单位代码：10293 密 级：

南京邮电大学

硕士学位论文

论文题目：无线传感器网络中的蠕虫检测与异常数据检测方案研究

学　　号	1016041018
姓　　名	郑文添
导　　师	吴蒙，杨立君
学科专业	信息安全
研究方向	无线传感器网络安全
申请学位类别	工学硕士
论文提交日期	2019-02-18

Research on Worm Nodes Detection and Outliers detection In Wireless Sensor Network

Thesis Submitted to Nanjing University of Posts and Telecommunications for the Degree of

Master of Engineering

By

Wentian Zheng

Supervisor: Prof. Meng Wu

February 2019

南京邮电大学学位论文原创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得南京邮电大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

本人学位论文及涉及相关资料若有不实，愿意承担一切相关的法律责任。

研究生学号：_____ 研究生签名：_____ 日期：_____

南京邮电大学学位论文使用授权声明

本人授权南京邮电大学可以保留并向国家有关部门或机构送交论文的复印件和电子文档；允许论文被查阅和借阅；可以将学位论文的全部或部分内容编入有关数据库进行检索；可以采用影印、缩印或扫描等复制手段保存、汇编本学位论文。本文电子文档的内容和纸质论文的内容相一致。论文的公布(包括刊登)授权南京邮电大学研究生院办理。

涉密学位论文在解密后适用本授权书。

研究生签名：_____ 导师签名：_____ 日期：_____

摘要

随着物联网技术和无线通信技术的发展，无线传感器网络(Wireless Sensor Network,WSN)越来越广泛的运用于现实世界中的数据采集和分析中。WSN 是典型的 Ad-hoc 网络，由数目众多的传感器节点和基站构成，传感器节点负责收集数据，与基站通信，它们体积小，能量和内存有限。WSN 需要根据实时监测的数据来做出决策，然而由于各个节点部署在无人看守的野外环境中，节点的能量受限，会引起一系列的网络安全问题。

目前传感器网络安全的研究主要分为两个方向。一方面，为了防止攻击者夺取 WSN 的控制权，需要对 WSN 中的恶意节点进行实时检测和清除。WSN 通常部署在无人值守的野外环境中，传感器节点极易被攻击者捕获。目前 WSN 中针对移动节点攻击和副本节点攻击的方案已经很成熟了，然而蠕虫节点攻击相关的研究却非常少。蠕虫攻击对网络的危害是最大的，它能够迅速感染传感器节点，使网络迅速瘫痪。另一方面，为了基站能够做出精准的判断，需要传感器节点能够传输可靠的数据值。因此，需要检测并剔除网络中由噪声或者传感器节点内部错误等原因引起的异常数据。然而，受限于传感器节点的计算资源和内存资源，相关的检测方案并不能在资源和检测效率上得到折衷的效果。

本文针对以上两点，进行了相关的研究。以蠕虫传播特性和序贯概率比检验(Sequential Probability Ratio Test,SPRT)为基础，结合偏向采样和随机值采样提出了 SPRT-Biased-Random 蠕虫检测方案来加速 WSN 中蠕虫节点的检测。实验结果表明 SPRT-Biased-Random 方案能够高效地检测 WSN 中的慢蠕虫传播，并在 5~18 个时隙内检测到所有蠕虫节点，使得传感器网络最终受感染的节点数维持在 2%~5%之间。针对 WSN 的异常数据检测，以 PCA 算法为基础结合马氏距离提出改进型分布式主成分异常数据检测方案(Improved Distributed PCA-Based Outlier Detection Method, IDPCA)。方案使用双重检测机制，有效的提高了网络中异常数据的检测性能并降低了网络中的通信资源开销。实验结果表明该模型能够达到 96%-97%的异常数据检测率并维持 2%的异常数据误报率，同时能够对异常数据进行溯源，让网络管理者能够快速定位网络问题。为了弥补 IDPCA 方案无法对线性不可分数据进行高效检测的缺陷，提出了基于马氏内核函数的分布式核主成分异常数据检测方案 (Improved Distributed KPCA-Based Outlier Detection Method, IDKPCA)。该方案将监测数据集映射到高维空间，能够很好的对线性不可分数据进行检测。通过马氏距离衡量数据向量到数据中心的距离，利用测量数据维度之间的相关性并训练最优的核参数，IDKPCA 方案对线性不可分的数据能够达到近乎 98%的异常数据检测率和 2%的数据误报率。

关键词：无线传感器网络，恶意蠕虫节点，异常数据检测，序贯概率比检验，主成分分析

Abstract

Abstract: With the continuous development of Internet of Things and wireless communication technology, Wireless Sensor Networks (WSNs) are widely used in data collection and analysis in the real world. WSN is a typical ad hoc network which posesses a large number of sensor nodes and base stations. Sensor nodes are responsible for collecting data measurments and communicating with base stations, they are small and energy-limited. WSN needs realtime data measurments to make decisions, However, due to the deployment of harsh enviroment, the limited energy, there exist a series of network security problems in WSN.

The security problems of wireless sensor network are mainly divided into two parts. On the one hand, how to identify and remove malicious nodes in the network and ensure the right control of the network is particularly important. Due to the deployment of harsh environment, sensor nodes are easily compromised by attackers. Up util now, schemes for mobile compromised node and replica node attacks in WSN are very mature, but few researches on worm compromised nodes have been taken. Worm attacks have the greatest damage to wireless sensor network, they could infect sensor nodes and paralyze the network quickly. On the other hand, in order for the base station to make an accurate judgment, it is important for the sensor node to transmit reliable data. Therefore, it is necessary to detect and eliminate outliers caused by noise or internal errors of sensor nodes in the network. However, due to the limited computing resources and memory resources of sensor nodes, the proposed detection schemes cannot achieve a compromise between resources and detection efficiency.

In view of the above two points, this paper has carried on the related research. Based on the characteristics of worm propagation and sequential probability ratio test, a SPRT-Biased-Random worm detection method is proposed combining with biased sampling and random value sampling to accelerate worm node detection in WSN. Experimental results show that SPRT-Biased-Random method can efficiently detect worm propagation in WSN and find all worm nodes in 5 to 18 time slots, so that the number of nodes eventually infected in the sensor network remains between 2 % and 5 %. Based on the PCA algorithm and Mahalanobis distance, an Improved Distributed PCA-Based Outlier Detection Method (IDPCA) is proposed for the outlier detection in WSN. By exploiting the double detection mechanism, IDPCA greatly improves the detection performance of outliers and reduces the communication overhead in the network. Experimental results show that IDPCA can

achieve an outlier detection rate of 96 % - 97 % and maintain a false alarm rate about 2 %. Moreover, the method can trace the source of outliers and enable network managers to locate network problems quickly. In order to solve the problem that IDPCA scheme can not be well applied to non-linear data measurements detection, this paper proposes an improved distributed kernel principal components analysis based on Mahalanobis kernel function, which performs well in detecting non-linear ouliers by mapping the data measuments to a high-dimensional space. The Mahalanobis distance is proposed to measure the similarity of data vectors, by leveraging the correlation among the measured data vectors and training a proper kernel parameter, about 98% outlier detection rate and 2% false alarm rate can be achieved by IDKPCA model.

Key words: Wireless Sensor Network**,** Worm Compromised Nodes**,** Oulier Detection**,** Sequential Probability Ratio Test**,** Principal Components Analysis

目录

第一章 绪论

1.1 研究背景及意义

无线传感器网络(Wireless Sensor Network,WSN)因其能够部署在各种恶劣环境下完成任务，受到越来越多专家和学者的关注。它们往往用于执行人类很难执行的任务，如火山检测，生物危害检测和森林火灾防控等。除此之外，WSN 还广泛运用于战场监视、边境监控、核攻击和化学攻击检测、入侵检测等场景[1]。为了执行各种任务，需要在监测环境中部署基站和一组小型传感器设备，传感器设备间形成自组织网络，相互协作以感测与分配的任务相关联的现象，然后将感测数据发送到基站[2]。为了帮助传感器节点高效地执行任务，研究人员提出了各种网络服务和通信协议。目前，已经为网络服务提出了节点定位、区域覆盖、数据压缩和聚合等协议。

然而，WSN 通常以无人值守的方式部署，这些协议将会面临各种攻击，例如拒绝服务攻击、路由中断、虚假数据注入攻击和网络服务中断攻击[3-5]等。为了保护 WSN 免受各种攻击，许多学者给出了相应的解决方案。文献[6-7]提出了安全路由方案来减轻传感器网络中的路由中断攻击，文献[8-9]通过信任节点计算信任值来检测虚假数据注入攻击，文献[10]基于贝叶斯检测算法检测网络中的虚假数据注入攻击。安全数据聚合协议[11-15]用于抵御攻击者破坏传感器节点对数据的聚合。这些方案偏向于使 WSN 协议具有抵御攻击的能力，能够减轻恶意攻击对网络服务和通信协议的威胁，但是需要大量的时间和精力根据新类型攻击来增强协议的健壮性。此外，由于很难预测新类型的攻击，协议只有在被新类型的攻击破坏后才能增强。因此，我们需要尽早发现和移除攻击源，以降低网络的损失。各种攻击的主要来源来自被俘获的传感器节点，攻击者可以利用 WSN 无人值守的特性来俘获传感器节点，对网络进行任何恶意活动。蠕虫病毒能够在网络中快速传播，让攻击者轻而易举的获取整个网络的控制权，是攻击效率最高的一种方式。然而，WSN 中针对恶意蠕虫节点检测的方案并不成熟，迫切需要一种高效的检测方案来检测恶意蠕虫节点。

另一方面，WSN 在许多领域的应用通常需要高效、准确、实时的分析来做出决策，准确的分析和决策主要依赖于传感器数据的质量。在监测过程中，传感应用会产生大量数据，在处理海量数据时，一个值得关注的问题是异常值问题[16]。在 WSN 中，异常值不可避免地会出现在几乎所有的数据集中，原因有两个:监控区域中的“罕见”事件以及由于 WSN 资源有限、软件或者硬件故障和恶劣的部署环境导致的感应错误。在 WSN 中，环境条件可能会随着

时间的推移而变化，因此，预定义的异常数据检测模型不一定能够对未知数据进行异常检测。因此，异常数据检测中的一个关键点是以可接受的检测精度动态检测异常值，同时将通信开销和能耗降至最低[17]。除此之外，对于网络管理者而言需要根据异常值来迅速判断网络的问题所在，他们需要根据异常值的起因(恶意事件、节点内部错误、网络噪声)来做出决策[18]。

在数据挖掘、数据库，统计决策等领域中，异常数据检测方案已经非常完善。然而对于资源受限的 WSN，这些解决方案并不适用。WSN 由大量小型自组织节点组成，这些节点的 CPU、带宽和能量资源有限，其他领域的解决方案并不能直接应用，因为这些方案会给网络带来巨大的的通信损耗和计算损耗[19]。已有的 WSN 异常数据检测方案并不能在资源消耗和检测效率上得到折衷的效果，迫切需要寻找一种有效的异常数据检测方案来检测 WSN 中的异常数据并对异常数据进行溯源。

1.2 国内外研究现状

1.2.1 无线传感器网络恶意节点检测相关研究

为了检测 WSN 中的恶意节点，文献[19-23]提出了多种检测方案。文献[19-20]提出了基于信誉值的信任管理方案，该方案根据单个节点的行为来计算其信誉值，通过计算所得的信誉值来判断节点是否被攻击者俘获。该方案能够有效的检测恶意节点，但同时也存在相当大的节点误报，会使网络中的大量有效节点不可用。事实上，所有基于信誉值的检测方案都存在误报，当信任管理方案被重复应用于网络中的节点时，很大一部分良性的节点可能会被误认为是恶意节点，直接清除这些节点将带来非常昂贵的代价。

检测 WSN 中恶意节点的另一种方案是软件认证[21-23]。该方案的核心思想是：传感器节点通常会运行存储在闪存中的同一份可执行代码，邻居节点通过验证节点上运行的闪存代码的完整性来检测被破坏的节点。这种方案的恶意节点误报率很小，但是为了保证恶意节点的高检测率，需要每个传感器节点进行周期性的软件认证。因此，该方案会给 WSN 带来大量的通信资源开销和计算资源开销。

为了克服上述两种方案的缺点并结合两种方式的优点，文献[24]在软件认证方案的基础上结合节点区域信誉值，提出了一种基于区域信誉值的信任管理方案。该方案的核心思想是将整个传感器网络分为不同的区域，然后对每一个区域设置信誉值，通过序贯概率比检验(Sequential Probability Ratio Test,SPRT)来检测非信任区域。最后，在这一部分区域运行软件认证，检测恶意节点并移除。通过这种区域性检测方案，能够快速定位非信任区域并运行软件

认证方案，相比无条件的软件认证方案，该方案能够快速的检测网络中的异常节点并移除，同时能够大大减小软件认证的节点数量，减少网络的资源消耗。但是文献[24]提出的这种方案有一些极端情况，如果每个区域都存在不信任节点，也即每个区域都是不值得信任的区域，那么该情况下每个节点都需要进行软件认证，计算开销和通信开销也非常大。针对这种情况，文献[25]提出了仅仅基于非信任节点的软件认证方案，在保证恶意节点检测率的同时，大大降低了网络的计算资源消耗和通信资源消耗。

然而，基于区域信誉值的信任管理方案是针对静态恶意节点的，如果恶意节点在网络中处于移动状态，那么会造成大量恶意节点漏报。检测移动恶意节点的最简单方法是在网络部署前，在每个传感器节点中预分配其邻居节点列表，并让每个节点拒绝与其列表中未包含的节点通信。该方案能够有效地检测移动恶意节点，但是需要网络管理者在部署阶段做更多的工作，因为网络的拓扑结构必须事先确定。此外，由于使用自动部署时总是存在部署错误，因此实际情况下每个节点的邻居节点很可能与预分配的邻居列表不同，导致许多良性节点与邻居节点通信失败，导致网络的大范围瘫痪。文献[26]表明，使用这种检测方案会导致 WSN 中的传感器节点发生严重的网络连接问题。

文献[27-30]提出的位置区分方案可以高效地检测 WSN 中的移动恶意节点。在这些方案中，接收器基于接收信号强度来识别无线网络中发送者的位置变化。然而，攻击者可以在移动过程中关闭节点的无线电，每次移动足够远的距离，以防止任何一个接收器监听到恶意节点从两个不同的位置发送数据包。文献[31]提出了一种基于 SPRT 算法的分布式检测方案来检测和移除 WSN 中的移动恶意节点。该方案的核心思想是：静态传感器节点能够频繁的与邻居节点通信并建立邻居节点表，而移动恶意节点因为节点位置会实时变化并不会每次都和同一个节点进行通信，当节点发现其邻居表中有不与它通信的节点时，可以将该节点标记为可疑的移动恶意节点。该方案可以高效地检测 WSN 中的移动节点攻击。

文献[32-33]通过椭圆曲线加密算法对网络之间传递的数据进行认证，能够有效地防止恶意数据的注入。但是，如果一个节点的密钥材料被攻击者获取，那么攻击者可以通过该节点进行合法的网络通信，窃取 WSN 的信息，该攻击称之为副本节点攻击。副本节点攻击是指攻击者从捕获的节点中获取节点的通信密钥和节点的 ID，生成大量共享节点密钥材料和 ID 的攻击者控制的副本，然后将这些副本散布在整个网络中。副本节点由攻击者控制，但有密钥材料，使它们看起来像网络中的授权参与者。用于安全传感器网络通信的协议将允许副本节点与其他节点和基站创建成对的共享密钥，从而使节点能够加密、解密和认证其所有通信。

检测副本节点攻击的一个直接解决方案是通过为移动节点配备防篡改硬件来防止对手从移动节点中提取秘密密钥材料。然而，尽管防篡改硬件会使从捕获的节点中提取密钥材料变

得更加困难和耗时，但如果有充分的时间和专业知识，攻击者仍有可能在少数节点中绕过防篡改。攻击者可凭借单个节点生成许多副本，这意味着副本攻击相对于其他类型的恶意节点来说更加危险，因此基于软件的方案来保护移动传感器网络免受副本节点攻击非常重要。文献[34-35]针对静态传感器网络提出了几种基于软件的副本节点检测方案。这些方案使用的主要方法是让节点报告位置声明，以识别它们的位置。如果同一个 ID 的节点发送的位置报告中显示它们的位置不一致，那么判断该节点为一个副本节点。整个网络对该节点进行通信限制，移除副本节点。在静态网络中，该方案能够高效快速地检测副本节点，但是由于这种方法需要固定节点位置，因此当节点移动时，该方案不再适用。针对该缺陷，文献[36]提出了一种基于 SPRT 的副本节点攻击检测方案来检测移动传感器网络中的副本节点攻击。该方案的核心思想认为一个良性移动节点的移动速度永远不应该超过系统配置的最大速度。因此，只要我们采用误差率低的速度测量系统，良性移动传感器节点的测量速度至多是系统配置的最大速度。然而，副本节点的移动速度会比良性节点快得多，因为它们需要同时位于两个(或更多)不同的位置，因此它们的测量速度可能会远远超过系统配置的最大速度。通过该特性结合 SPRT 检验，该方案能够快速并准确的检测移动网络中的副本节点攻击。

然而，通过物理捕获大量传感器节点需要花费很多时间和精力，并且可能会使攻击者面临被发现的风险。对于攻击者来说，一个更好的选择是只捕获和破坏几个节点，然后使用自传播恶意代码(即蠕虫)来感染网络中的其他节点。通过简单地将受感染的节点重新引入网络，蠕虫会迅速传播并感染大量传感器节点，让攻击者获取整个网络的控制权。

尽管 WSN 中蠕虫节点的检测和预防十分重要，但是相关方面的研究却很少。由于传感器网络缺乏相应的硬件资源设施，已有的互联网蠕虫检测和预防方案[37-40]并不能直接应用于传感器网络。文献[37][39]中的签名检测算法通过对传感器节点发送的包进行签名认证，识别网络中的恶意蠕虫代码。该方案需要大量的计算开销来生成和维护签名，这不适用于资源受限的传感器网络。文献[38]介绍了互联网蠕虫检测的常见方案，这些方案中结合了贝叶斯网络模型、神经网络模型来做蠕虫识别，需要大量的计算资源开销并不适用于 WSN。杨等人[41]提出了基于软件多样性技术的方案来防止蠕虫传播，该方案的核心思想是将整个网络分成一系列的网格单元，然后在每一个节点中分配不同的闪存程序，当攻击者通过某个闪存程序的漏洞捕获某个节点之后，由于节点之间使用的是不一样的闪存程序，那么很大概率的情况下，蠕虫将无法进行传播。上述机制能够工作的前提条件是，每个节点的闪存程序漏洞是不一样的，否则无法阻止蠕虫节点的传播。文献[42]扩展了文献[41]中的思路，能够以较小的计算开销阻止相邻两个节点之间的蠕虫传播，进而阻断整个网络的蠕虫传播。然而，现实中很难使得闪存程序具有不同的漏洞，如果我们能够发现闪存程序的漏洞就可以主动修复它们，而不需要再

等待蠕虫节点来感染。另一方面，如果在两个或更多版本的代码中发现不同的漏洞，蠕虫代码可以利用所有漏洞进行编程。这种蠕虫可以首先感染一个节点，然后切换漏洞，感染相邻的一个节点，从而在网络中传播。文献[43]研究了当节点的活动状态周期由随机节点调度确定时，使用软件多样性技术对传感器节点预防蠕虫感染的影响。文献[44]提出了一种软件多样性技术结合基于角色的图像着色方案来检测蠕虫节点。这些方案都是基于软件多样性技术的，所以它们的缺陷也和文献[41-42]所提出的相同。如果蠕虫传播不容易被中断，那么可以通过检测蠕虫节点然后移除来保证传感器网络的安全。文献[45-49]引入了远程软件认证方案有效的检测蠕虫节点。远程软件认证方案的核心思想是在软件或者硬件的基础上证明节点程序的完整性，由于任何节点都可以针对其他节点进行软件认证，并且无需使用专用硬件即可检测出受损节点，因此远程软件认证方案在资源受限的传感器网络中检测蠕虫节点有很大的优势[49]。但是，如果周期性地使用软件认证方法来检测传感器网络中的恶意节点，那么会给网络带来大量的通信资源开销。文献[50]提出了一种适用于无线传感器网络的软件认证方案，其主要思想是让每一个节点在每一个时隙中随机选择一组节点来进行认证，当该节点收到这组节点发送的数据包时，对它们进行远程认证，从而检测被蠕虫感染的节点。尽管这种方法能够在一定程度上检测网络中的蠕虫节点，但是由于无法检测未包含在随机选择的集合中的感染节点导致检测能力下降。因此，寻找一种高效的蠕虫节点检测方案检测并剔除恶意蠕虫节点至关重要。

1.2.2 无线传感器网络异常数据检测相关研究

在 WSN 中，异常数据定义为与正常数据特征差别较大的数据[51]。低成本和低质量的传感器节点具有严格的资源限制，例如能量(电池功率)、内存、计算能力和通信带宽。有限的资源使得传感器节点产生的数据不够可靠和准确，特别是当电池电量耗尽时，产生错误数据的概率将迅速增长[52]。另一方面，传感器节点易受环境影响,大规模高密度无线传感器节点部署在恶劣的环境下(多达数百甚至数千个节点)，在这种环境中一些传感器节点不可避免地会出现故障，这可能会导致噪声、故障、丢失和冗余数据。此外，传感器节点容易受到恶意攻击，如拒绝服务攻击、黑洞攻击和窃听[53]等，在这些攻击下，传感器监测的数据将被攻击者篡改或窃取，导致网络数据的泄露或者大规模失真。

WSN 已经广泛运用于各种生活场景，例如农业灌溉，医疗手术，森林火灾防控等。这些场景需要根据监测的数据来进行实时的状态分析，因此要求传感器网络传递的数据值足够可靠。为了保证基站接收数据的可靠性，必须对 WSN 中的异常数据进行剔除。检测异常数据最

简单的方法就是对正常的监测数据集进行建模，然后将模型应用于新的监测数据，如果新到达的监测数据和所建模型的偏差超过一定的阈值，那么可以将该测量数据定义为异常数据进行丢弃。目前，已经有相当多的学者针对 WSN 中异常数据检测提出了解决方案，但是受限于传感器节点的资源限制，很多研究结果并不能在资源和检测效率上得到折衷的效果。另一方面，如何根据异常数据来追踪引起异常数据发生的原因这一问题亟待解决。本文旨在寻找一种有效的方案使得传感器网络在满足指定的异常检测率下，尽可能少地降低传感器节点消耗的资源，并对引起传感器节点数据异常的原因进行分析、溯源。

（1） 异常数据检测的相关研究

异常数据检测方案主要分为两种类型：集中式异常数据检测方案和分布式异常数据检测方案。在集中式异常数据检测方案中，各个节点将数据发送到基站统一处理，由基站进行数据的异常检测和剔除。这种方案会给整个网络带来巨大的通信开销，同时网络的故障率会大大提高，一旦基站因为数据处理崩溃或者直接被攻击者攻击，那么整个网络将崩溃。分布式异常数据检测方案中，数据的检测由各个节点自身完成，这种方案的优点是每个节点都能参与检测，网络的容错率高，能够大大降低节点之间的通信损耗。但是这种方案的缺点也很明显，每个节点单独运行检测算法会给节点的计算资源和内存资源带来消耗，加速传感器节点的能量耗尽，但是正常情况下，节点之间的通信能耗要远远大于计算能耗，所以主流的研究方向偏向于分布式检测方案。

根据是否需要标签训练样本，检测算法分为监督算法，半监督算法和无监督算法。监督算法和半监督算法训练模型的时候需要将训练数据按照标签分为正常数据和异常数据，而无监督算法不需要对数据进行分配标签，通过给定数据进行模型训练。在现实场景中要获取标签数据是相对困难的，会增加额外的工作量。文献[18]详细地归纳了机器学习算法在传感器网络异常数据检测中的运用，图 1.1 给出了传感器网络中异常数据检测方案的研究成果。

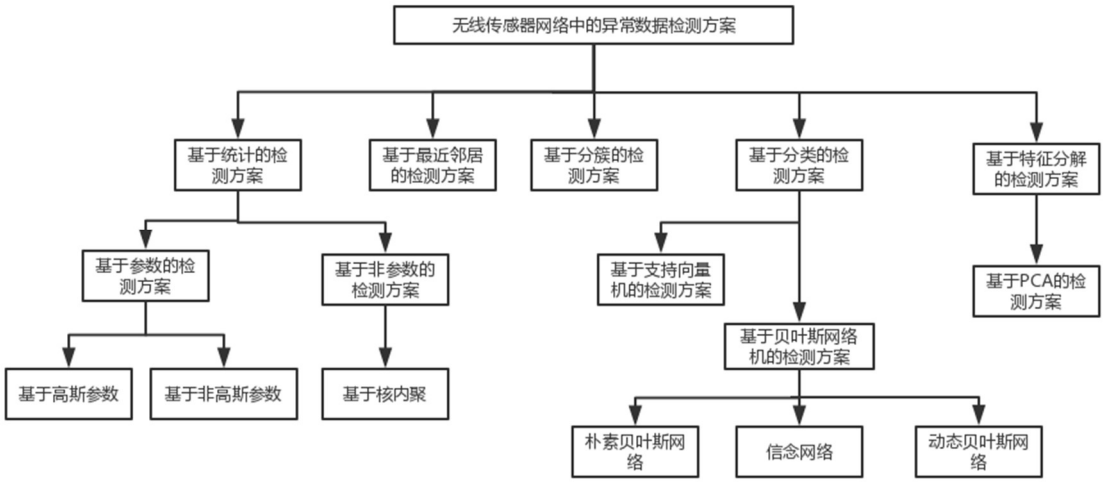


图 1.1 无线传感器网络异常数据检测方案综述

基于数据统计的异常检测方案是最早提出用于处理异常数据检测问题的方案。该类检测方案的核心是根据已知数据进行统计建模，生成一个概率模型表征数据的产生规律，如果一个数据值在该模型下的生成概率特别小，那么将该数据归为异常数据，否则归为正常数据，这是一类无监督的检测算法。文献[54]提出了一种基于高斯分布的概率统计模型来检测异常数据，该检测模型并没有利用节点数据的时域相关性因而异常数据的检测率较低。文献[55]提出了一种基于核函数的概率密度统计模型，该模型能够很好地检测高维异常数据，然而这种方法需要定义一个合适的门限值作为异常数据值的区分标准，找到该门限值非常难。如果能够建立合适的概率模型表征数据的生成规律，那么基于统计模型的检测方案能够很好地检测出异常数据。然而现实场景下，数据的统计分布模型是很难预估的，这是该方案最大的缺点。

