线异常分析策略	17
<b>第四章 基于曲线分析的定位数据异常检测</b>	19
4.1 基于差分的异常检测算法	19
4.1.1 差分算法	19
4.1.2 结果分析	19
4.2 基于小波变换的异常检测算法	20
4.2.1 离散序列小波变换	20
4.2.2 结果分析	21

4.2.3 结果处理优化	21
4.3 基于极大似然估计的异常检测算法	22
4.3.1 极大似然估计与 $3\sigma$ 准则	22
4.3.2 结果分析	22
4.4 局部异常因子检测算法	24
4.4.1 局部离群因子	24
4.4.2 结果分析	25
4.5 曲线异常检测算法总结	26
<b>第五章 基于曲线的定位数据预测与异常检测</b>	<b>27</b>
5.1 异常的检测与预测	27
5.2 基于神经网络的曲线定位数据预测	28
5.2.1 神经网络分类	28
5.2.2 定位数据预测	30
5.2.3 分析结果	30
<b>第六章 基于区域的定位数据异常检测</b>	<b>33</b>
6.1 基于图像的异常区域检测	33
6.1.1 相邻帧间差分法	33
6.1.2 高浮动区域检测	33
<b>第七章 总结与展望</b>	<b>37</b>
7.1 内容总结	37
7.2 未来展望	38
<b>参考文献</b>	<b>39</b>
<b>致    谢</b>	<b>40</b>
<b>附    录</b>	<b>41</b>
附录 1 缩略语表	41
附录 2 数学符号	41

# 第一章 绪论

本章主要介绍了定位数据的异常事件检测的课题背景及其研究意义，其次介绍了异常检测这一领域的基本概念及常用算法分类，最后对论文的主要研究工作进行了总结并阐述了本文的行文章节安排。

## 1.1 课题背景

随着 GPS 定位，传感器网络和高速无线通信等技术的日益发展，越来越多的终端定位数据被收集和保存在应用服务器，它们是各地人口密度的一个衡量依据。除了定位数据本身所体现的人口密度空间特征，在相同的空间位置不同的时间点上进行记录还可以得到定位数据的人口密度时序特征。而通过分析某片区域上的时序定位数据，可以得到该区域上人口密度的变化特征<sup>[1]</sup>，例如图 1-1 所示从北京市每日的滴滴打车的定位数据可以明显地总结出以下特征：滴滴车辆在早高峰时将大量住在郊区的人群运送至各大工作区（例如中关村和国贸区域），而在晚高峰时它们又将人群从工作区运送回家。这种区域性的时序定位数据反应出了北京市的日人口密度变化特征。

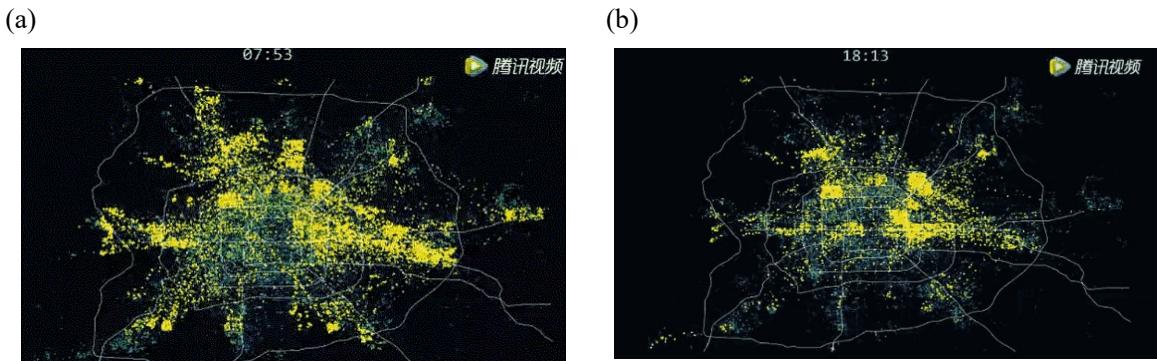


图 1-1 北京市滴滴打车定位数据图

而在这种区域性的时间序列中，也会在某些时间点上出现这样的观测点，它们较以往同时段的数据，例如某个月的温度与前几个月的温度变化来说，会有一个明显的波动误差，这就是时序数据点中的异常值，如图 1-2（横坐标代表时间，纵坐标代表温度）所示， $t_1$  处和  $t_2$  处的取值是一样的，但是  $t_2$  属于异常点，而  $t_1$  是正常的温度。分析这些异常点也是一个很重要的课题，通常对于区域性时序特征模型的建立，这些异常点是应当被剔除的噪声，它们会对模型的预测功能产生极大的阻碍。但同时异常点也可以作为一些突发事件（如异常气象，交通管制等）的判断因素，在异常事件发生时，区域的时序定位数据会在某一个时间间隔中出现较大的落差，即异常波动，通过异常检测算法将上述波动检出并分析，可以使有关部门察觉到异常状况的发生并及时做好应急响应。

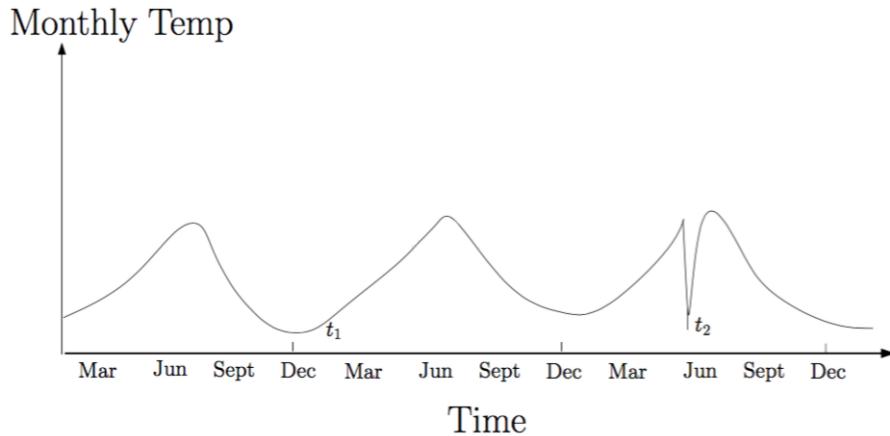


图 1-2 时序数据中的异常值

本课题基于腾讯定位数据，对已知的异常-某一整天确定区域的台风袭来时的定位数据进行研究，研究能够检测出该天整块区域的异常的算法，并对检测过程中的一些问题进行讨论，随后检测区域定位数据异常最大的子区域。

## 1.2 异常检测研究现状

### 1.2.1 基本概念与挑战

“异常”是指数据特征不符合该特征一般所隶属区间的现象<sup>[2]</sup>，如图 1-3 所示。寻找异常是一个非常困难的课题，其难点主要来源于以下两个角度：首先，“异常”通常情况下只是一个定性的概念，偏离正常数据多少可以被界定为异常没有一个定量的比例数值，那么对于那些处于异常非异常边界线附近的异常数据来说，完全可以把边界线略微移动，使其能被归类为正常的数据；再者，用于划定数据特征正常区间的正常样本中有时也会存在异常数据，导致划定边界线偏差或是训练出的预测模型不准确。同时，考虑到正常的数据量远大于异常数据，使用机器学习的方法进行训练时很容易使网络结构偏向于正常数据的分布，即过拟合导致无法检测出异常。

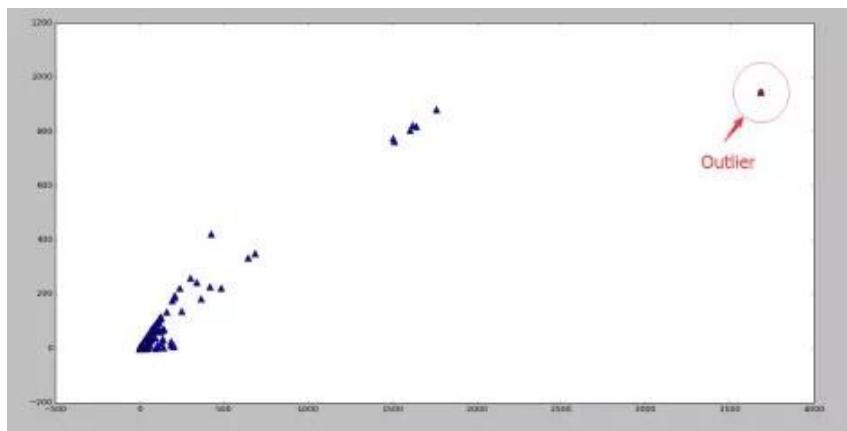


图 1-3 点异常

为了便于分析，异常也有多种分类：通常情况下直观理解的异常指的是点异常，其含义是多个数据实体中，如果存在一个实体对于其他实体来说有极大的偏差，那么这个实体数据所对应的特征就表明一种点异常。而另外一种异常被称为环境异常，它与点异常中的异常概念是一样的，表征的是一个数据实体在特定环境中的异常，存在某种限定条件。这种异常类型的数据实体有两部分组成：环境属性与行为属性。环境属性表征了数据实体所处的环境，例如时间序列数据的时间点，空间数据的地理坐标；行为属性表征了在上述特定环境属性下区分数据实体的属性，类似于地理数据的某地降雨量，行为属性即固定了环境属性后的数据特征，例如已知某地理坐标上的定位数据量。在本课题中，数据属于时序性的定位数据，而其中的异常是一种环境异常。环境属性即是时空坐标与地理坐标，行为属性是在某时间点上某地理坐标下的定位终端数量。

### 1.2.2 异常检测算法分类

异常检测是找出数据特征严重不同于预期对象的一个检测过程。传统检测异常的方法分为以下几类<sup>[3]</sup>：基于分类的异常检测方法，基于最近邻的异常检测方法，基于聚类的异常检测方法，基于统计的异常检测方法。

1. **基于分类的异常检测方法：**分类是一种从一组已做好标注的数据实例（训练）中学习模型（分类器），然后使用学习模型（测试）将测试实例分类到其中一个类中的方法。基于分类的异常检测算法以类似的两阶段方式生成一个分类模型从而判断数据的特征是否异常：在训练阶段通过使用已进行标记的样本来训练分类器的模型和参数；测试阶段使用分类器将测试实例分类为正常或异常。基于分类的异常检测算法基于以下假设下实现：在给定的数据特征空间中学习可以区分正常类和异常类的分类器是可行的。
2. **基于最近邻的异常检测方法：**近邻分析的概念已用于多种异常检测算法，这些算法都基于以下关键假设：正常的数据实例发生在密集的邻域中，而异常发生在离它们最近的邻居很远的地方。最近邻的异常检测算法需要两个数据实例之间所定义的距离或相似度衡量，而这些距离又针对不同类别的属性有不同的衡量标准。对于连续的属性，欧几里得距离的效果优秀，可以表征出两个数据间的联系性，但在不同情况下也可以使用其他方法计算。对于分类属性，通常使用简单的匹配系数，但也可以使用更复杂的距离度量。对于多变量数据实例，通常为每个属性计算距离或相似度，然后进行合并。
3. **基于聚类的异常检测方法：**聚类用于将类似的数据实例分组到集群中。尽管聚类是无监督式学习，但聚类在半监督式学习中的应用也被最近探讨。尽管聚类和异常检测看起来彼此根本不同，但是已经有几种基于聚类方法被应用于异常检测：
  - 第一类基于聚类的算法依赖于以下假设：普通数据实例属于数据中的一个集群，而异常不属于任何集群。基于此假设的技术将已知的基于聚类的算法应

用于数据集，并将任何不属于任何聚类的数据实例声明为异常。

- 第二类基于聚类的技术依赖于以下假设：正常的数据实例靠近它们最接近的集群质心，而异常距离它们最近的集群质心很远。基于这种假设的技术由两个步骤组成。在第一步中，数据使用聚类算法进行聚类。在第二步中，对于每个数据实例，计算它到最近的集群质心的距离作为其异常分数。
  - 如果数据中的异常自身形成聚类，这些技术将无法检测到这种异常。为了解决这个问题，已经提出了第三类基于聚类的算法，它依赖于以下假设：普通数据实例属于大型且密集的集群，而异常集群属于小型集群或稀疏集群。基于此假设的技术将属于大小和/或密度低于阈值的集群的实例声明为异常。请注意，如果数据中的异常自身形成集群，则这些技术将无法检测到这种异常。
4. **基于统计的方法：**统计异常检测技术基于以下关键假设：正态数据实例出现在随机模型的高概率区域，而异常发生在随机模型的低概率区域。统计技术将一个统计模型（通常用于正常行为）与给定数据相匹配，然后应用统计推断测试来确定一个看不见的实例是否与该模型相符。基于应用的测试统计信息从学习模型中生成概率较低的实例被声明为异常。

### 1.3 论文主要工作

本文基于以上课题背景以及研究现状，基于腾讯地图所提供的时序定位终端地图数据，在已知某一天整块定位数据区域为异常天（台风过境）的前提下，研究并实现检测出该天整块区域为异常天的算法，实现了根据现有的前几日确定时刻时序定位数据预测当前相同时刻定位数据的生成模型来处理局部时刻异常，最后使用图像处理的方法，根据异常时间点检测结果识别出造成异常的主要区域。我们对这几块内容进行了讨论，具体实现内容包含以下几部分：

1. **数据解析及预处理：**对腾讯定位数据进行解析及预处理。首先，由于研究的异常为台风过境时某区域的定位数异常，将定位数据的区域统一标定在该地域的经纬度；其次，对定位数据进一步作图分析，观察在相同位置处定位终端数量一天内的变化、每天同时段的变化，确定了分析已知异常的策略，讨论了处理局部时刻异常的方法；最后，根据数据特点进行预处理便于分析。
2. **基于曲线的异常检测分析：**对经过处理后的数据采取曲线分析的形式进行区域性整天异常检测。采用了诸如小波变换，极大似然估计法，差分分析法等传统方法以及一些混合改进算法，对这些算法的效果进行对比分析，根据数据结果讨论在该定位数据下对于检测整天整块区域的异常各个算法的表现。
3. **基于曲线的时序数据预测：**根据数据的分析，设计并实现曲线预测局部时刻定位数据值模型。采用动态神经网络使用现有数据中的一部分进行训练，并使用后续

补充的数据进行验证，采用预测模型的方法可以有效避免只分析整天异常而无法分析小时的弊端。

4. 基于图像的异常区域检测：在定位数据异常能够被成功检出的基础上，通过使用图像分析中的帧间差分法以及不同大小的滑动窗计算区域定位数据变化程度来确定造成整体异常的区域。

## 1.4 论文章节安排

**第一章 绪论：**本章主要介绍了定位数据的异常事件检测的课题背景及其研究意义，其次介绍了异常检测这一领域的基本概念及常用算法分类，最后对论文的主要研究工作进行了总结并阐述了本文的行文章节安排。

**第二章 异常检测算法基础：**本章主要介绍了异常检测的算法理论基础，涵盖后续几章所用的主要算法的一些理论背景，包括了极大似然估计，小波变换，神经网络等相关概念。

**第三章 定位数据：**本章主要介绍了本课题所研究的腾讯定位数据的形式，并在 MATLAB 中作图分析其数据特征，包含时序特征及地理特征，最后根据分析结果讨论如何进行检测。

**第四章 基于曲线分析的定位数据异常检测：**本章主要基于第三章数据分析的结果从曲线的角度对数据进行异常检测，研究了在数据中找出异常的方法。使用了例如小波变换、极大似然估计和邻域的方法进行了分析，并对这些方法的效果进行了讨论。

**第五章 定位数据的预测分析：**本章主要对异常检测的另一种思路进行了探讨，即实时判断数据是否异常的检测方法。采用了动态神经网络训练时序数据预测模型，并根据预测与实际值比对的结果对异常检测的效果进行了分析。

**第六章 基于区域的定位数据异常检测：**本章基于前述两章的异常检测结果，检测造成异常的重点区域。采用了图像分析的策略，例如帧间差分法，将异常的核心区域变化特征凸显并使用滑动窗标记。

**第七章 总结与展望：**本章主要对基于腾讯定位数据的异常检测算法进行分析总结，针对实验结果进行分析，获得本文方法存在的缺点并且提出解决存在问题的有效方法，提高异常检测的准确率和平台使用的有效性。

## 第二章 异常检测算法基础

本章主要介绍了本课题所研究的异常检测的算法基础，因为在后续几章中要根据本课题所研究的数据对象对算法进行改写，所以在本章中主要介绍理论基础，后续章节中再具体阐述实际应用策略。

### 2.1 时序曲线离群点检测

#### 2.1.1 极大似然估计

极大似然估计是一种反推样本模型参数的统计方法<sup>[4]</sup>。其通过对已知的样本信息建模，在数据符合某种特定分布下对分布中的参数进行似然估计。当给定了足够多的观测数据情况下，利用这些大量的试验结果去计算参数值为多少才最有可能导致这样的实验样本结果，即已经知道骰子的大量独立投掷结果去计算扔到骰子各个面的概率参数值。例如已知数据集的采样是独立且为正态分布时，我们可以使用极大似然估计法求出参数  $\mu$  和  $\sigma$  的值。

设已知样本集为： $D = \{x_1, x_2, \dots, x_N\}$

似然函数(Link 函数)：联合概率密度函数  $P(D|\theta)$  称为相对于  $\{x_1, x_2, \dots, x_N, \dots\}$  的  $\theta$  的似然函数。

$$l(\theta) = P(D|\theta) = p(x_1, x_2, \dots, x_N|\theta) = \prod_{i=1}^N P(x_i|\theta) \quad \text{式 (2-1)}$$

$\hat{\theta}$  值是未知的参数值，我们要对其进行估计，如果它的值可以在范围内使前述似然函数  $l(\theta)$  的值最大，则  $\hat{\theta}$  是造成这样的试验样本的最大可能值，即是  $\theta$  参数的最大似然估计量。它是样本集的函数，记作：

$$\hat{\theta} = p(x_1, x_2, \dots, x_N) = d(D) \quad \text{式 (2-2)}$$

在式2-2 中， $\hat{\theta}(x_1, x_2, \dots, x_N)$  被称作极大似然函数的估计值。

基于上述定义，如果已知试验样本服从独立的正态分布  $N(\mu, \sigma^2)$ ，则试验样本的似然函数如下式所示：

$$L(\mu, \sigma^2) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} = (2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2} \sum_{i=1}^n (x_i-\mu)^2} \quad \text{式 (2-3)}$$

它的对数：

$$\ln L(\mu, \sigma^2) = -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2 \quad \text{式 (2-4)}$$

求导，得方程组：

$$\begin{cases} \frac{\partial \ln L(\mu, \sigma^2)}{\partial \mu} = \frac{1}{\sigma^2} \sum_{i=1}^n (x_i - \mu) = 0 \\ \frac{\partial \ln L(\mu, \sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2 = 0 \end{cases} \quad \text{式 (2-5)}$$

联合解得：

$$\begin{cases} \mu^* = \bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \\ \sigma^{*2} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 \end{cases} \quad \text{式 (2-6)}$$

在上述条件下，由方程式2-6 似然方程存在唯一的解  $(\mu^*, \sigma^{*2})$ ：而且它一定是最值点，这是因为当  $|\mu| \rightarrow \infty$  或  $\sigma^2 \rightarrow \infty$  或  $0$  时，非负函数  $\ln L(\mu, \sigma^2) \rightarrow 0$ 。于是  $\mu$  和  $\sigma^2$  的极大似然估计为  $(\mu^*, \sigma^{*2})$ 。

### 2.1.2 离散序列小波变换

时序信号或序列存在大量信息，但是仅仅从时序的角度去分析信号会损失大量有效信息。Fourier 变换提供了一种变换域分析的方法，它利用大量的三角基去构造信号，从频域的角度对信号分析，得出更多信号的信息。

$$F(\omega) = F[f(t)] = \int_{-\infty}^{\infty} f(\omega) e^{-i\omega t} d\omega \quad \text{式 (2-7)}$$

美中不足的是，Fourier 变换只能反映信号的频域特征，即将整个时序信号的频域分量提出，而不能反映各个时间点上的频域特征。短时 Fourier 变换 (STFT) 对此进行了改进，使用了定长的窗口对信号的时间进行了限制从而可以将每个短时进行 Fourier 频谱分析，最终得到时间轴上的频域特征。

$$X(t, \omega) = \int_{-\infty}^{\infty} \omega(t - \tau) x(\tau) e^{-j\omega\tau} d\tau \quad \text{式 (2-8)}$$

小波变换是对短时 Fourier 变换的进一步改进<sup>[5]</sup>，与后者不同的是，小波并非使用固定的窗口去限定时间，而是使用随时间变化的窗口-小波基去构造短时的 Fourier 分析。它可以有效地改善 STFT 中对于随时间信号幅度变化很不均匀的信号无法妥善处理的缺

点。

$$W_f(a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} f(t) \overline{\Psi} \frac{t-b}{a} dt \quad \text{式 (2-9)}$$

小波变换和 Fourier 变换类似，都有应用于连续信号及离散信号的分析方法。本课题所研究的时序数据应当使用离散序列小波变换，它基于 Mallat 算法将时序信号按照低频信号与高频信号进行逐层分解，每一层的低频信号被继续分解为低高频信号，以此类推<sup>[6]</sup>，具体的分解关系图 2-1 所示：

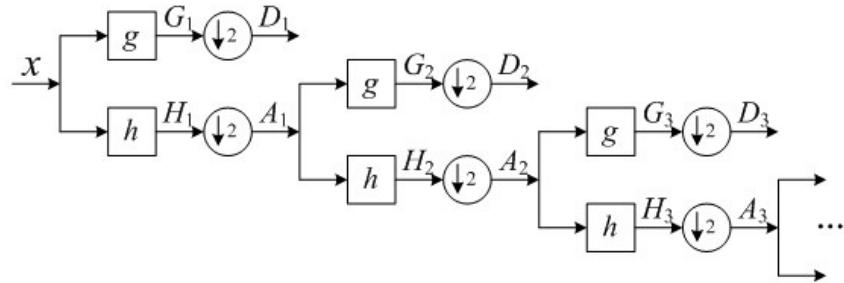


图 2-1 离散小波变换逐层分解图

时序离散序列的初始向量系数  $x$  在经过一层分解后得到高频部分  $D_1$ （系数的高频部分）与低频部分  $A_1$ （系数的低频部分），这两个分量是原始信号分别经过高通与低通滤波器后下采样得到的，代表着原始信号的细节以及近似部分。而后，因为  $D_1$  反应的是原始信号的细节部分，继续分解没有意义，所以我们分解近似部分  $A_1$ ，得到二层分解的  $D_2$  和  $A_2$ ，它们是  $A_1$  信号分别经过高通与低通滤波器后下采样得到的。我们仍可以继续对  $A_2$  进行分解，以此类推。由于在离散小波分解时滤波器系数 G 和 H 保持不变，所以带宽减半，但因为经过了下采样，所以每一部分仍然可以近似估计原始信号。将上述信号分解后的系数，经过正交小波基，可以还原原始信号的近似值。

### 2.1.3 最近邻及邻域算法

如图 2-2 对于  $C_1$  集合里的点，它们虽然互相之间相隔较远（相对于  $C_2$  来说），但它们的互相之间的间距以及分散情况是均匀的，可以认为是统一集合而不是异常的离散点。 $C_2$  集合显然是统一集合，而  $O_2$  虽然相对于  $C_1$  集合内的点的距离是不会被认为是孤立点，但是其距离最近的  $C_2$  集合相对于  $C_2$  集合是较远的，即  $O_2$  是异常点。LOF 算法从最近邻的思想展开，提供了一种检测异常的手段。

首先，我们介绍 LOF 算法的前序概念：

1.  $d(p, o)$ : 两点  $p$  和  $o$  之间的距离。
2. **k-distance** (第 k 距离)

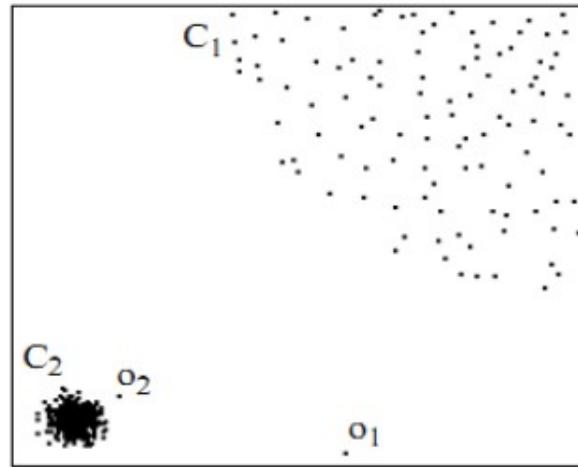


图 2-2 离散小波变换逐层分解图

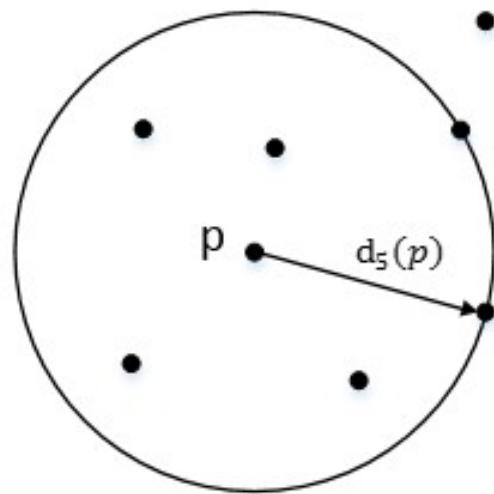
对于点  $p$  的第  $k$  距离  $d_k(p)$  定义如下：

$$d_k(p) = d(p, o) \quad \text{式 (2-10)}$$

上式满足以下两个条件：

- (a) 在集合中至少有不包括  $p$  在内的  $k$  个点  $o' \in C\{x \neq p\}$ , 满足  $d(p, o') \leq d(p, o)$ ;
- (b) 在集合中最多有不包括  $p$  在内的  $k-1$  个点  $o' \in C\{x \neq p\}$ , 满足  $d(p, o') < d(p, o)$ ;

