



中国科学院大学
University of Chinese Academy of Sciences

硕士学位论文

基于流量分类的智能家居设备识别技术研究

作者姓名: 胡伟业

指导教师: 舒敏 研究员

国家计算机网络应急技术处理协调中心

学位类别: 电子信息硕士

学科专业: 网络与信息安全

培养单位: 中国科学院信息工程研究所

2023 年 6 月

**Research on smart home device identification technology based
on traffic classification**

A thesis submitted to
University of Chinese Academy of Sciences
in partial fulfillment of the requirement
for the degree of
Master of Electronic Information
in Network and Information Security

By

HU Weiye

Supervisor: Research Fellow SHU Min

**Institute of Institute of Information Engineering, Chinese Academy of
Sciences**

June, 2023

中国科学院大学

学位论文原创性声明

本人郑重声明：所呈交的学位论文是本人在导师的指导下独立进行研究工作所取得的成果。承诺除文中已经注明引用的内容外，本论文不包含任何其他个人或集体享有著作权的研究成果，未在以往任何学位申请中全部或部分提交。对本论文所涉及的研究工作做出贡献的其他个人或集体，均已在文中以明确方式标明或致谢。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日 期：

中国科学院大学

学位论文授权使用声明

本人完全了解并同意遵守中国科学院大学有关收集、保存和使用学位论文的规定，即中国科学院大学有权按照学术研究公开原则和保护知识产权的原则，保留并向国家指定或中国科学院指定机构送交学位论文的电子版和印刷版文件，且电子版与印刷版内容应完全相同，允许该论文被检索、查阅和借阅，公布本学位论文的全部或部分内容，可以采用扫描、影印、缩印等复制手段以及其他法律许可的方式保存、汇编本学位论文。

涉密及延迟公开的学位论文在解密或延迟期后适用本声明。

作者签名：

导师签名：

日 期：

日 期：

摘要

重新写摘要

关键词：物联网设备识别，智能家居，增量学习，迁移学习

Abstract

Key Words: IoT device recognition, smart home, incremental learning, transfer learning

目 录

第1章 绪论	1
1.1 研究背景	1
1.2 研究意义	2
1.3 研究内容和创新点	3
1.4 论文组织结构	5
第2章 相关研究工作综述	7
2.1 基于指纹匹配的识别	8
2.1.1 基于经验构造指纹库的方式	8
2.1.2 基于指纹自动化生成的方式	8
2.1.3 研究工作小结	9
2.2 基于机器学习的识别	10
2.2.1 基于流量统计特征的识别	10
2.2.2 基于网络协议栈特征的识别	12
2.2.3 基于有效载荷的识别	13
2.2.4 研究工作小结	14
2.3 本章小结	14
第3章 基于流量时序特征的设备类型识别及迁移能力研究	17
3.1 引言	17
3.2 基于流量时序特征的物联网设备类型识别	18
3.2.1 数据预处理模块	18
3.2.2 特征提取模块	21
3.2.3 分类及迁移模块	23
3.3 实验评估	24
3.3.1 实验设置	24
3.3.2 设备类型识别实验结果	27
3.3.3 迁移实验结果	27
3.3.4 对比实验	30
3.3.5 基于有效载荷模型的迁移能力解释实验	31
3.4 本章小结	35

第 4 章 基于自动数据标注与类别增量学习的设备厂商识别 ······	37
4.1 引言 ······	37
4.2 基于域名的自动数据标注 ······	38
4.2.1 基于域名的数据标注的研究动机 ······	38
4.2.2 基于域名的自动数据标注算法 ······	39
4.3 基于网络协议栈特征的设备厂商增量识别 ······	42
4.3.1 网络协议栈特征提取 ······	42
4.3.2 基于类别增量学习的设备厂商识别 ······	44
4.4 实验评估 ······	44
4.4.1 实验设置 ······	44
4.4.2 自动数据标注 ······	45
4.4.3 设备厂商增量识别 ······	48
4.5 本章小结 ······	51
第 5 章 面向智能家居环境的物联网设备识别原型系统 ······	53
5.1 引言 ······	53
5.2 系统总体设计 ······	53
5.3 系统模块实现 ······	54
5.3.1 数据采集层 ······	54
5.3.2 存储层 ······	55
5.3.3 业务层 ······	56
5.3.4 接口层和表示层 ······	57
5.4 系统测试 ······	58
5.4.1 测试环境 ······	58
5.4.2 非物联网设备流量过滤模块测试 ······	59
5.4.3 集成测试 ······	59
5.5 本章小结 ······	61
第 6 章 总结与展望 ······	63
6.1 本文工作总结 ······	63
6.2 未来展望 ······	64
参考文献 ······	65
致谢 ······	69
作者简历及攻读学位期间发表的学术论文与其他相关学术成果 ······	71