基于最近邻居的方法是数据挖掘和机器学习社区中最常用的分析数据实例的方法。该方案使用明确的距离概念来计算两个数据实例之间的相似度。具体方案是通过计算当前节点的数据向量同邻居节点的数据向量的距离来判断当前节点数据是否为异常值，对于单个变量通常使用欧式距离描述，对于多元变量通常使用马氏距离来描述。文献[56-57]详细介绍了最近邻居方案，文献[58-59]使用最近邻居检测方案来检测异常数据，但是给网络带来了比较大的通信开销和计算开销。对于多元数据而言，反复计算节点数据之间的相似度是一个计算量比较大的工作，同时如何确定相似度的阈值也是一个复杂的工作，因此最近邻居检测方案在WSN 中并不常用。

基于聚类的方法是数据挖掘领域中用于将相似的数据实例分为同一个分组的常见方案。如果某个数据实例不属于某个分组，或者某个分组的大小明显小于其他分组，则它们被认定为异常数据值，通常使用欧式距离来衡量某个数据向量到中心向量的距离。文献[60]中使用 k

均值聚类的方法将训练数据集分成 k 类，对监测数据进行异常检测。该方案下异常数据的检测率是变化的，随着 k 值和每个组的组距变化，性能也大不相同，如何选取 k 值和分组组距使得检测模型的性能达到最佳非常困难，需要耗费大量的计算资源。

基于分类的异常检测方法是数据挖掘和机器学习领域中重要的学习算法。通过使用一组数据实例训练分类模型，并将一个未知实例分类到一个学习过的(正常/异常)类别中。无监督的分类算法不需要标签数据进行训练，一类非监督学习算法通过寻找正常值和异常值的边界来进行异常数据的检测。目前主流的分类算法主要包括两种类型：基于支持向量机(SVM)和基于贝叶斯网络的分类方法。文献[61-64]通过 SVM 算法去计算正常数据和异常数据的分界面（超平面或者超曲面，超椭圆面）进行数据的检测，这种方案的计算复杂度过高，而且需要选取合适的内核函数进行高维空间映射，实现比较复杂。不仅如此，这些文献中提出的 DQSSVM,CQSSVM,EOOD,EAOD 方案只有在各自特定的数据类型下才能保持高效地检测，例如 QSSVM 方案适用于按照球面分布的数据的检测，而 EOOD 方案适用于数据类型是椭球形分布的数据检测。文献[65-66]通过训练数据集来得到贝叶斯分类器，对数据进行检测。通过分类器能够很好地检测异常数据，但是训练合适的贝叶斯分类器需要大量的数据向量和计算资源开销。

以上的方案中仅仅考虑了传感器节点之间的空间相关性和节点内部数据的时域相关性，并没有考虑数据向量之间的维度相关性，这导致了多余的计算复杂度并且在一定程度上降低了异常数据检测效率。同时以上这些方案并没有考虑到数据维度较大的情形，这在分布式检测方案中会引入巨大的通信资源开销。最后，以上这些方案并没有很好地利用传感器节点的空间相关性来解决异常数据源的分析。

主成分分析（Principal Component Analysis,PCA）是数据降维的重要方法，广泛地运用于模式识别的领域。该方法的核心思想是找到一个变换使得坐标系旋转能够将数据的主要信息集中在少数几个维度上。基于 PCA 的异常检测模型不仅能够很好地处理低维数据向量而且也非常适用于高维的数据向量。文献[68]介绍了 PCA 算法在 WSN 中异常节点检测和异常数据检测中的运用，实验结果表明基于 PCA 算法的异常数据检测方案能够很好地检测异常数据，但是该方案基于基站来建立模型属于集中式检测方案，会给网络带来巨大的通信开销。文献[69]提出了一种基于固定组距的分布式 PCA 异常数据检测方案(DPCA)，该方案下每个传感器节点通过训练数据集进行 QR 分解将 R 矩阵发送给簇头节点计算全局的主成分值并计算判决门限值发送给各个节点。该方案的异常数据检测效率能够达到 94%以上，但是网络中的通信资源消耗依然很大。文献[70]在计算完主成分分量之后并不使用欧式距离来计算判决门限值而是使用非相似度来判定异常数据，该方案的异常数据检测率能够达到 90%以上，但是如何

获取最佳门限值是限制该方案的主要因素。基于 PCA 算法的检测模型在处理线性可分的数据时性能非常好，但是在非线性可分数据模型下往往不能找到最佳的映射坐标轴来反映原数据的特征，因此引入了基于核函数的主成分分析法(Kernel Principle Components Analysis,KPCA)将数据映射到高维空间中寻找最佳的映射坐标达到最好的检测效果。文献[71-72]表明 KPCA 检测方案在非线性系统中能够达到非常好的检测效果。文献[73]对 KPCA 方法进行了详细地描述，文献[74-75]给出 KPCA 方案在传感器网络中的应用并提出了集中式 KPCA 检测方案，该方案能够有效地检测网络中的异常数据，但是引入了大量的通信开销。

（2） 异常数据源检测的相关研究

由于 WSN 的资源限制和部署环境复杂，很难确定是什么引起了异常数据的发生。传统的异常数据检测方案并不区分异常数据发生的原因，这往往会导致网络的管理者无法获悉一些恶意事件的发生。传感器网络的异常数据主要由 4 个原因引起(1)部署环境中的噪声影响(2)传感器节点内部发生错误，比如节点的能量不足或者节点损坏(3)恶意事件发生(4)传感器节点的恶意攻击[76]。环境噪声对数据的影响是随机的，这意味着一个传感器节点检测到的数据是一个非连续的异常数据值，那么它有可能是一个噪声引起的。如果某个传感器节点从某个时隙开始监测的数据一直都是异常值，那么它有可能是节点内部异常引起的。如果一个组中的节点在连续的时隙内，节点产生的数据都是异常数据，那么有可能是网络中出现了恶意事件。文献[52]提出了一种 ODS 方案能够有效地检测恶意事件和恶意攻击，但是不能识别由节点损坏或者噪声引起的异常。文献[53]中提出了一种 OPTICS 的异常数据检测方案能够有效地区分异常数据源为恶意事件或者非恶意事件，但是不能确定非恶意事件发生的具体原因。综上所述，异常数据源的追溯至关重要。但是相关的研究却不能很好地区分引起异常数据产生的原因，本文提出的 IDPCA 方案能够很好地解决该问题。

1.3 研究内容及创新点

本文针对无线传感器网络中的恶意节点检测和异常数据检测提出了新的解决方案，论文的主要工作如下：

（1） 以往的无线传感器网络恶意节点检测方案中并没有很好地解决蠕虫节点感染问题。本文利用蠕虫节点传播的特性，以 SPRT 蠕虫检测模型为基础，结合偏向采样方案加速蠕虫节点的检测。同时为了更好地解决蠕虫节点的慢传播问题，结合随机值采样方案，提出了 SPRT-Biased-Random 蠕虫检测方案。方案能够在 5~18 个时隙内检测到蠕虫节点，并使得传感器网络最终受感染的节点维持在 2%~5%之间。

（2） 在 PCA 算法的基础上提出了一种改进型分布式 PCA 异常数据检测模型(Improved Distributed PCA-Based OutlierDetection Method, IDPCA)。通过充分利用传感器节点间监测数据的空间相关性和节点内数据的时域相关性我们提出双重检测机制（本地检测和全局检测）进行异常数据的检测和异常数据溯源。该模型充分考虑了数据维度之间的相关性，利用马氏距离来衡量数据间的相似度。对于网络中的异常数据检测率能够达到 96%-97%并且整个网络的通信资源相对以往的分布式检测方案要小，同时能够很好地识别异常数据发生的原因。

（3） 在 IDPCA 异常数据检测模型的基础上提出了一种基于马氏内核函数的主成分异常数据检测模型(Improved Distributed Kernel Principal Components Analysis Method,IDKPCA)，该方案能够很好地处理线性不可分的数据集。通过马氏内核函数将线性不可分的监测数据集一一映射，该模型能够很好的将数据在高维空间中进行区分。为了克服 IDPCA 异常数据检测模型中频繁地数据模型更新，引入艾宾浩斯遗忘曲线来实现固定窗口模型更新。实验表明 IDKPCA 异常数据检测模型能够达到近乎 98%的异常数据检测率和 2%的数据误报率，同时该模型下网络中的通信资源消耗趋近于常量级别，有利于延长整个传感器网络的寿命。

1.4 本文组织结构

本文由六个章节组成，各个章节的主要内容如下：

第一章介绍了研究背景和意义，简述了国内外对 WSN 恶意节点检测和异常数据检测的的研究现状，重点对蠕虫节点检测和异常数据检测技术进行了分析介绍。同时简单介绍了一下本文的研究方法和研究内容。

第二章详细地说明了本文用到的一些基础算法和网络的模型结构，如节点的预分配方案，SPRT 检测算法，蠕虫模型等。

第三章详细地阐述了所提出的 SPRT-Biased-Random 检测方案。该方案能够快速的检测网络中的蠕虫节点，并通过软件认证方案快速的将蠕虫节点移除，能够使得网络受到的影响最小。

第四章研究了 WSN 中异常数据的检测方案以及异常数据溯源。如何在合理利用传感器网络节点能量的同时，对网络中的异常数据和恶意事件进行准确的检测，以主成分分析法为基础提出 IDPCA 异常数据检测方案，能够在较小的通信资源开销下，准确的检测传感器网络中的异常数据和异常数据源。

第五章对第四章中 PCA 算法无法很好地处理线性不可分的数据这一缺陷做出弥补，提出基于马氏内核函数的分布式主成分异常数据检测模型 IDKPCA，能够很好地对线性数据和非线性数据进行异常数据的检测。同时引入了固定窗口模型更新策略，减小模型的更新频率，降低节点的计算资源消耗。

第六章对整篇文章的工作进行了概述和总结，针对文章中方案的不足之处提出了未来的研究方向。

第二章 无线传感器网络相关算法概述

WSN 已经广泛运用于各种生活场景，例如农业灌溉，医疗手术，森林火灾防控等[59]。这些场景需要根据监测的数据来进行实时的状态分析，因此要求传感器网络传递的数据值足够可靠。然而无线传感器网络部署在无人监控的野外，传感器节点很容易被攻击者捕获，攻击者通过捕获节点仿造数据值达到欺骗基站的目的，同时传感器节点能量有限，网络发送数据时极易受到环境中噪声的影响，造成数据的失真。无线传感器网络中的安全问题主要分为两种类型，第一种为传感器节点的异常行为检测即传感器节点遭受攻击，第二种类型为传感器节点异常数据的检测。本章节讨论文章中使用的一些模型结构和相关算法。

2.1 蠕虫传播模型

流行病传播模型能够很好地预估网络中受感染节点的数量，被广泛的运用于互联网蠕虫传播建模中，本文也同样采用该模型来模拟传感器网络中的蠕虫传播。文献[36]给出了离散时间下的简单流行病传播模型(在不同时隙下，而非连续时间)，式 2.1 给出了该模型下的蠕虫感染率：

$$\frac{dI_t}{dt} = \rho I_t (N - I_t) \tag{2.1}$$

蠕虫病毒的感染数量随时间的变化关系如式 2.2 所示，其中时间 t 的单位为一个时隙。

$$I_t = (1+\rho N) I_{t-1} - \rho I_{t-1}^2 \tag{2.2}$$

I_t 代表时刻 t 网络中受感染的节点数量，N 代表网络中的总节点数量，ρ 代表传感器节点彼此之间的感染率[36]，I_0 代表初始的蠕虫节点的数量，第 t 个时隙中受感染的节点数为 $I_t - I_{t-1}$。

2.2 序贯概率比检验

序贯概率比检验(Sequential Probability Ration Test,SPRT)是一种统计决策方案[76],该方案也被称作序列化假设检验。该方案和其他假设检验模型的区别在于序贯概率比检验的采样数在检验前是不固定的，它根据检验的结果动态增加采样样本数，换句话说在 SPRT 算法中采样的样本数目是随机的。这一特点使得 SPRT 检测算法能够在满足给定的误报率和漏报率的条件下更快的结束检测。文献[6-7]中将 SPRT 检测算法描述成具有上限值和下限值的一维随机游

走策略，在该策略开始时先定义空假设对应下限值，定义选择性假设对应上限值。SPRT 算法在上下限区间的某个值开始逐渐向上限值或者下限值靠拢，如果新到达的样本值使得 SPRT 计算的值低于下限值那么 SPRT 结束检验，接受空假设。如果新到达的样本值使得 SPRT 计算的值大于上限值，那么 SPRT 结束检验接受选择性假设。否则，需要再增加一个样本值进行检验。

以故障半导体设备检验为例，详细阐述 SPRT 检测方案。在该情形中我们认为故障机器产生故障半导体的概率要远远大于非故障机器产生故障半导体的概率。令 o_i 为设备 E 生成的第 i 个半导体，o_i 即为 SPRT 算法中的一个样本，令 S_i 为随机伯努利变量，定义为式 2.3 所示：

$$S_i = \begin{cases} 1 \ \ if \ o_i \ is \ defective \\ 0 \ \ if \ o_i \ is \ indefective \end{cases} \tag{2.3}$$

伯努利分布为 1 的概率定义为：$\Pr(S_i = 1) = 1 - \Pr(S_i = 0) = \lambda$。如果 λ 小于一个预定义的门限值 λ'，那么该设备很有可能不是一个故障设备，反之如果 $\lambda > \lambda'$ 那么该设备很有可能是一个故障设备。判断半导体设备是否为故障设备的问题可以归结为 $\lambda < \lambda'$ 和 $\lambda \geq \lambda'$ 的假设检验问题，然而假设检验的结果有可能出错，我们需要指定能够承受的最大容错率。因此将上面的假设检验模型修改为 $\lambda < \lambda_0$ 时接受空假设，当 $\lambda > \lambda_1$ 时接受选择性假设。当 $\lambda < \lambda_0$ 时如果我们接受选择性假设那么定义为误报，当 $\lambda > \lambda_1$ 时我们接受空假设那么定义为漏报。我们通过自定义配置最大容错的误报率为 α'，最大容错的漏报率为 β'。

为了进一步理解 SPRT 算法如何检测，我们演示设备 E 如何通过 n 个采样样本进行决策。我们首先假设设备没有故障为空假设 H_0，假设设备有故障为选择性假设 H_1。定义 L_n 为 n 个采样样本下的对数概率比，定义如式 2.4 所示：

$$L_n = \frac{\Pr(S_1, S_2, ...S_n \mid H_1)}{\Pr(S_1, S_2, ...S_n \mid H_0)} \tag{2.4}$$

根据对数概率比 L_n，SPRT 检测算法接受假设 H_0 和假设 H_1 的规则如式 2.5 所示：

$$\begin{cases} L_n <= \ln\dfrac{\beta'}{1-\alpha'}: \text{接收}H_0\text{假设终止}SPRT\text{算法} \\ L_n >= \ln\dfrac{1-\beta'}{\alpha'}: \text{接受}H_1\text{假设终止}SPRT\text{算法} \\ \ln\dfrac{\beta'}{1-\alpha'} < L_n < \ln\dfrac{1-\beta'}{\alpha'}: \text{根据新的样本值继续}SPRT\text{算法} \end{cases} \tag{2.5}$$

2.3 静态无线传感器网络模型

本小节将介绍静态传感器网络的分簇结构和位置分布函数。

本文提出的方案利用了传感器网络的分簇结构[77]，该结构示意图如图 2.2 所示。如该图所示，每一个集群拥有一个集群头和若干个邻居节点，集群头能够和邻居节点间通信也能和其余的集群头节点(包括基站)进行数据通信，基站监控和管理着整个传感器网络。传感器节点的异常数据检测需要经过双层节点检测即节点本身的局部检测和集群头的全局检测，通过集群头节点来寻找异常数据发生的原因，如果节点遭受了恶意攻击或者监控区域发生了恶意事件，那么集群头向基站发出警报，基站做出相应的处理。除此之外，集群头节点还能够进行数据聚合，重复数据剔除，节省与基站间的通信带宽。

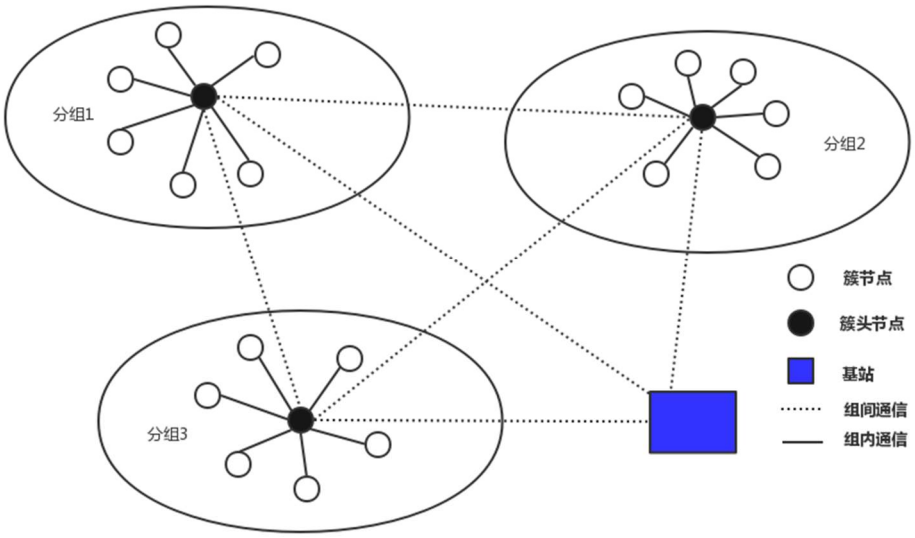


图 2.2 网络的分簇结构

将传感器节点按照分组进行部署，采用的分组策略是将节点按照二维高斯分布分散在区域中，所满足的二维高斯分布密度函数满足式 2.6。其中(x_g, y_g)是每个传感器分组的组头节点，σ为标准差。

$$f(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{(x-x_g)^2+(y-y_g)^2}{2\sigma^2}} \tag{2.6}$$

2.4 主成分分析与核主成分分析

主成分分析（Principal Component Analysis,PCA）是降维的重要方法，该方法的核心思想是找到一个变换使得坐标系旋转能够将数据的主要信息集中在少数几个维度上。每一个主成分都是数据在某一个方向上的投影，在不同的方向上这些数据方差的大小由其特征值决定。一般会选取最大的几个特征值所在的特征向量，这些方向上的信息丰富，能够包含数据向量的绝大部分信息。由[78]我们可知求解最大主成分分量即求解最优化问题：$v=\underset{v\in R^D,\|v\|=1}{\arg\max} v^TCv$，其中$C$为向量间的协方差矩阵，通过拉格朗日乘数法有：

$$f(v,\lambda)=v^TCv-\lambda(v^Tv-1) \tag{2.7}$$

式 2.7 对v,λ求导有：

$$\begin{aligned}&\frac{\partial f}{\partial v}=2Cv-2\lambda v=0 \quad Cv=\lambda v\\&\frac{\partial f}{\partial \lambda}=v^Tv-1=0 \quad ||v||=1\end{aligned} \tag{2.8}$$

所以求解主成分分量相当于求解$Cv=\lambda v$，从而得到λ的特征值对角阵和对应特征向量也即新的投影坐标方向。假设我们的训练样本是$X^T=[x_1,x_2,x_3.....x_N]$，每个样本值$x_i$是$d$维向量，那么训练样本的协方差矩阵$C$为式 2.9 所示，我们需要求解该矩阵的特征向量矩阵。

$$C=\frac{1}{N}\sum_{i=1}^{N}x_ix_i^T=\frac{1}{N}(x_1,x_2,.....x_N)\begin{pmatrix}x_1\\x_2\\....\\x_N\end{pmatrix}=\frac{1}{N}X^TX \tag{2.9}$$

在核主成分分析(Kernel PrincipalComponents Analysis,KPCA)方法中，认为原有数据有更高的维数，可以在更高维的空间（Hilbert Space）中做 PCA 分析（即在更高维空间里，把原始数据向不同的方向投影）这样做的优点在于：对于在低维空间中难以线性分类的数据点，我们有可能再更高维度上找到合适的高维线性分类平面。方案的核心还是求解协方差矩阵C的特征值和特征向量，但是协方差矩阵是高维空间下的协方差矩阵，具体可参考文献[78].

$$C=\frac{1}{N}\sum_{i=1}^{N}\phi(x_i)\phi(x_i^T)=\frac{1}{N}[\phi(x_1),\phi(x_2),.....\phi(x_N)]\begin{bmatrix}\phi(x_1^T)\\ \phi(x_2^T)\\ .....\\ \phi(x_N^T)\end{bmatrix} \tag{2.10}$$

2.5 本章小结

本章介绍了文章中所使用的一些网络模型和算法，为后文的方案奠定了一定的的技术基础。

第三章 基于 SPRT 算法的恶意蠕虫节点检测方案

3.1 引言

在 WSN 中，攻击者可以很容易的通过物理手段来俘获传感器节点，然后通过俘获的节点发动各种攻击。然而寻找和俘获大量的传感器节点需要大量的时间和精力，同时会让攻击者很容易暴露。因此，对于攻击者而言，更好的选择是通过俘获少数几个节点，向这些节点注入自我传播的恶意代码（即蠕虫）来感染其余节点。通过简单的方式，将受感染的节点重新引入网络，蠕虫可以在网络中迅速传播，感染网络中绝大部分的良性节点，让攻击者获取网络的控制权。

蠕虫病毒传播是传感器网络安全的一个主要问题。正如在互联网中看到的那样，蠕虫可以迅速传播并且造成巨大的破坏。尽管检测蠕虫节点对于 WSN 安全至关重要，但是相关方面的研究却很少。已有的互联网蠕虫检测方案需要大量的计算资源开销，并不适用于与节点能量受限的 WSN。杨等人提出了基于软件多样性技术的方案来防止蠕虫传播[41]，该方案的核心思想是将整个网络分成一系列的网格单元，然后在每一个节点中分配不同的闪存程序，这样的话，当攻击者通过某个闪存程序的漏洞捕获某个节点之后，由于节点之间使用的是不一样的闪存程序，那么很大概率的情况下，蠕虫将无法进行传播。上述机制能够工作的前提条件是，每个节点的闪存程序的漏洞是不一样的，否则无法阻止蠕虫节点的传播。文献[45-49]引入了远程软件认证方案有效的检测蠕虫节点。远程软件认证方案的核心思想是在软件或者硬件的基础上证明节点程序的完整性，由于任何节点都可以针对其他节点进行软件认证，并且无需使用专用硬件即可检测出受损节点，因此远程软件认证方案在资源受限的传感器网络中检测蠕虫节点有很大的优势。文献[50]提出了一种适用于 WSN 的软件认证方案，其主要思想是让每一个节点在每一个时隙中随机选择一组节点来进行认证，当该节点收到这组节点发送的数据包时，对它们进行远程认证，从而检测被蠕虫感染的节点。尽管这种方法能够在一定程度上检测网络中的蠕虫节点，但是由于无法检测未包含在随机选择的集合中的感染节点导致检测能力下降。

为了解决以上方案中的缺陷，高效地检测 WSN 中的蠕虫节点，本章提出了一种基于 SPRT 算法的蠕虫传播检测方案 SPRT-Biased-Random。该方案结合随机采样策略和偏向采样策略，能够大大的提高蠕虫节点的检测效率。该方案基于以下事实：蠕虫节点以逐跳方式传播，会产生一条链式的通信链路, 因此当一个蠕虫节点传播时可以观测到一条“蠕虫链”慢慢增长，

该链路连接了传感器网路中的多个节点。相较而言，良性节点之间的通信模式偏向于多对一的通信模式，多个数据源节点向一个数据聚合器发送数据，在这种模式下很难观测到“链式”通信模式，因此在正常的网络通信模式中我们会很难发现包重传现象。基于以上理论，SPRT 检测方案以网络中是否有包重传现象作为采样样本，根据样本类型动态地配置最低阈值和最高阈值。定义空假设为蠕虫没有在网络中传播，选择性假设为蠕虫在网络中传播，当 SPRT 检测到当前的网络中“包重传”数量低于最低阈值时，接受空假设。当“包重传”现象数量高于最高阈值时，接受选择性假设。

本章提出的 SPRT-Biased-Random 蠕虫传播检测方案的优势在于能够快速地检测出网络中的蠕虫节点，该方案相对于其他方案更加健壮可靠，对基本上的蠕虫类型都有效。理论分析和仿真实验表明该方案能够高效地检测 WSN 中的蠕虫节点，极大地限制攻击者。实验结果表明，该方案能够在 5~18 个时隙中检测出网络中的所有蠕虫节点，并限制网络最终被蠕虫感染的节点在 2%-5%之间。本章的内容主要包括模型的假设，模型的具体描述，模型的分析和仿真。

3.2 网络拓扑结构和攻击者模型

本小节详细介绍 SPRT-Biased-Random 方案下 WSN 的拓扑结构和攻击者模型。

（1）网络拓扑结构

假设传感器网络是静态网络，其中的传感器节点在部署完成之后位置不再发生改变，其网络结构如图 2.1 所示。同时节点之间是双向链路，能够彼此通信，彼此监听。一个传感器节点要被蠕虫感染，需要接收多个数据包。在当前的传感器网络中，网络分组包的大小为 28 比特，一个自传播的恶意蠕虫代码至少为 1024 比特，因此想要注入一个恶意的代码，需要发送多个数据包。根据文献[79]中的实验结果，当蠕虫代码在网络中传播时需要大约 50-100 的分组来传送数据包。假设正常传感器节点之间不会彼此发送大量的数据包，因为如果节点之间频繁的通信，会使得节点的能量快速的消耗，缩短整个网络的寿命。

（2）攻击者模型

假设攻击者能够捕获 WSN 中的一些传感器节点，并将这些节点设置为蠕虫病毒的发起者在网络中传播蠕虫。假设一个传感器节点只有两种状态:易感染和已感染。所有传感器节点最初都处于易感染状态，除了已感染的蠕虫发起者。一旦易感染节点被蠕虫病毒感染，它的状态就会变成已感染。我们不仅要模拟蠕虫节点传播的速度，同时还要模拟蠕虫如何通过网络传播。假设攻击者采用逐跳传播策略，每个感染节点将蠕虫传播到其相邻的易感染节点。

WSN 通常使用本地化协议进行聚类、数据聚合和其它活动，任意多跳对等通信在 WSN 中是非常少见的。如果蠕虫不是以邻居到邻居的方式传播，将表现出与正常网络流量截然相反的特征，那么蠕虫节点将很容易被检测到。因此，为了降低被检测到的机率，攻击者将依赖逐跳传播策略，使得蠕虫病毒传播看起来更像是正常的本地流量。然而，当这种逐跳传播策略与流行病模型一起使用时，可能无法满足给定时隙内感染节点数的增长，因为感染节点在某个时隙内的邻居节点可能已经都成为了感染节点。在这种情况下，我们假设感染节点通过从网络中随机选择敏感节点并将蠕虫传播给它们来维持感染率。

3.3 SPRT-Biased-Random 蠕虫节点检测方案

本小节详细地阐述 SPRT-Biased-Random 蠕虫节点检测方案。该方案的核心思想在于：蠕虫节点的通信模式异于正常节点的通信模式，会在网络中形成一条通信“链”。