$p$  的第  $k$  距离, 也就是距离  $p$  第  $k$  远的点的距离, 不包括  $p$ , 如图 2-3。

图 2-3 点  $p$  的第 5 距离

### 3. k-distance neighborhood of $p$ (第 $k$ 距离邻域)

点  $p$  的第  $k$  距离邻域  $N_k(p)$ , 就是  $p$  的第  $k$  距离以内的所有点, 其中也包括第  $k$  距离。因此点  $p$  的第  $k$  距离邻域内的点个数  $N_k(p) \geq k$ 。

#### 4. reach-distance (可达距离)

点  $o$  到点  $p$  的第  $k$  可达距离定义为:

$$\text{reach-distance}_k(p, o) = \max\{k - \text{distance}(o), d(p, o)\} \quad \text{式 (2-11)}$$

也就是说, 点  $o$  到点  $p$  的第  $k$  可达距离, 至少是点  $o$  的第  $k$  距离, 或者是  $o$  和  $p$  之间的真实距离。同时这也意味着, 离点  $o$  最近的  $k$  个点,  $o$  到它们的可达距离可以认为是相等的, 且都等于  $d_k(o)$  如图 2-4, 点  $o_1$  到点  $p$  的第 5 可达距离为  $d(p, o_1)$ , 点  $o_2$  到点  $p$  的第 5 可达距离为  $d_5(o_2)$

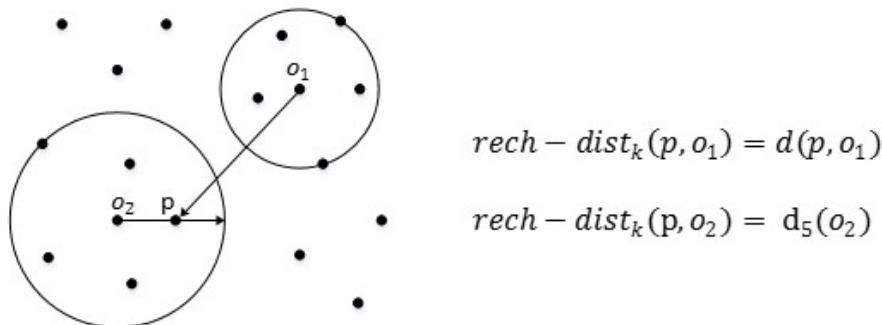


图 2-4 点  $d(p, o_1)$  与点  $d(p, o_2)$  到点  $p$  的第 5 距离

#### 5. local reachability density (局部可达密度)

点  $p$  的局部可达密度表示为:

$$\text{lrd}_k(p) = 1 / \frac{\sum_{o \in N_k(p)} \text{reach-distance}_k(p, o)}{|N_k(p)|} \quad \text{式 (2-12)}$$

该式的含义表示点  $p$  的第  $k$  邻域内点到  $p$  的平均可达距离的倒数。该式经常会被误解是  $p$  到  $|N_k(p)|$  的可达距离, 其实它实际表示的是  $p$  的邻域点  $|N_k(p)|$  到  $p$  的可达距离。并且, 如果有重复点, 那么分母的可达距离之和有可能为 0, 则会导致局部可达密度变为无限大。

这个值的含义可以这样理解, 首先这代表一个密度, 密度越高, 我们认为越可能属于同一簇, 密度越低, 越可能是离群点。如果  $p$  和周围邻域点是同一簇, 那么可达距离越可能为较小的  $d_k(o)$ , 导致可达距离之和较小, 密度值较高; 如果  $p$  和周围邻居点较远, 那么可达距离可能都会取较大值  $d_k(o)$ , 导致密度较小, 则很可能 是离群点。

## 2.2 时序曲线预测

### 2.2.1 神经网络

在现代信息产业中，神经网络是人们根据人脑中神经细胞中的运作原理（虽然实际上复杂的多）所模拟出来的计算系统，它通过大量的近似模拟去解决普通计算机程序无法解决的复杂问题，例如模式识别这类对人来说相对轻松的任务<sup>[7]</sup>。

神经网络通常涉及大量处理器并行运行并按层排列的处理器。第一层接收与人类视觉处理中的视神经相似的原始输入信息，例如人眼接受的光信号在计算机中被表示为图像。每个连续的层次都接收来自其前一层的输出，而不是重新采样，越后层的神经元接受离它越近的前序神经元传来的信号。最后一层产生系统的输出。在这些一层层的神经元节点中，每一个神经元处理节点都在网络中扮演着自己的角色，即是无法去理解这些节点实际对信息做了什么处理，但所有的节点高度相连后，整个神经网络的输出便和输入存在某种对应关系（例如输入图像，输出判别结果）。

神经网络在模型结构确立后，是根据训练样本来修改自身网络结构参数的，其最基本的学习集中在每一个神经元都根据前序输入进行加权，并不断权衡参数权重，使网络能够更加可能获得正确答案。通常情况下，一个网络需要经过大量已标注数据来进行训练，通过这些标注过的数据告诉网络在某种输入情况下理应输出什么，提供答案可让模型调整其内部权重，以了解如何更好地完成工作。

最初，系数是随机的；网络创建出来时对于结构化数据一无所知。每个节点的激活函数决定了节点对一个输入或一组输入的输出。所以节点点亮与否，取决于它所接收的刺激的强度（输入和系数的返回值）是否超过了激活的阈值。在一个所谓的致密或完全连接层，每个节点的输出会传递到后续层的所有节点。这后续会通过所有隐藏致密层，直到输出层为止，即该输入达成的决策的地方。在这个输出层，该网络关于输入的决策是针对预期决策的评估（比如，这个图片中的像素表示的是只猫还是只狗？）。通过比对网络的猜测和包含在测试集中的实际答案评估出误差，然后使用这些误差更新网络的系数，从而改变网络为图片中不同像素赋予的重要性的程度。目标是减少所形成的输出与预期输出之间的错误，正确地标识出来是狗还是猫。

通过提供大量的网络活动日志（每行日志都有一个时间戳）给递归神经网络，递归神经网络将了解正常预期的网络活动是什么样的。当把来自于网络的陌生活动提供给训练过的网络时，它就能够区分这个活动是正常预期的，还是入侵的了。

训练一个神经网络去识别预期行为有一个得天独厚的优势，因为它拥有大量的异常数据，或者还不足以精确分类的所有异常行为。我们在自己拥有的正常数据上训练网络，以便它能提醒我们注意未来的非正常行为。而针对攻击的训练也是在充分的数据上训练的。

## 第三章 定位数据

本章主要介绍了本课题所研究的腾讯定位数据的基本形式，并在 MATLAB 中作图分析其数据特征，包含时序特征及地理特征，最后根据分析结果讨论如何进行异常检测。

### 3.1 腾讯定位数据的形式

本课题所给定的腾讯定位数据是使用 MATLAB 的 Mapping Toolbox 生成的 GeoTIFF 格式的图像文件，每张图像文件的分辨率为 113\*150，文件中的 Reference 信息包含图像所表示的地理位置信息。本类图像数据经过处理后可以确定该图像单位像素上的值表示实际地图上的 0.01 经度与纬度覆盖面积上（约为平方一千米）定位终端数量，其中横坐标表示经度，纵坐标表示纬度。根据上述地理信息可以在 MATLAB 中画出该区域的定位终端热力图，为表现特征，采用归一化后终端定位数据作出该区域定位终端密度热力图如图 3-1 所示：

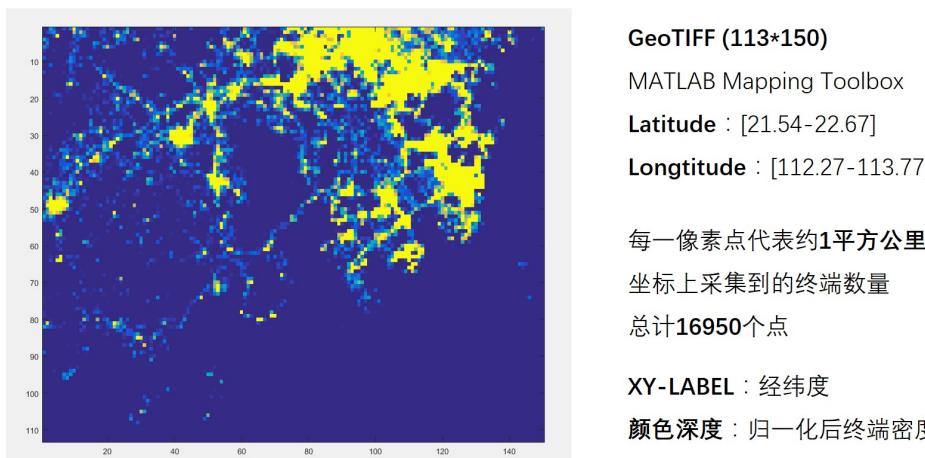


图 3-1 腾讯定位数据基本信息

将地理位置信息与实际世界地图进行比对，大致确定数据坐标为广东省珠海市沿海一带，如图 3-2 所示。同时，本数据集记录了 8 月 14 日至 9 月 30 日总计 48 天每天的每一小时区域定位终端数量。其中，该数据中已知的异常事件为 8 月 23 日的台风过境，由于台风袭来势必会导致图上的终端定位数量发生显著改变，本课题通过分析该时段的终端定位数据来研究定位数据的异常检测方式。

### 3.2 数据分析及应用

对于本课题，数据的维度涵盖时间与空间，直观上要分析单日的区域性台风影响带来的异常是困难的，我们首先需要对定位数据的规律进行分析，以便确定异常检测的算

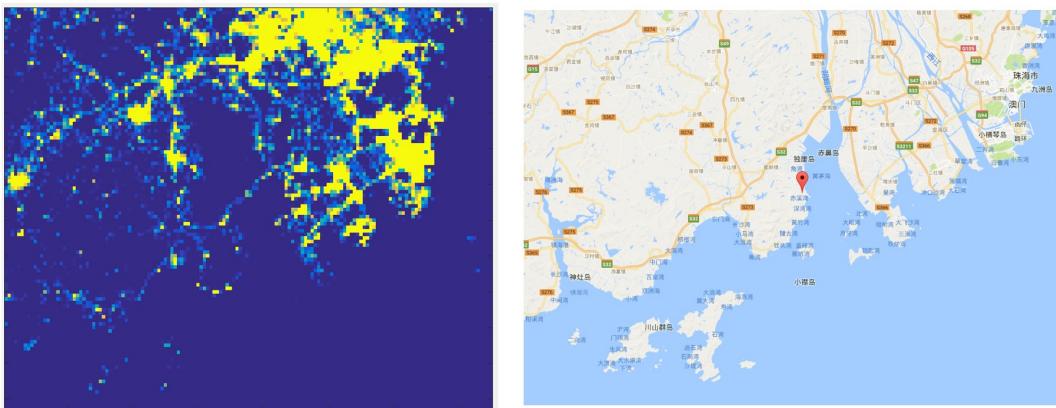


图 3-2 腾讯定位数据基于的实际地理坐标上地图

法思路。

### 3.2.1 数据的小时变化规律

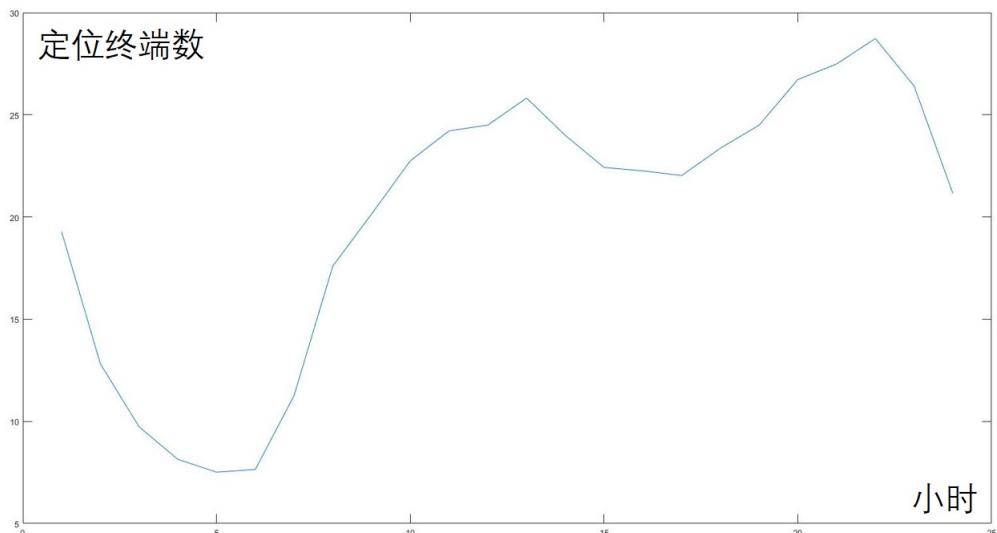


图 3-3 珠海市平均定位数据一天中随小时变化的曲线图

定位数据在时间上以小时的单位进行采样，可以通过观察某地一天 24 小时的终端数量值得出定位数据的小时变化规律。为便于观察，应选取终端数量较多的区域从而得出普适规律，而在 3.1 中我们通过比较已经确定该区域的实际地理位置，可以选择图中人口密度相对较高的珠海市进行研究。将珠海市的地理坐标范围确定后，取该区域的定位终端数量平均值并绘制出其从午夜 0 点至次日午夜 0 点的小时变化曲线图，如图 3-3 所示。

由图 3-3 中可以看出：午夜至清晨时间段为定位终端数量最少的时间段，大多数终端处于关闭状态，这是由于在这期间用户正处于睡眠状态造成的；而随着清晨至午间及晚间的推移，用户逐渐起床、工作、娱乐，定位终端数量也能观察到数量上的上升，而

后从午夜开始再次下滑。

为确定此规律符合每一个正常的自然天，而不是工作日或休息日的特殊情况或是误采了某个节日的数据，再取该区域的定位终端数量平均值并在一张图内用不同颜色的曲线绘制出其一周每一天 24 小时内的小时变化曲线，如图 3-4 所示：

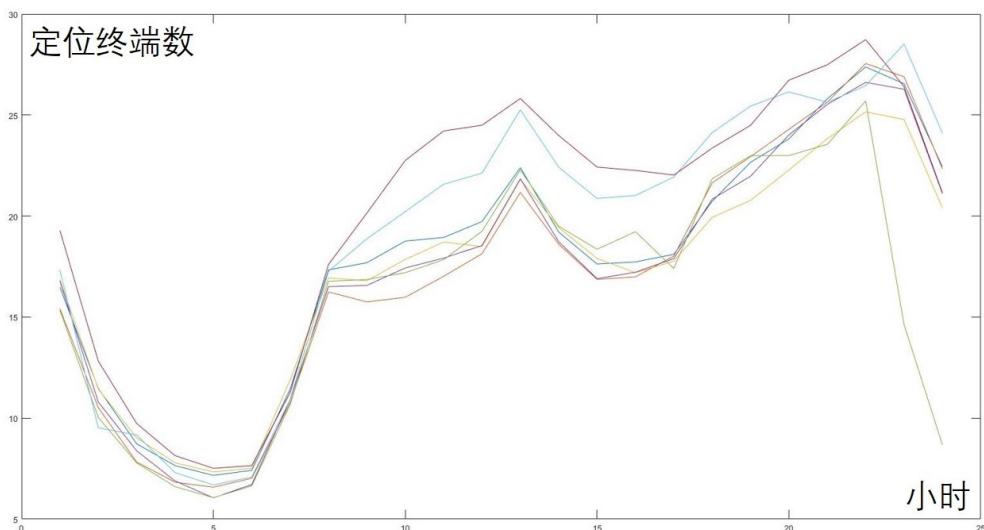


图 3-4 珠海市平均定位数据一周中各天随小时变化的曲线图

接下来我们再将一个正常天的 24 小时内终端数量变化的曲线与台风天的曲线进行比较，如图 3-5 所示：

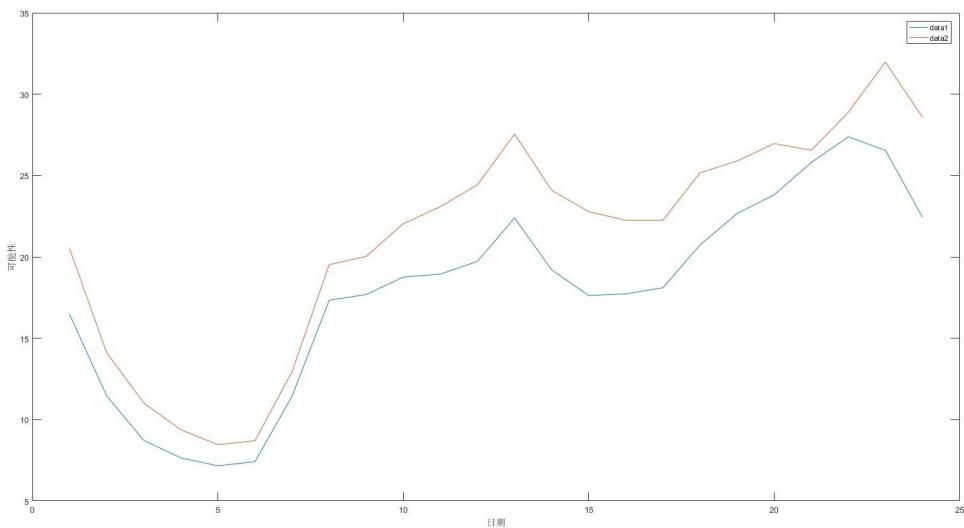


图 3-5 珠海市平均定位数据在正常日与台风日随小时变化的曲线图

由图 3-5 可以分析得到数据的小时变化规律：定位终端数据在确定的地点内一周中每天的变化趋势大致相同，而对于我们所已知的台风天异常，所造成的异常影响将不仅限于某小时带来的影响，而是会对整天各个小时的数据产生大的波动。

### 3.2.2 数据的日变化规律

在上一小节中，我们绘制出了台风天与正常天一天内 24 小时定位终端数量的变化并且观察得到台风天在一天内的值较正常值有较大的偏差。由于定位数据又在时空坐标上以自然天的单位进行采样，我们可以用同样的思路挖掘数据的日变化规律。为便于观察，同样选择图中人口密度相对较高的珠海市进行研究并选择一天当中定位终端数量较大的 13:00 时刻进行研究，取该区域每一天 13:00 的定位终端数量平均值并绘制出其在数据范围的 48 天内的日变化曲线图，如图 3-6 所示：

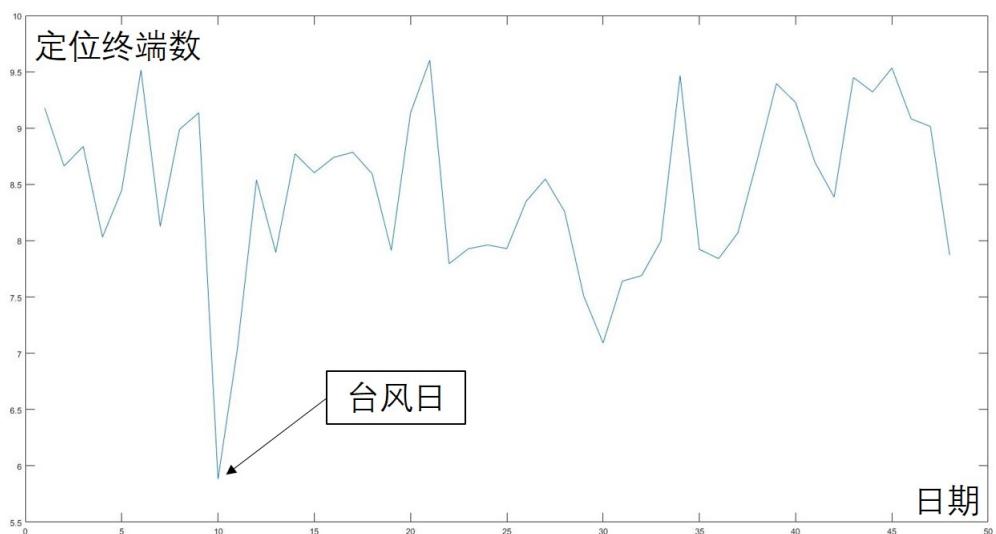


图 3-6 珠海市数据范围所有日期的 13:00 定位数据量变化曲线图

由图??中可以分析得到：在该地的每天 13:00 时刻，台风天相较于正常天的定位终端数有明显的偏差，是上一小节中 24 小时的定位终端变化曲线中的 13 点时刻偏差值的天数扩展。那么，如果能够通过选取任意的小时时间节点来代表整一天的定位终端数量值，台风天异常检测问题将会转化为每一天中的某确定小时的曲线异常检测问题。我们对此做进一步验证及讨论。

### 3.2.3 数据的总时空特征

将上述两章所分析的小时变化规律及日变化规律进行汇总，以 X 轴为数据范围内的自然日，Y 轴为自然日内的每一小时，在 Z 轴绘出 XY 形成的<日-时>时间节点上的定位终端数量，如图 3-7 所示：

由该三维图的多角度观察可以分析得到：一天 24 小时内的定位数据变化确实大致相似，并且定位数据在 8 月 23 日中整体出现了一个明显的沟壑。但是，与前两小节的小部分时间节点分析不同的是，在这张图上某些小时时间点上出现了数据的突变暴露出来。如果仅依据每天中单个小时的数据对整天进行分析，过大的单小时点可能会对算法的判断产生误差；另外，由于图中仅显示了某固定点的定位数据图，需要对每个地点的时空特征进行统计，得到总体的判断依据。

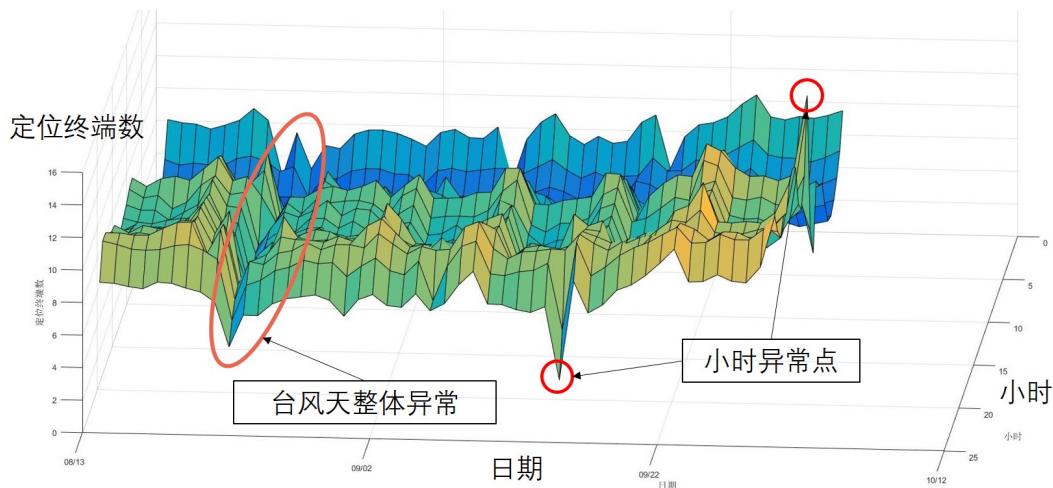


图 3-7 珠海市数据范围所有日期中定位数据量随小时变化曲线图

### 3.3 数据预处理及异常分析策略

#### 3.3.1 数据预处理

经过 3.2 对数据的分析，我们得到了定位数据的基本形式同时分析了其变化规律。在 3.2 中，我们选取的大多是极具代表性的区域（人口密度较高的珠海市）进行分析。但实际由于该定位坐标沿海，或是从任意时间节点上的定位终端矩阵或是绘制的区域热力图中也能观察得出：位于海面上的坐标终端数值存在大量接近零的点，如图 3-8 所示。这些点无论对于分析数据规律或是检测异常都是冗余的，比如海面上某点两时刻的值从 1 到 2 有 100% 的变化，会极大地影响基于变化率的检测方法，需要将这些点进行剔除。