图目录

图 1-1 IoT Analytics 对物联网设备规模的预测	1
图 1-2 典型的智能家居环境	2
图 1-3 研究内容概况	4
图 1-4 论文组织结构	5
图 2-1 物联网设备识别方法分类	8
图 3-1 研究内容结构图	18
图 3-2 不同类型的物联网设备的包长分布统计图.	19
图 3-3 特征提取模块网络结构图	21
图 3-4 设备类型识别混淆矩阵.	28
图 3-5 迁移实验混淆矩阵.	29
图 3-6 迁移场景下验证集的 loss-acc 曲线.	30
图 3-7 物联网设备类型识别结果对比	31
图 3-8 迁移实验结果对比	32
图 3-9 相同厂商不同类型设备的载荷字段比较举例	33
图 3-10 基于有效载荷的模型迁移实验	34
图 4-1 研究内容结构图	38
图 4-2 物联网设备在通信流量中暴露域名信息	39
图 4-3 物联网设备流量中的域名词云图	40
图 4-4 基于域名的自动数据标注示意图	41
图 4-5 数据集 UNSW-TMC2018 上单个新类增量识别混淆矩阵.	49
图 4-6 数据集 CUHK-AsiaCCS2020 上单个新类增量识别混淆矩阵.	49
图 4-7 数据集 CICIoT-DataSet2022-Idle 上单个新类增量识别混淆矩阵....	50
图 4-8 持续性识别新类实验效果图	50
图 4-9 与传统离线学习方法对比	51
图 5-1 原型系统技术架构图	54
图 5-2 非物联网设备流量过滤模块执行流程图	57
图 5-3 非物联网设备流量过滤实验混淆矩阵	59
图 5-4 原型系统的流量视角观察设备识别效果截图	60
图 5-5 原型系统的设备视角观察设备识别效果截图	61

表目录

表 2-1 物联网设备识别和传统设备识别对比	7
------------------------------	---

表 2-2 基于指纹匹配的物联网设备识别研究现状 ······	10
表 2-3 基于机器学习的物联网设备识别 ······	14
表 3-1 特征提取模块超参数设置 ······	23
表 3-2 实验中使用的数据集情况 ······	26
表 3-3 设备类型识别结果 ······	27
表 3-4 迁移实验结果 ······	29
表 3-5 迁移实验结果平均值 ······	32
表 3-6 基于有效载荷方法进行迁移实验的数据集 ······	34
表 4-1 不同厂商设备的网络协议栈字段的实现情况 ······	42
表 4-2 网络协议特征 ······	43
表 4-3 标注算法在 UNSW-TMC2018 的执行结果 ······	46
表 4-4 标注算法在 CUHK-AsiaCCS2020 的执行结果 ······	46
表 4-5 标注算法在 CICIoT-DataSet2022-Idle 的执行结果 ······	47
表 4-6 单个新类增量识别结果 ······	48
表 5-1 ClickHouse 数据库流表结构 ······	55
表 5-2 系统测试数据集 ······	58
表 5-3 原型系统设备类型识别结果 ······	60
表 5-4 原型系统厂商识别结果 ······	61

第1章 绪论

1.1 研究背景

自从国际电联于 2005 年在信息社会世界峰会（WSIS）上正式提出“万物互联”的概念^[1]，越来越多的物联网（Internet of Things, IoT）设备被应用于各行各业，密切关系到每个人的方方面面。特别是最近几年，随着智能芯片、5G 等技术的高速发展，接入互联网的物联网设备规模急速增长，商业分析公司 IoT Analytics 的报告^[2] 显示，到 2025 年全球物联网设备的数量预计将达到 210 亿，如图 1-1 所示。