考虑这样一个场景，蠕虫节点 A 向易感染节点 B 发送大量的蠕虫包进行感染，当 B 节点被感染了之后，它会将蠕虫包传递给 C 节点进行感染。如果对 A 发送到 B 的蠕虫包和 B 发送到 C 的蠕虫包进行抽样检测的话，不难发现肯定有包是重复发送的。定义这种传播模式为“链”,“链”和多跳通信模式的区别在于“链”的每两个节点都是一组源节点和目的节点，而多跳通信则是一组源节点和目标节点通过多个中间节点进行通信。图 3.1 描述了“链”和多跳模式通信的区别，其中“A-B-C 链“有两组源节点和目的节点，A-B 通过三个中间节点进行通信，属于多跳通信模式。

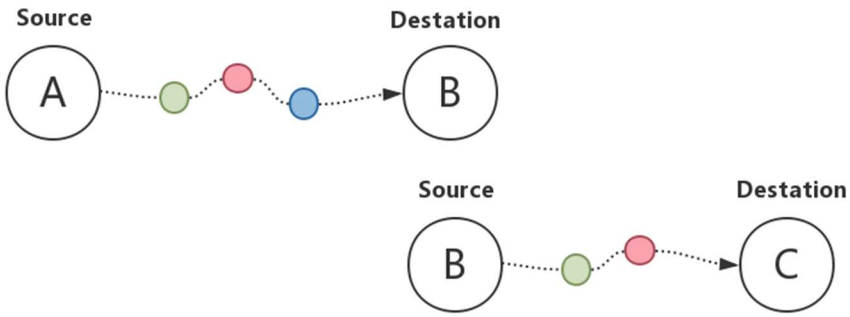


图 3.1 多跳通信模式和“链“通信模式示意图

3.3.1 网络预处理

在 WSN 部署前需要对每一个传感器节点分配唯一的 ID 号同时分配私钥进行节点之间的通信。通过密钥管理方案可以防止源节点伪造，当一个节点接收到另一个节点发来的分组包

时，首先通过密钥验证节点之间是否可以通信，如果认证不通过那么丢弃分组，通过这种方式可以防止蠕虫病毒仿造节点 ID 增加蠕虫检测的误报率。在节点部署完成之后，每个节点需要找到自己的邻居节点并以 P_d 的概率周期性地将自己选举为监控节点，因此一个组中的节点有可能依次成为监控节点。通过周期性地选举监控节点可以让攻击者无法精确掌握监控节点因而无法准确攻击，另一方面可以让每个节点依次成为监控节点，节约每个节点的能量。

3.3.2 通信模式收集存储

每当节点 u 接收到来自节点 v 的数据包时，节点 u 检查分组包的目的地址是不是自身，节点 v 是不是自身节点的邻居节点，如果满足上述条件，那么节点 u 将分组包的源节点 ID 和目标节点 ID 以概率 P_f 广播发送给它的邻居节点。将 $< pkt.sourceID, pkt.destinationID >$ 定义为通信模式，u 节点的邻居节点在接收到通信模式的时候，如果该邻居节点是监测节点那么该节点接收并保存该通信模式，否则丢弃该通信模式。包预处理(Packet Preprocessing Unit,PPU)的伪代码如表 3.1 所示,该算法工作在 MAC 层。

表 3.1 分组包预处理单元算法

输入：接收到的数据包 packet
If pkt.destination==u and pkt.source==u's neighbor then
Broadcast $< pkt.sourceID, pkt.destinationID >$ to neighbors with probability P_f

每个监测节点 w 以一定概率接收邻居节点广播的通信模式，进行蠕虫检测。更进一步，节点 w 将整个检测时域分成一系列的时隙，在每一个时隙中保存通信模式信息。每当节点 w 接收到一个通信模式 (s_i, d_i)，首先检测 s_i 和 d_i 是否是其邻居节点，如果是的话节点 w 在内存中存储 (s_i, d_i)，否则将其丢弃。接着监测节点 w 检测内存中是否已经存在 (s_j, d_j)，其中 $d_i = s_j$ 或者 $s_i = d_j$，那么可以组合成融合通信模式 (s_i, d_i, d_j) 或者 (s_j, d_j, s_i)，同时对计数器 M 进行加一操作。其中计数器 M 用于统计融合通信模式的合成次数，在每一个时隙开始时 M 初始化或重置为 0，每当有通信模式融合计数器加一。我们以节点中是否存在融合通信模式来作为数据包重复传输的重要理论依据。在正常的网络中，监测节点 w 基本上不可能产生融合通信模式，因此 M 计数器很大概率的值为 0。而在蠕虫节点传播下监测节点 w 会有大量的融合通信模式产生，因此 M 的值至少大于 1。因此每一个时隙中 M 的值都可以作为节点 w 的邻居区域是否有蠕虫传播的依据。在每一个时隙中，如果 M=0，我们就直接接受空假设 H_0 网络中没

有蠕虫传播，如果 M>1 那么我们根据 M 值的大小选取 M 个采样值来加速接受H_1网络中有蠕虫传播，我们把该方案定义为偏向采样方案。偏向采样方案的优势在于，能够产生更多的样本加速 SPRT 算法接受H_1假设，因此能够快速的监测到网络中的蠕虫传播区域。

3.3.3 蠕虫节点检测

SPRT 算法是一种统计决策算法，它并不预先规定观测样本群的数目，而是在检测过程中不断地增加样本数，一直到满足某个门限值时终止算法。假设蠕虫没有在网络中传播为空假设H_0，蠕虫在网络中传播为选择性假设H_1。本节首先结合偏向采样方法定义 SPRT 蠕虫检测方案，假设M_k为第k个时隙时监测节点计数器 M 的值。通过M_k定义伯努利随机变量A_k如式 3.3 所示。

$$\begin{cases} A_k = 0 \quad if\ M_k = 0 \\ A_k, A_{k+1}, A_{k+2}, \ldots\ldots A_{k+M_k-1} = 1 \ \ if \ \ M_k > 0 \end{cases} \tag{3.3}$$

假设伯努利随机变量成功的概率为γ，也即网络中有融合通信模式出现的概率为：

$$\gamma = \Pr(A_k = 1) = 1 - \Pr(A_k = 0) \tag{3.4}$$

因为假设H_0和假设H_1在整个样本空间中是独立的，所以我们得到式 3.5：

$$\Pr(A_k) = \Pr(A_k \mid H_0) \times \Pr(H_0) + \Pr(A_k \mid H_1) \times \Pr(H_1) \tag{3.5}$$

如果单独考虑$\Pr(A_k = 1)$的话，可得式 3.6。由该式我们可以推出$\Pr(A_k = 1 \mid H_0)$随着$\Pr(A_k \mid H_1) \times \Pr(H_1)$减小而增大，$\Pr(A_k = 1 \mid H_1)$随着$\Pr(A_k \mid H_0) \times \Pr(H_0)$的减小而增大，这意味着蠕虫节点的检测率越高那么漏报率就越低。给定预定义的两个门限值γ_0和γ_1，如果监测节点运行 SPRT 算法检测到$\gamma < \gamma_0$那么网络中很可能没有蠕虫传播，如果检测到$\gamma > \gamma_1$那么很有可能网络中发生了蠕虫传播。所以判断传感器网络中是否有蠕虫传播可以归结成一个假设检验问题，问题的空假设为H_0，选择假设为H_1。

$$\begin{aligned} \Pr(A_k = 1 \mid H_0) &= \frac{\Pr(A_k = 1) - \Pr(A_k = 1 \mid H_1) \times \Pr(H_1)}{\Pr(H_0)} \\ \Pr(A_k = 1 \mid H_1) &= \frac{\Pr(A_k = 1) - \Pr(A_k = 1 \mid H_1) \times \Pr(H_1)}{\Pr(H_1)} \end{aligned} \tag{3.6}$$

基于该问题的描述，给出检测节点w如何通过n个样本结合 SPRT 算法做出决策，其中

A_k 被视为一个采样样本，SPRT 的对数概率比 R_n 计算方法如式 3.7 所示：

$$R_n = \ln \frac{\Pr(A_1,.....A_n \mid H_1)}{\Pr(A_1,.....A_n \mid H_0)} \tag{3.7}$$

因为每个时隙间产生融合通信模式是互相独立的，假设每个样本 A_k 之间是独立同分布的，那么 R_n 可以改写为式 3.8：

$$R_n = \ln \frac{\prod_{k=1}^{n} \Pr(A_k \mid H_1)}{\prod_{k=1}^{n} \Pr(A_k \mid H_0)} = \sum_{k=1}^{n} \ln \frac{\Pr(A_k \mid H_1)}{\Pr(A_k \mid H_0)} \tag{3.8}$$

定义 δ_n 为 n 个采样样本中 $A_k = 1$ 的情况，$\gamma_0 = \Pr(A_k = 1 \mid H_0), \gamma_1 = \Pr(A_k = 1 \mid H_1)$，可以推出式 3.9。

$$R_n = \delta_n \ln \frac{\gamma_1}{\gamma_0} + (n - \delta_n) \ln \frac{1-\gamma_1}{1-\gamma_0} \tag{3.9}$$

基于对数概率比 R_n, SPRT 的运算规则如式 3.10 所示：

$$\begin{cases} R_n <= \ln \dfrac{\beta'}{1-\alpha'}: \text{接收} H_0 \text{假设终止} SPRT \text{算法} \\ R_n >= \ln \dfrac{1-\beta'}{\alpha'}: \text{接受} H_1 \text{假设终止} SPRT \text{算法} \\ \ln \dfrac{\beta'}{1-\alpha'} < R_n < \ln \dfrac{1-\beta'}{\alpha'}: \text{根据新的样本值继续} SPRT \text{算法} \end{cases} \tag{3.10}$$

其中 α' 代表用户配置的最大容许的误报率，β' 代表用户配置的最大容许的漏报率，根据式 3.9，令 $l_0(n)$ 和 $l_1(n)$ 分别为 3.11 和 3.12 所示：

$$l_0(n) = \frac{\ln \dfrac{\beta'}{1-\alpha'} + n \ln \dfrac{1-\gamma_0}{1-\gamma_1}}{\ln \dfrac{\gamma_1}{\gamma_0} - \ln \dfrac{1-\gamma_1}{1-\gamma_0}} \tag{3.11}$$

$$l_1(n) = \frac{\ln \dfrac{1-\beta'}{\alpha'} + n \ln \dfrac{1-\gamma_0}{1-\gamma_1}}{\ln \dfrac{\gamma_1}{\gamma_0} - \ln \dfrac{1-\gamma_1}{1-\gamma_0}} \tag{3.12}$$

那么 SPRT 算法的运算规则可以修改为式 3.13 所示：

$$\begin{cases} \delta_n <= l_0(n): & \text{接收}H_0\text{假设终止}SPRT\text{算法} \\ \delta_n >= l_1(n): & \text{接受}H_1\text{假设终止}SPRT\text{算法} \\ l_0(n) < \delta_n < l_1(n): & \text{根据新的样本值继续}SPRT\text{算法} \end{cases} \tag{3.13}$$

如果 SPRT 算法接受了假设 H_0，那么监测节点重启 SPRT 算法继续监测。如果接受假设 H_1 那么监测节点 w 向邻居节点发送广播说明本地可能有节点被蠕虫感染。然后监测节点和其邻居节点使用软件认证方案[49,80]分别对其邻居节点进行蠕虫检测，在检测到蠕虫节点之后使网络中的节点不再与该节点进行通信。

虽然基于偏向采样的 SPRT 蠕虫检测算法能够快速准确地检测蠕虫攻击，但是在一些特殊的场景下蠕虫检测将会失败，比如说蠕虫病毒可以动态地改变其在网络中的节点感染率快慢，导致 SPRT 算法出现大量的漏报。具体来说，如果一个蠕虫节点感染邻居节点的速率较慢，那么在网络中就不会频繁地出现融合通信模式，这样一来很多蠕虫节点会被当做是良性节点。为了解决这个问题，我们需要动态地改变采样样本的门限值。由上面给出的定义我们知道 γ_0 代表网络中没有蠕虫病毒传播，但是监控节点出现了融合通信模式的概率。该概率也即代表了传感器节点在判断网络中没有蠕虫病毒传染的时候能够最大承受的监控节点有融合通信模式发生的概率，如果蠕虫病毒能够掌握这个信息，那么它可以动态地改变自己的成对感染率，将自己伪装成良性节点。因此我们引入随机参数 κ 动态修改该概率值，即将 γ_0 替换成 γ_0^{κ}，很明显参数 κ 的值应该大于 1 也即 $\gamma_0^{\kappa} < \gamma_0$，这意味着我们对样本值为假设 H_0 的样本信任权重较低，接受假设 H_0 的概率降低，能够在慢蠕虫传播的情况下加速 SPRT 决策向假设 H_1 靠近。将该方案定义为 SPRT-Biased-Random 方案，如果 SPRT 决策接受假设 H_0 那么网络中基本不可能出现融合通信模式(由良性节点构成融合通信模式的概率非常小)，这样的话即使是慢蠕虫传播也能检测出来。我们假设参数 κ 服从 $[1,\theta_{\max}]$ 的均匀分布，在每一个时隙运行 SPRT 算法时随机选取 κ 值，使得攻击者无法动态改变蠕虫的成对感染率。在该假设下 SPRT 的对数概率比扩展为式 3.14 所示：

$$R_n^{'} = \varphi_n \ln\frac{\gamma_1}{\gamma_0^{\kappa}} + (n-\varphi_n)\ln\frac{1-\gamma_1}{1-\gamma_0^{\kappa}} \tag{3.14}$$

因此 SPRT 的修改规则相应的修改为：

$$\begin{cases} \delta_n <= L_0(n): & \text{接收}H_0\text{假设终止}SPRT\text{算法} \\ \delta_n >= L_1(n): & \text{接受}H_1\text{假设终止}SPRT\text{算法} \\ L_0(n) < \delta_n < L_1(n): & \text{根据新的样本值继续}SPRT\text{算法} \end{cases} \tag{3.15}$$

其中 $L_0(n), L_1(n)$ 如式 3.16，3.17 所示：

$$L_0(n) = \frac{\ln \frac{\beta'}{1-\alpha'} + n \ln \frac{1-\gamma_0^{\kappa}}{1-\gamma_1}}{\ln \frac{\gamma_1}{\gamma_0^{\kappa}} - \ln \frac{1-\gamma_1}{1-\gamma_0^{\kappa}}} \tag{3.16}$$

$$L_1(n) = \frac{\ln \frac{1-\beta'}{\alpha'} + n \ln \frac{1-\gamma_0^{\kappa}}{1-\gamma_1}}{\ln \frac{\gamma_1}{\gamma_0^{\kappa}} - \ln \frac{1-\gamma_1}{1-\gamma_0^{\kappa}}} \tag{3.17}$$

3.4 安全性分析

本节首先分析了 SPRT-Biased-Random 方案的蠕虫节点检测率，接着给出当 SPRT 算法终止时传感器网络中受感染节点的数量。

3.4.1 蠕虫节点检测率

假设 α 和 β 分别代表实际场景下蠕虫节点的误报率和漏报率。α 定义为 SPRT 接受假设 H_1 而实际情况是 H_0 的概率，β 定义为 SPRT 接受假设 H_0 而实际情况是 H_1 的概率，根据文献[72]有如下关系式：

$$\alpha <= \frac{\alpha'}{1-\beta'}, \quad \beta <= \frac{\beta'}{1-\alpha'} \tag{3.18}$$

α' 代表用户配置的最大容许的误报率，β' 代表用户配置的最大容许的漏报率。由式 3.18 可得单个检测节点的蠕虫检测率的下限值为：

$$DR = 1-\beta >= \frac{1-\alpha'-\beta'}{1-\alpha'} \tag{3.19}$$

由 3.19 可知，α' 和 β' 配置的值越小，单个节点的检测率就越高。如果网络中有 N 个监测节点，那么传感器网络的蠕虫检测率如式 3.20 所示：

$$DR(N)=1-\beta^{N}>=1-(\frac{\beta^{'}}{1-\alpha^{'}})^{N} \tag{3.20}$$

接着，我们探究监测时隙个数对单个节点蠕虫检测率的影响。在每个检测时隙，我们首先随机选取 κ 值来防止蠕虫节点的慢传播，针对单个检测节点我们提出定理 1。

定理 1：假设在前 r 个时隙检测中选取的 κ 值分别是 $\kappa_1,\kappa_2,.....\kappa_{r-1},\kappa_r$，同时假设 SPRT 算法终止时检测了 n 个样本，那么蠕虫节点在第 $r+1$ 个时隙前检测到的概率 $P_r=1-e^{-\sum_{i=1}^{r}P_S^i}$。其中 P_S^i 为每一轮时隙中蠕虫未被检测到的最大概率，如式 3.21 所示：

$$P_S^i=1-\frac{\ln\frac{1-\beta^{'}}{\alpha^{'}}+n\ln\frac{1-(\gamma_0)^{\kappa_i}}{1-\gamma_1}}{n(\ln\frac{\gamma_1}{(\gamma_0)^{\kappa_i}}-\ln\frac{1-\gamma_1}{1-(\gamma_0)^{\kappa_i}})} \tag{3.21}$$

证明：在每一轮时隙中，SPRT 算法在 $\delta_n>=L_1(n)$ 时接受 H_1 假设，检测到蠕虫传播。因此在每一轮的时隙中，蠕虫不被检测到的概率为：

$$P_S^i=\frac{n-L_1(n)}{n}=1-\frac{\ln\frac{1-\beta^{'}}{\alpha^{'}}+n\ln\frac{1-(\gamma_0)^{\kappa_i}}{1-\gamma_1}}{n(\ln\frac{\gamma_1}{(\gamma_0)^{\kappa_i}}-\ln\frac{1-\gamma_1}{1-(\gamma_0)^{\kappa_i}})} \tag{3.22}$$

如果蠕虫节点在 $r+1$ 个时隙前的任意一个时隙中检测到，那么蠕虫节点将被检测到，所以蠕虫节点在前 $r+1$ 个时隙被检测到的概率为 $P_r=1-\prod_{i=1}^{r}(1-P_S^i)$，由于 $(1+x)\le e^x$ 因此有：

$$P_r=1-\prod_{i=1}^{r}(1-P_S^i)\ge 1-e^{-\sum_{i=1}^{r}P_S^i} \tag{3.23}$$

定理 1 证明完毕。

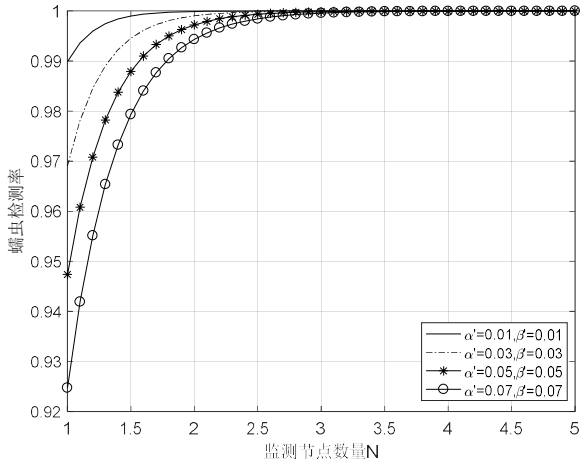


图 3.2 网络中监测节点个数和检测率的关系

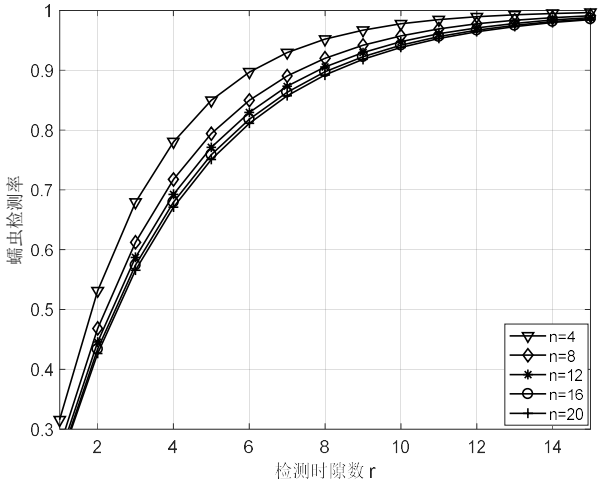


图 3.3 检测时隙个数和检测率的关系

我们首先给出了 $DR(N)$ 随检测节点 N 变化的关系， N 从 1 个节点逐渐增加到 5 个节点，结果如图 3.2 所示。由图 3.2 分析可知，WSN 蠕虫节点的检测率随着 α' 和 β' 的减小而增大，这与所分析的结论一致。同时整个网络的蠕虫检测率随着网络中监测节点的数量增加而增大，且在较小数量的监测节点下依然能保持高的蠕虫检测率。

图 3.3 描述了蠕虫检测率随着检测样本数和检测时隙的变化关系。在该仿真中为了方便讨论蠕虫检测率的性能，我们假设在每一轮时隙中 κ 值相等，令 $\kappa_1=\kappa_2,.....\kappa_{r-1},\kappa_r=\kappa_{\max}=3$，那么就有 $P_S^1=P_S^2=.....=P_s^{r-1}=P_S^r$，同时我们令 $\alpha'=\beta'=0.01,\gamma_0=0.1,\gamma_1=0.9$。由仿真结果可知随着采样样本数目的增加和检测时隙的增加，蠕虫检测率都随之增加，且只需要通过 5 个时隙的检测就能达到 0.8 的检测率，这说明了方案的高效性。

3.4.2 蠕虫节点感染的上限

在这一节中，分析 SPRT-Biased-Random 检测方案检测到蠕虫节点时，网络中受感染节点的数量。

定理 2：令 η 为监测节点 n 个通信模式采样样本中由蠕虫节点引起的融合通信模式的比例，那么监测节点 w 在其邻居节点至少被感染 $\eta'*n$ 个时，能够检测出蠕虫节点。其中 η' 如式 3.24 所示：

$$\eta'=\frac{\ln\frac{1-\beta'}{\alpha'}+n\ln\frac{1-(\gamma_0)^{\kappa_i}}{1-\gamma_1}}{n(\ln\frac{\gamma_1}{(\gamma_0)^{\kappa_i}}-\ln\frac{1-\gamma_1}{1-(\gamma_0)^{\kappa_i}})*n} \tag{3.24}$$

证明：因为 φ_n 代表融合通信模式在 n 个通信模式采样样本中的个数，那么有 $\varphi_n=\eta*n$。根据 SPRT 算法的终止条件有：$\varphi_n\geq L_1(n)$，那么可以重写为 $\eta\geq\eta'$， η' 如式 3.24 所示。值得注意的是一个融合通信模式的出现代表有一个节点被感染，为了简单起见，我们假设蠕虫节点之间不会在彼此传染。因此，有 $\varphi_n=\eta*n$ 个融合通信模式出现，说明就有 $\eta*n$ 个节点被感染，那么当监测节点 w 的邻居节点至少有 $\eta'*n$ 个被感染时， w 能监测出该区域有蠕虫节点。

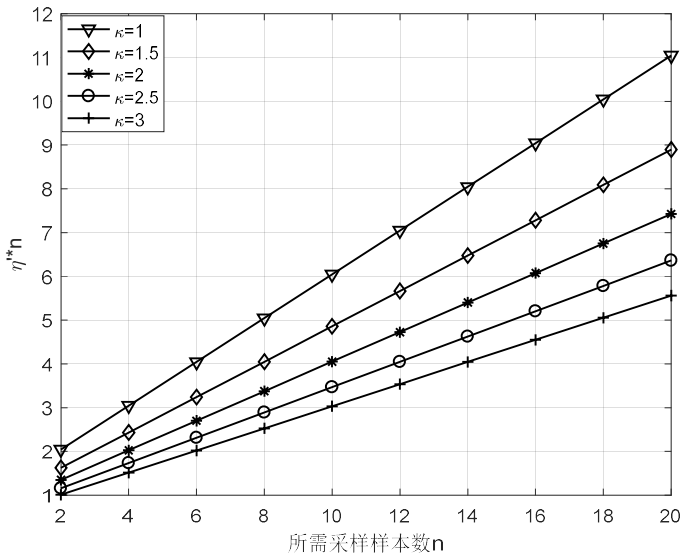


图 3.4 参数κ和n对w节点检测出蠕虫节点所需要的最小感染节点数的影响

根据定理 2，图 3.4 描述了参数κ和n与监测节点w检测出蠕虫节点所需要的最小感染节点数的关系。我们令$\alpha^{'}=\beta^{'}=0.01,\gamma_0$=0.1,$\gamma_1=0.9$，可以发现最小感染节点的数量与采样数$n$成正比，这说明需要检测的时间越久，网络中受感染的节点就会越多。因此如果能够通过尽量少的采样样本n检测蠕虫节点的话，对传感器网络的影响会最小，与此同时我们发现参数κ越大，传感器节点最后受到感染的节点越少，这是因为参数κ越大,SPRT 算法接受假设H_1的概率越大，可以加速对蠕虫节点的检测。

3.5 性能分析

在本小节，我们首先给出了 SPRT 蠕虫检测模型决策需要的样本均值，接着讨论了整个方案中产生的通信开销和内存开销。

3.5.1 决策平均采样样本数

我们将n定义为 SPRT 算法做出决策的样本数目，n是一个随机变量，该变量由样本类型决定,根据文献[72]平均采样样本数计算如式 3.25 所示：

$$E(n)=\frac{E[R_n]}{E\left[\ln\frac{\Pr(A_k\mid H_1)}{\Pr(A_k\mid H_0)}\right]} \tag{3.25}$$

由式 3.25 计算$E(n\mid H_0)$和$E(n\mid H_1)$如式 3.26 所示：

$$E(n|H_0)=\frac{(1-\alpha')\ln\frac{\beta'}{1-\alpha'}+\alpha'\ln\frac{1-\beta'}{\alpha'}}{\gamma_1\ln\frac{\gamma_1}{\gamma_0}-(1-\gamma_0)\ln\frac{1-\gamma_1}{1-\gamma_0}}$$
$$E(n|H_1)=\frac{\beta'\ln\frac{\beta'}{1-\alpha'}+(1-\beta')\ln\frac{1-\beta'}{\alpha'}}{\gamma_1\ln\frac{\gamma_1}{\gamma_0}-(1-\gamma_1)\ln\frac{1-\gamma_1}{1-\gamma_0}} \tag{3.26}$$

结合引入的随机参数κ， $E(n|H_0)$和$E(n|H_1)$ 可以改写为式 3.27 所示：

$$E(n|H_0)=\frac{(1-\alpha')\ln\frac{\beta'}{1-\alpha'}+\alpha'\ln\frac{1-\beta'}{\alpha'}}{\gamma_1\ln\frac{\gamma_1}{\gamma_0^{\kappa}}-(1-\gamma_0^{\kappa})\ln\frac{1-\gamma_1}{1-\gamma_0^{\kappa}}}$$
$$E(n|H_1)=\frac{\beta'\ln\frac{\beta'}{1-\alpha'}+(1-\beta')\ln\frac{1-\beta'}{\alpha'}}{\gamma_1\ln\frac{\gamma_1}{\gamma_0^{\kappa}}-(1-\gamma_1)\ln\frac{1-\gamma_1}{1-\gamma_0^{\kappa}}} \tag{3.27}$$