02	3	4	5	20	11	13	3	8	8
7	2	0	6	14	14	4	6	38	6
323	503	701	260	116	16	11	13	52	57
151	341	772	324	231	326	169	325	128	95
303	876	222	807	870	346	438	153	125	155
41	38	67	1343	1290	237	123	0	64	
220	243	157	979	390	145	71	0	0	0
296	177	143	129	64	122	112	120	4	0
200	54	180	119	122	29	3	323	506	112
90	73	50	205	16	0	4	24	22	28
24	119	282	126	3	0	2	3	2	167
0	2	0	0	0	0	0	4	22	49
0	0	0	0	0	0	0	1	3	17
0	0	0	0	0	0	0	2	5	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	8	5
0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
10	0	0	5	0	0	0	0	0	12
0	8	299	37	5	0	0	0	71	71
2	120	54	36	0	0	0	7	126	184

图 3-8 定位数据量中的高密度与低密度区域对比

平均数是一个衡量区域内定位终端数量量级的基本方法，但是考虑到海面上可能会

在某时刻突然出现了高额终端数这种极端异常情况，不能轻易地将其忽略。采用平均数阈值去衡量有效点可能会因为天数过多而将这种异常点舍去，因此本课题更适合采用最大值阈值的方法对数据进行预处理。

创建一个与定位数据地图相同大小的 0-1 矩阵表征定位数据图中像素点是否有效（以下称为有效矩阵），读取定位数据中每一个像素点在所有时刻的值，如果这些值中没有一个超过 10（1 平方公里的区域中没有一个时刻超过 10 个定位终端），则将有效矩阵相同位置处置为 0，否则置 1。经过这样处理后定位数据地图中只有约 2000 多个点有效，极大地加快检测速度的同时也避免了突变的错误舍去。

### 3.3.2 曲线异常分析策略

在第一章中我们讨论过异常的分类，而对于本课题所讨论的异常，应被归类为环境异常。环境属性即是时空坐标与地理坐标，行为属性是某地理坐标下在某时间点上的定位终端数量。台风天的检测目标即是输入所有的时空与地理坐标上的定位数据来检测出某一天的时空异常（出现台风的迹象）。

3.2 中我们讨论了数据的时空特征，对于某固定的地理坐标，其可能会在某个小时时间点上出现大的偏差，即使每天按照小时的变化定位数据的曲线大致相似，这些突然的抖动不能被忽略，所以应当使用一天中平均的定位数据来衡量。而在 3.3.1 中我们又对数据进行了预处理，减少了地理分析量，同时排除了一些会对算法造成影响的无效数据点。经过上述讨论，对于本课题所研究的台风天异常，对经过数据预处理后的筛选点进行时空维度上的曲线异常检测，使用每天的平均数据来分析，判断异常日期是哪一天或是全部为正常数据；再从地理上统计地图上所有已筛选点的异常日期，如果地图上的大部分点都指向某一天存在异常的，即可认为该天是异常天。流程图如图 3-9 所示

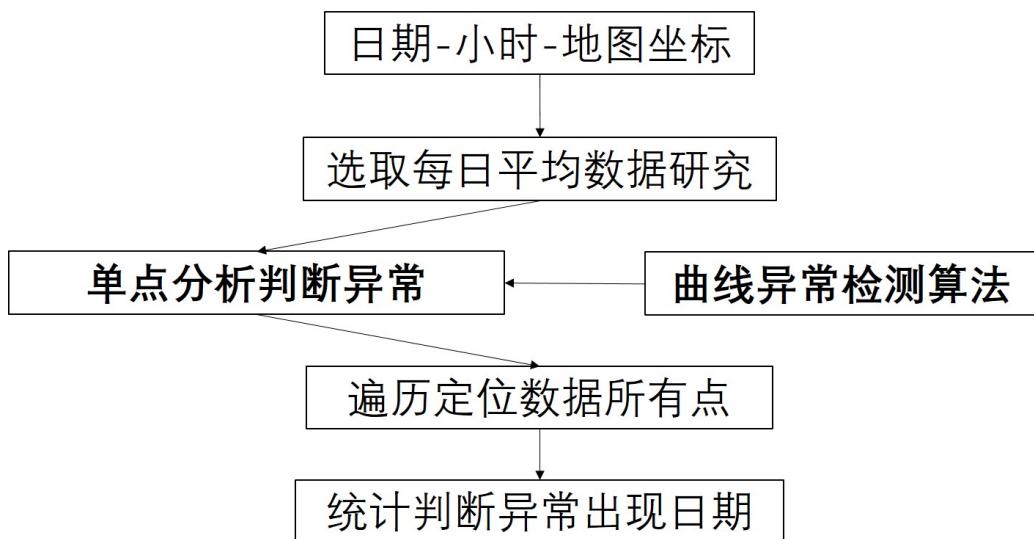


图 3-9 基于曲线的异常检测算法步骤

而对于其它类型的异常，例如某小时突然出现的剧烈抖动，由于会出现台风所造成整体异常偏差高过该抖动的偏差的可能性，以及该剧烈抖动发生的或许只是部分区

域，我们不能用上述整体区域整天的曲线分析异常检测算法来解决，对于可能的解决方案，我们将会在第 5 章中进行进一步讨论。

## 第四章 基于曲线分析的定位数据异常检测

经过在第3章中对数据的分析，我们将本身<时-空>的坐标分开分析，先对单个空间上的坐标点进行时空曲线异常检测，再统计空间上的规律，得出台风异常天的检测结果。在本章中，我们对每一种方法进行了设计与验证，查看其是否能够成功检出整个区域8月23日的台风异常结果，并分析了各个算法的优势以及弊端。

---

### 算法1 基于曲线分析的定位数据异常检测

**输入：**有效单个定位数据点上的日期时序定位数据值  $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_m\}$

**输出：**定位数据平面上各有效点在哪天被判定为异常的日期统计图

- 1: 计算有效单个定位数据点上的日期时序定位数据值的平均值绘制成曲线
  - 2: 使用曲线异常检测算法来检测出异常点  $x_a$  的日期  $t_a$
  - 3: 如果存在异常点  $x_a$ ，则将该日期  $t_a$  的数量值  $v_a$  加一
  - 4: 以  $t$  为横坐标， $v$  为纵坐标，绘制出定位数据平面上各有效点在哪天被判定为异常的日期统计图（横坐标-日期，纵坐标-该日期检测到的异常点数量）
- 

## 4.1 基于差分的异常检测算法

### 4.1.1 差分算法

异常是指某个数据严重偏离正常数据的范围之内，在时序数据中，如果某时刻的定位数据较其周围时刻的数据有很大的波动，该时刻的定位数据也是异常的。使用差分算法计算每一个点与其左右两个时间节点上的数据浮动比例，当这个比例超过某种阈值后，即可认为该点是异常的。

---

### 算法2 基于差分的异常检测算法

**输入：**有效单个定位数据点上的日期平均时序定位数据值  $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_m\}$

**输出：**该定位数据点被判定为在  $t_a$  出现异常或未出现异常

- 1: 计算每个数据点与周围时刻上点的差分平均值  $d_i = \frac{|x_i - x_{i-1}| + |x_i - x_{i+1}|}{x_i}$
  - 2: 如果存在某点  $x_a$  的差分平均值  $d_a \geq 20\%$ ，则认为该点的日期  $t_a$  出现了异常；如果有多个点出现异常，选择差分平均值最大的那个点作为异常日期；如果没有异常，则无视该点继续判断其它定位数据点
- 

### 4.1.2 结果分析

采用了20%的变化率阈值的差分算法得到了如图4-1所示的异常天分布情况。该图横坐标为日期，纵坐标为经过筛选后的定位点中有多少在某日期中被算法认为是异常

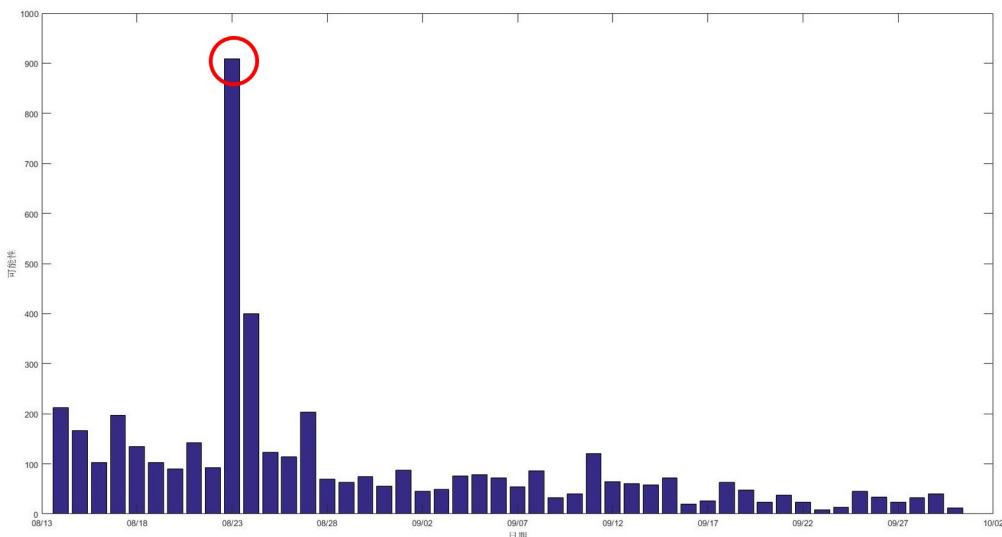


图 4-1 基于差分的异常检测算法定位数据异常天检出分布图

的。由此可见本算法成功检出了异常天，并且误检率已经较低，并且由于本算法完全是线性运算，计算量较低。

差分算法是一种更加针对平稳的时序数据算法，因为它只考虑每一个数据点相邻时序上的点，所以对于时序上突然抖动的数据更加敏感。而且，使用变化率来衡量的差分算法也对小数据的微量浮动很敏感。在本课题中，因为已经事先对数据进行预处理，排除掉了小数据的干扰才使得差分算法准确度较高。

另外，本算法中所涉及的差分采取的是时序上左右两点间的差分值，真实异常点的附近几点也会受到异常点的影响导致可能被归类为异常点，应扩大差分的范围，减少异常点对周围点的影响。并且，实际情况中可能会出现持续的异常，此时差分方法不再适用，它只会关注陡然的变化而忽视了变化之后连续的异常，甚至还会认为持续异常之后回到正常状态的变化是异常的。

## 4.2 基于小波变换的异常检测算法

### 4.2.1 离散序列小波变换

离散序列的小波变换基于著名的 Mallat 算法，离散序列值  $x$  与其第一层分解后的高频系数  $D_1$ （细节部分 Detail）的关系是  $x$  经过高通滤波器  $g$  滤波后再下采样，与低频系数  $A_1$ （近似部分 Approximate）的关系是  $x$  经过低通滤波器  $h$  滤波后再下采样；然后继续对低频系数  $A_1$  进行第二层分解，依此类推，即离散信号  $x$ ，经过多层分解后最后各分解系数合起来就是变换的结果。

对于本课题的数据，因为数据规模较小，直接采用一层分解即可从原始离散序列分离出有效高频部分  $D_1$ <sup>[8]</sup>，即原始信号的突变分量（异常分量）在  $D_1$  中体现。对  $D_1$  进行模糊处理并结合  $A_1$  重建信号与原始信号差分<sup>[9]</sup>，取最大值的横坐标（日期）即可得到该地理坐标下的异常日期。

**算法 3 基于小波变换的异常检测算法**

**输入:** 有效单个定位数据点上的日期平均时序定位数据值  $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_m\}$

**输出:** 该定位数据点被判定为在  $t_a$  出现异常或未出现异常

1: 计算定位数据序列  $\mathbf{x}$  的一层离散小波变换系数  $[\mathbf{D}_1, \mathbf{A}_1]$

2: 对高频系数  $\mathbf{D}_1$  进行模糊处理, 并结合低频系数  $\mathbf{A}_1$  重建信号  $\dot{\mathbf{x}}$

3: 将原始信号  $\mathbf{x}$  与重建信号  $\dot{\mathbf{x}}$  进行差分, 选择差分结果最大的那个点  $x_a$  的日期  $t_a$  作为异常日期。

**4.2.2 结果分析**

使用一层的离散小波变换成功检出了异常天, 如图 4-2 所示, 但是有数量相对较大的误检测, 并且小波分解计算量较大。

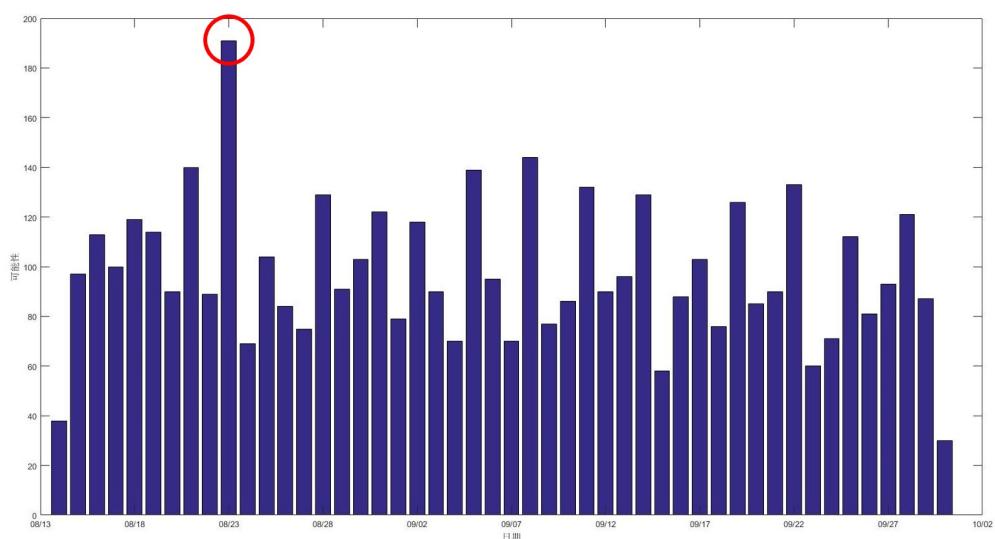


图 4-2 基于小波变换的异常检测算法定位数据异常天检出分布图

考虑到离散小波分解后是直接采用寻找最大值寻找异常天的横坐标的方法, 在某些地理位置上其变化幅度较小, 导致曲线本身就很平滑, 采用小波变换后提取到的信号高频特征不明显, 从而导致取最大值时发生错误造成了误检测。小波变换的确可以将离散的曲线信号中细微变化的部分突出, 但由于台风异常的变化本身就很剧烈, 导致数据的一层小波高频分量不明显。

**4.2.3 结果处理优化**

在 4.1.2 中我们讨论的小波变换能够将时序数据的高频分量提出, 进而放大原始数据的噪声异常点。但由于在这一方法中将异常放大后直接使用取最大值的策略导致很多本身没有异常的点也被错误的认为是异常点。所以, 我们对该方法进行了优化, 将原始时序数据经过小波分解后再进行差分运算, 得到异常天出现的日期。同时, 我们又对 4.1 中提到的差分运算运用到小波运算之中, 如图 4-3 所示。同样由于台风异常的变化

本身就很剧烈，使用小波运算提取时序数据中的高频分量后再进行差分的效果不如直接对原始数据进行差分的效果好。在普通的差分算法中，平滑的数据中出现了陡变分量则马上被检出。

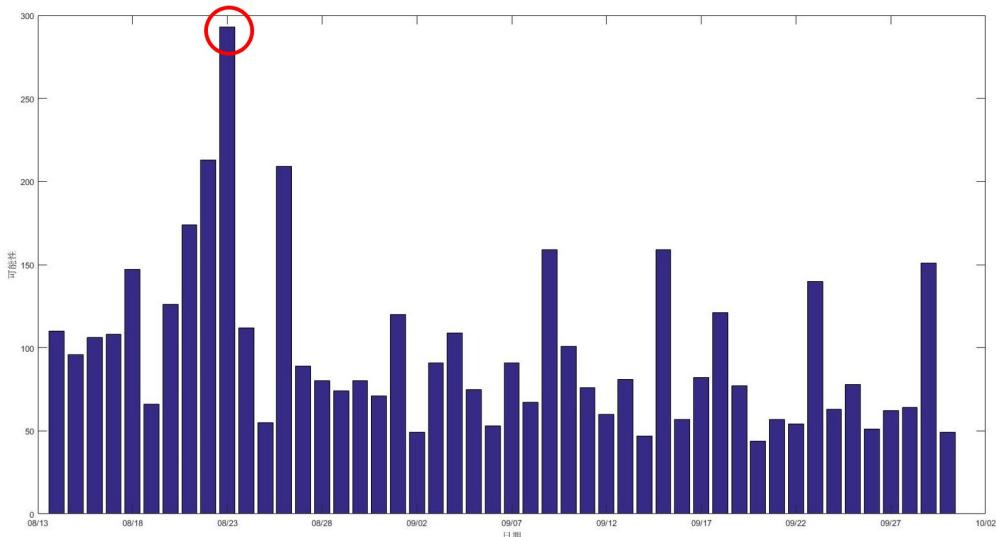


图 4-3 小波变换算法处理结果优化后的定位数据异常天检出分布图

## 4.3 基于极大似然估计的异常检测算法

### 4.3.1 极大似然估计与 $3\sigma$ 准则

在第 2 章中我们讨论的极大似然估计的基本含义，其是用来估计一个概率模型的参数的一种方法，即通过若干次试验，观察其结果，利用试验结果反推最有可能（最大概率）导致这样结果的参数值。经过第 3 章的讨论，我们已知某地理坐标上的定位终端数在没有异常事件到来的情况下浮动规模应大致符合正态分布。基于此假设后，对于某地的时间序列求解最大似然估计，求解正态分布下似然方程得到唯一解  $(\mu, \sigma^2)$ 。

在有了似然估计的解之后，我们得到了数据本身的一种拟合分布情况。在正态分布下，数值分布在  $(\mu - 3\sigma, \mu + 3\sigma)$  中的概率为 0.9973。可以认为，正态分布中 Y (在本课题中为定位终端数量) 的取值几乎全部集中在  $(\mu - 3\sigma, \mu + 3\sigma)$  区间内，超出这个范围的可能性仅占不到 0.3%。本课题中所涉及的异常数据如果较正常数据偏差大，很有可能会落在此小区间中被检出，即可得到异常天的日期，否则认为不存在异常天。

### 4.3.2 结果分析

使用极大似然估计拟合数据的分布，并基于  $3\sigma$  准则将异常值筛选出后，得到了如图 4-4 所示的异常天分布情况。本算法仍然检出了异常天，但误检率仍然较高，且由于要求解每一个坐标点的似然估计方程，计算量相对于小波变换更加大。

**算法 4** 基于极大似然估计的异常检测算法

**输入:** 有效单个定位数据点上的日期平均时序定位数据值  $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_m\}$

**输出:** 该定位数据点被判定为在  $t_a$  出现异常或未出现异常

- 1: 在正态分布下求解定位数据序列  $\mathbf{x}$  的似然方程唯一解  $(\mu, \sigma^2)$
- 2: 寻找定位数据序列  $\mathbf{x}$  中所有处于  $(\mu - 3\sigma, \mu + 3\sigma)$  之外的数据点
- 3: 如果存在这样的点  $x_a$ , 则认为该点的日期  $t_a$  出现了异常; 如果有多个点出现异常, 选择差分平均值最大的那个点作为异常日期; 如果没有异常, 则无视该点继续判断其它定位数据点

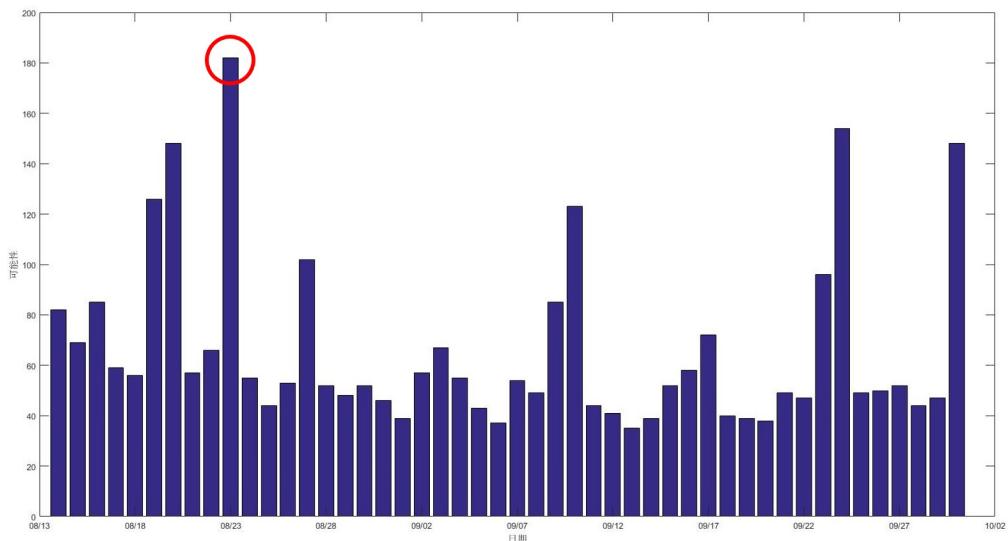


图 4-4 基于极大似然估计的异常检测算法定位数据异常天检出分布图

考虑到极大似然估计是基于现有的数据对原始分布进行拟合，需要较为庞大的数据量支撑以便充分拟合，才能忽略个别噪声的影响。而在本课题所涉及的数据中，数据量较小且这些用来估计的现有数据中也包括了异常的数据，如果异常值偏离很大，估计出的参数会极为不准确，从而导致误检。另外，为了便于计算，本算法认为原始数据基于正态分布，实际情况下需要进行长时间的统计，对区域的定位终端数量有一个充分的采样继而判断该地的数据符合哪一种分布，最后对这种分布的参数进行极大似然估计，才能得到较为准确的模型以及判断异常的条件。

## 4.4 局部异常因子检测算法

### 4.4.1 局部离群因子

在第 2 章中，我们提及了局部可达密度的概念，点  $p$  的局部可达密度表示为：

$$lrd_k(p) = 1 / \frac{\sum_{o \in N_k(p)} \text{reach} - \text{distance}_k(p, o)}{|N_k(p)|} \quad \text{式 (4-1)}$$

Breunig 等人提出了局部离群因子的概念<sup>[10]</sup>，它表示点  $p$  的邻域点  $N_k(p)$  的局部可达密度与点  $p$  的局部可达密度之比的平均数。使用公式表示如下：

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} = \frac{\sum_{o \in N_k(p)} lrd_k(o)}{|N_k(p)|} / lrd_k(p) \quad \text{式 (4-2)}$$

式 4-2 即局部离群因子公式的含义是点  $p$  的邻域点  $N_k(p)$  的局部可达密度与点  $p$  的局部可达密度之比的平均数。

如果比值该接近 1，则  $p$  的邻域点密度相似， $p$  可能是邻域中的一个簇。如果局部离群因子的值小于 1，则  $p$  的密度高于邻域点密度， $p$  是稠密点。如果比值大于 1， $p$  的密度小于其邻域点密度， $p$  更可能是异常点。

局部离群因子算法检测异常的主要思想是通过比较每个点  $p$  及其邻域点的密度来确定点是否是一个异常。如果点  $p$  的密度较低，则越有可能被识别为异常。至于密度，它是由点之间的距离来计算的。点越远，密度越低，距离越近，密度越高，这完全符合我们的理解。此外，由于离群因子的值是通过点的  $k$  邻域而不是全局计算来衡量的，所以称为“局部”异常因子，因此，对于图 2-2 所示的两种类型数据集  $C_1$  和  $C_2$ ，局部异常因子算法可以完全正确地处理，而不会因为数据密度分散情况不同而错误地将正常点判定为异常点。

在本课题中的定位数据中，数据是基于时间坐标上的一维数值，但使用局部异常因子算法求解时，无需考虑数据的时刻位置，只需要关注在哪一天中数据的定位数据值与其他所有值都相差甚远，这一天即是异常出现的日子。在算法介绍的图例中考虑的距离是二维坐标上的距离，在本课题所研究的数据中，定位数据量的距离只需要用一维距离，即定位量之间的绝对值来表征。同时在使用局部异常因子算法进行求解时，可能会出现

由于重复点的情况所导致的局部邻域密度值为无穷大，我们对这种情况进行特殊处理，即将重复点往周围移动 1 个单位，即对重复点的定位数据值 +1，因为在数据分析里提及到预处理后的定位数据量至少有一天是大于 5 的，所以挪动 1 个单位对结论没有过大的影响。

#### 算法 5 局部异常因子检测算法

**输入：**有效单个定位数据点上的日期平均时序定位数据值  $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_m\}$

**输出：**该定位数据点被判定为在  $t_a$  出现异常或未出现异常

1: 根据第 2 章中式 2-11 计算  $\mathbf{x}$  中所有点的第 k 可达距离

2: 根据第 2 章中式 4-1 计算  $\mathbf{x}$  中所有点的局部可达密度

3: 根据本章中式 4-2 计算  $\mathbf{x}$  中所有点的局部离群因子。如果存在这样的点  $x_a$ ，它的值远大于 1，则认为该点的日期  $t_a$  出现了异常；如果有多个点的值符合条件，选择最大的  $x_a$  作为异常日期；如果没有值大于 1，则无视该点继续判断其它定位数据点

#### 4.4.2 结果分析

采用了局部离群因子来计算异常值的位置，本算法也成功检出了异常天，如图 4-5 误检率较低，但是由于局部异常因子算法需要对每个点局部异常因子进行计算，计算量复杂，效率较低。当 K 值（邻域范围）增加时，计算量进一步加大。