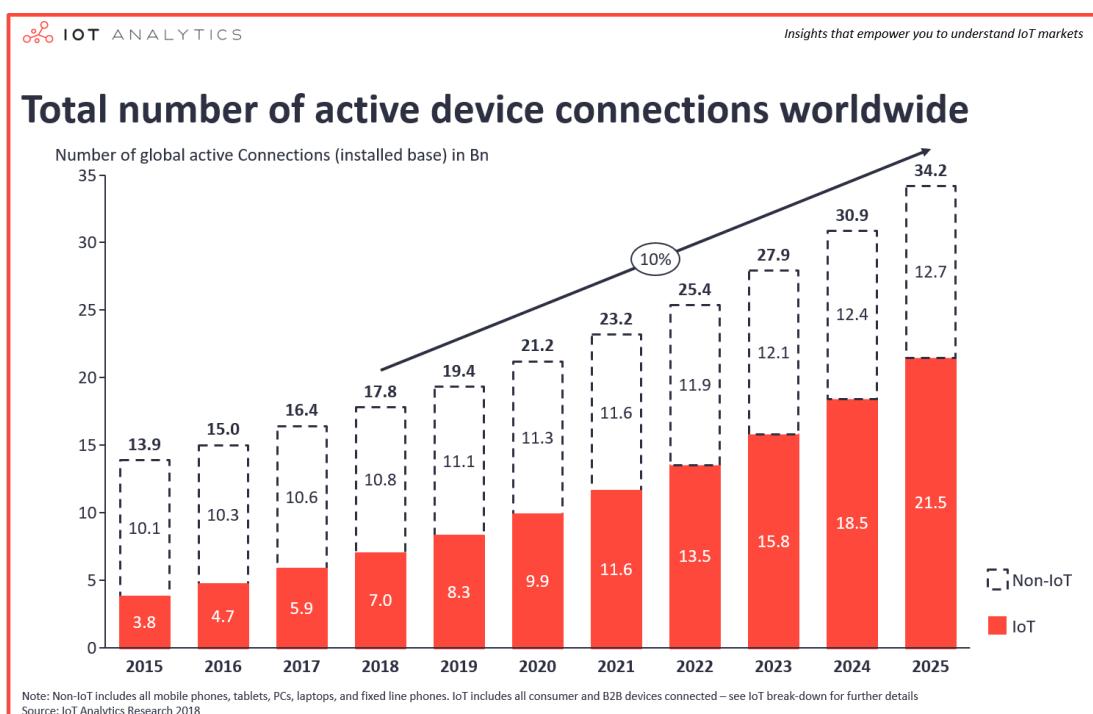


图 1-1 IoT Analytics 对物联网设备规模的预测

Figure 1-1 IoT Device Scale Forecast by IoT Analytics.

另一方面，随着人们物质生活水平的提高，消费者对日常起居的便捷性的需求也越来越高，以此为基础，“智慧生活”的概念逐渐崛起，越来越多的物联网设备正在占据我们的日常生活。为了与智能制造、水利电网等工业控制领域的物联网设备区分开，有关研究人员将其称为“智能家居设备”^[3]。以国内典型的智能家居设备提供商小米公司为例，截止 2021 年 11 月，小米已生产超过 4 亿台智能家居设备，国内拥有 5 件及以上小米智能家居设备的用户数达 800 万^[4]。

相较于工业控制领域的物联网设备，智能家居设备往往位于具有 NAT 功能的家庭网关之后，通过蓝牙、ZigBee、WI-FI、以太网等多种方式连接网关^[5]，不具有公网 IP，因此隐蔽性较强；此外，智能家居设备因其应用环境，在安全性和

可用性之间往往倾向于后者，因此可能具备安全隐患；最后，智能家居设备的数量已远大于那些具有公网 IP 并暴露在互联网上的物联网设备，因此管理监控海量的智能家居设备的挑战更大。如图 1-2 为典型的智能家居环境示意图。