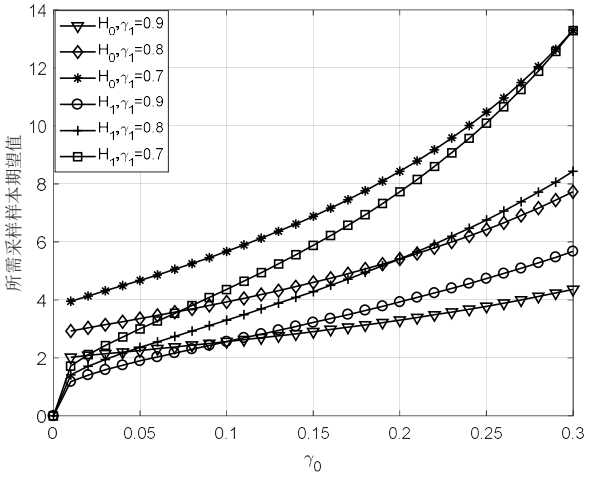


图 3.5. γ_0,γ_1和采样均值的关系

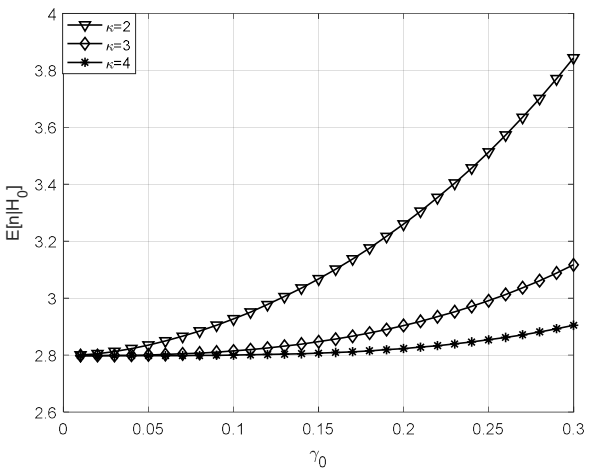


图 3.6 κ 值和 $E(n|H_0)$ 的关系

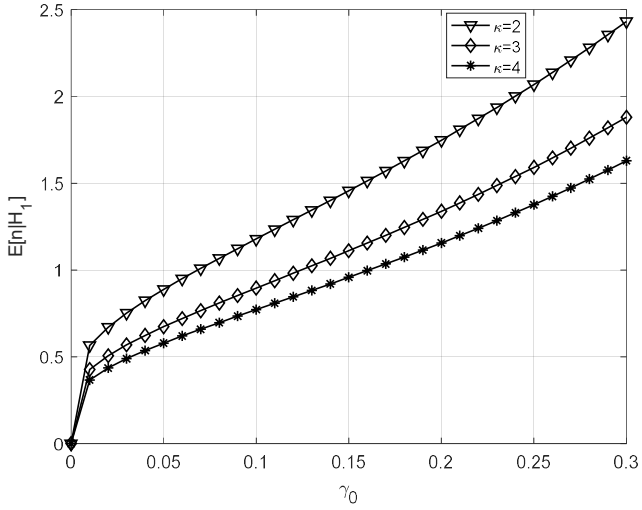


图 3.7 κ 值和 $E(n|H_1)$ 的关系

图 3.5 展示了 γ_0,γ_1 与采样样本均值的关系，通过结果图我们可以发现 γ_1 越大 SPRT 算法所需要的决策样本数目就越少，γ_0 越大所需要的决策样本数就越高。同时由图进一步分析我们可知 γ_0 较小时 $E(n|H_0)>E(n|H_1)$，当 γ_0 逐渐增大时两者之间的差距渐渐缩小，最终 $E(n|H_0)<E(n|H_1)$。这表明 γ_0 越小，SPRT 检测算法能够通过少量样本，迅速检测出蠕虫节点。图 3.6 和 3.7 分别展示了 κ 值和 $E(n|H_0)$、$E(n|H_1)$ 的关系，在该仿真中我们假设 $\alpha'=\beta'=0.01,\gamma_1=0.9$，$\gamma_0$ 从 0.0 逐渐增加至 0.30。由仿真结果分析可知，随着 γ_0 的增大，平均采样值都有所增加，且随着 κ 增大，平均采样样本逐渐减少，这是因为 $\gamma_0^{\kappa}<\gamma_0$，该值越小 SPRT 算法决策所需要的平均样本数就越少。

3.5.2 通信资源开销和内存资源开销

我们定义 SPRT 检测算法中的通信损耗为每个时隙中节点间发送和接收的平均通信模式量。假设整个传感器网络中有 T 个节点，那么整个网络在一个时隙中有 $T*P_d$ 个监测节点，P_d 为监测节点的选举概率。假设节点 u 拥有 b 个邻居节点，同时每个时隙中平均有 ϖ 个数据包的目的地址为节点 u，如果该节点为非监测节点，那么该节点在每个时隙中将会发送 $\varpi*p_f*p_n$ 个数据包，其中 p_f 为数据包的抽样概率，p_n 为包的源节点为节点 u 的邻居节点的概率。如果该节点为监测节点那么它在每个时隙中最多需要接收 $b*\varpi*p_f*p_n$ 个数据包，对于整个传感器

网络而言每个时隙中，监测节点的数量为$T*P_d$，非监测节点的数量为$T*(1-P_d)$，那么一个时隙中整个网络的最大通信开销为$T*P_d*b*\varpi*p_f*p_n+T*(1-P_d)*\varpi*p_f*p_n$。我们定义 SPRT 检测算法中的内存资源开销为每个时隙中监测节点需要存储的通信模式的平均值。在最坏的情况下，每一个监测节点最多需要存储的样本数目为$b(b-1)/2$，网络中的监测节点数量为$T*P_d$，那么网络中总的最大内存资源开销为$T*P_d*b*(b-1)/2$。在本文中我们讨论通信资源开销和内存资源开销随检测时隙增加的累计和。需要注意的是每个时隙中被检测出的蠕虫节点不在进行通信，因此ε逐渐减小。

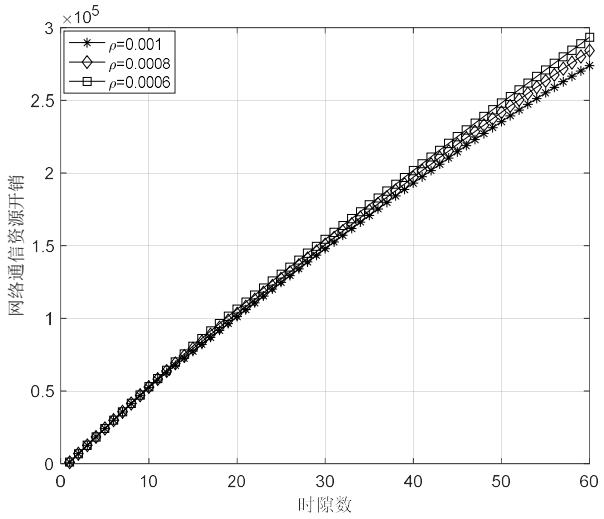


图 3.8 不同感染率下检测时隙和通信资源的关系

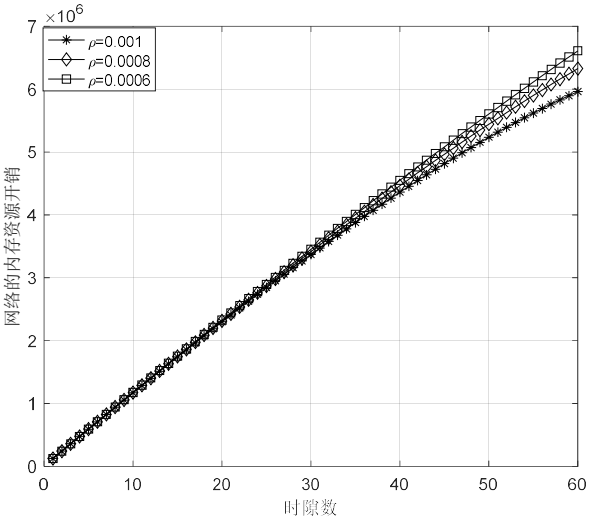


图 3.9 不同感染率下检测时隙和内存资源的关系

图 3.8 和图 3.9 分别描述了不同感染率下传感器网络的通信资源和内存资源随检测时隙的变化关系。很明显，随着时隙的增加整个网络的内存资源和通信资源都在增加，但并不是呈现简单的线性增加关系而是增长率慢慢下降的趋势，这是因为随着检测时隙的增加，网络中能够通信的节点减少（被检测到的蠕虫节点不在通信），总的通信量增长减小。同时我们发现感染率越大，相对的资源开销越小，这是因为感染程度越高，蠕虫节点检测速度加快，总体资源开销下降。

3.6 实验分析

本节首先给出了仿真环境的设定，然后给出相应的仿真结果，分析方案的性能。

3.6.1 仿真环境设定

我们通过简单的仿真实验来评估所提出模型的性能。在仿真实验中，500 个传感器节点部署在$500m\times500m$的方形区域内，传感器节点的通信半径为$25m$。在之前的分析中我们将传感器节点按照分组进行部署，采用的分组策略是将节点按照二维高斯分布分散在区域中，所满足的二维高斯分布密度函数满足式 3.23。其中(x_g,y_g)是每个传感器分组的组头节点，σ为标准差，在仿真实验中$\sigma=25$和通信半径相同，我们将 500 个传感器节点分成 20 组，每一组 25 个传感器节点，其它仿真参数设计如下：$\alpha'=\beta'=0.01,\gamma_0=0.1,\gamma_1=0.9,\kappa_{\max}=5$，监测节点的自我选举概率$p_d=0.1$，分组包随机广播的概率$p_f=0.02$。

$$f(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{(x-x_g)^2+(y-y_g)^2}{2\sigma^2}} \tag{3.28}$$

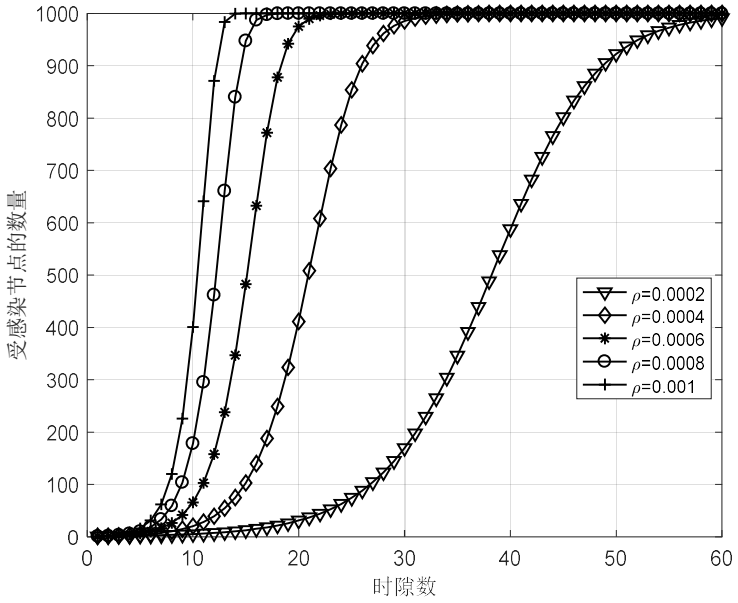


图 3.10 不同感染率下感染节点随时隙的变化

假设每个节点要么只单独传播良性的分组包，要么就仅仅传送蠕虫节点包。在良性传播的情形下，我们假设正常的网络流量生成满足泊松分布，包发送的间隔时间满足参数为$\lambda=1$的指数分布。对于蠕虫节点而言，假设它在每个时隙中发送的分组包为 50 字节以感染其他节点，文献[79]对这一假设进行了合理的分析。同时假定蠕虫节点以逐跳方式进行传播，在感染一个节点之后将恶意代码继续传播给下一个节点，假定蠕虫的传播模型为 2.1.3 节中讨论的离散型蠕虫传播模型，我们讨论蠕虫的感染率从 0.0002 变化到 0.001 的实验结果。对于每一组的性能指标，我们取 1000 次实验结果的均值作为最后结果进行比较。

3.6.2 仿真结果说明

我们通过以下几个性能指标来评估文章所提出的 SPRT 蠕虫检测方案（原始方案$\kappa=1$和κ为$[1,\kappa_{\max}]$随机值方案）。

i SPRT 蠕虫检测算法决策所需要的平均采样样本数

ii SPRT 蠕虫检测算法决策所需要的平均时隙数

iii 检测到蠕虫病毒传播时，网络中被感染的节点数

iv 通信资源和内存资源消耗

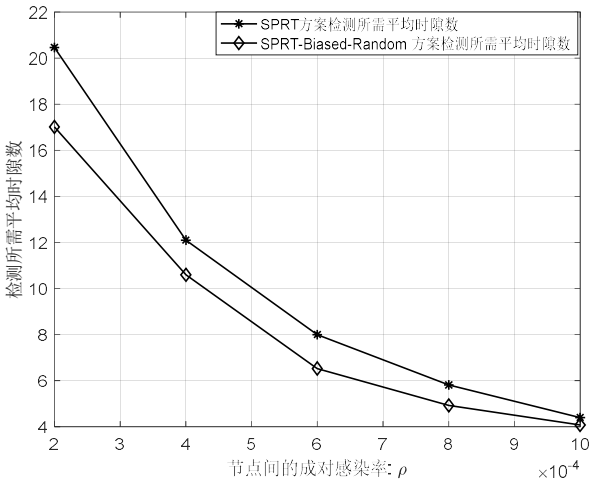


图 3.11 不同感染率下所需平均检测时隙

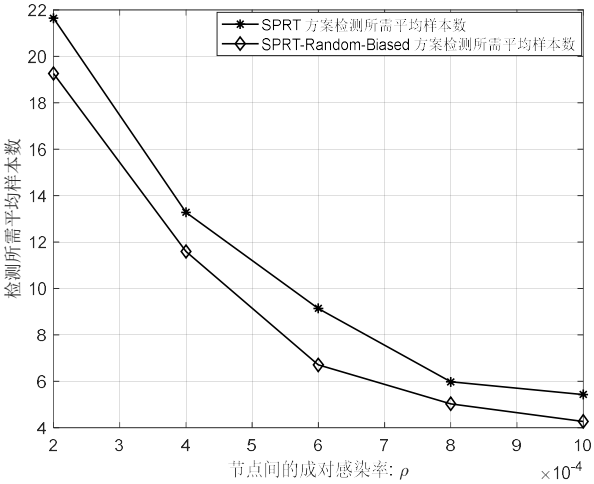


图 3.12 不同感染率下所需的平均采样数

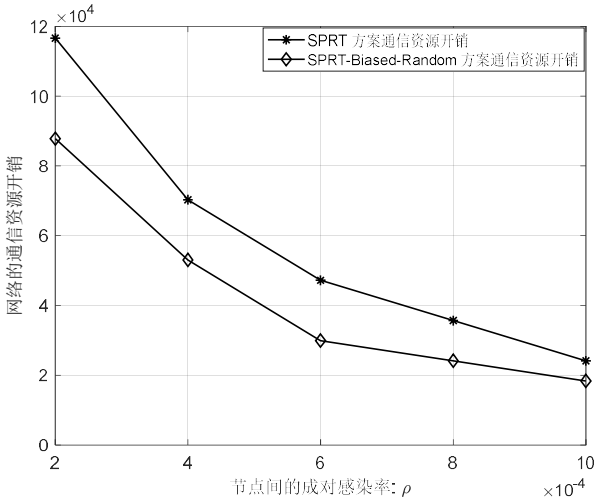


图 3.13 不同感染率下网络的通信资源比较

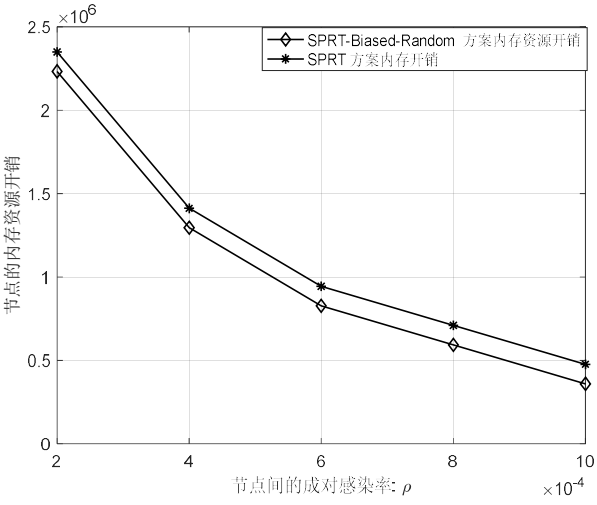


图 3.14 不同感染率下网络的内存资源开销

图 3.11 描述了不同感染率下 SPRT 蠕虫检测方案和 SPRT-Biased-Random 蠕虫检测方案检测蠕虫节点所需的平均检测时隙数。由 3.11 分析我们可知随着蠕虫病毒感染率的增长，整

个网络的检测时隙逐渐减少，这是因为蠕虫节点的感染率增高，网络中的蠕虫包传送频繁，会出现大量的包重传现象加速了 SPRT 检测方案对蠕虫的检测。同时，我们可以很明显的发现 SPRT-Biased-Random 方案相比 SPRT 方案能够更加有效的降低所需检测时隙数，更快的检测到蠕虫节点。蠕虫的感染率越低，SPRT-Biased-Random 的检测时隙数相对于 SPRT 算法的效果就越好，这突出了所提方案的优越性。

图 3.12 展示了不同感染率下 SPRT 蠕虫检测方案和 SPRT-Biased-Random 蠕虫检测方案检测并清除网络中的蠕虫节点所需要的平均样本数。随着感染率的增加，两种方案下平均检测样本数目都在减少，这是因为蠕虫感染率增加，网络中大量的发生包重传现象，网络中的大量样本为有效样本能够加速蠕虫节点的检测。同时我们可以知道 SPRT-Biased-Random 方案下需要的平均样本数要小于 SPRT 蠕虫检测方案，可以简单理解为需要的检测时隙少，所需要的平均样本值相对较少。且当蠕虫节点的成对感染率为 0.001 时，SPRT-Biased-Random 方案只需要约为 5 个样本值就能检测到网络中的蠕虫节点。

图 3.13 和 3.14 分别描述了不同感染率下整个网络的通信资源损耗和内存资源损耗。随着蠕虫感染率的增加，通信资源和内存资源都逐渐降低。这是因为随着感染率的增加，网络对蠕虫节点的检测加快，所需要的平均检测时隙数减少，网络间的通信开销和内存资源开销随之降低，具体可以参考 3.5.2 中的分析。同时 SPRT-Biased-Random 检测方案在各个蠕虫感染率下的资源消耗要比 SPRT 检测方案低，这也是所提方案的优越性所在。

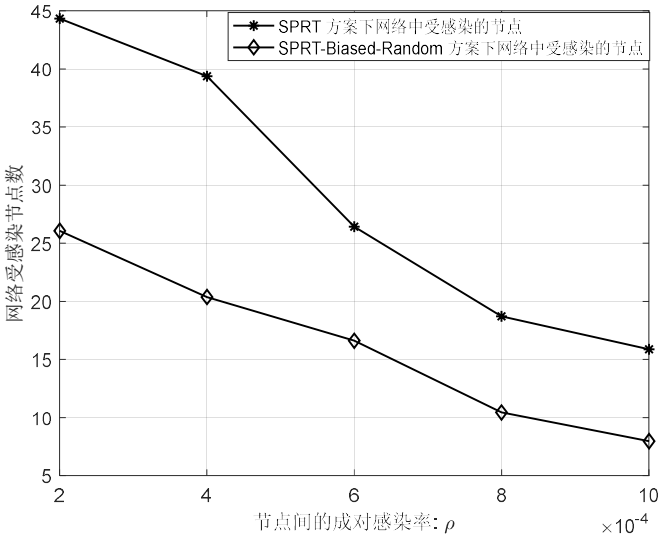


图 3.15 不同感染率下传感器网络检测到蠕虫节点时受感染的节点数量

图 3.15 展示了两种方案在不同感染率下传感器网络检测到蠕虫节点时受感染的节点数量，由图分析我们可知，蠕虫节点的感染率越高，就越容易被侦测到，其在网络中感染的节点就会相对更少。我们也可以看到所提出的 SPRT-Random-Biased 检测方案能够有效的限制感染节

点的比例在 2%~5%左右，比 SPRT 检测方案要更加优越。

3.7 本章小结

本章基于 SPRT 算法构建了蠕虫检测模型，并在其基础上构建了有效解决慢蠕虫传播的 SPRT-Biased-Random 蠕虫检测模型。通过进一步的分析和实验，我们提出的模型能够有效地检测传感器网络中的蠕虫节点，极大的限制蠕虫传播给网络带来的影响。实验结果说明我们提出的 SPRT-Biased-Random 检测方案能够通过尽可能少的时隙和采样样本数量来检测到蠕虫节点，该检测方案能够将蠕虫感染节点限制在 2%~5%之间，同时该方案能够减少网络中的通信损耗和内存损耗，高效地检测传感器网络中的蠕虫节点。

然而，在该研究中我们并没有研究恶意节点既传播良性数据又传播恶意代码的情况，如果蠕虫节点能够既发送良性数据又发送蠕虫包，那么它完全有可能将自己伪装成良性节点进行隐藏。在本章中并没有对这一个情况进行考虑，这是值得研究的一个方向点。

第四章 改进型分布式 PCA 异常数据检测方案

4.1 引言

WSN 要求网络传输的数据值足够可靠，然而其部署在无人监控的野外，传感器节点很容易被攻击者捕获。攻击者通过捕获节点仿造数据值可以很容易的达到欺骗基站的目的，同时传感器节点能量有限，网络发送数据时极易受到环境中噪声的影响，造成数据的失真。因此为了保证基站接收到数据的可靠性，必须对异常数据进行检测。上一章中我们针对蠕虫节点进行检测，文献[24]针对移动恶意节点进行检测，文献[26]针对复制节点进行检测，文献[34]针对移动复制节点进行检测。通过恶意节点检测能够很大程度上减小网络被攻击的概率，本章我们并不考虑由恶意攻击引起的异常数据检测，而将重点放在由恶意事件、传感器内部错误和环境噪声引起的异常数据检测。

在 WSN 中，异常数据定义为与正常的监测数据特征差别较大的数据。因此检测异常数据最简单的方法就是对正常的数据进行建模，然后将模型应用于新的监测数据。如果新到达的监测数据和模型的偏差超过一定的阈值，那么可以将该监测数据定义为异常数据进行丢弃，否则认为该数据值是一个正常的数据值。文献[68,81]详细的概述了 WSN 中的异常数据检测方案, 这些方案普遍存在以下的缺点：一方面，这些方案很多是集中式方案，不仅会给整个网络带来巨大的通信资源开销，还会给网络中的一些节点带来巨大的计算资源开销。另一方面，这些方案并不能在资源消耗和检测效率上得到折衷的效果，这对检测精度要求高的应用而言并不适用。最后，以上这些方案仅仅解决了如何检测传感器网络中的异常数据，却没有涉及如何追溯异常数据发生的原因，对于网络的管理者而言，仅仅知道网络中是否有异常数据发生是不够的，掌握异常数据发生的原因有助于网络管理者更好地保护网络的安全性。

PCA 被广泛运用于模式识别领域，是一类非常高效的模式识别算法[82]。本章基于 PCA 算法，对数据向量进行降维之后使用马氏距离来衡量数据向量之间的相似性，提出了 IDPCA 异常数据检测方案。该方案下异常检测分为局部检测和全局检测，大大的提高了网络的检测性能。同时充分利用传感器网络的空间相关性，本章创造性的提出了异常数据源检测方案，主要检测网络中的异常数据是由恶意事件引起的还是由传感器节点内部错误或者噪声引起的。通过拟合数据和英特尔伯克利实验室侦测的数据进行仿真分析，仿真结果表明 IDPCA 异常数据检测方案能够达到 96%~97%的异常数据检测率和 2%的数据误报率，能够有效地识别网络中异常数据发生的原因并且降低网络的整体通信资源消耗。

4.2 网络拓扑结构假设

本节对 WSN 的部署规则进行相应的说明以及假设。假设整个传感器网络是一个静态网络，节点部署完成之后不再改变位置，节点之间可以进行双向通信且节点之间的密钥对已经提前匹配，非合法密钥之间的节点无法进行通信。整个网络部署在无人看守的野外环境下，传感器节点之间彼此同步。将节点按照组分散开来，每个组有一个组节点，假设同一个组中的节点监测的数据值是相近的，且都在组节点的通信范围内。令 $N(s_l)=\{s_i, i=1,2\cdots b\}$ 代表一组以 s_l 节点为组节点的一组传感器节点集合。在每一个时隙间隔 Δt，集合中的每一个节点测量一组数据值，令 $x_l, x_1^l, x_2^l, \cdots, x_b^l$ 分别代表 $s_l, s_1, s_2, \cdots s_b$ 各个节点测量的数据向量。每一个数据向量拥有 d 维的数据：$x_k^l=\{x_{k1}^l, x_{k2}^l, \cdots, x_{kd}^l\}, x_k^l \in \Re^d$。图 4.1 展示了一个以 s_0 为中心节点的邻居节点关系图。

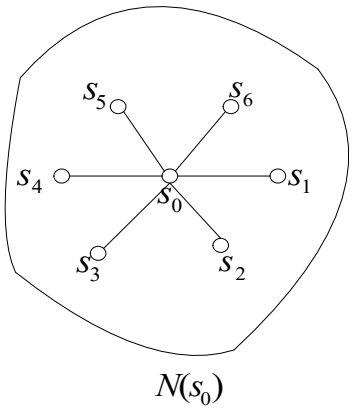


图 4.1 以 s_0 为中心节点的邻居节点

根据不同的应用场景，异常数据可以分为全局异常数据和局部异常数据。局部异常数据代表的是每一个节点根据自身的测量值和检测算法检测出的异常数据。全局异常数据是指通过一组节点中各个节点的共同作用，检测出的异常数据值。我们的目的在于当某个节点 s_i 检测到一个新的数据测量值，通过局部检测和全局检测对该数据进行实时的检测。另外，通过充分利用传感器节点之间的空间相关性，我们进一步研究了异常数据产生的原因。

4.3 IDPCA 异常数据检测模型

本节详细的介绍 IDPCA 异常数据检测方案，该方案主要包括四个部分：模型训练阶段、异常数据检测阶段、异常数据源检测阶段、模型更新阶段。

4.3.1 IDPCA 异常数据检测模型训练

模型训练阶段旨在为每一个节点单独地建立起正常数据模型。假设 $X_i(n_i)$ 是传感器节点 i 在 m 个时隙窗口中收集到的 n_i 个检测数据向量， $X_i(n_i)$ 可以表示为式 4.1 所示。

$$X_i(n_i)=(x_i(1),x_i(2),\cdots,x_i(n_i))^T \tag{4.1}$$

其中每一个 $x_i(k)\in X_i(n_i)$ 有 d 维分量。节点 s_i 首先将 $X_i(n_i)$ 中每一个向量值进行标准化，然后计算以列为中心的矩阵：

$$\hat{X}_i(n_i)=X_i(n_i)-e_{ni}\bar{X}_i(n_i) \tag{4.2}$$

$\bar{X}_i(n_i)$ 为每一列的均值向量，$e_{ni}=(1,1,1,\cdots 1)$ 为长度 n_i 的向量。接着计算协方差矩阵如式 4.3 所示：

$$C=\hat{X}_i^T(n_i)\hat{X}_i(n_i) \tag{4.3}$$

求解协方差矩阵对应的特征值和特征向量，对应的特征向量即为主成分分量，其中特征值按照从大到小排列，最大特征值对应的特征向量为第一主成分分量定义为 $\varphi_i(0)$。

$$CV=\lambda V \tag{4.4}$$

节点 s_i 计算所有测量向量值 $x_i(k)\in X_i(n_i)$ 到第一主成分 $\varphi_i(0)$ 的投影距离 d_p 如式 4.5 所示，示意图如图 4.2 所示。

$$d_p(x_i(k),\varphi_i(0))=(\|x_i(k)-\bar{X}_i(n_i)\|^2-(\varphi_i^T(0)\cdot(x_i(k)-\bar{X}_i(n_i)))^2)^{\frac{1}{2}} \tag{4.5}$$

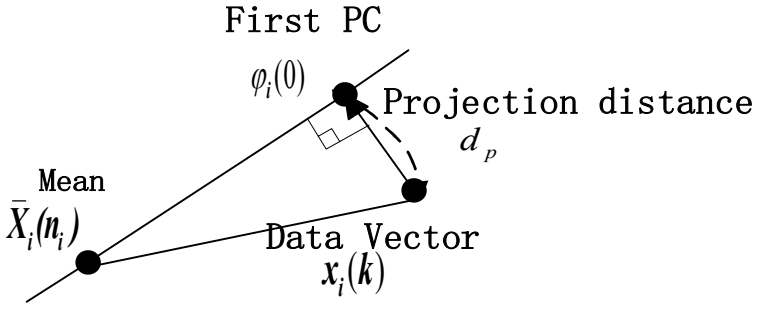


图 4.2 测量向量值 $x_i(k)\in X_i(n_i)$ 到第一主成分 $\varphi_i(0)$ 的投影距离 d_p.