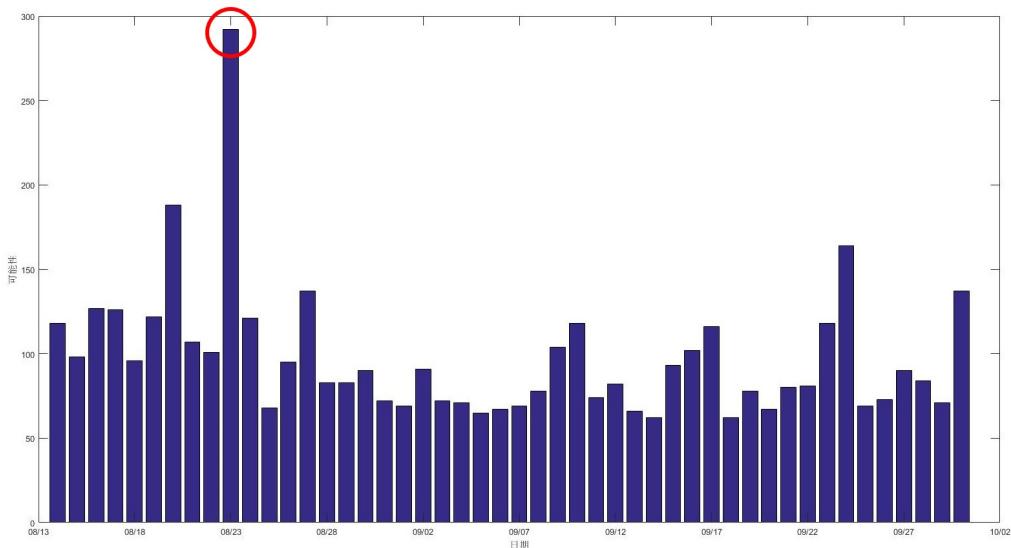


图 4-5 局部异常因子检测法定位数据异常天检出分布图

局部离群因子算法与差分算法等考虑时序数据的连贯性不同，它无论时序数据还是纯粹的行为异常点都可以处理，它仅仅将数据点放置在同一张平面，用数据之间的联系来计算某点是否异常。对于 K 值的选择问题，由于该值的选择标准没有一个普适的定量方法，本算法采取了 K=2 的值<sup>[11]</sup>。

## 4.5 曲线异常检测算法总结

表 4-1 曲线异常检测算法检测情况表

检测算法	异常点数	运行速度(秒)
差分运算	909	1.14
离散小波变换	191	7.38
小波变换-结果改进	293	8.75
最大似然法( $3\sigma$ 准则)	182	14.32
局部异常因子算法	292	18.91

本章节对台风天的异常检测算法进行了讨论，将大范围的地理坐标下的整天异常问题细化为每一点的整天平均定位数据异常问题。在这个基础之上，我们采用 4 种算法对该问题进行了研究。对于本课题所研究的数据，时序关联性很强，每一天的平均定位数据量值相仿，而台风当天相对于邻近日整体的变化浮动相对较大，是一个极其明显的抖动。差分分析法对于本课题所研究的范围性异常检测问题快速有效，因为其能够捕捉到明显陡变的时序数据，如表 4-6。但是如果对于持续时间较长的异常，差分算法不再有效。由表 4-1 可见除了差分算法的检出量较高之外，其余方法的检出量大致处于同一水平。由于本课题的数据量较小，使用统计的方法来衡量异常的偏差较为困难；局部异常因子算法则存在和差分分析一样的问题，当异常点不再孤立时较难检出，但其对于数据的时序性没有要求，只是检出一个数据是否在一系列数据中是异常的。

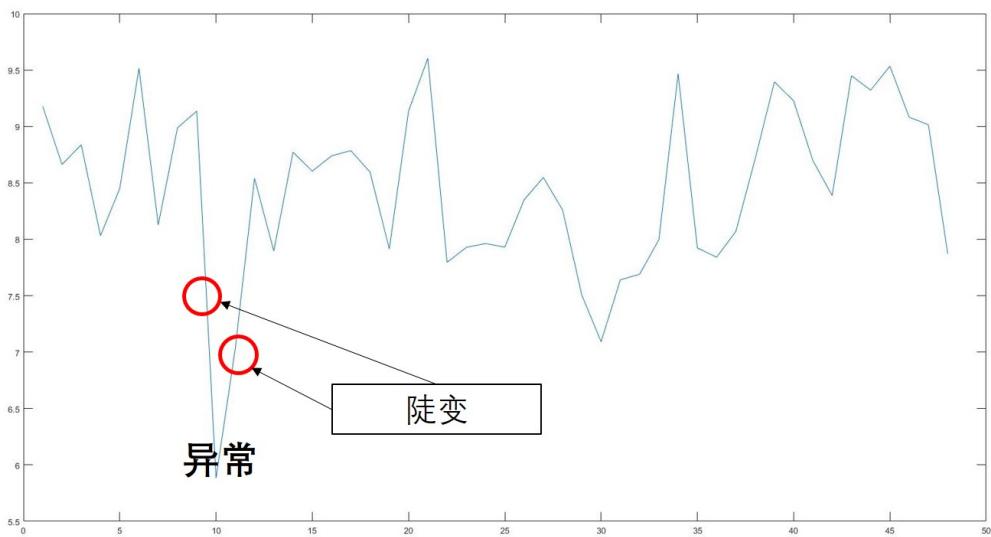


图 4-6 差分算法能够检出异常时刻的数据陡变

我们在第 3 章最后讨论了某小时时刻的数据陡变情况，由于这些陡变的值被更大的台风天变化所忽略或是只出现在少部分地理位置，所以本章的算法没有将这些异常值检出，我们将会在下一章中对其进行进一步讨论。

## 第五章 基于曲线的定位数据预测与异常检测

本章主要基于第4章中的整体区域台风天异常检测基础，提出了时序数据的预测估计方法，并根据神经网络训练模型预测当前时间节点的值与实际值进行比对，进一步讨论了异常检测的另一种思路，探索一种方法去完善异常检测算法。

### 5.1 异常的检测与预测

时间序列是根据时间顺序得到跟时间相关的变量或者参数的观测数据<sup>[12]</sup>。对时间序列的研究主要是挖掘其中有价值的信息，找到其中变化的内在规律<sup>[13]</sup>。在第4章中我们所讨论的问题都是基于现有的<空-时>数据，从其中检出哪一天存在区域性的异常，检测是一个判别的过程，是从已知数据中检测出异常的模型，这种检测方法更倾向于数据清洗。同时，异常检测在生活中也多用于即时的判断，例如网络流量的异常检测可以对DDoS攻击进行相应，使网络运营商做好应急预案。对于本课题所研究的定位数据来说，如需实际应用那么检测方法更加强调实时性，即输入一组新的数据后基于现有的数据能够判别新数据是否存在异常。



图 5-1 使用曲线预测方法来检出异常的步骤

在前序章节中我们曾提及本课题所研究的腾讯定位数据存在一些局部地理位置上的小时点突然异常，而第4章的方法更加适合检测整天整块区域的异常，这一章中我们更关注这些异常，通过预测的方法，预测当前时刻的定位数据值，并与实际值比较来检出异常，如图5-1所示。时间序列预测是指根据现有的和历史的时间序列的数据，建立能反映时间序列中所包含的动态依存关系的数学模型<sup>[14]</sup>。我们可以通过建立模型来预测新的数据理应符合什么区间，并与实际数据进行比较，判断其是否异常。

## 5.2 基于神经网络的曲线定位数据预测

### 5.2.1 神经网络分类

神经网络是一种重要的机器学习技术。它是模仿生物神经网络（动物的中枢神经系统，特别是大脑）的结构和功能的数学模型或计算模型。它被用来估计或近似函数。神经网络是由大量的人工神经元连接来计算的。在大多数情况下，人工神经网络可以在外部信息的基础上改变内部结构，它是一个自适应系统。通过校正每个层的权重（学习）来创建模型的过程被称为自动学习过程（训练算法），通过训练样本的校正。由于网络结构和模型的不同，具体的学习方法不同，并使用反向传播算法（反向传播/反向传播/反向传播，使用差分增量规则来修改权值）来验证该方法。通过这样学习（训练与验证）的过程，它可以对目标函数进行相对完整的模拟，如图 5-2 所示。

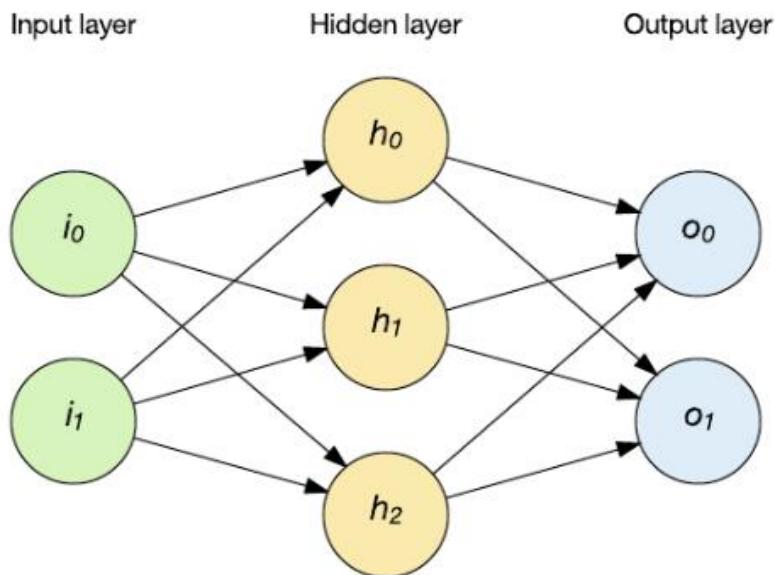


图 5-2 前馈神经网络示意

神经网络可分为前馈神经网络和递归神经网络，根据其是否包含延迟或反馈，将具有延迟或反馈环节的神经网络称为递归神经网络。前馈神经网络由前向传播的神经元组成<sup>[15]</sup>，连接仅馈送到后续层，因为没有回馈的因素（即同一列神经元间没有相连的关系），其主要被用于静态判断或预测，例如卷积神经网络被用来识别静态图像。递归神经网络中一个显著的特点就是神经元间的输出回馈到前序神经元中，神经元之间的联系使得神经网络又可以解决前馈神经网络无法解决的时序问题，由于 RNN 包含循环，它们可以在处理新输入的同时存储信息<sup>[16]</sup>。这种存储器使得它们非常适合于处理在输入之前必须考虑的任务，例如时间序列数据。在 4.1 中，我们曾用差分算法检测异常值，差分算法的本质在于通过相邻时间节点的浮动变化率对异常点进行判断，即根据变化趋势来检测是否存在异常。上述神经网络也需要使用训练出的模型并根据前几项时间节点上的值，来预测当前时间节点上的值，如图 5-3 所示。

对于本课题所讨论的定位数据异常检测，动态神经网络起到了先预测当前值的作

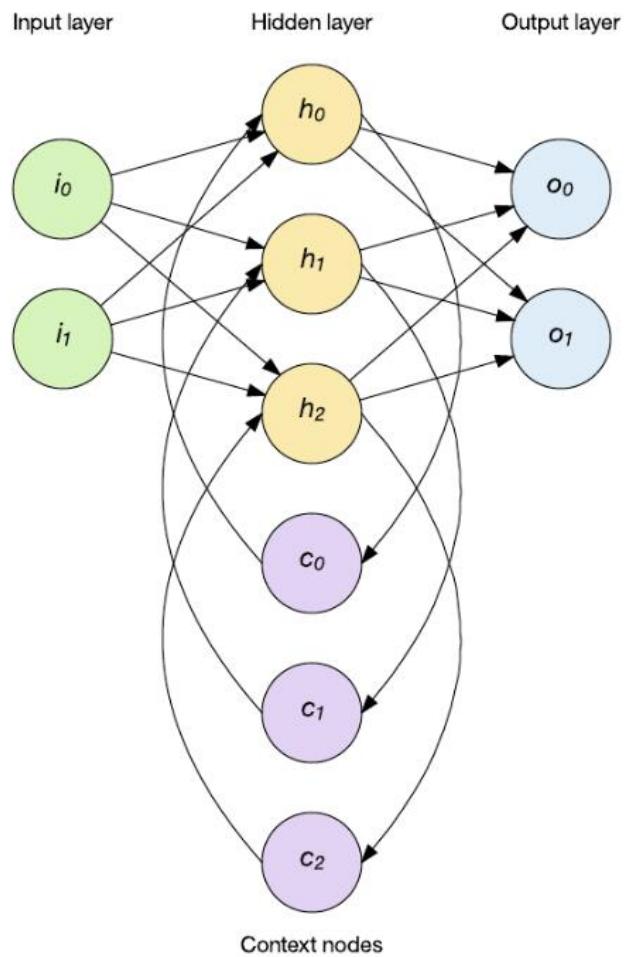


图 5-3 带有回馈的递归神经网络

用，而对于这个预测值，我们只需要和当前实际值进行比较，如果他们相差的幅度达到一定规模，则认为当前值存在异常，应当启动应急方案来应对。

### 5.2.2 定位数据预测

本章节将使用 MATLAB 软件中的神经网络工具箱，以某地区的过往定位数据时间序列为输入，该地区当前时间节点的定位为输出，依据输入的过往时空数据构建神经网络模型，以时间序列预测方法来进行当前定位数据量预测<sup>[17]</sup>。借助神经网络的非线性问题处理能力，根据不同的实际情况，对数据进行训练模型，预测当前定位数据值并与实际定位数据对比，验证方法的可行性，如图 5-4 所示。

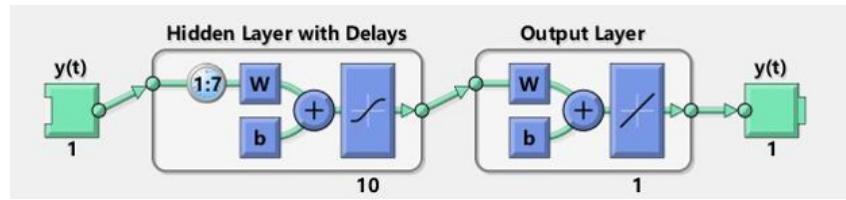


图 5-4 MATLAB 工具箱中非线性自回归模型

在 4.2 的极大似然估计中，我们提到异常的数据会对极大似然估计造成很大的误差，本章我们所讨论的定位数据预测同样需要考虑这个问题，应当将异常的数据进行剔除，使用正常的数据训练网络结构并使用它来预测当前时间节点上的值。接下来的部分和第 4 章一样，将经过预处理后的定位数据导入 MATLAB 中，使用 ntstool 命令进入时间序列工具箱进行训练。采用动态神经网络非线性自回归模型，网络训练时把数据分为三类：训练数据、验证数据和测试数据，三者比例设置为：70%、15%、15%。由于每一个自然周的变化大致相似，所以采用了前 7 个点作为预测的前序值。通过训练数据和验证数据来训练神经网络的模型并自动反向传播调整网络参数，如图 5-5 所示。

### 5.2.3 分析结果

---

#### 算法 6 基于神经网络的曲线定位数据预测及局部异常检测

---

**输入：**待预测有效单个定位数据点的日期  $t_a$  与其前 7 日同时刻的定位数据值  $\{x_{a-1}, x_{a-2}, \dots, x_{a-7}\}$

**输出：**该定位数据点  $x_a$  是否出现异常

- 1: 向模型中输入  $t_a$  与  $\{x_{a-1}, x_{a-2}, \dots, x_{a-7}\}$
  - 2: 模型给出输出预测结果  $\hat{x}_a$
  - 3: 计算预测结果  $\hat{x}_a$  与实际值  $x_a$  间的绝对值，如果超过某个阈值，则认为其为阈值
- 

如图 5-6 所示，根据训练的模型，对 9 月 23 日至 9 月 30 日中的午间实际定位数据进行预测，可以观察到前几日的预测与实际值差较为相近，而 9 月 30 日中出现了较大的变化，预测值与实际值相差极大，因被归类为异常。

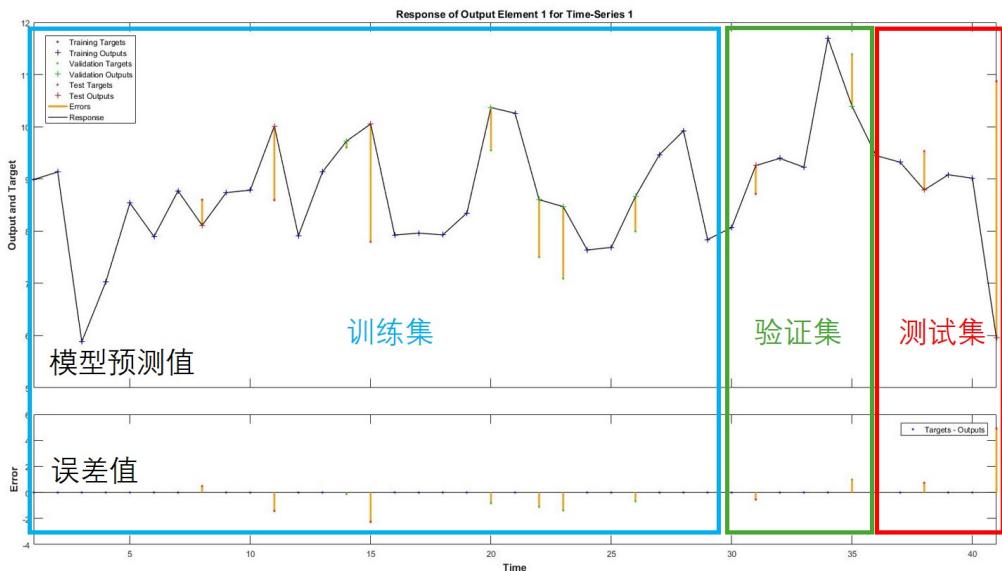


图 5-5 数据的不同部分被用于训练与验证网络

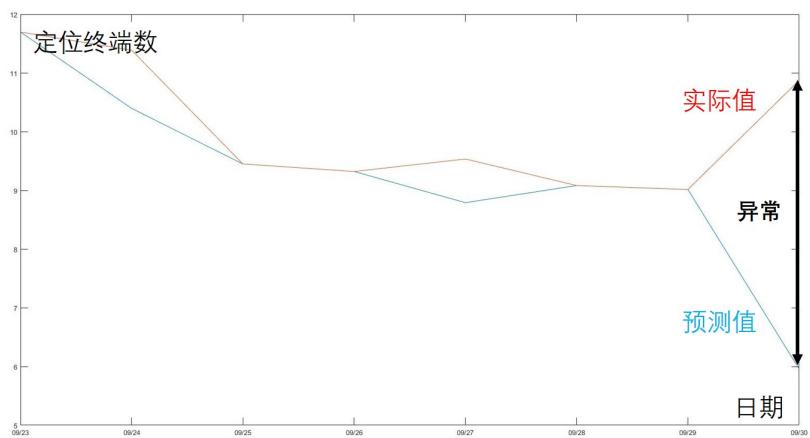


图 5-6 非线性自回归网络被用于预测小时时刻的值来检出异常

又如图 5-7 所示, 针对日期平均数据重新训练模型, 对 8 月 21 日至 8 月 27 日的平均定位数据进行预测, 可以观察到前几日的预测与实际值差较为相近, 而 8 月 23 日中出现了较大的变化, 预测值与实际值相差极大, 说明使用预测的方法也对类似于台风这样的区域性整天的异常有效。

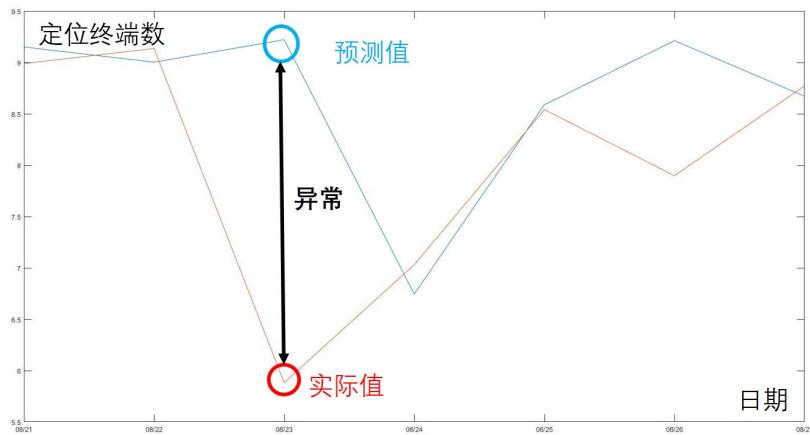


图 5-7 非线性自回归网络被用于预测整天整块区域的平均值来检出异常

本章的实验结果图是反复训练了多次模型才得到的, 可见神经网络对于小规模数据点的规律拟合较差, 当正常数据点的训练样本足够多时, 模型能够进行充分训练, 预测结果将更加优秀, 也更容易衡量实际值与预测值相差多少时被划定为异常的门限。

## 第六章 基于区域的定位数据异常检测

前述几章我们主要讨论了基于腾讯时序定位数据的异常检测及预测算法，可以对异常发生的整体区域进行分析，在第3章中我们也提及该定位数据中存在大量海面上的无效数据，实际上发生异常的可能只是部分区域。在这一章中，我们根据已知的异常时刻，与正常时刻的定位数据图进行比较，尝试去寻找具体发生异常的子区域。

### 6.1 基于图像的异常区域检测

#### 6.1.1 相邻帧间差分法

相邻帧间差分法是一种视频分析上检测物体运动的方法。由于摄像机采集的视频序列具有连续性的特点。如果场景内没有运动目标，则连续帧的变化很微弱，如果存在运动目标，则连续的帧和帧之间会有明显地变化。帧间差分法(Temporal Difference)就是借鉴了上述思想。由于场景中的目标在运动，目标的影像在不同图像帧中的位置不同。该类算法对时间上连续的两帧或三帧图像进行差分运算，不同帧对应的像素点相减，判断灰度差的绝对值，当绝对值超过一定阈值时，即可判断为运动目标，从而实现目标的检测功能<sup>[18]</sup>，如下图6-2所示。

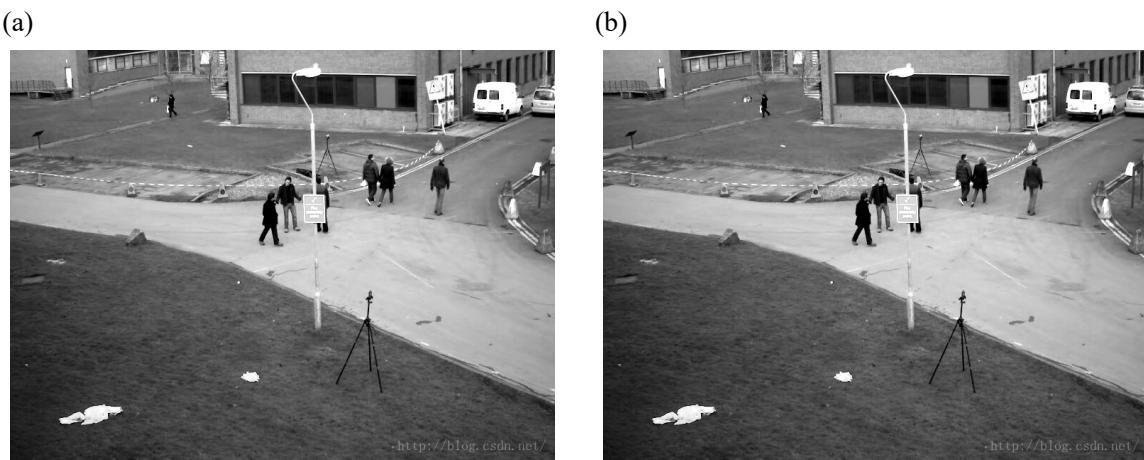


图 6-1 相邻帧间差分法原图

#### 6.1.2 高浮动区域检测

对于本课题的定位数据，帧间差分法所实现的功能即是检测出由于异常导致的子区域定位数据的剧烈抖动。由于台风过境这类异常事件的产生，使得图中某一部分的定位数据相较于正常数据产生较大的变化，这些变化可以使定位数据产生的图像之间差分后