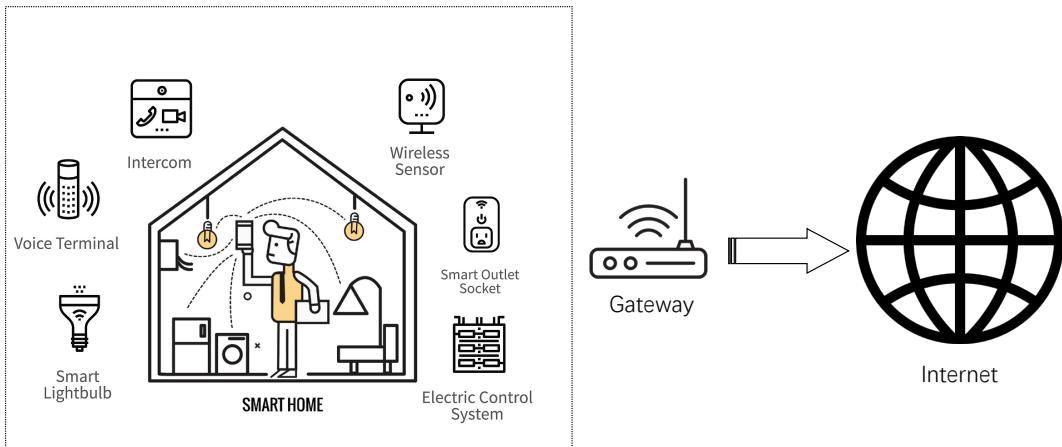


图 1-2 典型的智能家居环境

Figure 1-2 Typical Smart Home Environment.

经调研发现，同一厂商、类型的智能家居设备往往具有相同的安全漏洞，这是因为同一厂商或类型的智能家居设备往往会复用系统代码。例如，2018 年 9 月，NUUO 旗下的多种型号的摄像头被发现存在漏洞^[6]，该漏洞可以使远程攻击者在未授权的情况下以 root 身份执行代码，提升系统权限，完全控制 NUUO 摄像头。2019 年 11 月，亚马逊旗下的智能视频门铃产品 Ring 被爆出存在漏洞^[7]，Ring 在启动后会将 WI-FI 密码以明文形式传输，黑客可以利用该漏洞获取密码，进而远程控制 Ring 对用户家庭进行监视。如果不能及时发现并修复智能家居设备的安全漏洞，那么大量不安全的设备势必会给整个网络环境带来巨大的威胁，发生于 2016 年的 Mirai 病毒控制大规模物联网设备发起 DOS 攻击引发美国东部断网的严重事件^[8]，就是前车之鉴。

1.2 研究意义

智能家居设备已经成为物联网设备生态的重要组成部分，对网络安全的稳定性具有重要影响力。此外，如上一节所述，智能家居设备的安全漏洞与其厂商、类型密切相关，因此准确识别智能家居设备的厂商与类型信息是保障网络安全的第一步，对提升网络安全监控能力具有重要意义。具体地说，智能家居设备识别的工作：

- (1) 有助于建立物联网设备安全态势图，帮助网络服务提供商（Internet Service Provider, ISP）了解到最新的物联网设备接入情况，增强其资产管理能力。

(2) 有助于网络管理员有针对性地制定安全策略，例如网络管理员可以根据接入设备的厂商、类型配置防火墙规则，验证其是否修复了已知的安全漏洞。

(3) 有助于促进智能家居设备生产厂商增强安全意识，推动厂商主动加强其品牌设备的安全性、保密性，更好地服务消费者。

综上所述，智能家居设备识别为住宅网络安全防护的工作提供基础保障。

1.3 研究内容和创新点

本文的研究内容是基于流量分类的智能家居设备识别技术，本文研究的物联网设备全部属于智能家居环境，识别目标是每个物联网设备最基本的两个属性：厂商与类型。相较于传统的物联网设备识别，智能家居环境下的设备识别具有其独特的问题：

(1) 隐蔽性：智能家居设备大多位于具有 NAT 功能的家庭网关之后，NAT 将每个出站数据包的本地源 IP 地址替换为网关的唯一公共 IP 地址^[9]，隐藏了设备 MAC 等信息，因此从 NAT 外部检测连接到 NAT 后面的智能家居设备是个具有挑战的任务。