然而欧式距离在计算数据向量的相似度时并不会考虑数据维度之间的相关性，马氏距离在计算数据向量的相似度时结合数据向量维度之间的相关性更加适用于多元数据向量的相似度分析[62]。同时，在最大主成分方向上的投影并不能完全地反应数据的原有特征，因此我们采用新的距离计算方案来建立模型。令 sum 为特征值的的总和，即：

$$sum=\lambda_1+\lambda_2+\lambda_3.....+\lambda_N \tag{4.6}$$

为了能够获取数据的绝大部分特征，使用主成分分量的得分来确定选择的主成分个数。实验表明当主成分的得分大于 0.9 时，新的映射坐标能够代表原数据的特征，主成分个数的获取由式 4.7 确定。

$$\begin{cases} PC\ Score = \sum_{i=1}^{\chi} \lambda_i / sum & 1 \le i \le N \\ PC\ Score >= 0.9 \end{cases} \tag{4.7}$$

这也就意味着当我们将数据进行 PCA 处理之后原来的 d 维数据 $x_k^l = \{x_{k1}^l, x_{k2}^l, \cdots, x_{kd}^l\}, x_k^l \in \Re^d$ 将变成 $(x_k^l)_{PCA} = \{x_{k1}^l, x_{k2}^l, \cdots, x_{k\chi}^l\}$ 的 χ 维数据向量。定义映射后的数据向量的协方差矩阵为 Σ，均值 $\mu = (\mu_1, \mu_2 ..... \mu_\chi)$，那么各个数据向量到数据中心的马氏距离计算如式 4.8 所示：

$$MD(x) = \sqrt{(x-\mu)\Sigma^{-1}(x-\mu)^T} \tag{4.8}$$

$MD(x)$ 定义了各个向量投影之后到中心节点的特征距离，能够很好的定义数据之间的相似度。定义 $MD_i(\max) = \max\{MD(x_k)\}, k = 1, 2, \cdots n_i$ 为邻居节点 s_i 中数据向量距离数据中心最大的马氏距离。以六元组 $(\mu, \lambda, v, \chi, \Sigma, MD(\max))$ 作为节点 s_i 的本地异常检测模型。

当一个数据向量在本地被检测为异常数据时需要再次进行全局的异常检测，全局异常检测由邻居节点的簇头节点负责检测，该节点的异常检测模型是由邻居节点共同作用获得的。每个邻居节点将训练得到的三元组 $(\mu, \Sigma, MD(\max))$ 发送给簇头节点，簇头节点根据特定的融合算法[83]来建立全局异常检测模型，模型的计算规则如式 4.9 所示。

$$MD(global) = (MD(x_1) + MD(x_2) + ...MD(x_d)) / d$$

$$\mu(global) = \frac{MD(x_1)}{MD(global) * d} * \mu_1 + ... \frac{MD(x_d)}{MD(global) * d} * \mu_d \tag{4.9}$$

$$\Sigma(global) = \sum_{i=1}^{d} \frac{MD(x_i) - 1}{MD(global) * d - 1} * \Sigma_i + \sum_{i=1}^{d} \sum_{j=1}^{d} \frac{MD(x_i) MD(x_j)}{(MD(global) * d - 1)(MD(global) * d)} [(\mu_i - \mu_j)(\mu_i - \mu_j)^T]$$

综上所述，IDPCA 异常数据检测模型的伪代码如表 4.1 所示。

表 4.1 异常检测模型的建立

for $i=1:b$
s_i *collects* n_i *data vectors*
s_i *use PCA to compute eigv and eigm* : (λ, v)
s_i *select* χ *principal components by PC Score*
s_i *compute* $MD(\max)$ *by Mahalanobis distance*
s_i *get training* mod*el* : $(\mu, \lambda, v, \chi, \sum, MD(\max))$ *for local* det*ection*
end
then
every neibour nodes send $\{\mu_i, \sum_i, MD(\max)_i\}$ *to group node*
group node get $(\mu(global), \sum(global), MD(global))$ *for global* det*ection*
end
each group operates like that

4.3.2 IDPCA 异常数据检测

在密集分布的传感器网络中，充分利用传感器节点之间的数据相关性，每个节点都能够提供充足的数据来进行异常检测。检测阶段分为局部检测和全局检测。初始化阶段，每个传感器节点通过它在 m 个时间窗口中获得的监测数据计算主成分向量和主成分个数进一步计算最大马氏距离 $MD_i(\max)$。接着每个节点将它的三元组 $(\mu, \sum, MD(\max))$ 广播到其相邻的簇头节点中，簇头节点根据各个邻居节点发送的三元组 $(\mu, \sum, MD(\max))$ 通过融合算法来计算 $(\mu(global), \sum(global), MD(global))$。当一个新的数据向量 $x_i(t)$ 到达节点 i ,节点 i 首先运行本地异常检测算法。根据其六元组 $(\mu, \lambda, v, \chi, \sum, MD(\max))$ 计算当前数据向量到数据中心的马氏距离 $MD_i(c)$，然后比较 $MD_i(c)$ 和 $MD_i(\max)$。如果 $MD_i(c) < MD_i(\max)$，那么当前数据向量被认为是一个正常的数据。否则，该数据向量被标记为可疑数据，节点 i 将降维之后的数据 $x_i(t)_{PCA}$ 发送到簇头节点并根据三元组 $(\mu(global), \sum(global), MD(global))$ 计算相似度 $MD_i(\text{global})$ 与 $MD(global)$ 进行比较，如果 $MD_i(global) > MD(global)$，那么 $x_i(t)$ 最终被认为是一个异常数据，否则将其可疑标记去除，认为其是一个正常数据。因此，最终的异常数据判别函数如式 4.10 所示:

$$f(x) = sgn(MD_i(c) - MD_i(\max)) \oplus sgn(MD_i(global) - MD(global)) \tag{4.10}$$

根据式(4.10)，如果 $f(x)$ 的最终结果等于-1，那么该数据被认为是一个异常数据,否则该数据为正常数据。

$$f(x)=\begin{cases}1 & \text{正常数据}\\ -1 & \text{异常值}\end{cases} \tag{4.11}$$

异常数据检测的伪代码如表 4.2 所示。

表 4.2 异常数据的检测流程

```
when x_i(t) arrives at s_i
    s_i computes MD_i(c) according to (μ, λ, v, χ, Σ, MD(max))
    if (MD_i(c) < MD(max))
            x_i(t) indicates an normal data
    else {
        s_i transfer x_i(t)_PCA to group node to compute MD_i(global)
        if (MD_i(global) > MD(global))
                x_i(t) indicates an outlier
                operate SourceOfOutlier()
        else
            x_i(t) indicates an normal data
}
end
```

4.3.3 异常数据源检测

区分是何种原因引起了网络中的异常数据是一项非常重要的工作。无线传感器网络中，引起数据异常的原因主要包括以下几种：通信链路中的环境噪声、传感器内部节点错误、真实的事件发生和恶意攻击。本方案着重讨论如何区分异常数据是由简单的噪声或者内部错误引起还是由真实事件导致。主要思路如下：簇头节点中维护初始变量 $count=0$，当邻居节点监测数据为异常数据时 $count$+1,当所有邻居节点都检测完毕，如果 $count \geq b/2$,那么当前网络中有真实的事件发生(比如森林火灾发生时，会有一大批异常数据同时发生)。否则，当前的异常数据只是由单纯的噪声或者内部错误引起的，在这种情况下节点 s_i 收集随之而来的 t 个时隙窗口中的 p 个数据向量，计算其到数据中心的平均马氏距离 $\overline{MA}$。如果 $\overline{MA} < MA_i(\max)$，那么此时的数据异常是由噪声引起的，否则是由于传感器节点内部异常导致的数据异常。

异常数据可以由传感器内部的异常和通信链路中的噪声引起，为了便于分析我们假设这两者的产生概率相同为 p_E，p_E 的值理论上应该是一个非常小的数，否则传感器网络将无法正常运行，$b \times p_E$ 是每个簇中产生异常数据的节点数量。异常数据源归结为恶意事件的条件满足 $count \geq b/2$，这意味着有一半以上的节点的数据为异常数据，所以如果异常数据源被虚报为

恶意事件的话，那么一个簇节点中至少有 $b/2-p_E\times b$ 的节点被误报为异常数据。假设异常数据的误报率为 p_{FAR}，那么异常数据源被误报为恶意事件的最大概率为：$C_{b(1-P_E)}^{b(1/2-P_E)}(p_{FAR})^{b(1/2-P_E)}\times(1-p_{FAR})^{b/2}$。假设某个节点周围发生了恶意事件，我们假设一个簇中的节点检测的数据都是异常数据，这个时候如果没有检测出有异常事件发生，说明至少有一半以上的节点漏报了异常数据，也即假设异常数据的检测率为 p_{Dec}，那么漏报率为 $1\text{-}p_{Dec}$，至少有一半以上的节点漏报，所以异常数据源为恶意事件的最小检测概率为 $1\text{-}C_b^{b/2}(p_{Dec}(1\text{-}p_{Dec}))^{b/2}$。异常数据源的检测算法伪代码如表 4.3 所示:

表 4.3 无线传感器网络异常数据源检测

```
s_l collects the Mahalanobis distance from its neighbor respectfully
measurement is outlier then count++
    if (count ≥ b/2)
            then means an event happened
    else
        s_i collects p data vectors within b time windows
        compute MA
        if (MA ≤ MA_i(max))
                then x_i(t) just an error caused by noise
        else if (MA > MA_i(max))
                then x_i(t) just an error caused by inner error
endif
return;
```

4.3.4 模型更新

随着 WSN 部署环境的变化，监测数据实时变化(温度的变化)，因此异常数据检测模型需要进行实时的更新。令 t 代表当前的时间窗口，为了更新全局检测模型 $(\mu(global),\Sigma(global),MD(global))$，每个传感器节点都需要根据当前时间窗口的前 m 个时间窗口中的正常数据重新计算六元组 $(\mu,\lambda,v,\chi,\Sigma,MD(\max))$。具体的数据模型更新过程如图 4.3 所示。异常数据检测模型的更新，能有效的提高异常数据的检测效率，该方案的核心思想是减少对旧数据的依赖性，根据实时的数据来提高模型的精确性。

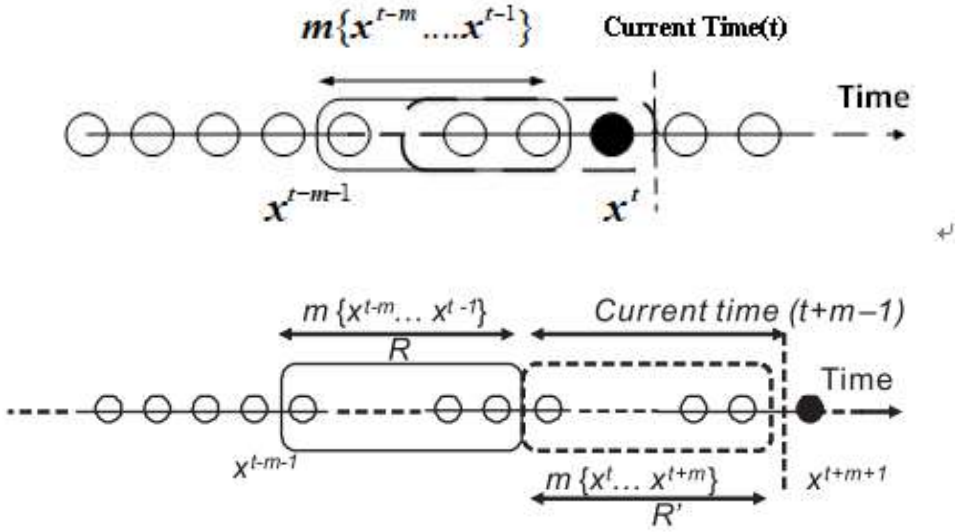


图 4.3 异常数据检测模型的更新过程，黑色节点代表当前的检测窗口

根据以上分析，给出完整的 IDPCA 算法检测的伪代码见附录 5。

4.4 性能分析

本节我们讨论 IDPCA 异常数据检测模型的通信资源消耗和计算资源消耗。

4.4.1 通信资源开销

我们假设通信资源开销为异常模型检测时，网络中因为 IDPCA 检测方案而引起的额外的通信开销。为了便于分析，我们以数据的维数为单位衡量通信的开销，例如传播一个数据向量 $x_k^l = \{x_{k1}^l, x_{k2}^l, \cdots, x_{kd}^l\}, x_k^l \in \Re^d$ 需要 d 的通信开销。在所提出的 IDPCA 方案中,每一个传感器分组拥有 b 个邻居节点，每个邻居节点在计算完成六元组之后将 $(\mu, \Sigma, MD(\max))$ 发送给簇头节点进行全局异常模型的建立。其中 μ 的维度是 $1\times\chi$，Σ 的维度为 $\chi\times\chi$，但是由于它是一个对称矩阵，所以其有效的维度为 $\chi(\chi+1)/2$，$MD(\max)$ 的维度为 1。所以，对于邻居节点而言其通信损耗为 $\chi(\chi+3)/2+1$，而簇头节点为 $b\times(\chi(\chi+3)/2+1)$。如果一个邻居节点 s_i 本地检测未通过，那么节点将当前向量值计算的 $x_i(t)_{PCA}$ 发送到簇头节点进行全局检测，这个过程中最坏的情况下每一个邻居节点需要额外增加 $1\times\chi$ 通信损耗，簇头节点需要额外增加 $b\times\chi$。因此在所提出的方案中邻居节点的通信损耗最多为为 $\chi(\chi+5)/2+1$,簇头节点的通信损耗为

$b\times(\chi(\chi+5)/2+1)$，总的通信损耗为$2b\times(\chi(\chi+5)/2+1)$。文献[68]提出的集中式 PCA 检测方案中，各个邻居节点将m个时隙窗口中收集到的n_i个数据向量集中发送到簇头节点，簇头节点根据这些数据向量建立检测模型，在异常数据检测时，每个节点将自己的数据向量值发送到簇头节点进行检测。因此在这种模式下邻居节点的通信量$(n_i+1)*d$，簇头节点的通信量为$\sum_{i=1}^{b}(n_i+1)*d$，总的通信损耗为$2\sum_{i=1}^{b}(n_i+1)*d$。文献[69]提出的分布式 PCA 检测方案中，每个邻居节点的通信量为$d(d+1)$，簇头节点的通信量为$b*d(d+1)$，总损耗为$2b*d(d+1)$。文献[62]中提出的基于 SVM 的分布式 EOOD 和 EAOD 方案在网络中的通信损耗为$b*(d+2)*(d+1)$，其中传递协方差矩阵的通信损耗为$b*d(d+1)$，传递均值矩阵和半径R_i的通信损耗为$2*b*(d+1)$。

4.4.2 计算资源开销

假设计算资源开销为异常模型检测时，网络中因为 PCA 检测方案而引起的额外的计算资源。在所提出的 IDPCA 检测方案中，模型建立阶段每个邻居节点需要对数据向量进行标准化，这个过程的计算资源耗费为n_i，接着计算协方差矩阵并通过特征值分解获取主成分分量，同时通过主成分得分值计算所需主成分分量的个数，这个过程的计算资源耗费为n_id^2。最后，再得到主成分分量之后需要计算坐标映射后的向量矩阵和其协方差矩阵并计算最大马氏距离$MD(\max)$值，这个过程的计算损耗为n_id^2，所以邻居节点的计算资源总消耗为$2n_id^2$。对于簇头节点而言，它需要额外的计算$(\mu(global),\sum(global),MD(global))$，这部分的计算开销为$2b+b^2$，所以簇头节点的计算开销为$2n_id^2+2b+b^2$，所以总的计算开销为$(b+1)*(n_id^2+2n_i)+2b+b^2$。文献[68]的集中式 PCA 检测方案，邻居节点的计算开销可视作为 0，簇头节点的计算开销为$(\sum_{i=1}^{b}(n_i+1))*(d^2+1)$。文献[69]中的 DPCA 异常数据检测模型中邻居节点的计算开销为$n_id^2+2n_i$，簇头节点的额外计算开销为$d^3\log_2 b$，所以簇头节点的计算开销为$n_id^2+2n_i+b(1+p_e)+d^3\log_2 b$，所以总的计算开销为：$(b+1)*(n_id^2+2n_i)+b(1+p_e)+d^3\log_2 b$。文献[62]中提出的基于 SVM 的分布式 EOOD 和 EAOD 方案在网络中的计算损耗分为协方差矩阵计算和均值计算，总的计算损耗为$(b+1)(n_id^2+d)$。

4.4.3 资源消耗对比

本节给出各个异常数据检测模型下，网络资源消耗的比较，其中邻居节点个数b为常量值不超过 50，具体分析如表 4.5 所示。

表 4.5 各个方案的通信性能损耗和计算资源损耗

	网络中通信损耗	邻居节点的计算损耗	簇头节点的计算损耗
CPCA 异常检测模型	$O(nd)$	-	$O(nd^2)$
DPCA 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$
IDPCA 异常检测模型	$O(b\chi^2)$	$O(n_i d^2)$	$O(n_i d^2)$
EOOD 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$
EAOD 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$

$$(b << n, \chi < d < n_i < n, bn_i = n)$$

由表 4.5 可知 IDPCA 异常数据检测模型在所给方案中拥有最小的通信损耗，且计算资源损耗和其它方案相近。为了细致比较各种方案的资源消耗，我们比较一个簇中总的计算资源开销和通信资源开销，如图 4.4,4.5,4.6 所示。

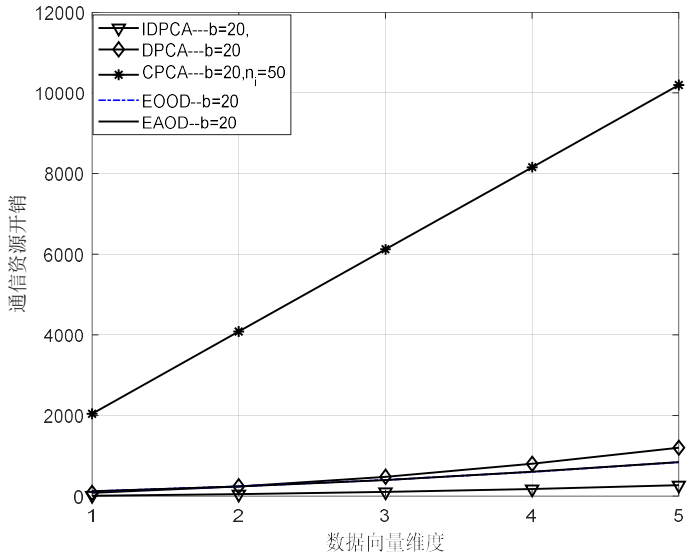


图 4. 4 监测数据维度和通信资源开销的关系

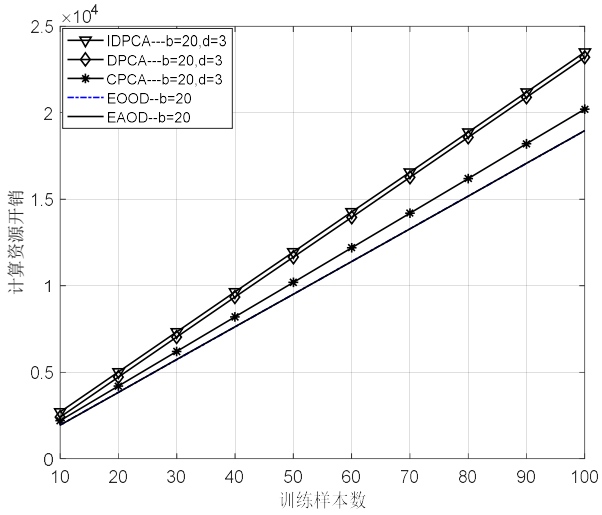


图 4.5 训练数据集和计算资源开销的关系

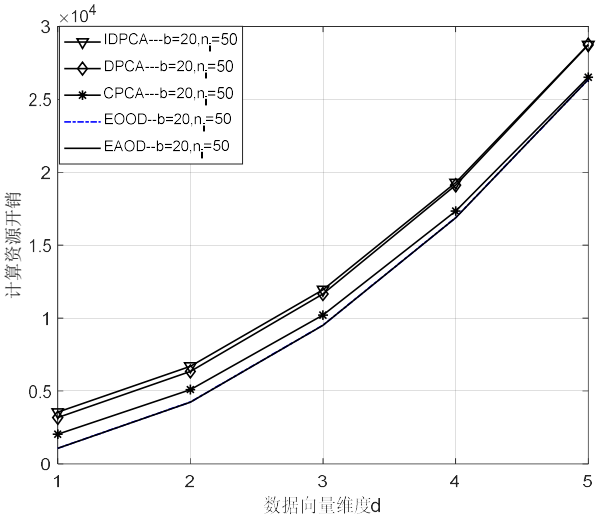


图 4.6 监测数据维度和计算资源开销的关系

图 4.4 描述了 5 种异常数据检测模型下，网络通信资源开销随测量数据向量维度变化的关系。CPCA 检测方案的通信资源开销远远超过了其余方案的通信资源开销，随着监测数据维度的增加各方案下网络的通信资源开销都慢慢增加，IDPCA 异常数据检测模型下网络的通信资源受数据维度变化的影响较小，拥有所给方案中最低的通信资源消耗。图 4.5 和图 4.6 展示了无线传感器网络中计算资源开销与模型训练数据样本数和监测数据维数的关系。随着训练样本数和数据维度的增大，总体计算资源开销随之增大，且 EAOD 和 EOOD 异常数据检测模型下，整个网络拥有最低的计算资源消耗，DPCA 方案和 IDPCA 方案的计算资源开销大致相等，IDPCA 稍微高一点。值得注意的是，在传感器网络中通信消耗的能量要远大于计算消耗的能量，因此 IDPCA 方案能够很好的节省传感器节点的能量，延长网络的寿命。

4.5 仿真分析

本节首先给出了仿真环境的设定，然后给出相应的仿真结果并分析方案的检测性能。在 WSN 中，通常选取两个指标来衡量模型的检测性能，即：异常数据的检测率(Detection Rate，DR)和误报率(False Alarm Rate, FAR)。异常数据的检测率指的是检测模型检测到的异常数据占总的异常数据的比例，误报率指的是检测模型将正确的数据误判为异常数据的比例，异常数据的检测率越高，误报率越低，检测模型的性能就越好。

同时在很多文献中以接收者操作曲线(Receiver Operating Characteristic,ROC) 来衡量异常数据检测模型的优劣性，ROC 曲线描述的是 FAR 和 DR 之间的关系，理论上该曲线和坐标轴所围面积最大为 1。评估某个检测方案的检测性能是否足够好可以通过 ROC 曲线所围面积是否趋近于 1 来判断和比较。本节我们通过比较不同 PCA 方案的 ROC 曲线和基于 SVM 异常

数据检测模型的 ROC 曲线来说明 IDPCA 检测模型的优越性，ROC 曲线与坐标轴所围面积(AUC)越大，该检测模型的性能就越好。

4.5.1 仿真环境设定

I 拟合数据集

在拟合数据集合中我们以一组节点进行仿真分析，簇头节点拥有 20 个邻居节点，监测数据为 3 维数据，每个维度的数据值服从高斯分布，方差为 0.03，均值为(0.3,0.35,0.4)中的随机值。每个节点中的异常数据定义为某个维度上的数据值服从均匀分布，取值范围在[0.5,0.7]。我们选取四个时间窗口中的 1000 正常数据来训练异常检测模型，接着在每个时间窗口中选取 200 个正常数据值和 50 个异常数据值，分析每一个时间窗口中，检测模型的异常数据检测率和误报率。拟合的数据集合如图 4.7 所示。