图 6-2 相邻帧间差分法差分图

在图中明显观察得到，之后再用滑动窗的方法去检测具体区域。正常情况下的数据帧间差分效果如图 6-3 所示：

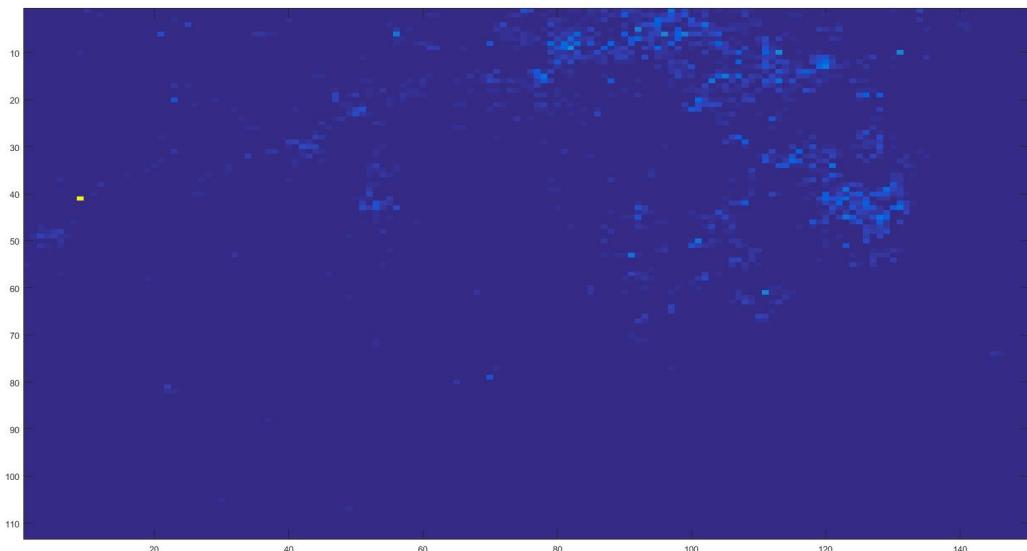


图 6-3 正常定位数据间相邻帧差分图

在第 5 章中我们检测出 9 月 30 日午间的定位数据存在异常，我们将其与 9 月 29 日的同时刻数据进行帧间差分，如图 6-4 所示：

由图可见，存在异常的时刻与正常时刻的定位数据图进行差分会在图中看到明显的“亮”区域，即变化幅值较大。我们再使用滑动窗的方法去检测高浮动区域，使用不同大小的正方形滑动窗对整个范围内进行搜索，自动标记出几个平均密度最高且超过阈值的区域，在图 6-5 中显示如下：

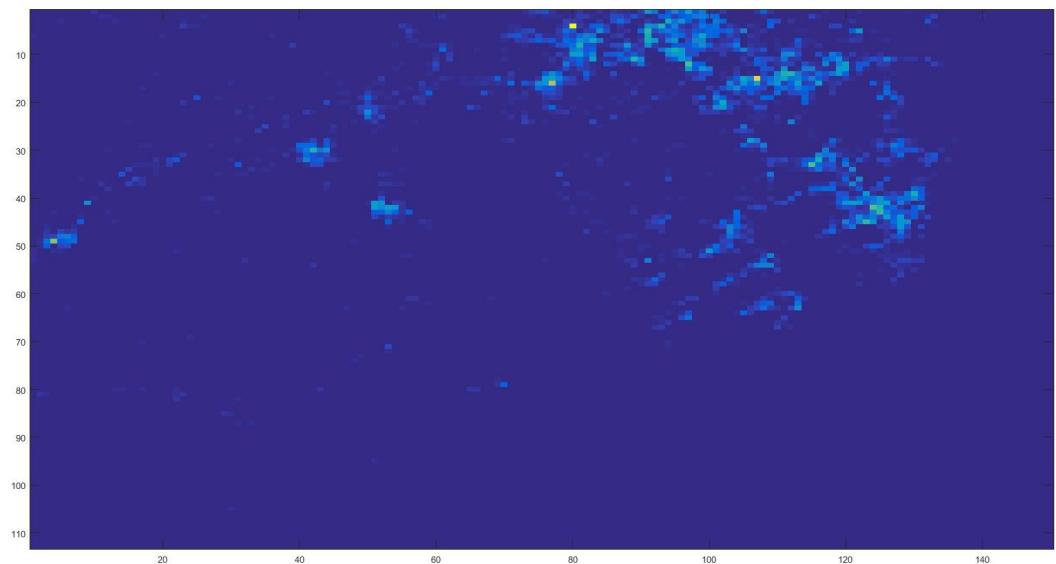


图 6-4 异常与正常定位数据间相邻帧差分图

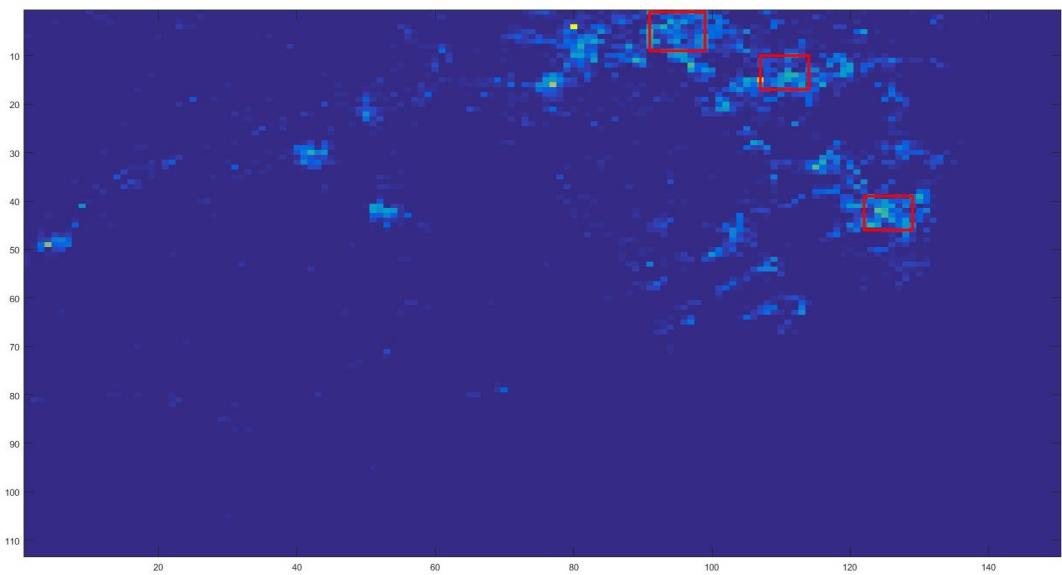


图 6-5 异常与正常定位数据间相邻帧差分图

图中所标记的异常浮动值主要集中在城市区域，由于9月30日是国庆节前一天，可能是游客的集中所造成的定位数据量增大。

## 第七章 总结与展望

### 7.1 内容总结

现代科学的发展，使得传感器网络越来越普及，让大数据分析成为可能。在这些数据之中，有一种数据尤为重要，其存在着很明显的时序性，描述着随着时间的推进中，某种事物的变化规律，这种数据常被称为时序数据。时序数据分析是对时间序列进行系统的分析并且简历合理的模型。其主要目的是考虑数据动态的波动情况并预报未来发生的事件。本文所讨论的定位数据就是基于某固定的空间坐标上的时序数据点，通过这种定位数据的时序分析，可以归纳总结出该地的人流变化特征。而对于存在明显规律的时序数据，我们可以对其进行建模并预测当前或未来时间节点上的值。但是，时序数据中经常会出现这样一些的观测点，它们的数据值较同性质时段上的数据存在着明显的偏差，我们称其为异常点。异常点的分析以及处理十分有意义：一方面异常点对于时间序列的建模来说是干扰很严重的噪音，较大的异常会使得拟合模型造成极大的偏差，需要将这些异常点检出并且删去使得模型的训练及预测更加精准；而另一方面，时序定位数据中的异常又反应出了该时间点上一定存在着某种外界干扰，比如定位数据急剧降低可能是由于台风过境人群大量躲在遮蔽物处而造成的。对于台风这种自然灾害，如果能根据时序定位数据的预测以及异常的判断，则能很好的做好预警，显得尤为重要。本文根据以上思路，基于腾讯的定位数据，对时序异常检测及相关内容完成了以下工作：

1. **数据分析：**对给定的腾讯定位数据进行处理后有了其基本的背景信息，以及为了后续的分析方便进行了预处理。给定的腾讯定位数据大致地理坐标为广东省珠海市沿海一带，而明确探查出的异常是某天的台风过境；由于地图上涵盖一大部分海面，而海面上的定位数据（定位终端数量）几乎为零，分析这些数据毫无意义，故通过最大值判断的方法进行了舍去；最后，通过数据的可视化，以及考虑到存在部分局部时刻异常的情况，我们得出了通过分析每天定位数据平均值来反应整天的定位数据信息，并统计分析图中所有地理坐标上的定位数据信息来确定某天是否存在异常的结论。
2. **基于曲线的异常检测分析：**我们将原本的整块区域异常天检测问题（本课题的异常为台风日）细化为各个有效定位数据点在数据范围日期内的一条曲线，并对整个定位数据点区域的每个点做曲线分析并统计结果；在曲线分析中，我们使用了基于统计、密度、差分、频域的几种曲线分析方法去检测异常并都成功将异常日期找出；最后，基于本课题的定位数据，我们分析了曲线异常检测的几种算法的特点并且讨论了本算法的局限性。
3. **基于曲线的时序数据预测：**由于前述曲线异常检测的算法对于检出局部时刻异常较为困难，我们采用预测并比较的方法去检测异常。神经网络能够经过数据的训

练来对网络结构中的参数进行调整，从而使网络近似趋近于时间序列的实际规律；我们对本课题所研究的腾讯时序定位数据中一部分进行了训练，使用该预测模型对国庆前的某一小时时刻的定位数据进行了预测及判断，预测与实际值有较大偏差，可以检出局部时刻的异常。

4. **基于图像的异常区域检测：**本章基于上述两章的检测结果，研究在检测区域中哪一块小区域出现了大浮动从而导致了异常；使用了图像相邻帧间差分法，观察到异常时刻与正常时刻的定位数据差分图较正常时刻之间的差分图颜色更深，并使用了不同大小的矩形窗计算差分定位数据图中平均异常变化幅度，将最大的几块区域圈出。

## 7.2 未来展望

本文的实验表明，在基于腾讯地图的时序定位数据上，我们能够成功检出已知的台风天异常并建立预测模型达到实时异常检测的目的，并且能够根据异常的时间点找出异常子区域。但对于实际定位数据的分析来说，本文在以下方面可以进行改善：

1. **高地理精度定位数据的异常检测：**本文所研究的课题所涉及的定位数据是大区域的粗精度数据，一个像素点约代表平方一千米以上的量级，算法对于分析大范围内的整体异常是有效的。但实际情况中有时研究者所接触到的定位数据是更高精度的，此时本文所讨论的对于整体区域的异常检测不再有效，高精度下应该更关注地理位置上的部分区域，应做更进一步探索。另外，本文对数据所做的简化处理也无法再适用于高精度的数据，无论是时空上或是地理上高精度的定位数据有更多信息可以挖掘，无法直接从统计角度进行整体分析，应当对数据进行更加细致的分析研究可实现的算法。
2. **大规模数据的曲线异常检测：**本文所实现的曲线异常检测算法中，部分算法对于本课题所研究的小规模数据量表现不佳，例如极大似然估计法这类统计方法，当数据量足够大时可能会把数据模型的参数拟合的较好，从而能更好地判断数据是否异常。同样，对于第5章所讨论的神经网络时序数据预测，数据量小时可能会导致网络模型拟合不佳，对于数据的拓展预测不利。
3. **异常值偏离正常值的门限：**本文所实现的曲线异常检测算法中，尤其是根据对已知的台风异常检测，设定了一个固定的门限来判断是否存在异常。实际情况中，定位数据会根据各种外在因素导致偏差，无法断定异常的偏差值设为多少才合适。应当具体问题具体分析，对当地的数据有一个大规模的统计，才能划定出最适合该地的异常值门限。

## 参考文献

- [1] 陈佳, 胡波, 左小清 et al. 利用手机定位数据的用户特征挖掘 [J]. 武汉大学学报: 信息科学版. 39 (6). 2014: 734–738.
- [2] Patcha Animesh, Park Jung-Min. An overview of anomaly detection techniques: Existing solutions and latest technological trends [J]. Computer networks. 51 (12). 2007: 3448–3470.
- [3] Chandola Varun, Banerjee Arindam, Kumar Vipin. Anomaly detection: A survey [J]. ACM computing surveys (CSUR). 41 (3). 2009: 15.
- [4] Dempster Arthur P, Laird Nan M, Rubin Donald B. Maximum likelihood from incomplete data via the EM algorithm [J]. Journal of the royal statistical society. Series B (methodological). 1977: 1–38.
- [5] Portnoff M. Time-frequency representation of digital signals and systems based on short-time Fourier analysis [J]. IEEE Transactions on Acoustics, Speech, and Signal Processing. 28 (1). 1980: 55–69.
- [6] 孙延奎. 小波分析及其工程应用 [M]. 北京: 机械工业出版社, 2009.
- [7] 韩力群, 康莘. 《人工神经网络理论, 设计及应用》——神经细胞, 神经网络和神经系统 [J]. 北京工商大学学报: 自然科学版. 23 (1). 2005: 52–52.
- [8] [S. l.]: 南京大学, 2013.
- [9] 张仁辉, 杜民. 小波分析在信号去噪中的应用 [J]. 计算机仿真. 22 (8). 2005: 69–72.
- [10] Breunig Markus M, Kriegel Hans-Peter, Ng Raymond T et al. LOF: identifying density-based local outliers [C]. In ACM sigmod record. 2000 : 93–104.
- [11] 杨营辉. 基于密度的样本裁剪算法的改进及在 kNN 中的应用研究. 2010.
- [12] 布罗克韦尔, 戴维斯, 铮. 时间序列的理论与方法 [M]. 高等教育出版社, 2001.
- [13] Keogh Eamonn. Data mining and machine learning in time series databases [J]. Tutorial in ICML. 2004.
- [14] [S. l.]: 哈尔滨: 哈尔滨工业大学, 2010.
- [15] Bebis George, Georgopoulos Michael. Feed-forward neural networks [J]. IEEE Potentials. 13 (4). 1994: 27–31.
- [16] Medsker LR, Jain LC. Recurrent neural networks [J]. Design and Applications. 5. 2001.
- [17] 韩路跃, 杜行检. 基于 MATLAB 的时间序列建模与预测 [J]. 计算机仿真. 22 (4). 2005: 105–107.
- [18] 李波, 姚春莲, 李炜 et al. 利用相邻帧和背景信息的运动对象检测 [J]. 电子学报. 36 (11). 2008: 2154–2159.

## 致 谢

本次毕业设计是在别老师的悉心指导下完成的。从课题的选择、设计到论文的撰写一直到最后一稿，期间遇到很多问题，都得到了别老师全力细心的指导。在此，向别老师表示衷心的感谢！

感谢信息与通信工程学院所有老师多年来对我的培养、帮助，使本人在本科学习中不仅学到了必备的专业知识技能和思考解决问题的方法，还学到了严谨治学的科研精神和积极进取的人生态度。

同时，感谢信息与通信工程学院的所有同学，给我创造了一个团结进取，充满温暖，充满爱的大集体，使我快乐而且充实地度过了人生中最美好的大学时光。

感谢提供此论文模板的开源项目 <https://github.com/sqyx008/BUPTBachelorThesis>

## 附录

### 附录 1 缩略语表

表 附-1 定位数据相关名词解释

特征	描述	形式与理论范围
定位数据值	确定地理坐标在某时刻检测到多少定位终端数量	数值, $\mathbb{N}$
时序定位数据值	确定地理坐标在离散时间区间内各时刻的定位数据值	数组, $\mathbb{N}$

### 附录 2 数学符号

#### 数和数组

$a$	标量 (整数或实数)
$a$	向量
$t_a$	$a$ 时刻
$x_a$	$a$ 时刻的定位数据值
$d_a$	$a$ 时刻的定位数据值与相邻时刻值间差分

## 外 文 译 文

### Yahoo 大规模时列数据异常检测技术及其高性能可伸缩架构

Nikolay Laptev, Saeed Amizadeh, Ian Flint

Yahoo Labs

## 摘要

本文介绍了一种用于大规模时序数据自动异常检测的通用和可扩展框架。及早发现异常情况对维护个人数据的一致性和保护公司免受恶意攻击者起着关键作用。目前最先进的异常检测方法受可扩展性，用例限制，使用困难和大量误报的影响。我们在雅虎(EGADS)的系统设计了基于异常检测和预测模型构建的异常过滤层，以实现时序精确和可扩展的异常检测。我们将我们的方法与具有不同时间序列特征的真实和合成数据的其他异常检测系统进行比较。我们发现，我们的框架可以使各种用例的精度和召回率提高 50-60%，并且数据和框架都是开源的。

## 第一章 概述

虽然计算硬件和软件方面的快速发展带来了强大可靠的应用程序，但仍然有数百个软件错误和硬件故障在大型集群中继续发生，从而影响用户体验并随后恢复。不间断系统具有严格的正常运行时间要求，对这些系统的持续监控至关重要。从数据分析的角度来看，这意味着不停地监测大量的时间序列数据，以便检测潜在的故障或异常情况。由于问题的规模很大，人们对这些数据的监测实际上是不可行的，这导致我们使用机器学习和数据挖掘技术进行自动异常检测。

异常或异常值是与其余数据显着不同的数据点。一般来说，大多数应用程序中的数据是由一个或多个反映系统功能的生成过程创建的。当潜在的生成过程以不正常的方式运行时，会产生异常值。对这些异常值进行快速有效的识别对于许多应用非常有用，这些应用包括：侵入检测，信用卡欺诈，传感器事件，医疗诊断，执法等。自动化异常检测中的现有方法源于大量误报，这些误报在实践中禁止了这些系统的实用性。用例或类别特定的异常检测模型，可能对特定应用程序的误报率较低，但当时间序列的特征发生变化时，这些技术在没有适当的再训练的情况下表现不佳。第 6.3 节在实践中证明了“一刀切”原则的缺陷。我们在雅虎的系统被称为 EGADS (可扩展通用异常检测系统)，它可以准确，可扩展地检测时间序列异常。EGADS 将预测，异常检测和警报分为三个独

立的组件，允许用户将自己的模型添加到任何组件中。EGADS 使用一组经过调整的默认模型来减少误报数量，这对于普通用户来说本身就足够了。然而，更高级的用例需要系统捕获某些类型的异常，而忽略其他异常。值得引起注意的异常可能在幅度，严重程度或其他未知的先验参数上有所不同，并取决于用例。为此，EGADS 的警报组件使用机器学习为消费者选择最相关的异常情况。EGADS 是首个灵活、准确且可扩展的异常检测综合系统。EGADS 框架与异常检测基准数据已经一同开源，旨在帮助学者和行业合作开发新的异常检测模型。在雅虎，EGADS 每天被许多团队用于数百万次的时间序列异常检测工作。在第 2 部分中，我们描述了 EGADS 体系结构。算法和警报模块分别在第 3 节和第 4 节中介绍。以前的工作将在第 5 节中介绍。实验将在第 6 节中讨论，然后分别讨论第 7 节和第 8 节中的实际用例和结论。

## 第二章 整体架构

GADS 框架由三个主要部分组成：时间序列建模模块（TMM）、异常检测模块（ADM）和报警模块（AM）。给定一个时间序列，TMM 组件模拟产生时间序列，由 ADM 和 AM 组件进行消费处理，分别计算误差并过滤不感兴趣的异常。这些组件在第 3 和 4 节中有详细描述。

EGADS 被构建为一个框架，可以轻松地集成到现有的监控基础设施中。在雅虎，我们内部的雅虎监控服务（YMS）每秒处理数百万个数据点。因此，对 YMS 进行可扩展、精准且自动化的异常检测至关重要。接下来我们将详细描述 YMS 的具体细节。

### 2.1 系统集成

EGADS 作为独立平台运行，可用作大型系统中的库。因此，设计 EGADS 和雅虎监控服务（YMS）之间的接口至关重要。EGADS 与 YMS 的集成架构图如图附-1 所示。

此外，异常检测还需要几个支撑组件来驱动完成。首先，所有的异常检测模型都是离线批处理（batch）产生的，然后应用到实时环境（real time）。其中批处理由三个步骤组成：

1. 监测数据（即监控的时间序列数据）批量存储在 Hadoop 集群上
2. 批量模型生成器针对这些数据运行，并为目标时间序列构建模型
3. 模型存储在模型数据库中

然后在线实时流使用这些存储的模型，具体步骤如下：

1. 数据流入 Storm 进行流式处理
2. 集群中的一个模块调用 EGADS ADM，根据存储在模型数据库中的模型来评估输入数据点

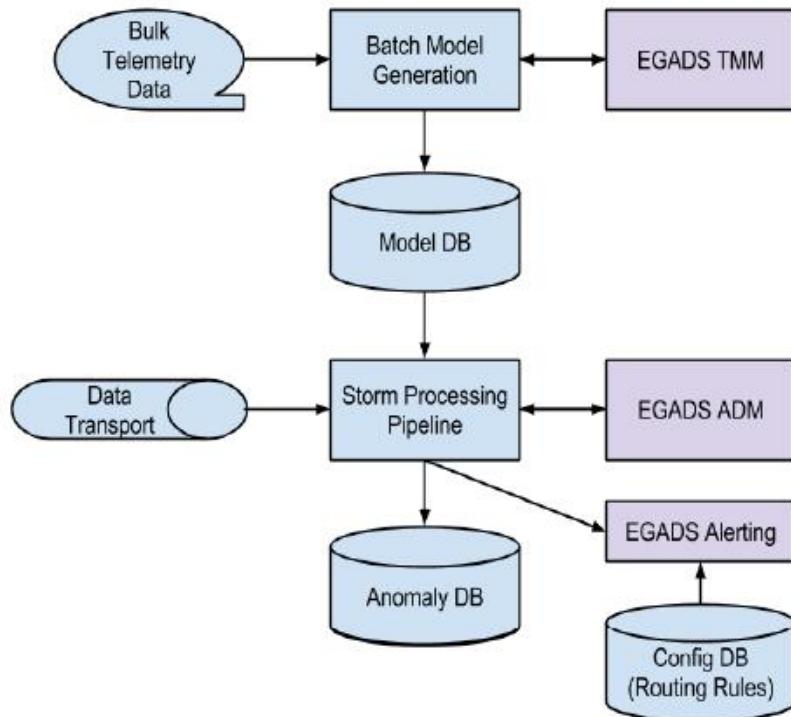


图 附-1 EGADS-YMS 整体架构

3. 如果存在异常，则将其发送到由组合规则和其他包含特定逻辑组成的辅助规则流（见第4节）
4. 根据规则，如果异常是警报事件，则生成事件，存储在状态数据库中，并转发到警报路由系统
5. 警报路由系统应用路由配置规则将警报发送给相应的处理人员

## 2.2 可拓展性

EGADS 的监控需要每秒分析超过百万级数据点和亿级别时间序列。要求在 CPU 负载、I/O 和内存占用方面具有可扩展性，并且数据点的处理需要尽可能高效。这意味着需要预先计算尽可能多的模型。从磁盘读取模型是不切实际的，会降低性能，因此模型应该存储在内存中。另一方面，为了控制成本，模型应尽可能小。

## 第三章 异常检测算法

在本节中，我们给出了 EGADS 支持的异常检测算法。目前，EGADS 能够检测出三类异常：

- **异常值：**给定输入时间序列  $x$ ，异常值为时间戳值对  $t, x_t$ ，其中观察值  $x_t$  与此时的时间序列的期望值显着不同，即  $E(x_t)$ 。

- **波动点**: 在给定输入时间序列  $\mathbf{x}$  的情况下, 变化点是在时间戳  $t$  上, 使得时间序列的行为在  $t$  之前和之后显着不同。
- **异常时间序列**: 给定一组时间序列  $\mathbf{X} = \{\mathbf{x}^i\}$ , 异常时间序列  $\mathbf{x}^j \in \mathbf{X}$  是在  $\mathbf{X}$  上与大多数时间序列不一致的部分。

在以下章节中, 我们给出了 EGADS 当前用于检测上述异常类型的方法。

## 3.1 异常检测

检测异常值是许多监控应用中最重要的功能。EGADS 提供了两类用于检测输出的算法, 本节对这两类算法进行了描述。

### 3.1.1 插件方法

EGADS 中异常值检测的第一类方法称为插件方法。为了模拟输入时间序列的正常行为, 可以业务和时序数据的特点来插入大量的时间序列模型和预测模型 (例如 ARIMA、指数平滑、Kalman 滤波、状态空间模型等)。这就是为什么我们将这个总体策略称为插件方法。应该注意的是, 所有这些模型都在 EGADS 中用于时间序列预测, 这是我们框架的另一个特征; 然而, 由于本文的重点是异常检测, 更多关于 EGADS 的建模和预测特征的细节我们不进行更深入的讨论。

我们提出的 Plug-in 框架包含两个主要组件: 时间序列建模模块 (TMM) 和异常检测模块 (ADM)。给定时间序列  $\mathbf{X} = \{\mathbf{x}_t \in \mathbf{R} : \forall t \geq 0\}$ , TMM 提供在时间  $t$  的  $\mathbf{x}_t$  的预测值, 用  $\mathbf{u}_t$  表示。我们也把这个数量称为  $\mathbf{x}_t$  的期望值 (不要与期望的数学概念混淆)。TMM 可以是预测的机器学习模型, 基于一些训练数据或基于规则的系统, 在时间  $t$  上挖掘数据点  $\mathbf{x}_t$  的具体表现特征 (如波动或异常)。在本文中, 我们不对 TMM 做任何假设; 也就是说, TMM 只是我们提出的生成预测的方法中的一个黑盒模块。从这个意义上说, 我们提出的框架是通用的, 并不依赖于任何特定的时间序列建模框架。