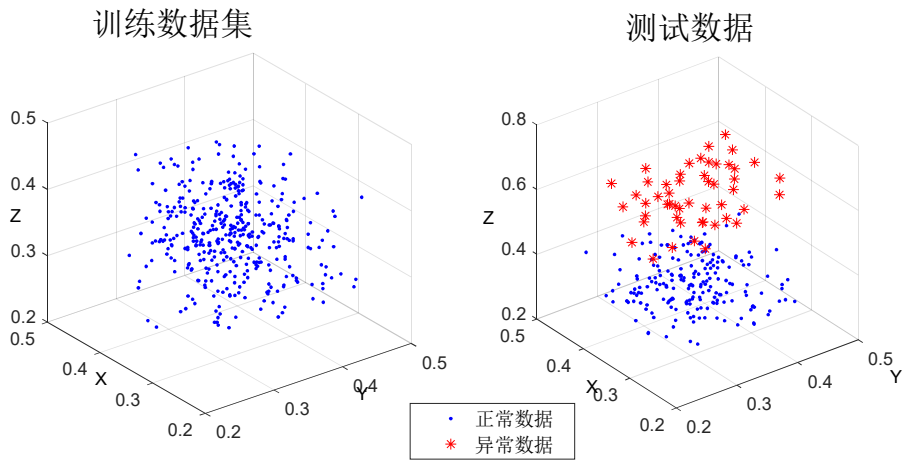


图 4.7 训练数据集和待检测数据集

II IBRL 数据集

英特尔伯克利实验室采集的传感器数据集(IBRL)是评估无线传感器网络中异常数据检测模型性能的主要数据源。该网络测量的数据包括：环境温度，湿度，光照强度和电压，本节我们使用湿度值和温度值两个属性进行仿真分析。该网络每隔 31 秒收集一次数据，我们的仿真数据来源于图 4.8 中以节点 35 为簇头节点的一组节点，其邻居节点为 1，2，3，34，36，37。

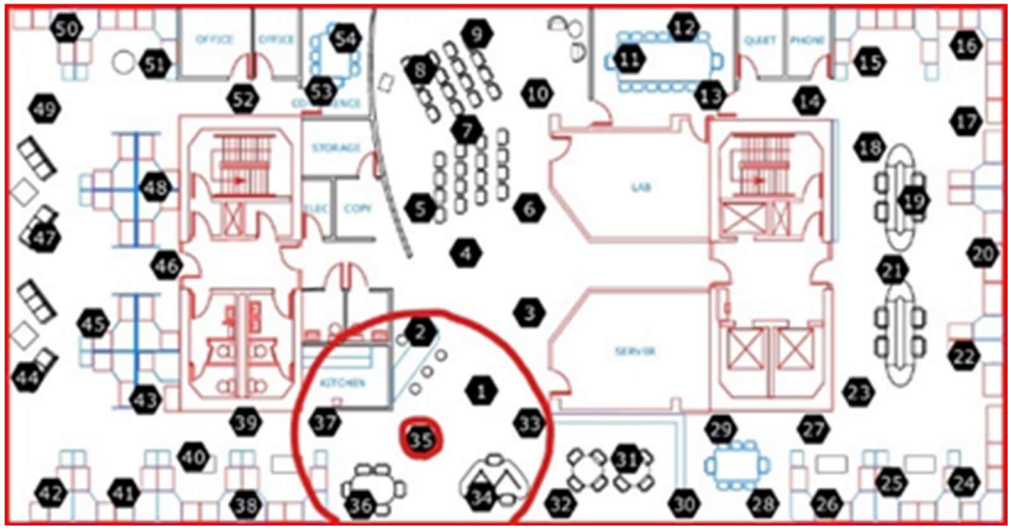


图 4.8 IBRL 无线传感器网络模型

我们使用 2004 年 2 月 28 号上午 9 点到下午 5 点的数据进行模型训练，如图 4.9 所示，2 月 29 号到 3 月 3 号的检测数据进行检验，如图 4.10 所示。为了更好的评估模型的性能，我们随机选取 50 个数据向量，对其温度值加上 ϑ服从均匀分布$U(2,4)$ 值进行仿真实验。同时，为了直观的比较三种模型下异常数据的误报率，对正常数据引入高斯白噪声，高斯白噪声的强度由信噪比(SNR)表示。

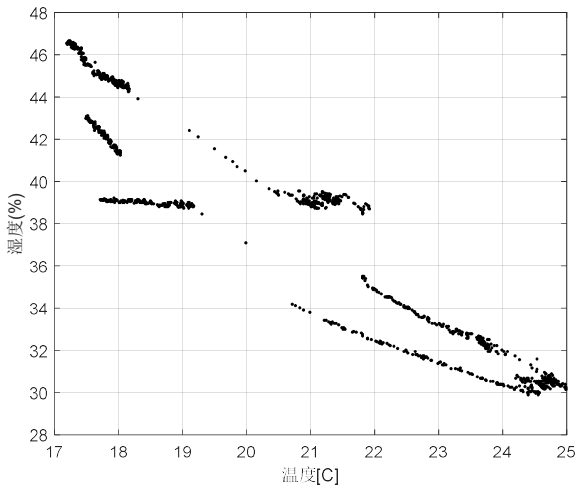


图 4.9 IBRL 训练数据集

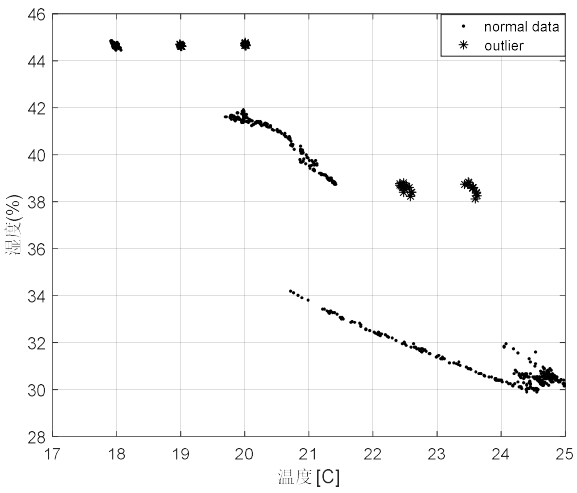


图 4.10 IBRL 测试数据集

4.5.2 仿真结果分析

本节首先给出了拟合数据集下，IDPCA 异常数据检测模型的 DR 和 FAR 随时间窗口的变化，接着给出了恶意事件的检测率和误报率随单个传感器节点异常数据 DR 和 FAR 变化的关系。最后在实际环境中，比较了 IDPCA 方案的检测性能和其它方案的检测性能。

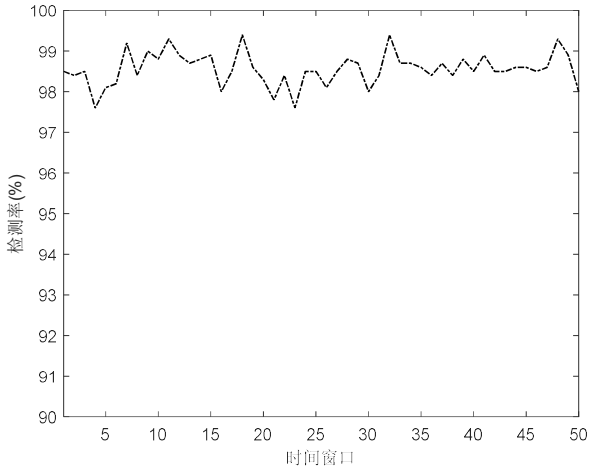


图 4.11 拟合数据集下 IDPCA 异常数据检测率

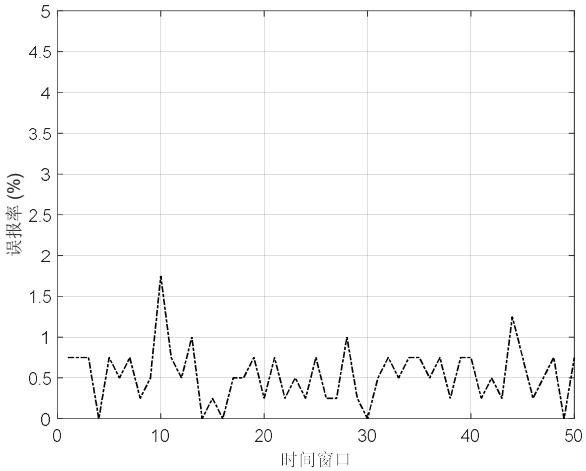


图 4.12 拟合数据集下 IDPCA 数据误报率

图 4.11 和 4.12 展示了 IDPCA 异常数据检测模型在拟合数据集下，不同时间窗口中的 DR 和 FAR。由图 4.11 可知 IDPCA 算法能够将 DR 稳定在 97%以上，平均的异常检测率能够达到 98%左右。由图 4.12 分析可知 IDPCA 检测模型能够达到一个较小的 FAR，基本能够维持在 0.5%~1%之间，是一个较小的误差值。

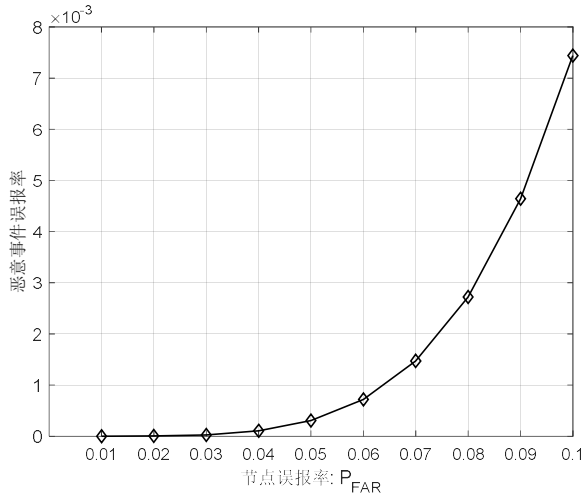
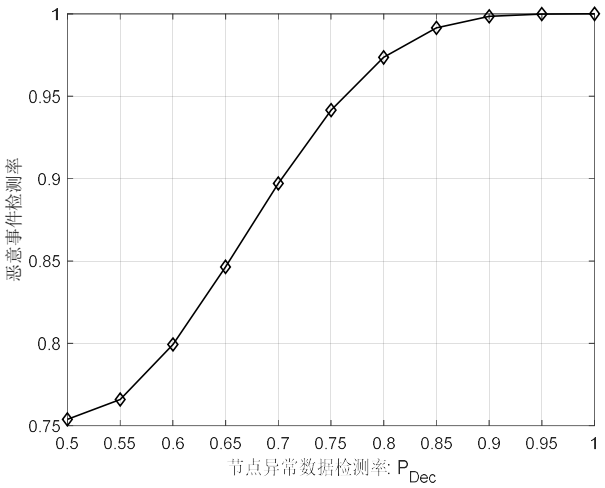


图 4.13 节点误报率和恶意事件误报率的关系



4.14 节点检测率和恶意事件检测率的关系

图 4.13 描述了单节点的异常数据误报率和异常数据源为恶意事件的误报率，由图分析可知网络中恶意事件的误报率随着单节点的误报率增大而增大，但是始终维持在一个非常小的值，说明方案对网络中的恶意事件存在极小的误报情况。图 4.14 描述了网络中恶意事件的检测率随每个节点异常数据检测率变化的关系，由图分析可知恶意事件的检测率随着节点异常数据检测率增加而增加且当节点的异常数据检测率为 0.7 以上时就能够达到 0.9 的恶意事件检测率，当节点的检测率超过 0.9 之后，恶意事件的检测率趋近于 1，这说明如果我们的方案能够达到 90%以上的异常数据检测率那么我们能够很准确的判断网络中是否有恶意事件发生。

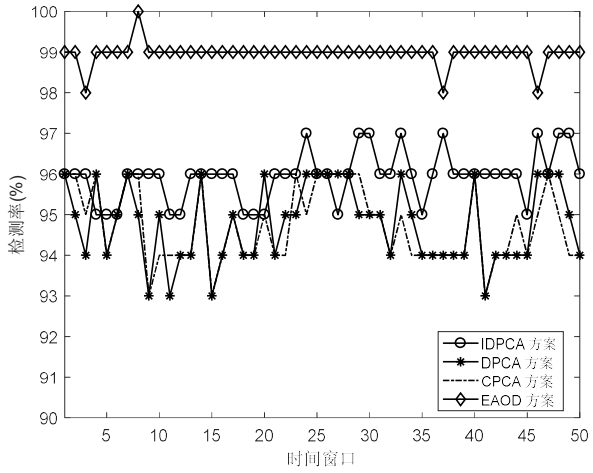


图 4.15 IBRL 数据集合下异常数据检测率对比

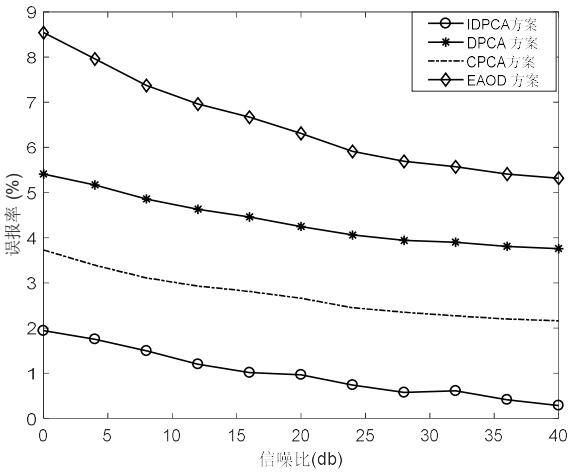


图 4.16 IBRL 数据集合下异常数据误报率对比

图 4.15 的结果说明了 IDPCA 异常检测模型能在实际场景中取得很好的效果，它能够达到近乎 96%-97%的异常数据检测率，比已提出的 DPCA 方案和 CPCA 方案达到的效果要更好，EAOD 方案使用的是基于 SVM 的异常检测方案，该方案的异常数据检测模型最高能够达到 99%的检测率，然而这并不能说明该方案的检测性能最优。另一方面，从图 4.16 我们可以看出，通过本方案提出的双重检测机制和模型更新策略，IDPCA 检测模型能够有效的降低异常数据的误报率，比已经提出的 DPCA 和 CPCA 方案更加的可靠,而 EAOD 方案虽然拥有最高的异常数据检测率，但是其异常数据的误报率在方案中也是最大的。由 4.4.3 小节分析结合本节实验结果我们可知 IDPCA 方案能够很好地检测网络中的恶意事件（DR>0.9）并且几乎不会出现误报的情况。

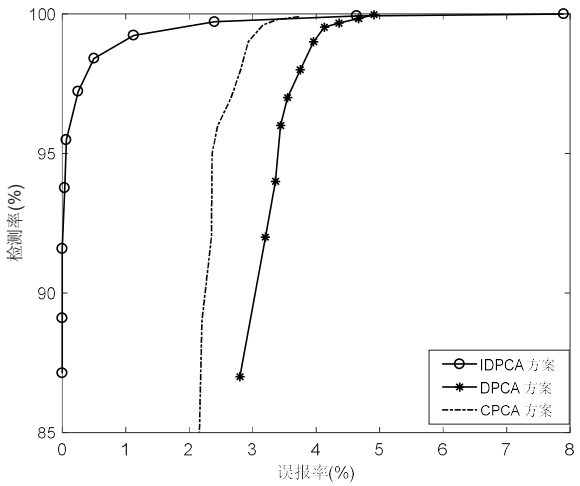


图 4.17 不同 PCA 方案的 ROC 曲线对比

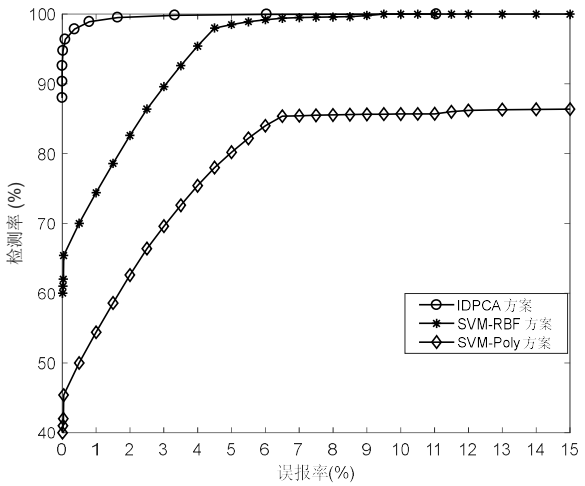


图 4.18 IDPCA 方案同 SVM 方案的 ROC 曲线对比

图 4.17 展示了 IDPCA 方案下 ROC 曲线和 DPCA,CPCA 的对比图，我们能够很轻易地看出 IDPCA 方案的性能要优于其余两种方案，它能在较低的误报率下保持较高的检测率, 更加

的符合实际场景。图 4.18 展示了 IDPCA 方案和 EAOD 方案[62]的 ROC 曲线对比，由图分析可知 IDPCA 方案能够在保证高检测率的同时保持低的误报率，它的检测性能要优于基于 SVM 的检测方案。综上所述，本方案在继承原有 PCA 异常检测模型的基础上，提高了模型的异常数据检测率、降低了误报率，同时对网络中的恶意事件能够很好地识别，该方案的可靠性更高，更加适合于实际场景。

4.6 本章小结

本章在已有的 PCA 异常数据检测模型的基础上提出了 IDPCA 异常数据检测模型，该模型充分利用传感器节点之间的空间相关性、节点数据间的时域相关性以及数据向量之间的维度相关性能够达到较高的异常数据检测率，同时维持较低的数据误报率。此外，通过提出的双重检测机制，我们能够很好地追踪异常数据产生的原因，并且高效地检测网络中的恶意事件。最后为了满足检测数据的实时变化，通过动态更新检测模型来保证模型的可用性。然而根据 4.4 节的分析我们可以发现 IDPCA 异常数据检测模型的计算步骤比较繁杂，而且网络中的通信消耗并不能达到常量级别，因此需要寻求更加高效地方案来解决该问题。同时针对部分线性不可分的数据集而言，PCA 算法并不能找到最佳的映射空间来反映数据的特征。针对以上问题我们在下一章中引入基于 KPCA 算法的分布式异常数据检测方案来检测传感器网络中的异常数据。

第五章 基于马氏内核的分布式 KPCA 异常数据检测方案

5.1 引言

在第四章中，我们提出了基于 PCA 算法的 IDPCA 异常数据检测模型，该模型能够很好的检测 WSN 的异常数据并追踪异常数据发生的原因。然而 PCA 检测算法只有在数据集为线性可分时能够达到的效果最好，对于线性不可分的数据向量，PCA 算法往往找到的映射坐标轴并不能达到最优。因此在 PCA 算法的基础上使用 KPCA 来处理监测数据为线性不可分的情况。KPCA 使用核函数将原始数据向量映射到高维特征空间，也就是把当前低维空间转换成高维空间，并在这个空间中计算特征向量。我们忽略特征值很低的向量，然后在这个变换的空间中训练模型。与 PCA 相比，KPCA 计算量大，耗时长得多。KPCA 中训练样本数目远高于 PCA，因此，需要估计的主成分分量也要大得多。与线性 PC 分析方法相比，KPCA 方法在处理非线性系统[71-72]时具有非常大的优势,文献[73]中详细介绍了 KPCA 算法的原理。本章提出基于马氏内核的 KPCA 方案用于 WSN 异常数据检测。为了检测异常值，使用马氏距离来衡量数据向量之间的相似度，通过模型训练得到最大的非相似度。如果新数据点的非相似度高于最大非相似度，观察值将被视为异常值。同时，借鉴第四章中的思路我们提出 IDKPCA 异常数据检测方案，使用双重检测机制来检测网络中的异常数据。最后，为了降低异常检测模型的更新频率，减小传感器节点的计算开销，通过固定时间窗口模型更新策略来更新异常检测模型，时间窗口大小由艾宾浩斯遗忘曲线来确定。实验结果表明：在英特尔伯克利实验室的数据下，IDKPCA 的异常数据检测率能够达到 98%且误报率在 2%左右，其检测效率要高于 IDPCA 方案和 EAOD 方案[62]，这是因为这些检测的数据是线性不可分的，KPCA 检测方案对于线性不可分的数据有更好的检测效果。

5.2 网络模型与马氏内核函数

假设 WSN 是一个静态网络，节点部署完成之后不再改变位置，节点之间可以进行双向通信且节点之间的密钥对已经提前匹配，非合法密钥之间的节点无法进行通信。整个网络部署在无人看守的野外环境下，传感器节点之间彼此同步。将节点按照组分散开来，每个组有一个组节点，假设同一个组中的节点监测的数据值是相近的，且都在组节点的通信范围内。令 $N(s_l)=\{s_i, i=1,2\cdots b\}$ 代表一组以 s_l 节点为组节点的一组传感器节点集合。在每一个时隙间

隔 Δt，集合中的每一个节点测量一组数据值，令 $x_l, x_1^l, x_2^l, \cdots, x_b^l$ 分别代表 $s_l, s_1, s_2, \cdots s_b$ 各个节点测量的数据向量。每一个数据向量拥有 d 维的数据：$x_k^l = \{x_{k1}^l, x_{k2}^l, \cdots, x_{kd}^l\}, x_k^l \in \Re^d$。通过寻找合适的核函数能将低维空间中线性不可分的数据映射到高维空间中进行线性分类，图 5.1 给出了 KPCA 的原理示意图。

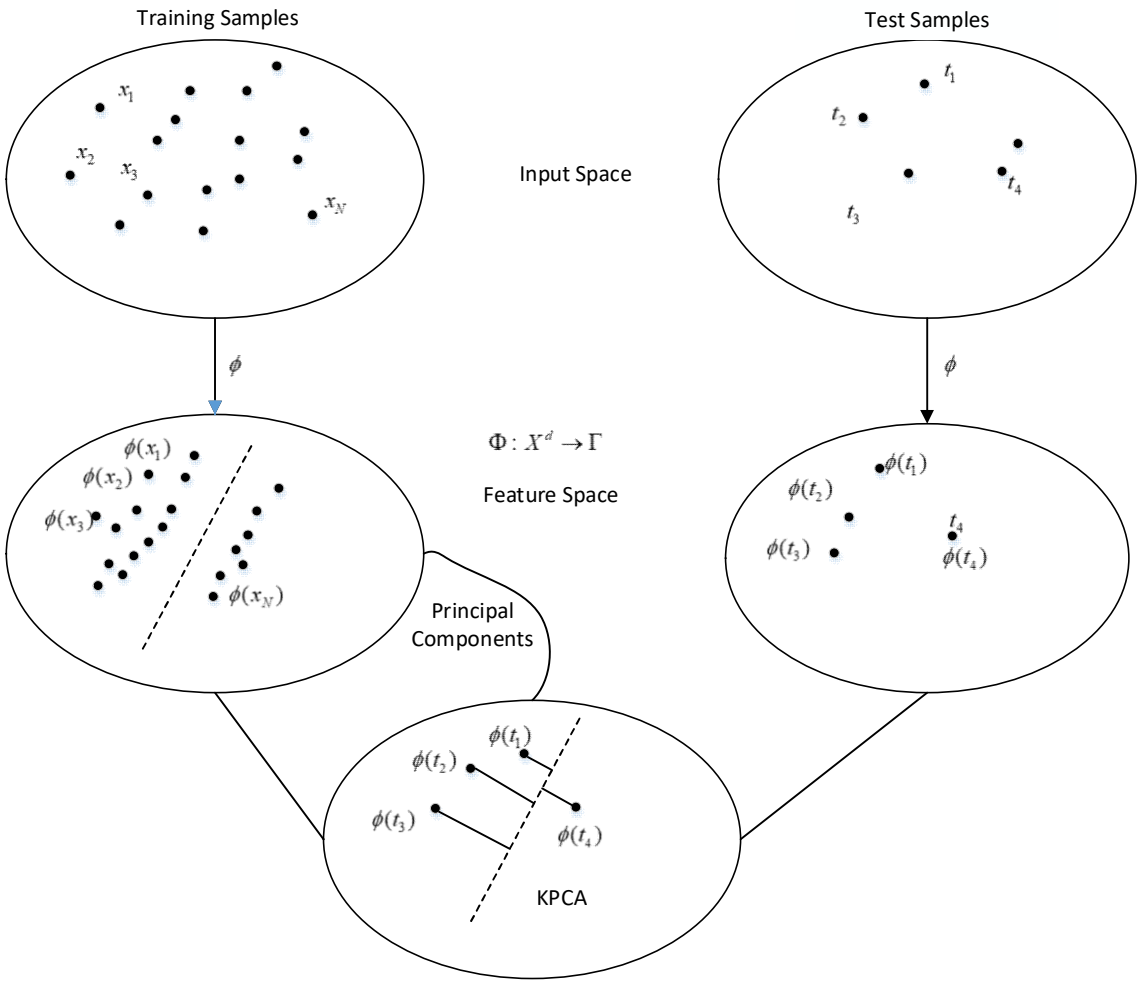


图 5.1 KPCA 原理示意图

经典的核函数包括径向基核函数(RBF)、多项式核函数等，马氏内核函数的应用并不广泛。马氏内核函数通过利用测量数据间数据值的相关性，将数据映射到高维空间中，马氏内核如式 5.1 所示：

$$K(x_i, x_j) = \exp\{\frac{-1}{2\sigma^2}(x_i - x_j)^T Q^{-1}(x_i - x_j)\} \tag{5.1}$$

令 $\{x_1, x_2, x_3 \ldots\ldots x_{n_i}\}$ 为节点 s_i 的 n_i 个 d 维数据向量，我们定义数据中心 c 为 $\frac{1}{n_i}\sum_{j=1}^{n_i} x_j$ 和协方差矩阵 Q 如式 5.2 所示：

$$Q=\frac{1}{n_i}\sum_{j=1}^{n_i}(x_j-c)^T(x_j-c) \tag{5.2}$$

某个数据节点到数据中心的马氏距离为：

$$d(x)=\sqrt{(x-c)^T Q^{-1}(x-c)} \tag{5.3}$$

定义马氏内核函数如式 5.4 所示:

$$A(x,x')=\exp(-(x-x')^T H(x-x')) \tag{5.4}$$

其中 H 为一个正半定矩阵，当 $H=\gamma I$，$\gamma>0$，I 为单位矩阵时，马氏内核可以看作是特殊的 RBF 内核函数，在实际场景中马氏内核一般定义为式 5.5 所示:

$$A(x,x')=\exp(-\frac{\delta}{m}(x-x')^T Q^{-1}(x-x')) \tag{5.5}$$

径向基核函数 RBF 也称为高斯核函数，其表达式如式 5.6 所示：

$$K(x_i,x_j)=\exp(\frac{-||x_i-x_j||_2^2}{2\sigma^2}) \tag{5.6}$$

马氏内核与高斯内核的不同之处在于，它为输入空间数据的每个维度定义了一个特定的深度值或权重。这使得计算出的判定边界相对于数据点的中心具有非球形形状。

5. 3 IDKPCA 异常数据检测模型

本节详细介绍所提出的 IDKPCA 异常数据检测方案，该方案主要分为四个阶段。异常数据检测模型训练阶段、异常数据检测阶段，异常数据源检测阶段、异常数据模型更新阶段。其中异常数据源的检测与第四章中的思路一致，本节不再单独提出。

5.3.1 IDKPCA 异常数据检测模型训练

模型训练阶段旨在为每一个节点单独地建立起正常数据模型。假设传感器节点 s_i 在 m 个时隙窗口中收集到的 n_i 个监测数据向量 $[x_1,x_2,x_3.....x_{n_i}]$。在高维空间中我们定义映射 $\Phi: X^d\rightarrow\Gamma$，这里 Γ 表示 Hibert 泛函空间，通过该映射每个监测数据映射为 $\phi(x_i)$,它们的维数理论上可以是无穷的，在这个新的空间中协方差矩阵可以定义为式 5.7 所示：

$$C=\frac{1}{n_i}\sum_{j=1}^{n_i}\phi(x_j)\phi(x_j{}^T)=\frac{1}{n_i}[\phi(x_1),\phi(x_2),.....\phi(x_{n_i})]\begin{bmatrix}\phi(x_1{}^T)\\ \phi(x_2{}^T)\\ .....\\ \phi(x_{n_i}{}^T)\end{bmatrix} \tag{5.7}$$

令 $X^T=[\phi(x_1),\phi(x_2),...\phi(x_{n_i})]$,那么 $C=\frac{1}{n_i}X^TX$，如果需要求解特征值的话那么需要计算矩阵 X^TX 的特征值，但是由于映射函数 ϕ 是未知的所以无法计算，根据 KPCA 的推导有如下的结果：

$$Ku_i=\lambda_i u_i \qquad \lambda_1\geq\lambda_2\geq\lambda_3.....\geq\lambda_{n_i} \tag{5.8}$$

$$v_i{}^T\phi(x_j)=\frac{1}{\sqrt{\lambda_i}}u_i{}^T\begin{bmatrix}K(x_1,x_j)\\ .....\\ K(x_{n_i},x_j)\end{bmatrix} \tag{5.9}$$

其中 K 是由核函数确定的内积值矩阵，如式 5.10 所示. $v_i{}^T\phi(x_j)$ 为高维空间中向量 $\phi(x_j)$ 向 v_i 方向的投影值。

$$K=\begin{bmatrix}K(x_1,x_1)\cdots\cdots K(x_1,x_{n_i})\\ K(x_2,x_1)\cdots\cdots K(x_2,x_{n_i})\\ .....\\ K(x_{n_i},x_1)\cdots\cdots K(x_{n_i},x_{n_i})\end{bmatrix} \tag{5.10}$$

在模型训练开始时，对数据向量 x_i 进行归一化，保留数据均值 μ 和标准差 Std，接着计算 K 矩阵的归一化矩阵 $\tilde{K}$ =K-$l_NK-Kl_N+l_NKl_N$ 并求出归一化矩阵对应的 v_i 和 λ_i。令 λ 为矩阵 $\tilde{K}$ 特征值的的总和，即：

$$\lambda=\lambda_1+\lambda_2+\lambda_3.....+\lambda_{n_i} \tag{5.11}$$

在高维空间中主成分个数比低维空间中要多，为了能够获取数据的绝大部分特征值，我们参照主成分分量的得分来确定选择的主成分个数。实验表明当主成分的得分大于 0.9 时，新的映射坐标能够代表原数据的特征，主成分个数的获取由式 5.12 确定。

$$\begin{cases}PC\ Score=\sum_{i=1}^{\chi}\lambda_i/\lambda & 1\leq i\leq n_i\\ PC\ Score>=0.9\end{cases} \tag{5.12}$$

构建模型的主成分分量个数由参数 χ 决定，χ 确定之后紧接着通过计算每个训练样本的非相

似度 Dis 并找出其中最大的非相似度值 $Dis_{\max}$ 来定义异常检测模型阈值,非相似度 Dis [70]的定义如式 5.13 所示：

$$Dis_j = \sum_{i=1}^{\chi} \frac{v_i^T \phi(x_j)\phi^T(x_j)v_i}{\lambda_i} \qquad 1 \le j \le n_i, 1 \le i \le \chi \tag{5.13}$$

最后，节点 s_i 通过保留六元组 $(\mu, Std, v, \lambda, \chi, Dis_{\max})$ 来构建本地异常数据检测模型。对于簇头节点，上一章使用相应的融合算法对协方差矩阵和均值进行了重构得到全局的异常检测模型。这种方案计算步骤比较繁琐而且涉及到协方差矩阵的传输，并没有对网络中的通信资源进行最合理的优化。因此结合文献[84]中的思路,令每个邻居节点将各自的 $Dis_{\max}$ 发送到簇头节点中，然后簇头节点计算全局最大非相似度 $Dis(global)=Median(Dis_{\max 1}, Dis_{\max 2}.....Dis_{\max b})$ 来定义全局异常数据检测模型。

表 5.1 IDKPCA 异常数据检测模型的建立

for $i = 1:b$
  s_i *collects* n_i *data vectors* $x_i.....x_{n_i}$
  x_i *normalization and keep mean value* μ *and Std*
  s_i *compute Matrix K by Mahalanobis kernel*
  s_i *get the eign value and first* χ_{th} *components*
  s_i *compute* $Dis_{\max}$
  s_i *use* $\{\mu, Std, v, \lambda, \chi, Dis_{\max}\}$ *as the local training model*
  s_i *send* $Dis_{\max}$ *to the group node* s_l
end
then
  s_l *compute* $Dis_{\max}(global)$

each group operates like that

5.3.2 IDKPCA 异常数据检测

检测阶段分为局部检测和全局检测。初始化阶段，每个传感器节点通过它在 m 个时间窗口中获取的训练数据计算六元组 $(\mu, Std, v, \lambda, \chi, Dis_{\max})$。接着每个节点将它能容忍的最大非相似度 $Dis_{\max}$ 广播到其相邻的簇头节点中，簇头节点根据各个节点发送的最大非相似度 $Dis_{\max}$ 计算全局最大非相似度 $Dis(global)$，通常情况下该全局最大非相似度取的是各个节点最大非相似度的中位数[84]。当一个新的数据向量 $x_i(t)$ 到达节点 s_i,节点 s_i 首先根据其 (μ, Std) 将当前的

数据向量标准化，然后通过核函数计算 PC Score 得到 $Dis(c)$，接着比较 $Dis(c)$ 和 $Dis_{\max}$。需要特别注意的是 $v_i^T\phi(x_j)$ 的计算，此处 $x_i(t)$ 先进行标准化得到 $x_i'(t)$。由式 5.1、5.9 和 5.13 得到式 5.14:

$$Dis(c)=\sum_{i=1}^{\chi}\frac{\sum_{j=1}^{N}\exp(\frac{-1}{2\sigma^2}(x_j-x_i'(t))^T Q(x_j-x_i'(t)))^2}{\lambda_i^{\,2}} \qquad 1\le j\le n_i, 1\le i\le \chi \tag{5.14}$$

如果 $Dis(c)<Dis_{\max}$，那么当前数据向量被认为是一个正常的数据，否则该数据向量被标记为可疑数据，并将其 $Dis(c)$ 发送到簇头节点中和 $Dis(global)$ 进行比较，如果 $Dis(c)>Dis(global)$，那么 $x_i(t)$ 最终被认为是一个异常数据，否则将其可疑标记去除，认为其是一个正常数据。因此，最终的异常数据判别函数如式 5.15 所示:

$$f(x)=sgn(Dis(c)-Dis_{\max})\oplus sgn(Dis(c)-Dis(global)) \tag{5.15}$$

根据式 5.5，如果 $f(x)$ 的最终结果等于-1，那么该数据被认为是一个异常数据。

$$f(x)=\begin{cases}1 & \text{normal data}\\ -1 & \text{outlier}\end{cases} \tag{5.16}$$

表 5.2 给出了 IDKCPA 异常数据检测模型下，异常数据的检测流程：

表 5.2 IDKPCA 异常数据检测流程

```
when x_i(t) arrives at s_i,
    s_i normalize x_i(t) to x_i'(t) by (μ, Std)
    s_i use (v, λ, χ, Dis_max) to computes Dis(c)

    if (Dis(c) > Dis(max) && Dis(c) > Dis(global))
      then x_i(t) indicates an oulier
      operate sourceOfOulier()
    end
    else
      x_i(t) indicates an normal data
    end
end
```

5.3.3 固定时间窗口模型更新

随着无线传感器网络部署环境的变化，异常数据检测模型需要进行相应的的更新。异常数据检测模型的更新，能有效的提高异常数据的检测效率，该方案的核心思想是减少对旧数

据的依赖性，根据数据的实时性来提高模型的精确性。在上一章中我们采用实时更新策略来保证模型的可靠性，在每个时间窗口中接受到新数据时根据新数据是否是异常数据来更新模型。一般情况下，网络中的正常数据值占的比例很大，这意味着模型更新的速度非常频繁，这会给网络带来大规模的计算资源开销。为了降低模型更新的速度且不影响模型的性能，我们提出了固定时间窗口模型更新策略，如图 5.2 所示。

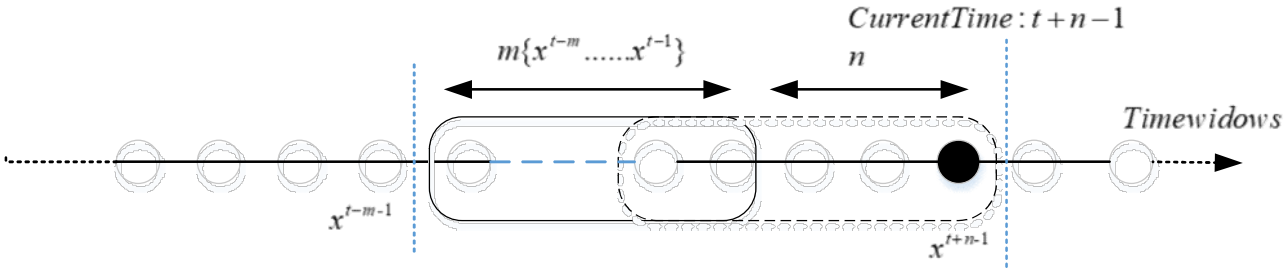


图 5.2 固定时间窗口模型更新策略

其中 n 表示经过 n 个时间窗口对模型进行更新，即重新计算全局的六元组 $(\mu, Std, v, \lambda, \chi, Dis_{\max})$。为了确定合适的 n 值，引入艾宾浩斯遗忘曲线。艾宾浩斯遗忘曲线揭示了人们在学习中的遗忘是有规律的，遗忘的过程并不是呈现简单的线性关系而是处于一种非均衡的变化。将艾宾浩斯遗忘曲线应用于机器学习中来舍弃旧数据，寻找合适的时间窗口间隔来进行模型更新，其表达式如式 5.17 所示。

$$R = e^{-t/s} \tag{5.17}$$

其中 s 代表的是记忆强度，t 代表记忆时间，R 代表记忆量。图 5.3 给出了记忆量与记忆时间窗口的关系，当 $R <= 0.5$ 时我们对模型进行更新。

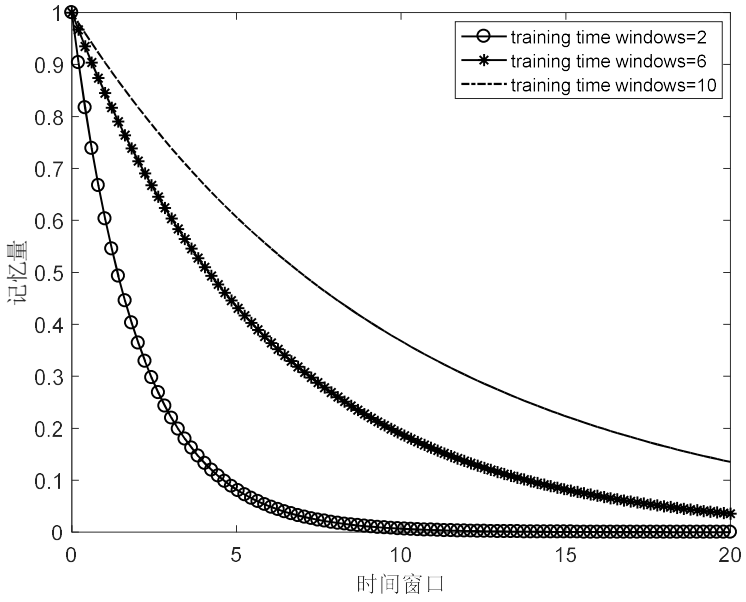


图 5.3 记忆量与记忆时间窗口的关系

5.4 性能分析

本节分析 IDKPCA 模型建立需要花费的通信资源和计算资源损耗。

假设计算资源开销为 IDKPCA 检测方案运行时，网络中因为 IDKPCA 检测方案而引起的额外的计算资源。模型建立阶段每个邻居节点需要对数据向量进行标准化，这个过程的计算资源耗费为 n_i（n_i 为节点的训练样本数），接着通过计算 K 矩阵进行特征分解得到 PC Score 计算 $Dis_{\max}$，这个过程中通过核函数计算得到 K 矩阵的资源耗费为 $n_i^2(d^2+d)$，马氏内核中的协方差矩阵的计算复杂度为 $n_i d^2$，最后在得到主成分分量之后需要计算 $Dis_{\max}$ 值，这个过程的计算损耗为 χn_i^2，所以邻居节点的计算资源总消耗为 $n_i^2(d^2+d+\chi)+n_i(1+d^2)$。对于簇头节点而言，它需要额外的计算 $d_l(global)$ 这部分的计算开销为 b，所以簇头节点的计算开销为 $n_i^2(d^2+d+\chi)+n_i(1+d^2)+b$，所以总的计算开销为 $(b+1)*(n_i^2(d^2+d+\chi)+n_i(1+d^2))$。

在所提出的 IDKPCA 方案中,每一个传感器分组拥有 b 个邻居节点，每个邻居节点在计算五元组 $(\mu, Std, v, \lambda, \chi, Dis_{\max})$ 之后将 $Dis_{\max}$ 发送给簇头节点进行全局的 $Dis(global)$ 计算，对于邻居节点而言其通信损耗为 1，而簇头节点为 b。如果任意一个邻居节点 s_i 本地异常检测未通过那么节点将当前向量值计算的 $Dis(c)$ 发送到簇头节点进行全局检测，这个过程中最坏的情况下每一个邻居节点需要额外增加 1 通信损耗，簇头节点需要额外增加 b。因此在所提出的方案中邻居节点的通信损耗为 2,簇头节点的通信损耗为 $2b$，总的通信损耗为 $4b$。

表 5.3 IDKPCA 方案的资源消耗同其它方案的比较

	网络中通信损耗	邻居节点的计算损耗	簇头节点的计算损耗
CPCA 异常检测模型	$O(nd)$	-	$O(nd^2)$
DPCA 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$
IDPCA 异常检测模型	$O(b\chi^2)$	$O(n_i d^2)$	$O(n_i d^2)$
EOOD 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$
EAOD 异常检测模型	$O(bd^2)$	$O(n_i d^2)$	$O(n_i d^2)$
IDKPCA 异常检测模型	$O(1)$	$O(n_i^2 d^2)$	$O(n_i^2 d^2)$

各个方案下的资源消耗如表 5.3 所示，其中 b 为常量值代表邻居节点的个数且不超过 50。IDKPCA 方案下异常数据检测模型的通信损耗是最少的达到了常量级别，虽然该方案的计算步骤简洁但是引入的计算资源开销却随着训练样本的增加而大幅度增加。为了克服频繁地模型更新带来的巨大计算资源开销，我们提出了 5.3.3 节中的策略有效地降低模型的更新频率。

5.5 仿真分析

本节我们分析所给出的 IDKPCA 方案的 DR 和 FAR 随核函数参数 σ 变化的关系，接着我们比较 IDKPCA 方案和 IDPCA、EAOD 方案的异常数据检测率，最后比较了这三种方案的 ROC 曲线和曲线所围面积(Area Under Curve,AUC)。

5.5.1 仿真环境设定

本节我们依旧使用英特尔伯克利的实验数据来进行模型分析，该测量数据值包括测量温度(temperature),环境湿度(humidity)，光照强度和传感器电压(Voltage)。在上一章中仅仅使用了环境温度和湿度来进行模型的性能分析，然而传感器节点的电压值也是一个很重要的因素，通过电压值异常能够检测出无效节点。传感器节点测量电压值的正常范围在[2,3]之间且其变化和环境中的温度密切相关[85],在本节中我们使用温度值、湿度值和传感器电压来进行模型的训练和检测。我们使用 2004 年 2 月 28 号上午 9 点到下午 5 点的 1200 个训练数据进行模型训练，如图 5.4 所示。

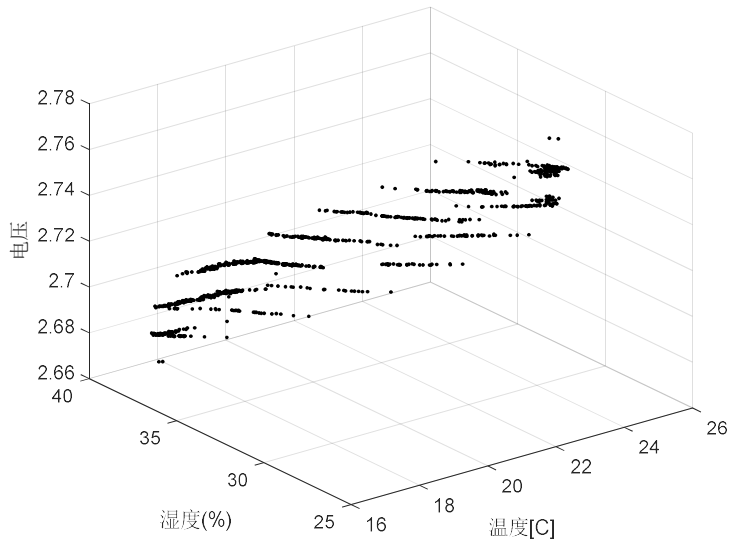


图 5.4 IBRL 训练数据集

为了突出 IDKPCA 方案的优势我们在测试数据集中随机选取 50 个数据向量插入电压异

常值服从均匀分布$U(3,3.2)$，同时选取 50 个数据向量(和电压异常向量不重复)温度值加上ϑ服从均匀分布$U(2,4)$值进行仿真实验，如图 5.5 所示。令$\gamma=-\frac{1}{2\sigma^2}$。为了寻找最佳的核参数使得模型的 DR 和 FAR 达到最佳，我们令γ的值从 1 逐渐递增到 2001，观察参数对模型性能的影响。

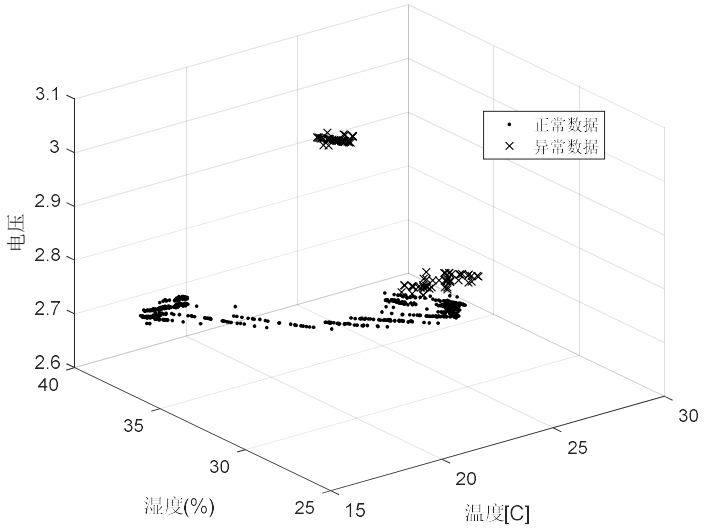


图 5.5 IBRL 测试数据集

5.5.2 仿真结果分析

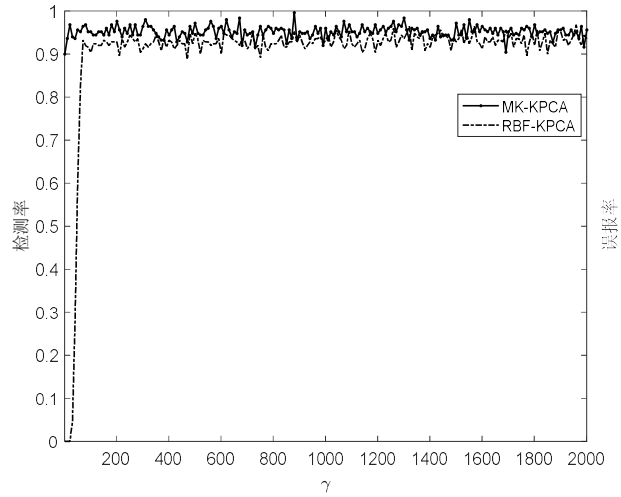


图 5.6 参数γ与模型检测率的关系

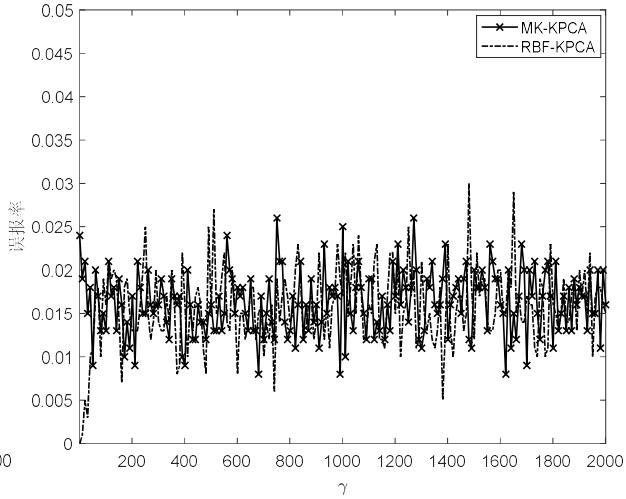


图 5.7 参数γ与模型误报率的关系

图 5.6 展示了基于高斯径向核函数的 KPCA 检测方案和基于马氏内核函数的 KPCA 检测

方案异常数据检测率随参数γ值的变化关系。由图分析可知，基于 RBF 高斯径向核函数当γ很小的时候模型的检测性能很差，而基于马氏核函数的异常数据检测方案的性能受该参数的影响较小，模型的异常检测率维持在 90%以上。同时，由图我们能发现基于马氏内核的 KPCA 检测方案的异常数据检测性能要优于基于 RBF 内核的 KPCA 检测方案，该方案的异常检测率要比 RBF-KPCA 平均高 0.01 到 0.02 左右，同时发现在参数γ=[800,1000]内出现检测率的峰值。

图 5.7 给出了基于高斯径向核函数的 KPCA 检测方案和基于马氏内核函数的 KPCA 检测方案异常数据误报率随参数γ值的变化关系。由该图分析可知，两种内核函数下的异常数据检测模型都能够达到很小的数据误报率，基本能维持在 3%以下，平均误报率在 1.5%左右。结合异常数据的检测率的峰值在[800,1000]的区间中出现，我们取参数值γ=901来进行其余实验，在该参数下基于马氏内核的 KPCA 检测方案的平均检测率为 98%，平均误报率为 1.4%。

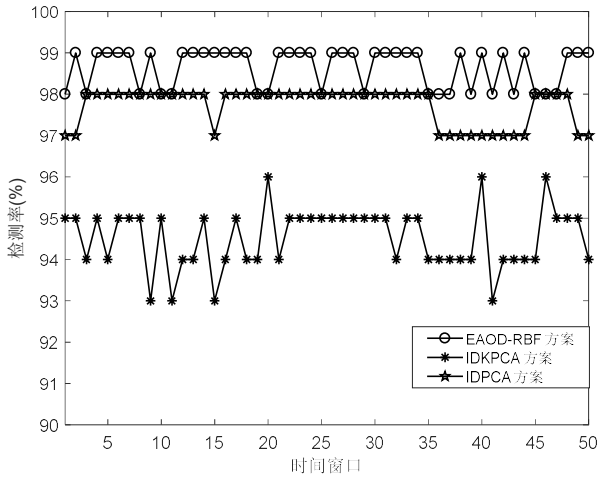


图 5.8 不同模型下异常数据检测率的比较

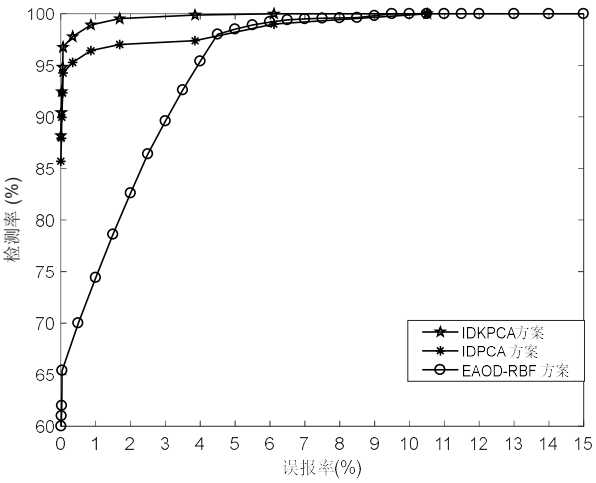


图 5.9 不同模型的 ROC 曲线比较

图 5.8 给出了γ=901时 IDKPCA 方案和 IDPCA 等方案的异常数据检测率对比图。由图分析可知在不同的时间窗口中 IDKPCA 方案下模型的异常数据检测率稳定在 97%-98%之间，是一个非常可靠的检测效率。同时，IDPCA 方案的异常数据检测率在 94%~95%之间，相比于线性可分的数据集而言，其检测效率下降了一些。EAOD 方案的异常数据检测率最高,达到了 98%~99%,但是这并不能说明该方案的检测效率是最好的。图 5.9 给出了 IDKPCA、IDPCA 和 EAOD 异常检测模型的 ROC 曲线，由图可知 EAOD 方案下模型的异常数据检测率高的同时异常数据的误报率也相对较大，而 IDKPCA 方案和 IDPCA 方案能够保证异常数据的检测率在 95%以上且数据的误报率维持在 2%左右，且在相同的误报率下 IDKPCA 方案的异常数据

检测率要高于 IDPCA 方案。图 5.10 进一步给出了三种异常检测模型下方案的 ROC 曲线所围面积在每个时隙中的平均值，AUC 越大模型的检测性能越好。由图 5.10 我们可知，IDKPCA 方案下模型的 AUC 值在 0.94-0.95 之间，IDPCA 方案的 AUC 值在 0.90-0.94 之间，EAOD 方案的 AUC 值在 0.87~0.88 之间。因此在检测性能这一方面，IDKPCA 方案能够达到最好的效果。

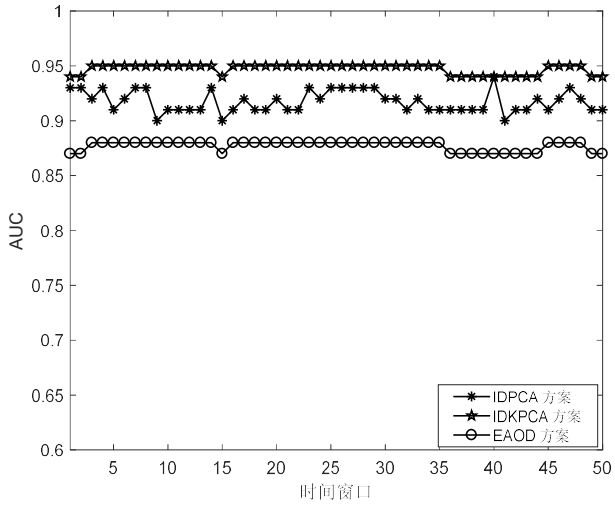


图 5.10 不同模型下 ROC 曲线所围面积的的比较

5.6 本章小结

本章在第四章的基础上，为了解决线性不可分的异常数据检测，提出了基于马氏内核的 IDKPCA 异常数据检测模型，该方案能够很好地检测线性不可分的异常数据，达到 98%的异常数据检测率的同时能够维持异常数据的误报率在 2%左右。同时，我们改进了第四章中全局异常检测模型的训练，有效的减少了网络中的通信资源消耗并且引入了固定时间窗口更新机制，大大降低了模型更新的频率，减小了每个节点的计算资源消耗。然而 KPCA 方案引入了核函数的计算并且需要计算最佳的核参数，这在一定程度上也加大了计算的复杂度。未来仍需要继续探究如何进一步的减小节点的计算复杂度。