给定预测值  $\mathbf{u}_t$  和实际观测值  $\mathbf{x}_t$ , ADM 计算出一些我们称之为偏差度量 (DM) 的偏差概念。最简单的偏差量度是预测误差, 即

$$\mathbf{PE}_t = \mathbf{x}_t - \mathbf{u}_t$$

如果错误超出某些固定阈值, 则会发出警报。这种简单的方法在某些情况下可能会起作用, 但是对于大多数的方法来说, 它不会是一个很好的策略, 因为它不能捕获到错误的具体信息。相对误差  $\mathbf{RE}_t$  被定义为  $\mathbf{u}_t$  的一个因素:

$$\mathbf{RE}_t = \frac{\mathbf{x}_t - \mathbf{u}_t}{\mathbf{u}_t} = \frac{\mathbf{x}_t}{\mathbf{u}_t}$$

通过对相对误差进行阈值处理，可以检测异常值，同时对所期望值的幅度进行归一化。虽然这些阈值确定了异常检测模块的敏感度，然而，很难确定异常检测的最佳度量。事实上，给定时间序列的最优度量的选择取决于时间序列的性质以及 TMM 性能。例如，如果我们处理一个非常规则的时间序列，我们有一个准确的模型，使用预测误差进行异常检测可能就足够了，因为它预期是正态分布的。在其他情况下，最佳度量可能在预测误差和相对误差之间存在某种差异。因此，EGADS 默认跟踪一组偏差度量，使用系统的人可以创建自己的错误度量。在第 4 节中描述的警报模块（AM）中使用这些错误度量以及其他功能（如时间序列特征）来了解用户的偏好并过滤不重要的异常。

### 3.1.2 基于分解的方法

EGADS 中第二类异常值检测方法是基于时间序列分解的思想，在时间序列分析中，通常将时间序列分解为：趋势、季节性和噪声三个要素。通过监测噪声分量，可以捕获异常值。更准确地说，如果点  $X_t$  的噪声分量的绝对值大于某个阈值，则可以认为  $X_t$  为异常值。

## 3.2 变点检测

在一些文献里有提到一种基于时间窗口的变点检测技术，在 EGADS 中，目前采用基于模型的方法。在这些方法中，时间序列的预期行为通过 3.1.1 节中提到的建模技术建模。我们结合 3.1.1 节中描述的插件方法来计算输入时间序列的残差序列（或模型预测的偏差）。然后我们对残差序列应用绝对变化点检测的方法来检测残差分布。我们使用内核密度估计（Kernel Density Estimation）、非参数估计残差分布和 Kullback-Leibler（KL 距离，常用来衡量两个概率分布的距离）来测量分布变化。

## 3.3 检测异常时间序列

EGADS 支持的另一类异常检测技术涉及检测异常时间序列。异常时间序列  $t$  被定义为与其他时间序列的平均偏差显着的时间序列。假设所有时间序列是均匀的并且来自相同的来源（即，是同一个群集的一部分），可以简单地计算相对于其他时间序列的时间序列 ( $i$ ) 的平均偏差。在 EGADS 中，我们目前的方法是根据各种时间序列特征（包括趋势和季节性，谱熵，自相关，平均欧几里德距离等）将时间序列聚类为一组聚类  $C$ 。在聚类后，我们执行簇内或簇间通过测量聚类质心和时间序列 ( $i$ ) 内部或之间的偏差来进行时间序列异常检测。这种 EGADS 异常检测类型的常见用例涉及分类。例如，如果网络工程师想要在数百万个时间序列中找到异常服务器，对于以前的方法来说可能是不切实际的，因为建模是按照时间序列来完成的，而不考虑其他度量的行为。这种异常检

测类型的另一个应用是发现类似的异常，这与以前的用例相反。

## 第四章 警报机制

异常检测的最终目标是产生准确和及时的警报。EGADS 通过两阶段过程实现这一点，首先通过阈值选择产生一组候选异常，然后根据给定的规则过滤不相关的异常。

### 4.1 阈值选择

阈值选择的作用是根据异常检测模块（ADM）产生的偏差度量选择合适的阈值。目前，EGADS 基于以下两种阈值选择算法实现：

- $K\sigma$  偏差
- 密度分布

第一种方法是参数化的，并假定数据正态分布，有明确的平均值和标准偏差。依靠高斯分布，并基于“3sigma 规则”（即：其中 99.73% 的样本位于平均值的三个标准偏差之内）。因此，根据  $K\sigma$  中的 K 值，可以确定在时间 t 观测样品的可能性。根据所需的敏感度，可以测量给定的样品是否在  $K = 2$  或 1 的所有样品的 95.45% 或 68.27% 之内。请注意，这里的假设是以我们的偏差度量是正态分布为前提的。第二种方法是非参数的，对于偏差度量不是正态分布的情况是有用的。基本思想是找到偏差度量分布的低密度区域。一种方法是使用诸如局部离群因子（LOF）的算法。通过将对象的局部密度与其邻居的局部密度进行比较，可以识别具有相似密度的区域，以及具有比邻居密度明显更低的密度的点，这些点被认为是异常值。

表 附-2 EGADS 所使用的时序数据特征表

时间序列功能	描述
周期（频率）	周期性对于确定季节性非常重要。
趋势	如果平均水平有长期变化，则其存在
季节性	时间序列受季节因素影响时存在，例如一年的月份或一周中的某一天
自相关	代表长期依赖。
非线性	非线性时间序列包含通常不由线性模型表示的复杂动态。
偏态	衡量对称性，或者更确切地说，缺乏对称性
峰度	相对于正态分布来衡量数据是否达到峰值或持平。
Hurst	衡量时间序列的长期记忆。
Lyapunov 指数	衡量附近轨迹发散率的指标。

## 4.2 过滤

过滤对异常进行最后阶段的后期处理，然后交付给消费者。虽然作为过滤阶段输入的候选异常具有统计意义，但并不是所有这些异常都会与特定用例相关。例如，一些消费者对时间序列中的峰值感兴趣，而其他消费者对下点感兴趣，而其他消费者则对变化点感兴趣。EGADS 提供了一个简单直观的界面，允许用户标记异常时间序列的区域。EGADS 将此反馈与时间序列和模型特征一起用于训练预测异常  $ai$  是否与用户  $uj$  相关的分类器。EGADS 跟踪的时间序列特征见表??。在第 6.2 节中，我们将看看这些时间序列特征如何影响模型性能。

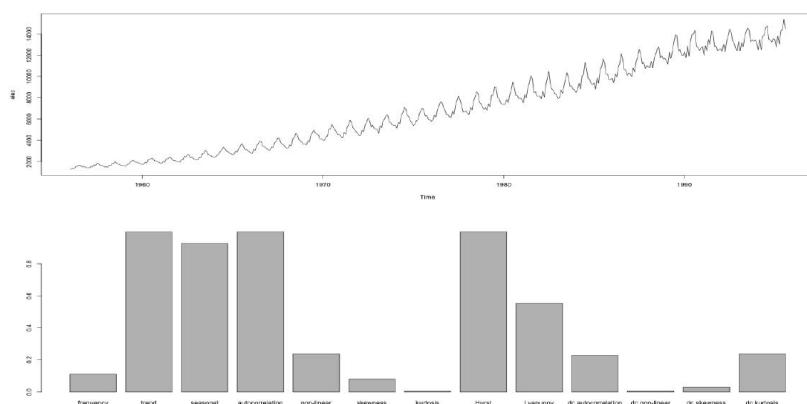


图 附-2 EGADS 提取的时间序列及其特征的一个示例

# 第六章 实验研究

## 6.1 数据

用于实验的数据集由 1:1 的合成数据和真实数据混合而成。我们创建了一个合成的时间序列生成工具，该工具已经随着框架和基准数据开源。使用该工具，通过指定长度、幅度、异常数、异常类型、异常大小、噪声水平、趋势和季节性来生成合成时间序列。真实数据集使用了雅虎会员登录数据 (YML)。合成和真实数据在时间序列都包含 3000 个数据点，YML 数据包含了 3 个月的数据点。除非另有说明，所有实验均以 1000 次随机选取的时间序列进行，结果取平均值。还要注意，合成和真实数据都有异常标签，方便测量精度和召回率。

## 6.2 建模实验

时间序列建模（由 EGADS 中的 TMM 组件捕获）是异常检测的基本部分。通常情况下，异常检测与底层时间序列模型一样好。由于大量的候选模型，模型选择变得至关重要，并且取决于时间序列特征和可用资源。在下面的实验中，我们展示了时间

序列特征对模型性能的影响，并显示了精度，实验中使用的模型和误差度量分别参见表附-3 和附-5。

表 附-3 用于建模实验的模型列表

模型	描述
Olympic Model	朴素的季节模型，其中下一点的预测是前 n 个周期的平滑平均值。
指数平滑模型	一种流行的模型，用于生成平滑的时间序列。
移动平滑模型	在这种模式下，预测基于一个人为地构建的时间序列。
回归模型	使用一个或多个变量来拟合 x 和 y 之间的关系进行建模。
ARIMA	自回归整体移动平均线。
(T)BATS 系列	(三角函数) 指数平滑状态空间模型与 Box-Cox 变换。
Kalman 谱滤波器	一个内部算法用于学习卡尔曼滤波器参数的快速多变量谱。

表 附-4 用于建模实验的指标列表

指标	描述
Bias	误差的算术平均值
MAD	平均绝对偏差
MAPE	平均绝对百分比误差
MSE	误差的均方
SAE	绝对误差的总和
ME	平均误差
MASE	平均绝对比例误差
MPE	平均百分比误差

## 6.2.1 时间序列特征和性能

为了证明时间序列特征对模型性能的影响，我们比较了不同模型的时间序列与不同特征的误差度量（见 4.2 节）。图附-3 显示了时间序列特征在模型行为中起着重要作用。例如，季节性模式的 Olympic 模型在数据集上表现不佳，没有季节性和强劲趋势。EGADS 能够跟踪历史时间序列特征和模型性能，使用这些历史信息，EGADS 选择由表 3 中描述的误差度量判断的最佳模型（给定时间序列特征）。实际上，基于数据特征进行模型选择比针对每个模型执行交叉验证要快得多。

## 6.2.2 时序模型的可扩展性

如第 2 节所述，时序模型需要高性能技术以支持大规模（例如每秒数百万点）数据流的实时计算，因此需要在模型大小、训练时间和准确性之间进行权衡。这种权衡在

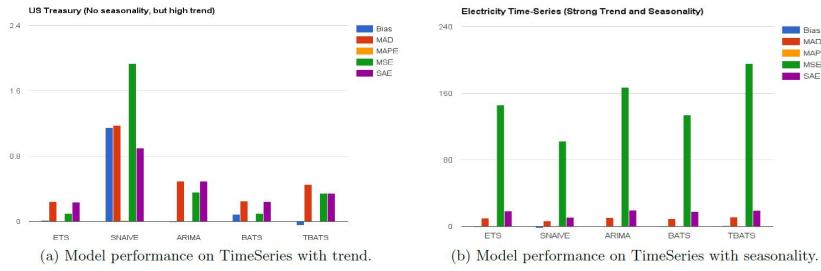


图 附-3 具有不同特性的时间序列性能

图 附-4 (a) 和 (b) 中可以看出。例如，从图中可以看出，季节性天性模型训练很快，但是具有较大的存储要求和较高的平均误差。在雅虎，首先设定了资源和模型训练时间的目标，然后选择相应的模型。换句话说，目标是在资源和模型建立时间限制下使得表 3 中的错误减到最小。



图 附-4 模型大小与训练时间的权衡

### 6.3 异常检测实验

在本节中，我们比较了开源系统与 EGADS，参考的开源系统如表??所示

表 附-5 建模实验的指标列表

模型	描述
EGADS Extremelow 密度模型异常值	EGADS 基于密度的异常检测
EGADS CP	基于 EGADS 内核的波动点检测
EGADS KSigmaModel 异常值	EGADS 重新实现了经典的 $K\sigma$ 模型
Twitter Outlier	基于广义 ESD 方法的开源 Twitter-R 异常检测库
Extreme&II R 异常值	开源单变异常值检测，阈值绝对值和残差检测异常
Breakout Twitter CP	来自 Twitter 基于 ESD 统计测试来检测变化点的一个库
ChangePt1 R CP	一个 R 库，实现各种主流和专门的变化点方法
ChangePt2&3 R CP	检测平均值和变量的变化

在图 附-5 显示了对 6.1 节中描述的数据的测试结果：

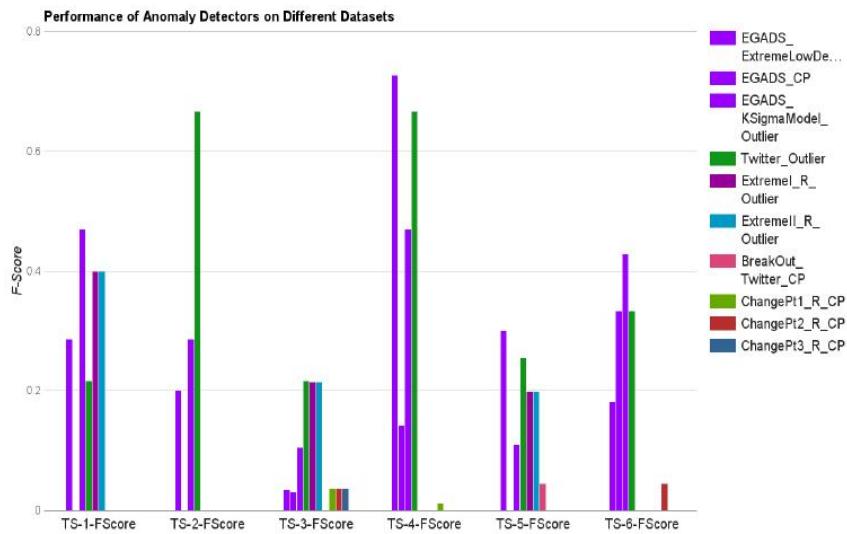


图 附-5 异常模型在不同数据集上的表现

结果可以看出，没有一个最佳的异常检测模型能适合所有业务场景，不同的算法需要结合检测不同类型异常来确定。例如，Twitter 在数据集上表现最好，而 ExtremeLow-Density 模型在上表现最好。但是，EGADS 的设计初衷是在用户对数据类型的时间序列和异常类型不了解的情况下，该系统能够优雅和稳健地处理数据中存在的各种异常现象。因此，EGADS 被构建为将一组异常检测模型组合成一个最佳框架的库。这些模型的异常被转发到过滤组件以进行精确的异常检测。

## 6.4 异常过滤实验

异常的重要性往往取决于实际应用场景。具体来说，一些用户可能对展示恶意攻击的时间序列行为感兴趣，而其他用户可能对收入有兴趣。

为了解决这个要求，过滤阶段扫描所有模型的所有异常，并使用分类作为真正的模型。在 YM 用例的过滤阶段使用的模型是基于 AdaBoost 的增强树模型。模型中使用的特征参见表 附-2。AdaBoost 的核心原理是在变化的数据上进行适合一系列 weak learners（例如，小决策树），最后的结果是通过组合加权多数表决产生的。图 6 中的实验结果表明，在滤波阶段使用不同的模型特征，得到比较好的精确率和召回率。

# 第七章 Yahoo 实例应用

## 7.1 系统度量

系统指标广泛地定义为服务系统中硬件组件的健康状况的度量。例子包括 CPU 利用率，可用磁盘空间，网络接口流量统计和内存利用率。其中一些度量标准（如 CPU 利用率）跟踪组件的总体流量，一些（如可用磁盘空间）与当前流量级别无关。系统指标与其他类型的指标不同之处在于，由于雅虎服务系统的冗余特性，阈值违反通常是服务

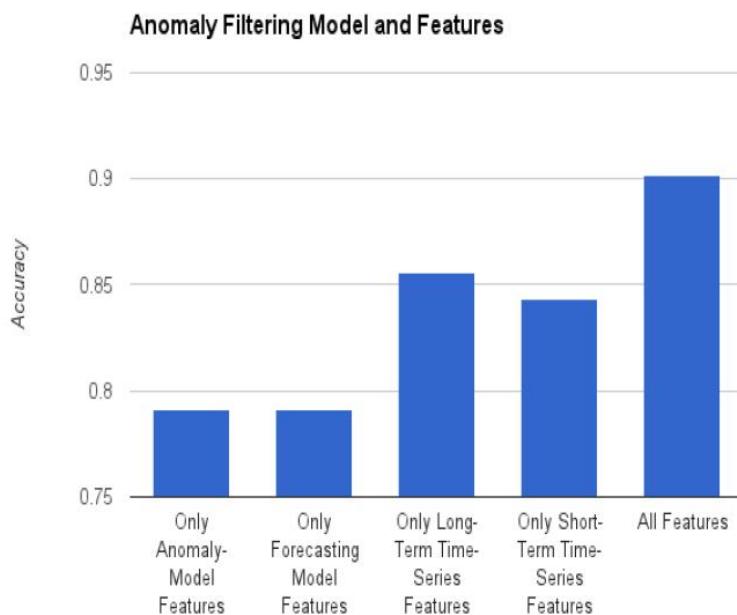


图 附-6 使用不同类型特征的过滤阶段的精度

问题的主要指标。因此，针对这些类型的指标的警报通常不会被视为中断，而是用于触发长期补救措施，如添加容量或清除磁盘空间。

## 7.2 业务绩效

业务 KPI 是直接反映客户与雅虎网站体验的指标。示例包括页面浏览量，服务延迟时间，服务错误，点击率和收入等。业务 KPI 几乎总是网站问题的尾随指标，定义反映了对雅虎服务能力的影响。因此，业务关键绩效指标的异常情况通常都会受到高度的关注。大规模业务关键绩效指标的性质是它们往往具有高度的可预测性，所以它们很适合自动化异常检测。雅虎在整合业务 KPI 方面取得了巨大成功，可通过自动化异常检测发现服务和收入问题。

## 7.3 相似度量组

雅虎的大部分基础架构都遵循横向扩展模型，每个服务层都有数十台到数百台服务器。在对可疑事件的原因进行分类和隔离时，可能很难调查数千台机器的基础设施以发现故障。自动异常检测可用于对基础设施各组成部分的相对异常情况或“兴趣度”进行排序，这些特征可按降序排列，以便操作员能够快速查看模式并隔离问题。

# 第八章 总结

异常检测是许多具有故障应用的实时监控系统的核心部分，比如检测、欺诈检测、网络入侵检测等等。尽管它至关重要，但实际上实施全自动异常检测系统是一项极具挑

战性的任务，这些挑战通常导致解决方案不可扩展或并非高度专业化的，也导致了较高的误报率。

在本文中，我们介绍了 EGADS——雅虎实现的通用异常检测系统，对不同的 Yahoo 属性和数百万个时间序列进行自动监控和警报。正如我们在本文中所描述的，Hadoop 上的 EGADS 并行架构以及 Storm 的流处理机制使得它能够在雅虎的数百万个时序数据集上执行实时异常检测。此外，EGADS 采用不同的时间序列建模和异常检测算法来处理不同的监控场景。通过将这一组算法与机器学习机制结合到警报模块中，EGADS 能自动适应对用户重要的异常检测用例。所有这些功能都有效地创建了一个功能强大的异常检测框架，它是通用且可扩展的。我们对真实数据集和综合数据集的展示实验表明，与其竞争对手的解决方案相比，我们的框架具有优越的适用性。

EGADS 的设计本质上是可扩展的，它为向系统插入新的模型和算法提供了一种简单的机制。我们的框架及其所有数据集都已经开源。

# 外文原文

## Generic and Scalable Framework for Automated Time-series Anomaly Detection

Nikolay Laptev  
Yahoo Labs  
Sunnyvale, CA, USA  
nlaptev@yahoo-inc.com

Saeed Amizadeh  
Yahoo Labs  
Sunnyvale, CA, USA  
amizadeh@yahoo-inc.com

Ian Flint  
Yahoo  
Sunnyvale, CA, USA  
ianflint@yahoo-inc.com

### ABSTRACT

This paper introduces a generic and scalable framework for automated anomaly detection on large scale time-series data. Early detection of anomalies plays a key role in maintaining consistency of person's data and protects corporations against malicious attackers. Current state of the art anomaly detection approaches suffer from scalability, use-case restrictions, difficulty of use and a large number of false positives. Our system at Yahoo, EGADS, uses a collection of anomaly detection and forecasting models with an anomaly filtering layer for accurate and scalable anomaly detection on time-series. We compare our approach against other anomaly detection systems on real and synthetic data with varying time-series characteristics. We found that our framework allows for 50-60% improvement in precision and recall for a variety of use-cases. Both the data and the framework are being open-sourced. The open-sourcing of the data, in particular, represents the first of its kind effort to establish the standard benchmark for anomaly detection.

### 1. INTRODUCTION

While rapid advances in computing hardware and software have led to powerful applications, still hundreds of software bugs and hardware failures continue to happen in a large cluster compromising user experience and subsequently revenue. Non-stop systems have a strict uptime requirement and continuous monitoring of these systems is critical. From the data analysis point of view, this means non-stop monitoring of large volume of time-series data in order to detect potential faults or anomalies. Due to the large scale of the problem, human monitoring of this data is practically infeasible which leads us to automated anomaly detection using Machine Learning and Data Mining techniques.

An anomaly, or an outlier, is a data point which is significantly different from the rest of the data. Generally, the data in most applications is created by one or more generating processes that reflect the functionality of a system. When the underlying generating process behaves in an un-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [Permissions@acm.org](mailto:Permissions@acm.org).

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.  
© 2015 ACM. ISBN 978-1-4503-3664-2/15/08 ...\$15.00.  
DOI: <http://dx.doi.org/10.1145/2783258.2788611>.

usual way, it creates outliers. Fast and efficient identification of these outliers is useful for many applications including: intrusion detection, credit card fraud, sensor events, medical diagnoses, law enforcement and others [1].

Current approaches in automated anomaly detection suffer from a large number of false positives which prohibit the usefulness of these systems in practice. Use-case, or category specific, anomaly detection models [4] may enjoy a low false positive rate for a specific application, but when the characteristics of the time-series change, these techniques perform poorly without proper retraining. Section 6.3 demonstrates the shortcoming of 'one size fits all' principle in practice.

Our system at Yahoo is called EGADS (Extensible Generic Anomaly Detection System) and it enables the accurate and scalable detection of time-series anomalies. EGADS separates forecasting, anomaly detection and alerting into three separate components which allows the person to add her own models into any of the components. Note that this paper focuses on the latter two components.

EGADS uses a set of default models that are tuned to reduce the number of false positives, which by itself suffices for the average user. More advanced use-cases, however, will require the system to capture some types of anomalies while ignoring others. The anomalies of interest may vary in magnitude, severity or other parameters which are unknown apriori and depend on the use-case. For this reason the alerting component of EGADS uses machine learning to select the most relevant anomalies for the consumer.

To the best of our knowledge EGADS is the first comprehensive system for anomaly detection that is flexible, accurate, scalable and extensible. EGADS is being open-sourced [19] along with the anomaly detection benchmarking data [18]. The open-sourcing of the data and the system will provide the first of its kind benchmarking data and the framework to help the academics and the industry collaborate and develop novel anomaly detection models. At Yahoo, EGADS is used on millions of time-series by many teams daily.

In Section 2 we describe the EGADS architecture. The algorithms and the alerting module are described in Sections 3 and 4 respectively. Previous work is described in Section 5. Experiments are discussed in Section 6 followed by the real-world use-cases and conclusion in Sections 7 and 8 respectively.

### 2. ARCHITECTURE

The EGADS framework consists of three main components: the time-series modeling module (TMM), the anomaly detection module (ADM) and the alerting module (AM).

Given a time-series the TMM component models the time-series producing an expected value later consumed by the ADM and AM components that, respectively, compute the error and filter uninteresting anomalies. These components are described in detail in Sections 3 and 4.

EGADS was built as a framework to be easily integrated into an existing monitoring infrastructure. At Yahoo, our internal Yahoo Monitoring Service (YMS) processes millions of data-points every second. Therefore, having a scalable, accurate and automated anomaly detection for YMS is critical. We describe the integration details into YMS next.

## 2.1 System Integration

EGADS operates as a stand-alone platform that can be used as a library in larger systems. Therefore, designing an interface between EGADS and an internal Yahoo monitoring service (YMS) is critical. A key constraint of YMS is scale; the platform needs to evaluate millions of data points per second. As a result, many of the integration architecture decisions are focused on optimizing real-time processing. The integration with YMS is shown in Figure 1.