第六章 总结与展望

6.1 文章总结

文章针对 WSN 中的恶意蠕虫节点检测和异常数据检测提出了新的解决方案。

首先，WSN 中的恶意节点是威胁传感器网络安全最主要的因素，一旦网络中存在恶意节点那么整个网络都将处于危险状态。已有的研究解决了很大一部分恶意节点的检测，但是对于恶意蠕虫节点并没有提出有效地检测方案。本文以 SPRT 算法为基础，结合偏向采样方案和随机值采样方案提出的 SPRT-Biased-Random 方案能够有效地检测 WSN 中的蠕虫节点，并在 5-18 个时隙内完全检测并移除传感器网络中的蠕虫节点，使整个网络受感染的节点维持在 2%到 5%之间。此外，该方案计算方式简单，所需存储资源和通信资源也相对较少，能够很好地适用于资源受限的 WSN。

其次，第四章提出了 IDPCA 异常数据检测方案来检测 WSN 中的异常数据。方案的核心思想是通过 PCA 来寻找数据的最大特征方向，使用马氏距离来衡量数据的相似度。同时我们充分的利用了传感器节点之间的空间相关性，通过局部检测和全局检测来提高模型的异常数据检测率、降低异常数据的误报率。IDPCA 方案能够有效地识别传感器网络中的异常数据源，我们针对网络中的恶意事件源做出了详细的分析，分析表明在单个节点的异常数据检测率达到 90%以上时，能够对网络中发生的恶意事件进行很好地检测。总结而言，基于 IDPCA 的异常数据检测方案能够降低网络中的通信负载，同时相比于其它方案 IDPCA 方案能够提高异常数据的检测率，降低数据的误报率[86]。

最后在第五章中我们基于 IDPCA 异常数据检测方案无法对线性不可分的数据进行最佳地检测，提出了基于马氏内核的 IDKPCA 异常数据检测方案，该方案中还引入了固定时间窗口模型来定期更新异常检测模型，大大减小了模型的更新频率，减小了传感器节点的计算资源开销。实验结果表明：IDKPCA 方案能够有效地检测线性不可分数据，异常数据的检测性能要较优于 IDPCA 方案，但是 IDKPCA 中引入了核函数，计算复杂度要比 IDPCA 方案高。

6.2 未来工作展望

虽然无线传感器网络安全的相关研究已经相当成熟。本文对针对无线传感器网络中的蠕虫节点检测和异常数据检测提出了新的方案，为了进一步提高无线传感器网络的安全，未来的工作可以从以下几个方面进行研究。

（1） 本文针对的是静态传感器网络中的蠕虫节点检测，但是并没有针对移动传感器网络进行检测。移动传感器网络中的蠕虫节点始终在移动，一旦定位之后，如果继续移动的话那么本文的方案并不能实时地去除该恶意节点，这样的话有可能蠕虫节点会持续感染整个网络使得网络变得不可用。

（2） 本文提出的基于 PCA 的异常数据检测方案基于矩阵的特征分解，计算复杂度相对较高。另外通过充分利用了节点之间的空间相关性和时域相关性本文中的方案能够达到高的检测性能，但是如果节点移动之后，检测数据的时域性和节点间的空间相关性就会大大改变，这样的话，本文所提出的模型就不再适合了。需要将模型扩展到移动网络中。

到目前为止，无线传感器网络安全的研究成果主要集中在理论分析和仿真分析。其中仿真软件像 NS2,OMNET 操作复杂，而且仿真的效果并没有很强。目前缺乏高效地仿真平台能够很好地进行仿真实验。

参考文献

[1] Yessembayev, A., et al. (2018). "Detection of Good and Bad Sensor Nodes in the Presence of Malicious Attacks and Its Application to Data Aggregation." IEEE Transactions on Signal and Information Processing over Networks 4(3): 549-563.

[2] Padmaja, P. and G. V. Marutheswar (2017). Detection of milicious node in wireless sensor networks. 2017 International Conference on Computer Communication and Informatics (ICCCI).

[3] Osanaiye, O. A., et al. (2018). "Denial of Service Defence for Resource Availability in Wireless Sensor Networks." IEEE Access 6: 6975-7004.

[4] Wang, Y., et al. (2017). An information-centric multiple-source routing scheme for wireless sensor networks. 2017 IEEE International Conference on Smart Energy Grid Engineering (SEGE).

[5] Tariq, M., et al. (2018). Detection of False Data in Wireless Sensor Network Using Hash Chain. 2018 International Conference on Applied and Engineering Mathematics (ICAEM).

[6] Hoang, T., et al. (2017). Adaptive routing in wireless sensor networks under electromagnetic interference. 2017 International Conference on Information Networking (ICOIN).

[7] Ahmed, A., Bakar, K.A., Channa, M.I. et al. Peer-to-Peer Netw. Appl. (2017) 10: 216.

[8] Sreevidya, B. and M. Rajesh (2018). False Data Injection Prevention in Wireless Sensor Networks using Node-level Trust Value Computation. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

[9] López-Valcarce, R. and D. Romero (2017). Defending surveillance sensor networks against data-injection attacks via trusted nodes. 2017 25th European Signal Processing Conference (EUSIPCO).

[10] Liu, X., et al. (2018). "Bayesian Test for Detecting False Data Injection in Wireless Relay Networks." IEEE Communications Letters 22(2): 380-383.

[11] Thompson, S. A. and B. K. Samanthula (2017). Optimized Secure Data Aggregation in Wireless Sensor Networks. 2017 15th Annual Conference on Privacy, Security and Trust (PST).

[12] Karthikeyan, B., et al. (2017). Energy efficient data compression and aggregation technique forwireless sensor networks. 2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2).

[13] Wang, T., et al. (2017). An Efficient and Secure Itinerary-Based Data Aggregation Algorithm for WSNs. 2017 IEEE Trustcom/BigDataSE/ICESS.

[14] Nehra, P. and A. Nagaraju (2017). Fault Tolerance using Quadratic-Minimum Spanning Tree (Q-MST) with Secure Data Aggregation in Wireless Sensor Networks. 2017 14th IEEE India Council International Conference (INDICON).

[15] Mukartihal, P. P. and D. I. Hatti (2017). Client Puzzle Approach for Securing Aggregated Data in Wireless Sensor Network. 2017 IEEE 7th International Advance Computing Conference (IACC).

[16] Ayadi, A., et al. (2017). Outlier detection based on data reduction in WSNs for water pipeline. 2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM).

[17] İ, K. and T. Ercan (2018). Anomaly Detection in Wireless Sensor Networks Data by Using Histogram Based Outlier Score Method. 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT).

[18] Dwivedi, R. K., et al. (2018). A Study on Machine Learning Approaches for Outlier Detection in Wireless Sensor Network. 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence).

[19] Zhang, W., et al. (2018). "A novel trust management scheme based on Dempster–Shafer evidence theory for malicious nodes detection in wireless sensor networks." The Journal of Supercomputing 74(4): 1779-1801.

[20] Amudha, G. and P. Narayanasamy (2018). "Distributed Location and Trust Based Replica Detection in Wireless Sensor Networks." Wireless Personal Communications 102(4): 3303-3321.
[21] Zeiser, M. and D. Westhoff (2016). Re-visited: On the Value of Purely Software-Based Code Attestation for Embedded Devices. Innovations for Community Services, Cham, Springer International Publishing.
[22] Lee, J. and T. Kwon (2016). "Secure dissemination of software updates for intelligent mobility in future wireless networks." EURASIP Journal on Wireless Communications and Networking 2016(1): 250.
[23] Y. Yang, X. Wang, S. Zhu, and G. Cao. Distributed software-based attestation for node compromise detection in sensor networks. In IEEE SRDS, October 2007.
[24] Jun-Won, H., et al. (2012). "ZoneTrust: Fast Zone-Based Node Compromise Detection and Revocation in Wireless Sensor Networks Using Sequential Hypothesis Testing." IEEE Transactions on Dependable and Secure Computing 9(4): 494-511.
[25] Thaile, M. and O. B. V. Ramanaiah (2016). Node Compromise Detection based on NodeTrust in Wireless Sensor Networks. 2016 International Conference on Computer Communication and Informatics (ICCCI).
[26] J. Ho, D. Liu, M. Wright, and S.K. Das. Distributed detection of replica node attacks with group deployment knowledge in wireless sensor networks. In Elsevier Ad Hoc Networks Journal, 7(8):1476–1488, 2009
[27] Rahman, M. U., et al. (2017). Lightweight detection of malicious nodes in mobile ad hoc networks. 2017 International Conference on Communication Technologies (ComTech).
[28] Z. Li, W. Xu, R. Miller, and W. Trappe. Securing wireless systems via low layer enforcements. In ACM WiSe, Pages:33–42, September 2006
[29] N. Patwari and S. K. Kasera. Robust location distinction using temporal link signatures. In ACM MobiCom, Pages:111–122, September 2007.
[30] J. Zhang, M. Firooz, N. Patwari, and S. K. Kasera. Advancing wireless link signatures for location distinction. In ACM MobiCom, Pages:26–37, September 2008.
[31] J.-W. Ho, ‘‘A framework for robust detection and prevention of wide-spread node compromise in wireless sensor networks,’’ Doctoral dissertation, Dept. Comput. Sci. Eng., Univ. Texas Arlington, Arlington, TX, USA, 2010.
[32] Al-Riyami, A., et al. (2016). "An Adaptive Early Node Compromise Detection Scheme for Hierarchical WSNs." IEEE Access 4: 4183-4206.
[33] Geedhabhanu, S. and P. Latha (2015). Excluding compromised node by tracing false data injected messages in Wireless Sensor Network. International Confernce on Innovation Information in Computing Technologies.
[34] M. Conti, R.D. Pietro, L.V. Mancini, and A. Mei. A Randomized, Efficient, and Distributed Protocol for the Detection of Node Replication Attacks in Wireless Sensor Networks. In ACM Mobihoc, pages:80-89, 2007.
[35] K. Xing, F. Liu, X. Cheng, and H.C. Du. Real-time Detection of Clone Attacks in Wireless Sensor Networks. In IEEE ICDCS, pages:3-10, 2008
[36] Jun-Won, H., et al. (2011). "Fast Detection of Mobile Replica Node Attacks in Wireless Sensor Networks Using Sequential Hypothesis Testing." IEEE Transactions on Mobile Computing (TMC) 10(6): 767-782.
[37] Rasheed, M. M., et al. (2017). Detecting and optimizing internet worm traffic signature. 2017 8th International Conference on Information Technology (ICIT).
[38] Koganti, V. S., et al. (2016). Internet worms and its detection. 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT).
[39] J. Newsome, B. Karp, and D. Song, ‘‘Polygraph: Automatically generating signatures for polymorphic worms,’’ in Proc. IEEE Symp. Secur. Privacy, May 2005, pp. 226–241.
[40] Ahmad, M. A., et al. (2016). Early containment of fast network worm malware. 2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS).
[41] Y. Yang, S. Zhu, and G. Cao, ‘‘Improving sensor network immunity under worm attacks: A software diversity approach,’’ in Proc. ACM MobiHoc, May 2008, pp. 149–158

[42] Y. Yang, S. Zhu, and G. Cao, ‘‘Improving sensor network immunity under worm attacks,’’ Ad hoc Netw., vol. 47, no. 1, pp. 26–40, Sep. 2016

[43] N. Gui, E. Zhai, J. Hu, and Z. Chen, ‘‘SWORDS: Improving sensor networks immunity under worm attacks,’’ in Proc. Web-Age Inf. Manage. (WAIM), Jul. 2010, pp. 86–96.

[44] Y. Liu, W. Zhang, S. Bai, and C. Wang, ‘‘Defending sensor worm attack using software diversity approach,’’ in Proc. IEEE Int. Conf. Commun. (ICC), Jun. 2011, pp. 1–5

[45] T. Abuhmed, N. Nyamaa, and D. Nyang, ‘‘Software-based remote code attestation in wireless sensor network,’’ in Proc. IEEE GLOBECOM, Nov. 2009, pp. 1–8.

[46] X. Kovah, C. Kallenberg, C. Weathers, A. Herzog, M. Albin, and J. Butterworth, ‘‘New results for timing-based attestation,’’ in Proc. IEEE Symp. Secur. Privacy, May 2012, pp. 239–253.

[47] Yang, P. and S. Yen (2017). "Memory attestation of wireless sensor nodes through trusted remote agents." IET Information Security 11(6): 338-344.

[48] Fu, D. and X. Peng (2016). "TPM-based remote attestation for Wireless Sensor Networks." Tsinghua Science and Technology 21(3): 312-321.

[49] Ahmed, N., et al. (2018). Program-flow attestation of IoT systems software. 2018 15th Learning and Technology Conference (L&T).

[50] Distributed Software-Attestation Defense against Sensor Worm Propagation,(2015)

[51] Haque, M. A. and H. Mineno (2018). Contextual Outlier Detection in Sensor Data Using Minimum Spanning Tree Based Clustering. 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2).

[52] Patil, S. D. and B. P. Vijayakumar (2017). A Novel Outlier Detection Scheme (ODS) in Wireless Sensor Networks. Computing and Network Sustainability, Singapore, Springer Singapore.

[53] Abid, A., et al. (2017). "Outlier Detection in Wireless Sensor Networks Based on OPTICS Method for Events and Errors Identification." Wireless Personal Communications 97(1): 1503-1515.

[54] W. Wu, X. Cheng, M. Ding, K. Xing, F. Liu, and P. Deng, Localized Outlying and Boundary Data Detection in Sensor Networks, IEEE Trans. Knowl. Data Eng., Vol. 19, No. 8, pp. 1145-1157, 2007.

[55] T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, Distributed Deviation Detection in Sensor Networks, ACM Special Interest Group on Management of Data, pp. 77-82, 2003

[56] E. Knorr and R. Ng, Algorithms for Mining Distance-Based Outliers in Large Data Sets, International Journal of Very Large Data Bases, pp. 392-403, 1998

[57] S. Ramaswamy, R. Rastogi, and K. Shim, Efficient Algorithms for Mining Outliers from Large Data Sets, ACM Special Interest Group on Management of Data, pp. 427-438, 2000.

[58] J. Branch, B. Szymanski, C. Giannella, and R. Wolff, In-Network Outlier Detection in Wireless Sensor Networks, Proc. IEEE ICDCS, 2006.

[59] K. Zhang, S. Shi, H. Gao, and J. Li, Unsupervised Outlier Detection in Sensor Networks using Aggregation Tree, Proc. ADMA, 2007

[60] Andrade, A. T. C., et al. (2016). Outlier detection using k-means clustering and lightweight methods for Wireless Sensor Networks. IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society.

[61] Shahid, N., et al. (2015). "One-class support vector machines: analysis of outlier detection for wireless sensor networks in harsh environments." Artificial Intelligence Review 43(4): 515-563.

[62] Zhang, Y., et al. (2013). "Distributed online outlier detection in wireless sensor networks using ellipsoidal support vector machine." Ad Hoc Networks 11(3): 1062-1074.

[63] Shahid, N., et al. (2012). Quarter-Sphere SVM: Attribute and Spatio-Temporal correlations based Outlier & Event Detection in wireless sensor networks. 2012 IEEE Wireless Communications and Networking Conference (WCNC).

[64] Rajasegarar, S., et al. (2010). "Centered Hyperspherical and Hyperellipsoidal One-Class Support Vector

Machines for Anomaly Detection in Sensor Networks." IEEE Transactions on Information Forensics and Security (TIFS) 5(3): 518-533.

[65] Hao, Y., et al. (2015). A Distributed Bayesian Algorithm for data fault detection in wireless sensor networks. 2015 International Conference on Information Networking (ICOIN).

[66] Titouna, C., et al. (2015). "Outlier Detection Approach Using Bayes Classifiers in Wireless Sensor Networks." Wireless Personal Communications 85(3): 1009-1023.

[67] Feng, H., et al. (2017). "Distributed outlier detection algorithm based on credibility feedback in wireless sensor networks." IET Communications 11(8): 1291-1296.

[68] V. Chatzigiannakis and S. Papavassiliou, " Diagnosing anomalies and identifying faulty nodes in sensor networks," IEEE Sensors J., vol. 7, pp. 637–645, 2007

[69] Livani, M. A. and M. Abadi (2010). Distributed PCA-based anomaly detection in wireless sensor networks. 2010 International Conference for Internet Technology and Secured Transactions.

[70] Rassam, M. A., et al. (2012). One-Class Principal Component Classifier for anomaly detection in wireless sensor network. 2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN).

[71] K, Kapitanova, S.H , Son and K. D, Kang. Event Detection in Wireless Sensor Networks. Second International Conference, ADHOCNETS 2010, Victoria, BC, Canada, August 2010.

[72] Y, Zhang, N, Meratnia, P. J.M, Havinga. Distributed online outlier detection in wireless sensor networks using ellipsoidal support vector machine‖. Ad Hoc Networks , December 2012

[73] Y, Zhang, N.A.S, Hammb, N, Meratnia, A, Steinb, M, Voorta. And P.J.M , Havinga. Statistics-based outlier detection for wireless sensor networks‖, Volume 26, Issue 8, April 2012

[74] Fernandes Jr, G., et al. (2015). "Autonomous profile-based anomaly detection system using principal component analysis and flow analysis." Applied Soft Computing 34: 513-525.

[75] Ghorbel, O., et al. (2015). A Novel Outlier Detection Model Based on One Class Principal Component Classifier in Wireless Sensor Networks. 2015 IEEE 29th International Conference on Advanced Information Networking and Applications.

[76] A. Francillon and C. Castelluccia. Code Injection Attacks on HarvardArchitecture Devices. In ACM CCS, Pages:15–26, October 2008

[77] Patil SD, Vijayakumar BP (2016) Clustering in mobile wireless sensor networks. In: International conference on innovations in computing and networking

[78] https://blog.csdn.net/baimafujinji/article/details/79376378

[79] C. Zou, L. Gao, W. Gong, and D. Towsley, ‘‘Monitoring and early warning for Internet worms,’’ in Proc. ACM CCS, Oct. 2003, pp. 190–199

[80] T. Park and K. G. Shin. Soft tamper-proofing via program integrity verification in wireless sensor networks. In IEEE Trans. Mob. Comput., 4(3):297–309, 2005

[81] McDonald, D., et al. (2015). "A Survey of Methods for Finding Outliers in Wireless Sensor Networks." Journal of Network and Systems Management 23(1): 163-182.

[82] Haque, M. A. and H. Mineno (2018). Contextual Outlier Detection in Sensor Data Using Minimum Spanning Tree Based Clustering. 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2).

[83] P. Kelly, An Algorithm for Merging Hyperellipsodial Clusters, Technical report, Los Alamos National Laboratory, 1994.

[84] Yang, Z., et al. (2008). An online outlier detection technique for wireless sensor networks using unsupervised quarter-sphere support vector machine. 2008 International Conference on Intelligent Sensors, Sensor Networks and Information Processing.

[85] http://db.csail.mit.edu/labdata/labdata.html

[86] Zheng W., Yang L., Wu M. (2018). An Improved Distributed PCA-Based Outlier Detection in Wireless

Sensor Network. In: Sun X., Pan Z., Bertino E. (eds) Cloud Computing and Security. ICCCS 2018. Lecture Notes in Computer Science, vol 11067. Springer, Cham；

[87] Sun, Q.-y., et al. (2018). "Study on fault diagnosis algorithm in WSN nodes based on RPCA model and SVDD for multi-class classification." Cluster Computing.

[88] Antonopoulos, C., et al. (2017). Event Identification in Wireless Sensor Networks. Components and Services for IoT Platforms: Paving the Way for IoT Standards. G. Keramidas, N. Voros and M. Hübner. Cham, Springer International Publishing: 187-210.

附录 1 程序清单

第三章 恶意蠕虫节点检测算法程序（Worm programme 文件夹）

sprt_computation_overhead_differentP：SPRT-Biased-Random 不同感染率下的计算资源

draw_worm_SPRT_Random_TimeSlot：SPRT-Biased-Random 平均检测时隙数

draw_worm_SPRT_Random_Samples：SPRT-Biased-Random 平均所需采样样本

draw_worm_Communication_OverHeads：SPRT-Biased-Random 通信资源开销

draw_worm_infected_nodes_DifferentP：SPRT-Biased-Random 不同感染率下的感染节点数

第四章 IDPCA 算法程序（IDPCA programme 文件夹）

IDPCA_Synthic_Distribute_Training：IDPCA 使用拟合数据进行异常检测模型训练

IDPCA_IBRL_Training：IDPCA 使用 IntelBerkerly 数据来进行异常检测模型训练

Draw_IDPCA_Intel_DR_FAR：IDPCA 方案下使用 IntelBerkerly 数据的检测效率

Draw_IDPCA_DPCA_CPCA_EAOD_ROC_Draw：不同方案的 ROC 曲线

Draw_FAR_SNR：IDPCA 方案下信噪比和模型的异常检测误报率的关系

第五章 IDKPCA 算法程序（IDKPCA programme 文件夹）

Draw_IDKPCA_DPCA_EAOD_AUC：不同方案的 AUC 比较

Draw_IDKPCA_IDPCA_EAOD_ROC：不同方案的 ROC 曲线对比

Draw_IDKPCA_IDPCA_EAOD_ROC：不同方案的异常数据检测率对比

KPCADR_RBF_Mha_Comparison：不同内核函数下 KPCA 的异常数据检测率

附录 2 攻读硕士学位期间撰写的论文

(1) Zheng W., Yang L., Wu M. (2018) An Improved Distributed PCA-Based Outlier Detection in Wireless Sensor Network. In: Sun X., Pan Z., Bertino E. (eds) Cloud Computing and Security. ICCCS 2018. Lecture Notes in Computer Science, vol 11067. Springer, Cham；
EI: 20184205957962

(2) Zheng, W., et al. (2017). "3D MIMO Kronecker channel modeling based on three types of typical attena arrays." Xi Tong Gong Cheng Yu Dian Zi Ji Shu/Systems Engineering and Electronics 39(6): 1366-1373.
EI: 20173404075779

附录 3 攻读硕士学位期间申请的专利

(1)“适用于无线传感器网络的异常数据检测方法”，公开号：-CN108650649A

(2)“一种基于 SPRT 算法的无线传感器网络恶意蠕虫节点检测方案”

附录 4 攻读硕士学位期间参加的科研项目

(1)国家自然科学基金，面向上下文感知数据的流计算复杂事处理技术研究(61572262)；

附录 5 IDPCA 算法伪代码

1. let $MD_i(\max)$ be the max Mahalanobis distance of node s_i ;

2. let $MD(global)$ be the global Mahalanobis distance of s_l 's neighboring nodes;

3. let n_i be the amount of data measurements for learning $(\mu, \lambda, v, \chi, \sum, MD(\max))$

4. let $x_i(t)$ be a new data measurement arrive at s_i ;

5. let $x_1^l(t), x_2^l(t), \cdots, x_d^l(t)$ be the data vectors arriving at s_l 's neighboring nodes at the same time interval

6. let $MD_i(c)$ be the Mahalanobis distance of $x_i(t)$ at node s_i in time windows t;

7. let $MD_i(\text{global})$ be the Mahalanobis distance of $x_i(t)$ at node s_i compute by group node s_l

8. procedure learning $(\mu, \lambda, v, \chi, \sum, MD(\max))$

①each node collects n_i data measurements for learning its own $(\mu, \lambda, v, \chi, \sum, MD(\max))$ by PCA, then broadcasts $(\mu, \sum, MD(\max))$ to its group node s_l ;

②each group node computes $(\mu(global), \sum(global), MD(global))$

③initiate IsOutlier($x_i(t)$) for each node;

return;

9. procedure IsOutlier($x_i(t)$) for s_i

when $x_i(t)$ *arrives at* s_i

s_i *computes* $MD_i(c)$ *according to* $(\mu, \lambda, v, \chi, \sum, MD(\max))$

if $(MD_i(c) < MD(\max))$

$x_i(t)$ *indicates an normal data*

else {

s_i *transfer* $x_i(t)_{PCA}$ *to group node to compute* $MD_i(global)$

if $(MD_i(global) > MD(global))$

$x_i(t)$ *indicates an outlier*

operate SourceOfOutlier()

else

$x_i(t)$ *indicates an normal data*

}

end

11. procedure SourceOfOutlier();

s_l *collects the Mahalanobis distance from its neighbor respectfully*

measurement is outlier then count++

if $(count \geq b/2)$

then means an event happened

else

s_i *collects p data vectors within b time windows*

compute $\overline{MA}$

if $(\overline{MA} \leq MA_i(\max))$

then $x_i(t)$ *just an error caused by noise*

else if $(\overline{MA} > MA_i(\max)$

then $x_i(t)$ *just an error caused by inner error*

endif

return;

12. procedure Updating training model

致谢

时间荏苒，岁月如梭，不知不觉，三年的研究生时光就接近了尾声，回想这三年，我经历了很多，也成长了很多，从入学时的迷茫，到毕业时的坚定。想到在学校的时光即将结束，心里充满了对学校，对老师，对同学的不舍与感激。

首先我要感谢我的导师吴老师和杨老师在这三年里对我的谆谆教诲，在科研和生活中都给了我很大的帮助。吴老师严谨的治学态度和渊博的知识一直是我学术生涯的榜样，而杨老师对学术研究的激情和创新精神则激励着我成长。在两位导师的指导下，我飞速成长，在学术领域取得了不错的成果，还养成了主动学习，独立思考的良好习惯，学会了发现问题并努力从实践中解决，为以后的人生之路打下了坚实的基础，没有虚度研究生的时光。

此外，还要感谢同教研室的师兄师姐，感谢他们慷慨地与我分享学术过程中的经验和教训，并教会我很多实践过后才能掌握的小技巧，让我少走了很多弯路。感谢教研室的师弟师妹，一起学习一起成长，互相讨论难题，交流学术，在教研室营造了浓厚的学术氛围。感谢我的室友谢豪、竺殊荣、张彦轩在生活中对我的帮助，和你们一起生活的时光非常开心。感谢我的小伙伴王慧健、吴警、孟阳、在我遇到困难挫折时候对我的安慰和鼓励。还要感谢陪伴了我上半生的篮球、感谢在学校陪我打了 7 年篮球的各位球友。

我要感谢我的父母，感谢他们对我的培养、教育以及无私的关怀，在父母的鼓励、引导和支持下，我才能顺利的完成学业。