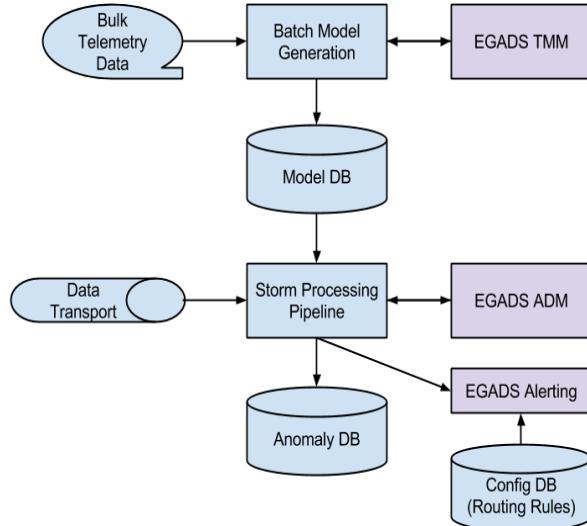


Figure 1: EGADS-YMS Architecture

Several support components are required to drive action based on detected anomalies. First of all, all anomaly detection models are generated in batch and applied in real time. The batch flow is comprised of three steps:

1. Telemetry (i.e. the monitored time-series) data are stored in bulk on a Hadoop cluster.
2. A batch model generator runs against these data and builds models for targeted time-series.
3. The models are stored in a model database.

The online flow then utilizes the stored models.

1. Data flows into a Storm [25] stream-processing topology.
2. One of the bolts (modules) in the topology calls the EGADS ADM to evaluate incoming data points based on models stored in the model database.

3. If an anomaly is present, this is fed to a secondary rule flow, consisting of combinatorial rules and other use-case specific logic (see Section 4).
4. Based on the rules, if the anomaly is an alert event, the event is generated, stored in a status database, and forwarded to an alert routing system.
5. The alert routing system applies routing configuration rules to send the alert to the appropriate support staff.

## 2.2 Scalability

The monitoring use case for EGADS requires the evaluation of millions of data-points per second, across over one hundred million time-series. This has scalability implications in terms of CPU load, I/O, and memory footprint. The evaluation of a datapoint needs to be as efficient as possible. This means that as much of the model as possible should be precomputed. It is not practical to read a model from disk each time a datapoint arrives because of the rate of inbound traffic. This suggests that the models should be stored in memory. In order to contain costs, the models should be as small as possible.

One optimization is to share models across multiple similar time-series. This is practical in the context of a large web serving environment, since applications are broken into horizontal tiers of similar servers. This optimization will reduce the memory footprint, the batch workload, and I/O against the model database.

Another possible optimization is to investigate self-tuning models; models that update themselves based on a stream of inbound data via online learning rather than requiring periodic batch generation. Models of this type may need to be initialized in batch, but overall they will reduce the batch workload. Depending on implementation, however, they may increase writes against the model database since they are being constantly refined.

Yet another optimization involves a trade-off between model size, training speed and accuracy. Depending on the characteristics of the time-series a light and fast forecasting model can provide similar accuracy as a more sophisticated one. We evaluate some of these optimization approaches in Section 6.2.2.

## 3. ANOMALY DETECTION ALGORITHMS

In this section, we give a big picture overview of the anomaly detection algorithms supported by EGADS. Currently, EGADS is capable of detecting three classes of anomalies:

- (a) **Outliers:** given an input time-series  $x$ , an outlier is a timestamp-value pair  $\langle t, x_t \rangle$  where the observed value  $x_t$  is *significantly* different from the expected value of the time-series at that time, i.e.  $\mathbb{E}(x_t)$ .
- (b) **Change points:** given an input time-series  $x$ , a change point is a timestamp  $t$  such that the behavior of the time-series is *significantly* different before and after  $t$ .
- (c) **Anomalous time-series:** given a set of time-series  $X = \{x^{(i)}\}$ , an anomalous time-series  $x^{(j)} \in X$  is a time-series whose behavior is *significantly* different from the majority of the time-series in  $X$ .

In the following sections, we give the general sketch of the methods that are currently used in EGADS for detecting the aforementioned anomaly types.

### 3.1 Outlier Detection

Detecting outliers is the most important functionality in many monitoring applications. For this reason the main focus of this paper is on outlier detection and unless it is explicitly specified, by anomalies, we refer to outliers by default.

EGADS offers two classes of algorithms for detecting outliers, which are described in this section.

#### 3.1.1 Plug-in methods

The first class of methods for time-series outlier detection in EGADS are called *plug-in* methods. These methods *explicitly* model the normal behavior of the time-series such that a significant deviation from this model is considered an outlier. To model the normal behavior of the input time-series we can plug-in a wide range of time-series modeling and forecasting models (e.g. ARIMA [30], Exponential Smoothing [11], Kalman Filter [9], State Space Models [6], etc.) depending on the application and the nature of time-series. That is why we refer to this general strategy as the plug-in methods. It should be noted that all these models are used in EGADS for time-series forecasting which is another feature of our framework; however, since the focus of this paper is on anomaly detection, we do not give more details on modeling and forecasting features of EGADS.

Our proposed Plug-in framework consists of two main components: the time-series modeling module (TMM) and the anomaly detection module (ADM). Given a time-series  $X = \{x_t \in \mathbb{R} : \forall t \geq 0\}$ , the TMM provides the predicted value of  $x_t$  at time  $t$ , denoted by  $u_t$ . We also refer to this quantity as the expected value of  $x_t$  (not to be confused with the mathematical notion of expectation). The TMM can be a machine learned model which makes predictions based on some training data or a rule-based system which encodes expert's knowledge about how  $x_t$  behaves at time  $t$ . In this paper, we *do not* make any assumption regarding the TMM; that is, the TMM is just a black box module in our proposed method that generates predictions  $u_t$ . In this sense, our proposed framework is *generic* and does not depend on any specific time-series modeling framework.

Given the predicted value  $u_t$  and the actual observed value  $x_t$ , the ADM computes some notion of deviation which we refer to as the deviation metric (DM). The simplest measure of deviation is the prediction error,  $PE_t = x_t - u_t$ . If the error falls outside some fixed thresholds, an alert is issued. This simple method may work in some cases, but it will not be a good strategy for most because it does not capture the relative error. The *relative error*,  $RE_t$  is defined as a factor of  $u_t$ :

$$RE_t = \frac{x_t - u_t}{u_t} = \frac{x_t}{u_t} - 1 \quad (1)$$

By thresholding the relative error, one can detect anomalies while normalizing out the dependence on the magnitude of the expected value. The values of these thresholds, indeed, determine how sensitive the anomaly detection module is. Various thresholding techniques are described in Section 4. Despite its common usage and effectiveness, however, there is no reason to believe the relative error is always the optimal metric for anomaly detection on a given time-series. In fact, the choice of the optimal metric for a given time-series highly depends on the nature of the time-series as well as the TMM performance. For instance, if we are dealing with a very regular time-series for which we have an accu-

rate model, using the prediction error for anomaly detection might be sufficient as it is expected to be Normally distributed. In other cases, the optimal metric might be something between the prediction error and the relative error. For this reason, EGADS tracks a set of deviation metrics by default and the person using the system can create her own error metrics. These error metrics, together with other features, such as the time series characteristics, are used in the alerting module (AM), described in Section 4, to learn consumer's preferences and filter unimportant anomalies.

#### 3.1.2 Decomposition-based methods

The second class of outlier detection methods in EGADS is based on the idea of time-series decomposition. In particular, in the time-series analysis literature, it is a common practice to decompose a time-series into three components: *trend*, *seasonality* and *noise*. By monitoring the noise component, one can capture the outliers. More precisely, if the absolute value of the noise component of point  $x_t$  is greater than a certain threshold, one can announce  $x_t$  as an outlier.

The decomposition of time-series can be done both in the time-domain via smoothing or in the frequency-domain via spectral decomposition. STL (Seasonal-Trend Decomposition based on Loess) [5] is a famous technique that uses Loess smoothing for decomposition. The frequency-domain methods can be further divided into parametric and non-parametric methods. For the parametric methods, the basis used for spectral decomposition has a known parametric form (such as Fourier transform [2] or wavelet transform [22]) whereas, for non-parametric methods, the basis is data-driven [21].

### 3.2 Change Point Detection

Change points are those points in time where the behavior of the time-series starts to deviate from what is expected. The big difference between change points and outliers is that change points correspond to more sustained, long-term changes compared to volatile outliers. A common strategy for detecting change points in the literature is to move two side-by-side windows on the time-series and compute the difference between the behavior of the time-series in the two windows as a measure of the deviation metric [12, 31, 20, 23]. The behavior of the time-series in each window is typically modeled by the distribution of the values, motifs, frequencies, etc. that are present in the time-series. We refer to these techniques as the *absolute* techniques because they do not make explicit assumptions regarding the expected behavior of the time-series.

In EGADS, currently we have taken a different approach which we refer to as the *relative* or *model-based* methods. In these methods, the expected behavior of the time-series is explicitly modeled through one of the modeling techniques mentioned in Section 3.1.1. In particular, we incorporate the plug-in approach described in Section 3.1.1 to compute the sequence of *residuals* (or deviations from the model expectation) for an input time-series. Then we apply the absolute change point detection methods on the series of residuals to detect a change in the distribution of the residuals. We have used Kernel Density Estimation [7] to non-parametrically estimate the distribution of the residuals and the Kullback-Leibler divergence [16] to measure the change in the distribution.

We believe the model-based change point detection methods are more useful than the absolute methods in the practical applications. This is because the change points are meaningful as much as our models cannot explain the behavior of the time-series after a certain time point. However, if the model can explain the time-series behavior even after an absolute change point, from the practical point of view, there is no reason for us to consider that time point as a change point. In other words, the change points are relative to the underlying model used to explain the behavior of the time-series, which in turn gives rise to the relative change-point detection techniques.

### 3.3 Detecting Anomalous Time-series

Another class of anomaly detection techniques supported by EGADS involves detecting anomalous time-series. An anomalous time-series  $T$  is defined as a time-series whose average deviation from the other time-series is significant. Assuming all time-series are homogeneous and come from the same source (i.e. are part of the same cluster) one can simply compute the average deviation for time-series ( $i$ ) relative to other time-series. In EGADS our current approach involves clustering the time-series into a set of clusters  $C$  based on various time-series features including trend & seasonality, spectral entropy, autocorrelation, average Euclidean distance etc. After clustering we perform intra or inter-cluster time-series anomaly detection by measuring the deviation within or among the cluster centroids and the time-series ( $i$ ). A common use-case for this EGADS anomaly detection type involves triaging. For example if a network engineer wants to find an anomalous server amongst millions of time-series, it can be impractical with the previous approaches because the modeling is done on the *per* time-series basis without taking into account the behavior of other metrics. Another application of this anomaly detection type is in finding similar anomalies, which is the inverse of the previous use-case.

## 4. ALERTING

The end-goal of anomaly detection is to produce accurate and timely alerts. EGADS achieves this via a two stage process by first generating a set of candidate anomalies by *threshold selection* and then *filtering* the irrelevant anomalies for a given use-case.

### 4.1 Threshold Selection

The job of threshold selection is to select appropriate thresholds on the deviation metrics produced by the anomaly detection module (ADM). Currently EGADS implements two algorithms for threshold selection based on (a)  $K\sigma$  deviation and (b) density distribution.

The first approach is parametric and assumes that the data is normally distributed with a well-defined mean and standard deviation. Relying on the Gaussian distribution we can apply a well known statistical tool called the ‘three-sigma rule’ which states that 99.73% of all samples lie within three standard deviations of the mean. Therefore, depending on the value of  $K$  in  $K\sigma$ , one can be confident as to the probability of observing a sample at time  $t$ . Depending on the desired level of sensitivity, one can measure if a given sample lies within the 95.45% or 68.27% of all the samples for  $K = 2$  or 1 respectively. Note that the assumption here was that our deviation metrics are normally distributed.

Time-series feature	Description
Periodicity (frequency)	Periodicity is very important for determining the seasonality.
Trend	Exists if there is a long-term change in the mean level
Seasonality	Exists when a time series is influenced by seasonal factors, such as month of the year or day of the week
Auto-correlation	Represents long-range dependence.
Non-linearity	A non-linear time-series contains complex dynamics that are usually not represented by linear models.
Skewness	Measures symmetry, or more precisely, the lack of symmetry.
Kurtosis	Measures if the data are peaked or flat, relative to a normal distribution.
Hurst	A measure of long-term memory of time series.
Lyapunov Exponent	A measure of the rate of divergence of nearby trajectories.

Table 1: Time-series features used by EGADS

The second approach is non-parametric and is useful for the cases when the deviation metric is not normally distributed. The basic idea is to find low density regions of the deviation metric distribution. One approach is to use an algorithm such as Local Outlier Factor (LOF) [3] which is based on a concept of a local density, where locality is given by nearest neighbors, whose distance is used to estimate the density. By comparing the local density of an object to the local densities of its neighbors, one can identify regions of similar density, and points that have a substantially lower density than their neighbors. These are considered to be outliers.

### 4.2 Filtering

Filtering performs the last stage post-processing on the anomalies which are then delivered to the consumer. While the candidate anomalies, which are the input to the filtering stage, are statistically significant, not all of them will be relevant for a particular use-case. For example some consumers are interested in spikes in the time-series, while others are interested in dips, yet others are interested in change points. EGADS provides a simple and intuitive interface which allows users to mark the regions of the time-series that are anomalous. This feedback is then used by EGADS together with time-series and model features to train a classifier that predicts if an anomaly  $a_i$  is relevant to user  $u_j$ . The time-series features tracked by EGADS are shown in Table 1 and are described in more detail in [29]. Section 6.4 explores the performance of a filtering module for a specific use-case. Like other components of EGADS, the filtering component is extensible in terms of models and features.

Figure 2 shows the feature profile of a sample time-series. Note that the metrics beginning with  $dc$  are obtained on the adjusted time-series (i.e. after removing trend and seasonality). In Section 6.2 we look at how these time-series characteristics impact the model performance.

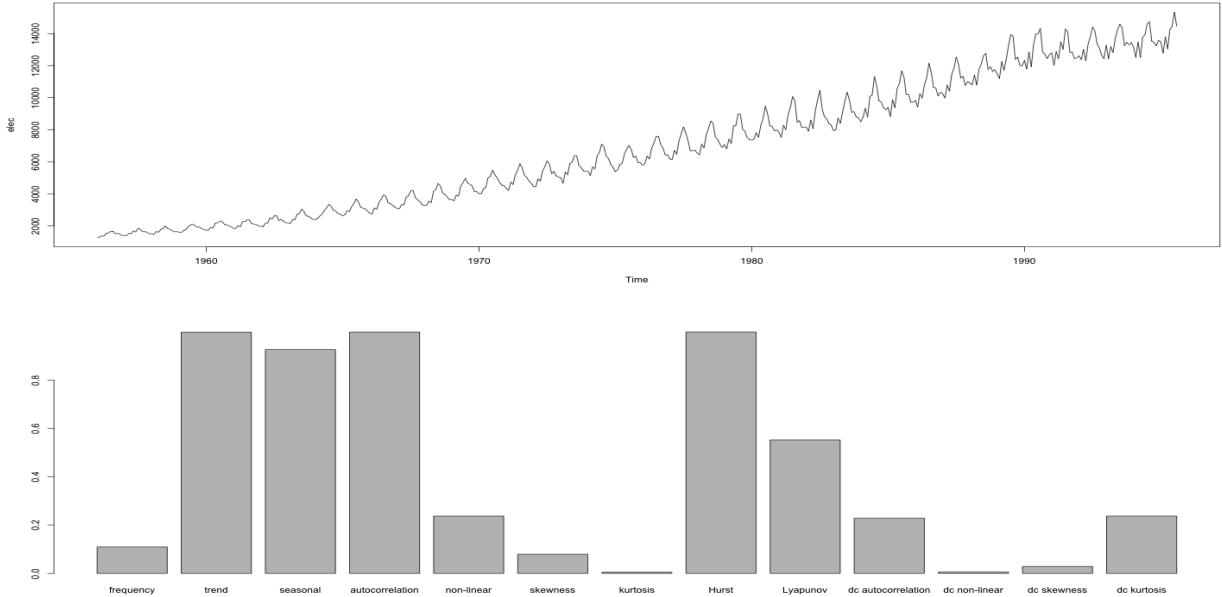


Figure 2: An example of the time-series and its characteristics extracted by EGADS. These characteristics are used by EGADS for filtering and model selection.

## 5. RELATED WORK

There are a number of anomaly detection techniques in the literature. The techniques range from point anomaly detection algorithms to change-point detection algorithms. In [24] authors propose an outlier detection technique based on hypothesis testing, which is very accurate at detecting extreme outliers. In fact Twitter, [26], uses [24] in conjunction with piecewise approximation of the underlying long-term trends to remove many of the false positives. Twitter’s approach is fast and enjoys an impressive precision and recall, however it is specific to the use-case of Twitter. There are also a number of open-source point anomaly detection techniques available including [27, 15].

Authors in [13] provide an anomaly detection technique that finds ‘Change Points’ or ‘Level Shifts’. Change Points (CP) are different from point anomalies or point outliers in that CP reflect a change in underlying statistic of the time-series (e.g., Mean shift). CP typically occurs in a time-series with a launch of a new product feature or a new platform. There are a number of open-source change point detection algorithms available including [14].

In our experience, a particular anomaly detection algorithm is usually applicable to only a specific use-case. As authors in [1] mention the anomalies will have typically a high anomaly score, but the high score alone is not a distinguishing factor for an anomaly. Rather, it is the analyst, who regulates the distinction between noise and anomaly. Similarly, authors in [4] provide a concise overview of the anomaly detection technique per category, citing the fact that only a set of anomaly models are most appropriate for a given anomaly category of interest. Therefore, based on the observation that ‘One Size Fits All’ is a myth in the anomaly detection world, EGADS uses a strategy where a

collection of well trained anomaly detection models with a post-processing use-case-specific anomaly filtering stage is used. Nevertheless, EGADS is not the only generic anomaly detection framework out there. Venkataraman et. al [28] proposed a ‘Black Box Anomaly Detection’ framework that can be applied to a variety of data sources. Although the proposed framework is generic, it is not fully automated because it still needs a significant degree of user involvement in setting the appropriate models and metrics for a given application. Besides, this framework assumes the input training data to the system is anomaly-free, which is an unrealistic assumption in many real-world use-cases. On the other hand, Lan et. al. [17] proposed a framework for anomaly detection in large-scale systems which is automated but not generic enough to be applied to a general time-series anomaly detection problem. EGADS, however, provides flexible and effective mechanisms which make it both generic, automated and scalable. Furthermore, from the industrial point of view, EGADS has been incorporated in large-scale monitoring systems across Yahoo.

## 6. EXPERIMENTAL STUDY

We present the experiments for the modeling, anomaly detection and alerting components of EGADS next.

### 6.1 Data

The dataset used for the experiments is comprised of a mixture (50/50) of synthetic and real data. We have created a synthetic time-series generation tool that is being open-sourced along with the framework [19] and the benchmarking data [18]. Using the tool, each synthetic time-series is generated by specifying the length, magnitude, number of anomalies, anomaly type, anomaly magnitude, noise level,

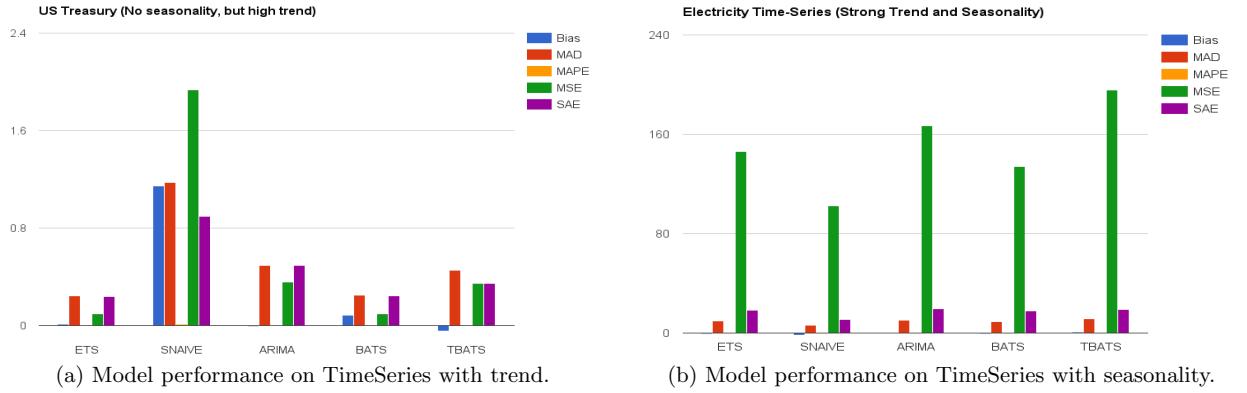


Figure 3: Model performance on time-series with varying characteristics.

trend and seasonality. These parameters are picked from a fixed distribution. The real dataset is comprised of Yahoo Membership Login (YML) data. The YML data tracks the aggregate status of user logins to the Yahoo network. Both the synthetic and real time-series contain 1400 data-points each, which for the YML data represent 3 months worth of data-points. Unless otherwise stated, all experiments were run on 1000 randomly picked time-series and the results were averaged. Also note that both the synthetic and real-time data have anomaly labels, that are either synthetically or editorially generated, allowing us to measure precision and recall.

## 6.2 Modeling Experiments

Time-series modeling (captured by the TMM component in EGADS) is a fundamental part of anomaly detection. It is often the case that the anomaly detection is as good as the underlying time-series model. Due to a large number of candidate models, model-selection becomes critical and depends upon time-series characteristics and available resources. In the experiments that follow, we demonstrate the impact of time-series features on the model performance and show the trade-off between accuracy, memory usage and training time. The models and the error metrics used in the experiments are described in Tables 2 and 3 respectively. More details about the models and the metrics can be found in [10] and [29].

### 6.2.1 Time-series Characteristics and Model Performance

To demonstrate the impact of time-series features on model performance we compare the error metrics of different models when fitting time-series with different features (see Section 4.2). Figure 3 shows that time-series characteristics play an important role in model behavior. For example the *Seasonal Naive Model*, performs poorly on a dataset with no seasonality and a strong trend. EGADS keeps track of the historic time-series characteristics and model performance. Using this historical information, EGADS selects the best model (given the time-series features) judged by the error metrics described in Table 3. In practice, performing model selection based on the data features is much faster than performing cross-validation for every model.

Model	Description
Olympic Model (Seasonal Naive)	The naive seasonal model where the prediction for next point is a smoothed average over previous $n$ periods.
Exponential Smoothing Model	A popular model used to produce smoothed time-series. Double and Triple exponential smoothing variants add trend and seasonality into the model. The ETS model used for the experiments automatically picks the best ‘fit’ exponential smoothing model.
Moving Average Model	In this mode, the forecast is based on an artificially constructed time series in which the value for a given time period is replaced by the mean of that value and the values for some number of preceding and succeeding time periods. The Weighted Moving Average and Naive Forecasting Model are special cases of the moving average model.
Regression Models	Models the relationship between $x$ & $y$ using one or more variable.
ARIMA	Autoregressive integrated moving average.
(T)BATS Family	(Trigonometric) Exponential smoothing state space model with Box-Cox transformation.
Spectral Kalman Filter	An in-house algorithm that implements a fast multi-variate spectral learning method for learning Kalman Filter parameters.

Table 2: Models Used for Modeling Experiments

### 6.2.2 Time-series Model Scalability

As discussed in Section 2 it is often prohibitive to build models for every time-series and optimization techniques are required to support real-time performance over massive (e.g., millions of points every second) data-streams. A fundamental optimization performs a trade-off between model size, training time and accuracy. Such a trade-off is shown in Figures 4(a) and 4(b). From the figure, for example, it is clear that the Seasonal Naive model is quick to train but has a relatively large memory requirement and a high average error. At Yahoo, a target in terms of resources and training

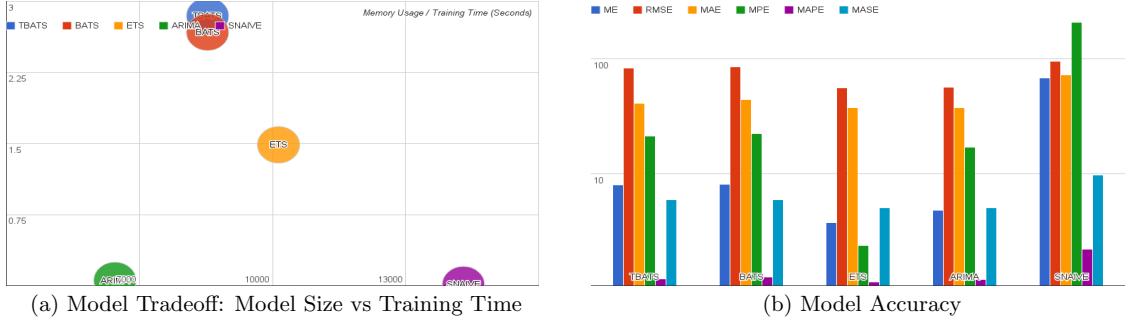


Figure 4: Model trade-offs

Model	Description
Bias	The arithmetic mean of the errors.
MAD	The mean absolute deviation. Also known as MAE.
MAPE	The mean absolute percentage error.
MSE	The mean square of the errors.
SAE	The Sum of Absolute Errors.
ME	Mean Error.
MASE	Mean absolute scaled error.
MPE	Mean percentage error.

Table 3: Metrics Used for Modeling Experiments

Model	Description
EGADS	EGADS density-based anomaly detection.
ExtremeLow-DensityModel Outlier	
EGADS CP	EGADS kernel-based change-point detection.
EGADS KSigmaModel Outlier	EGADS re-implementation of the classic k-sigma model.
Twitter Outlier	The Open-Source Twitter-R anomaly detection library based on the Generalized ESD method.
ExtremeI & II R Outlier	Open source univariate outlier detection that threshold the absolute value and the residual to detect anomalies.
BreakOut Twitter CP	A package from Twitter that uses an ESD statistics test to detect change points.
ChangePt1 R CP	An R library that implements various mainstream and specialized change-point methods for finding single and multiple change-points within data. Method I uses a change in variance.
ChangePt2 & 3 R CP	Detects a change in the mean and the variance.

Table 4: Open Source systems used for evaluation

time is first set and then the models are picked accordingly. In other words, the objective is to minimize the errors in Table 3 subject to the resource and model building time

constraints. Other optimization techniques including time-series sampling and model sharing are being investigated.

### 6.3 Anomaly Detection Experiments

In this section we compare open source system against EGADS. The open source systems considered are shown in Table 4. The results on the data described in Section 6.1 are shown in Figure 5. The results are compared in terms of the standard  $F_1\text{-Score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ . The results indicate that there is no best anomaly detection model for all use-cases. In particular different algorithms are best at detecting different types of anomalies. For example Twitter [13] performs best on the time-series labeled *TS-2* while ExtremeLowDensity model is best on *TS-3*. These datasets contain a mixture of anomaly types (e.g., outliers, change-points), and one might argue that comparing an algorithm that is only meant for change-point detection is not fair. Recall, however, that the motivation for EGADS was that the user should be agnostic to the type of time-series and the type of anomalies that are in the data. The system must be able to gracefully and robustly deal with a wide variety of anomalies present in the data. For this reason, EGADS is built as a library that combines a set of anomaly detection models into a single framework. The anomalies from these models are forwarded to the filtering component for accurate anomaly detection.

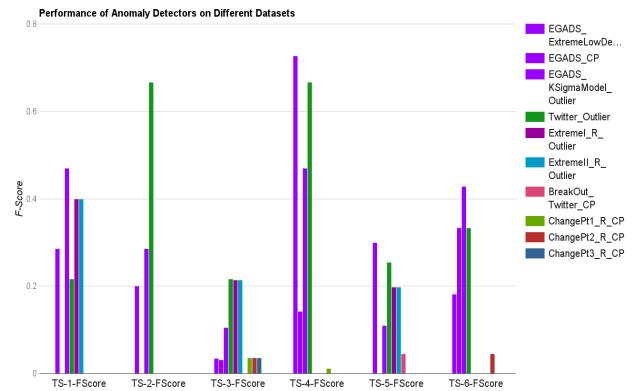


Figure 5: Anomaly model performance on different datasets. Observe that there is no single model that is best on all datasets.

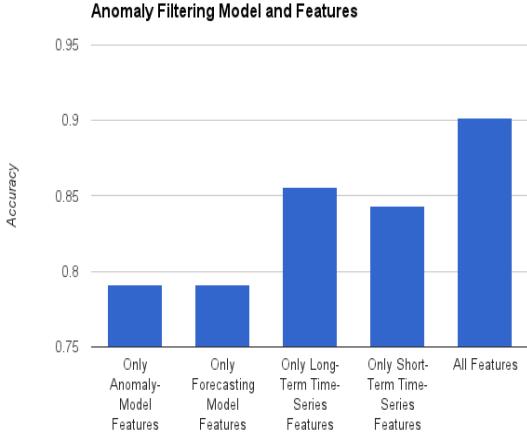


Figure 6: Accuracy of the filtering stage using different types of features.

## 6.4 Anomaly Filtering Experiments

The importance of an anomaly often depends on the use-case. Specifically, some users may be interested in the time-series behavior that exhibits a malicious attack, while others may be interested in revenue drops. Yahoo Membership (YM) use-case refers to the former set of users. Specifically for the YM use-case, editors supplied feedback to EGADS of instances that exhibited abnormal spikes and level shifts. *Abnormal* in the case of YM meant *seasonal* followed by *non-seasonal* behavior which characterizes most of the attacks. Also the YM editors did not care about *traffic-shift* behavior, where a large drop in traffic was observed in a time-series due to router table being updated.

To address this requirement the filtering stage scanned all anomalies  $a_i$  from all models and using a model classified if  $a_i$  was a true positive. The model used in the filtering stage for the YM use-case is a boosted tree model based on AdaBoost [8]. The features used in the model are described in Table 1. The core principle of AdaBoost is to fit a sequence of weak learners (e.g., small decision trees) on repeatedly modified version of the data. The final result is then produced via a combined weighted majority vote. On each iteration, the examples that are difficult to predict receive a higher importance in the next iteration and therefore each subsequent weak learner focuses on the examples that are missed by the previous learners in the sequence. Besides the time-series features described in Table 1 we use the model features described in Section 6.2. The experiments in Figure 6 indicate an impressive precision/recall even with just the time-series features compared to just using the model alone without the filtering stage. This experiment underlines an important principle and a critical component of any anomaly detection framework: an anomaly is use-case specific and must be learned automatically for a fully scalable and automated solution.

## 7. PRACTICAL USE-CASES AT YAHOO

A major use case for anomaly detection at Yahoo is the monitoring of system and business metrics in order to detect infrastructure and product issues. Yahoo currently tracks over one hundred million distinct timeseries emitted by its

production systems. In monitoring these timeseries, three broad use categories emerge; system metrics, business KPIs, and groups of like metrics.

### 7.1 System Metrics

System metrics are broadly defined as measurements of the health of a hardware component in a serving system. Examples include CPU utilization, free disk space, network interface traffic stats, and memory utilization. Some of these metrics, such as CPU utilization, track the overall traffic to a component, and some, like free disk space, are independent of the current traffic levels. What sets system metrics apart from other types of metrics is that due to the redundant nature of Yahoo’s serving system, a threshold violation is typically a leading indicator of serving problems. Because of this, alerting against these types of metrics is often not treated as an outage, but instead used to trigger longer term remediations such as adding capacity or clearing disk space.

### 7.2 Business KPIs

Business KPIs are metrics that directly reflect customers’ experiences with Yahoo sites. Examples include things such as page views, serving latency, serving errors, click-through rate, and revenue received. Business KPIs are almost always trailing indicators of site issues, and by definition reflect impact to Yahoo’s ability to serve. As a result, anomalies in business KPIs are normally treated with a high degree of urgency. The nature of business KPIs at large scale is that they tend to be highly predictable, so they lend themselves well to automated anomaly detection. Yahoo has had tremendous success in instrumenting business KPIs to discover serving and revenue issues using automated anomaly detection.

### 7.3 Groups of similar metrics

Most of Yahoo’s infrastructure follows a horizontal scaling model, with dozens to hundreds of individual servers making up each serving tier. When triaging and isolating the cause of a suspected incident, it can be difficult to survey an infrastructure of thousands of machines to find the fault. Automated anomaly detection can be used to rank the relative anomalousness, or “interestingness”, of each component of the infrastructure, and these characteristics can be ranked in descending order to enable operators to quickly see patterns and isolate issues.

## 8. CONCLUSION

Anomaly detection is a critical component at the heart of many real-time monitoring systems with applications in fault detection, fraud detection, network intrusion detection and many others. Despite its crucial importance, implementing a fully-automatic anomaly detection system in practice is a challenging task due to the large problem scale and the diverse use-cases residing in the real-world setting. These challenges typically result in solutions that are either not scalable or highly specialized, which would in turn result in a high rate of false positives when applied to other use-cases.

In this paper, we introduced EGADS, the generic anomaly detection system implemented at Yahoo to automatically monitor and alert on millions of time-series on different Yahoo properties for different use-cases ranging from fault detection to intrusion detection. As we described in the paper, the parallel architecture of EGADS on Hadoop as well as its

stream processing mechanism through Storm enable it to perform real-time anomaly detection on millions of time-series at Yahoo. Furthermore, EGADS employs different time-series modeling, and anomaly detection algorithms to handle different monitoring use-cases. By incorporating this array of algorithms combined with a machine-learned mechanism in the alerting module, EGADS automatically adapts itself to the anomaly detection use-case that is important to the user. All of these features effectively create a powerful anomaly detection framework which is both generic and scalable. Our showcase experiments on real and synthetic datasets have shown the superior applicability of our framework compared to its rival solutions.

Last but not least, EGADS by its very nature is extendable, providing an easy mechanism to plugin new models and algorithms into the system. This feature specifically creates an opportunity for the community to contribute to EGADS. Finally, to further engage with the anomaly detection and monitoring community, our framework together with all its datasets are contributed to the open source repository.

## 9. REFERENCES

- [1] C. Aggarwal. *Outlier Analysis*. Springer New York, 2013.
- [2] P. Bloomfield. *Fourier analysis of time series: an introduction*. John Wiley & Sons, 2004.
- [3] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, May 2000.
- [4] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.
- [5] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.
- [6] J. Durbin and S. J. Koopman. *Time series analysis by state space methods*. Number 38. Oxford University Press, 2012.
- [7] V. A. Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.
- [8] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1996.
- [9] S. S. Haykin, S. S. Haykin, and S. S. Haykin. *Kalman filtering and neural networks*. Wiley Online Library, 2001.
- [10] R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, pages 679–688, 2006.
- [11] R. H. Jones. Exponential smoothing for multivariate time series. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 241–251, 1966.
- [12] Y. Kawahara, T. Yairi, and K. Machida. Change-point detection in time-series data based on subspace identification. In *ICDM*, pages 559–564. IEEE, 2007.
- [13] A. Kejariwal and P. Kumar. Mitigating user experience from ‘breaking bad’: The twitter approach. In *Velocity*, New York, NY, Sept. 2014.
- [14] R. Killick. *changepoint, an R package that implements various mainstream and specialised changepoint methods.*, 2014.
- [15] L. Komsta. *outliers, an R package of some tests commonly used outlier detection techniques.*, 2011.
- [16] S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.
- [17] Z. Lan, Z. Zheng, and Y. Li. Toward automated anomaly identification in large-scale systems. *Parallel and Distributed Systems, IEEE Transactions on*, 21(2):174–187, 2010.
- [18] N. Laptev and S. Amizadeh. Online dataset for anomaly detection. <http://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70>, April 2015.
- [19] N. Laptev and S. Amizadeh. Egads source code. <https://github.com/yahoo/egads>, June 2015.
- [20] S. Liu, M. Yamada, N. Collier, and M. Sugiyama. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83, 2013.
- [21] V. Moskvina and A. Zhigljavsky. An algorithm based on singular spectrum analysis for change-point detection. *Communications in Statistics-Simulation and Computation*, 32(2):319–352, 2003.
- [22] D. B. Percival and A. T. Walden. *Wavelet methods for time series analysis*, volume 4. Cambridge University Press, 2006.
- [23] B. K. Ray and R. S. Tsay. Bayesian methods for change-point detection in long-range dependent processes. *Journal of Time Series Analysis*, 23(6):687–705, 2002.
- [24] B. Rosner. Percentage points for a generalized esd many-outlier procedure. *Technometrics*, 25(2):165–172, 1983.
- [25] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal, and D. Ryaboy. Storm@twitter. In *SIGMOD*, pages 147–156, New York, NY, USA, 2014. ACM.
- [26] O. Vallis, J. Hochenbaum, and A. Kejariwal. A novel technique for long-term anomaly detection in the cloud. In *USENIX*, Philadelphia, PA, June 2014. USENIX Association.
- [27] M. van der Loo. *extremevalues, an R package for outlier detection in univariate data*, 2010. R package version 2.0.
- [28] S. Venkataraman, J. Caballero, D. Song, A. Blum, and J. Yates. Black box anomaly detection: is it utopian? 2006.
- [29] X. Wang, K. Smith-Miles, and R. Hyndman. Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. *Neurocomput.*, 72(10-12):2581–2594, June 2009.
- [30] W. W.-S. Wei. *Time series analysis*. Addison-Wesley publ., 1994.
- [31] Y. Xie, J. Huang, and R. Willett. Change-point detection for high-dimensional time series with missing data. *Selected Topics in Signal Processing, IEEE Journal of*, 7(1):12–27, 2013.

# 北京邮电大学

## 本科毕业设计（论文）开题报告

学院	信息与通信工程	专业	通信工程	班级	2014211112
学生姓名	林文鼎	学号	2014210328	班内序号	07
指导教师姓名	别红霞	所在单位	信息与通信工程学院	职称	教授
设计（论文）	基于腾讯定位数据的异常事件检测算法				
题目	Anomalous Event Detection Based on Tencent Positioning Data				

### 1. 选题的背景和意义

随着 GPS 定位，传感器网络和无线通信等应用的日益普及，越来越多的定位数据被收集和保存在应用服务器，如何快速地从这些定位数据中挖掘出有效信息日益成为一个令人关注的研究课题。在地图的定位数据中，人流密度可以反映出当地的一些事件变化。在通常情况下，人流密度的分布应该会服从一个基于时间的规律变化。而在异常事件（如异常气象，交通管制等）出现时，定位数据较以往同时段的数值必然会有异常的波动，我们可以从这些波动中获取到异常事件发生时上述定位数据的异常特征，从而在相似事件发生时可以及时做好有关准备。

### 2. 研究的基本内容和拟解决的主要问题

通常在异常事件（如异常气象，交通管制等）出现时，定位数据较以往同时段的数值必然有异常的波动。本论文研究基于位置服务数据的异常事件检测算法，并开发实际应用。主要研究内容如下：

- (1) 首先针对当前定位数据的时代背景，提出异常检测的必要性和重要性。然后调查异常检测的研究概况与发展状况，对常用检测方法进行介绍。
- (2) 通过给定的腾讯定位数据，读入并分析在异常事件出现时的数据异常性特征，标出所要处理数据的异常。
- (3) 研究并实现基于位置服务数据的异常检测算法。
- (4) 研究并实现基于上述位置服务数据形成的图像异常检测算法。
- (5) 分析可能提高准确率的改进，集成上述算法开发实际应用。

基于上述研究内容，需要解决的主要问题如下：

- (1) 腾讯定位数据的读入，MATLAB 信号分析工具的使用，信号异常点特征的明确，数据的标定；
- (2) 信号异常检测的方法调查，输入曲线并识别出异常点的功能实现；
- (3) 基于上述位置服务数据形成图像，图像异常检测的算法调查，机器学习算法的运行框架选择。

### 3. 研究方法及措施：

- (1) 针对原始的腾讯定位数据，首先需要了解 TIF 格式的数据样式（维度、有效信息），将数据读入 MATLAB 中作信号分析以便于观察异常点特征并标注数据。在此过程中，需要熟悉 MATLAB 中常用的信号分析工具，在此基础上调研相关信号异常检测算法并尝试实现 demo。由于常规的定位数据应大致符合某一个变化曲线，可以通过基于统计的方法展开调查；同时也可以从信号的频谱特征或是聚类方法入手寻找可能解决问题的方法。当积累了一定的信号异常检测算法，可以对

实际数据进行测试，汇总各个算法的优劣，准备中期检查。

(2) 实现定位数据的曲线异常检测后，需要对数据进行图像的重建，从图像的角度分析异常。可以从以下两个角度进行处理：

①建立判别模型：和信号的处理方法类似，直接寻找异常点。基于这种角度可以通过主成分分析（PCA）与矩阵分解的方法将图像的高维数据映射到低维度空间，将异常点的特征最大化地表示在低维度空间中进行判断。

②建立生成模型：预测正常情况下的数据点分布并通过计算实际点差值来寻找异常点。近年来机器学习、神经网络在异常检测中的应用得到了广泛关注，递归神经网络（RNN）是一类神经网络，包括一层内的加权连接（与传统前馈网络相比，连接仅馈送到后续层）。因为 RNN 包含循环，所以它们可以在处理新输入的同时存储信息。这种记忆使它们非常适合处理必须考虑事先输入的任务（比如时序数据）。在充分调研该网络的理论以及最佳运行框架后，可以采用 RNN 去预测时序的定位数据并寻找异常点。

根据以上两种思路，寻找相应的算法应用到数据图像上，并对它们的有效性进行分析。

(3) 实现上述相关算法后，查找并实现一些相对复杂的改进策略，分析其复杂度和有效性；同时，调研并设计应用，整合输入输出与上述异常检测算法。汇总所有资料并核对任务完成情况，完成论文撰写及答辩工作。

#### 4. 研究工作的步骤与进度

01/01-01/26 资料查阅，论文开题；

01/27-03/04 阅读综述论文，调研相关算法；

03/05-03/19 技术路线确定；

03/20-04/15 完成异常数据的标定以及实现异常数据的检测算法；

04/16-04/20 准备中期检查相关材料；

04/21-05/10 完成异常数据生成的图像检测算法并开发应用；

05/11-05/20 核对任务完成情况，毕设论文撰写；

05/21-06/04 汇总毕设相关材料，准备答辩。

#### 5. 主要参考文献

- [1] Patcha A, Park J M. An overview of anomaly detection techniques: Existing solutions and latest technological trends[J]. Computer Networks, 2007, 51(12):3448-3470.
- [2] Chan P K, Mahoney M V, Arshad M H. A machine learning approach to anomaly detection[J]. 2003.
- [3] Malhotra P, Vig L, Shroff G, et al. Long short term memory networks for anomaly detection in time series[C]//Proceedings. Presses universitaires de Louvain, 2015: 89.
- [4] Eskin E. Anomaly detection over noisy data using learned probability distributions[C]//In Proceedings of the International Conference on Machine Learning. 2000.
- [5] Ringberg H, Soule A, Rexford J, et al. Sensitivity of PCA for traffic anomaly detection[J]. ACM SIGMETRICS Performance Evaluation Review, 2007, 35(1): 109-120.
- [6] Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey[J]. ACM computing surveys (CSUR),

2009, 41(3): 15.

- [7] Lee W, Xiang D. Information-theoretic measures for anomaly detection[C]//Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on. IEEE, 2001: 130-143.
- [8] Noble C C, Cook D J. Graph-based anomaly detection[C]//Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2003: 631-636.

指导教师签字		日期	年   月   日
--------	--	----	-----------

# 北京邮电大学

## 本科毕业设计（论文）中期进展情况检查表

学院	信息与通信工程学院	专业	通信工程	班级	2014211112
学生姓名	林文鼎	学号	2014210328	班内序号	7
指导教师姓名	别红霞	所在单位	信息与通信工程学院	职称	教授
设计（论文） 题目	(中文) 基于腾讯定位数据的异常事件检测算法 (英文) Anomalous Event Detection Based on Tencent Positioning Data				
目前已完成任务	<p>主要内容：(毕业设计(论文)进展情况，字数一般不少于1000字)</p> <p>异常检测是找出其行为严重不同于预期对象的一个检测过程。这些对象被称为异常点或者离群点。而地图的定位数据中，终端的数量变化可以反映出当地的一些事件变化。在通常情况下，终端的分布应该会服从一个基于时间的规律变化。而在异常事件（如异常气象，交通管制等）出现时，定位数据较以往同时段的数值必然会有异常的波动，我们可以从这些波动中获取到异常事件发生时上述定位数据的异常特征，从而在相似事件发生时可以及时做好有关准备。目前我主要在以下几方面取得了一定进展。</p> <p>(1) 调研异常检测的研究概况与发展状况，并对主要算法进行复现<sup>[1]</sup></p> <p>异常是指数据特征不符合该特征一般存在的区间的现象。寻找异常挑战来源于两个方面：首先，“异常”这个概念较为模糊，偏离正常数据中心多少可以被界定为异常没有一个定量的数值，甚至完全可以认为在划定的边界线附近的数据是正常的数据；再者，用于划定正常区间的数据中有时也会存在异常，导致划定边界线偏差，同时考虑到正常的数据往往远大于异常数据，使用机器学习的方法进行训练时很容易过拟合导致无法检测出异常。</p> <p>异常也有诸多分类。通常情况下我们所说的异常指的是点异常，其含义是多个数据实体中，如果存在一个实体对于其他实体来说是异常的，那么其就是点异常。对于本课题，异常应被认为是环境异常，其是一个数据实体在特定环境中的异常；数据实体有两部分组成：环境属性&amp;行为属性。环境属性表征了数据实体所处的环境，例如时间序列数据的时间点，空间数据的地理坐标；行为属性表征了在上述特定环境属性下区分数据实体的属性，类似于地理数据的某地降雨量。本课题中环境属性即是时间点与地理坐标，行为属性是在某时间点某地理坐标下的定位终端数量。</p> <p>传统检测异常的方法分为以下几类：基于分类的异常检测方法，基于最近邻的异常检测方法，基于聚类的异常检测方法，基于统计的异常检测方法。</p> <ul style="list-style-type: none"><li>• 基于分类的异常检测方法：该方法分为两个步骤。第一阶段通过已有的标签数据训练分类器。第二阶段使用该分类器对未知数据进行分类。</li><li>• 基于最近邻的异常检测方法：该方法基于“正常数据间的距离较近，异常数据与最近的数据点也较远”的假设展开，可以从密度的角度去区分正常点和异常点。</li><li>• 基于聚类的异常检测方法：该方法基于“正常数据通常聚集在一起，同分类下存在大量数据，而异常数据不属于任何一个小组或是某分类下的数据样本极少”的假设展开，但聚类的思想更适合寻找聚类，即正常数据。</li><li>• 基于统计的方法：对于一个统计模型，如果输入数据会处于统计概率中较低的</li></ul>				

位置，那么则认为其为异常数据。

(2) 读入并分析定位数据，并对数据进行预处理与分析

给定的腾讯定位数据的异常事件为一次台风过境，会导致地图上的终端定位数量发生显著改变。对于该定位数据，我分为以下几个步骤进行处理：

1. 通过文件的分析得到其真实地理坐标并与实际地图进行比对，大致确定为广东省珠海市沿海一带；
2. 数据样本为 TIF 格式文件，横纵坐标分别为经纬度，组成的每一个点其上的值代表了当前时刻的该点存在的终端数量；
3. 通过 MATLAB 对数据进行处理，作出一天内各小时，整体数据内每天的某一小时和整体数据内每天的每一小时的图像，对该时空异常有一个大致的判断。
4. 由于该定位坐标沿海，位于海面上的坐标点存在大量接近零的数值点，对于异常判断是冗余的，采用最大值判断进行剔除。
5. 通过作图，我观察得到每一天内的数值变化大致符合一个曲线，所以通过采样所有日期某一小时的地图数据可以得到异常的日期。
6. 对上述的地图数据上所有已筛选过的点进行曲线异常检测，判断异常日期是哪一天或是全部为正常数据。如果最后一张图上的大部分点都指向某一天存在异常，即可以认为该天是异常天

(3) 使用异常点检测算法对处理后的数据进行异常检测，并对效果分析

通过上述算法的调研，并根据数据的特征，我采用了以下方法检测异常日期：

1. 差分分析法 (Laplace 算子)：通过分析某点前后日期定位数值的变化比例，找到变化波动最大的那一天并且如果那一天的变化确实超过某个阈值，那么该点在那个时间点是异常的；对于数据量小且异常值较为明显的异常样本，该方法十分高效。
2. 小波分析法 (离散小波变化)：通过对某点随时间变化的曲线进行一层二进离散小波分解，可以得到信号的高频分量（剧烈的变化）以及低频分量（信号的大致波形），将高频分量减少。使用低频分量以及模糊过的高频分量进行小波重建，将重建后的信号和原始信号相减，并取出峰值（即最大的噪声），认为峰值日期即为异常日期；较差分分析法计算复杂，但对于数据量大时拥有更好的效果。
3. 局部异常因子算法：通过计算某点的局部密度，使用局部离群因子来表示某点的领域点的局部可达密度与某点的局部可达密度之比的平均数来表征某点是否与其邻域内的点为同一簇的可能性。如果这个比值远大于 1，则认为该点代表的日期为异常日期；无论数据量大小，只要异常样本相对明显，则十分有效，缺点是计算复杂。
4. 最大似然法 ( $3\sigma$ 准则)：假设正常的数据符合高斯分布，采用最大似然法去拟合数据的平均值与方差，找到数据当中超过  $\mu + 3\sigma$  的点并采用最大值，那么这个点代表的日期即是异常天。由于数据不完全符合高斯分布，且异常值可能存在于  $3\sigma$  内，所以该方法计算量大且效果不好。

参考文献：

[1] Chandola, Varun, A. Banerjee, and V. Kumar. Anomaly detection: A survey. ACM, 2009.

是否符合任务书要求进度：符合任务书要求进度

尚需完成的任务	采用更大量的数据分析，并对曲线异常检测算法做进一步更新； 尝试采用图像的方法直接对异常天进行检测； 整合现有功能，使用 Matlab 生成可执行文件，达到输入地图数据→判断异常天的功能。		
	能否按期完成设计（论文）：可以按期完成设计		
存在问题和解决办法	存在	(1) 如何判断多异常天数 (2) 地图数据中存在大量的冗余数据如果，采用图像的方法会显的低效。	
	拟采取的办法	(1) 更适合采用邻域或统计的方法而不是阈值的方法去判断。 (2) 采用聚类的方法将图中划分为几个子图像区域并对每个子图像区域进行分析。	
指导教师签字		日期	年 月 日
检查小组意见	负责人签字： 年 月 日		

注：可根据长度加页。