(2) 高可变性：智能家居环境具有高可变性。由于设备成本低廉等原因，在智能家居环境下频繁更替设备是一种常态。然而现有的研究工作大多是在一组固定的设备上进行实验的离线学习模型，当测试环境的设备发生变化时，模型的准确率将得不到保证，发生类似“概念漂移”^[10]的现象，此时可能需要重新训练分类模型。

(3) 异构性：智能家居环境具有异构性。由于设备多样性和个人生活习惯等因素，现实中任意两个家居的设备情况可能大不相同。然而现有的研究工作都是不具备迁移能力的孤立模型，此时在一个环境中训练好的模型将完全不能应用于另一个环境，这大大限制了设备识别模型的推广能力。

针对以上问题，本文进行了深入研究，如图 1-3 所示，详述如下：

(1) 基于流量时序特征的设备类型识别及迁移能力研究。该工作解决了智能家居设备类型识别的问题，并通过设计具有良好迁移能力的识别模型，克服了智能家居环境的高可变性和异构性问题。首先，从流量元数据中提取流量时序特征，使用 Bi-LSTM 和 1D-CNN 的混合神经网络学习不同类型物联网设备的网络行为模式，实现设备类型识别；其次，实验验证了该方法在不同的智能家居环境下具有较强的可迁移性——仅通过训练最后一层全连接神经网络便能使模型在新环境中保持较高的识别精度，大大降低了在异构的智能家居环境下从头训练模型所带来的资源浪费。

(2) 基于自动数据标注与类别增量学习的设备厂商识别。该工作解决了智能家居设备厂商识别的问题。首先，基于大部分智能家居设备会不定期与其厂商运营的服务器通信这一基本事实，实现了基于域名的自动数据标注算法，该算法通过提取流量中的域名信息与厂商名进行匹配，可以标注一部分流量，作为有监督学习的数据基础。其次，基于不同厂商对同类型设备在网络协议栈实现上的不

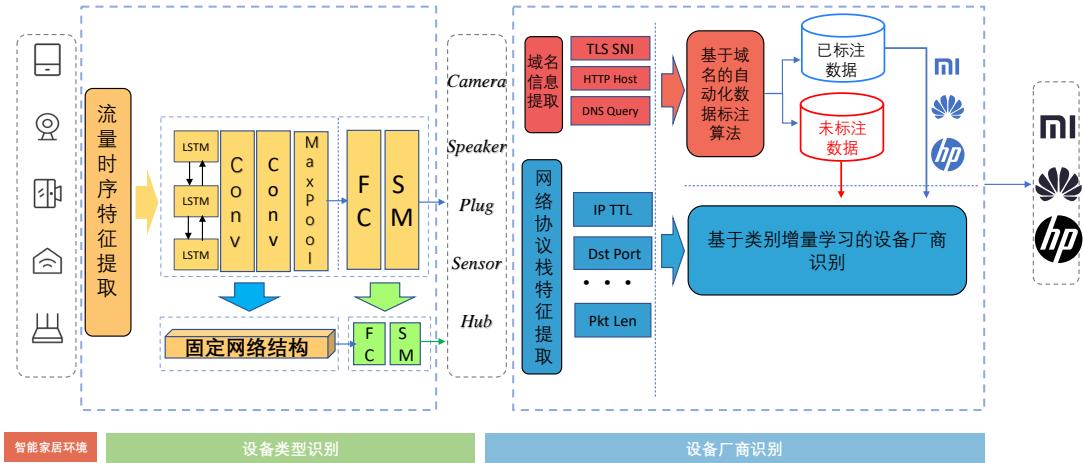


图 1-3 研究内容概况

Figure 1-3 Research Content.

同，提取特征训练机器学习分类器，识别无法被自动标注的流量。此外，为了应对智能家居环境的高可变性问题，引入了一种基于类别增量学习的随机森林方法——CIRF，使得分类器可以适应持续加入新设备的数据环境。基于域名的自动数据标注使分类模型可以自适应地在不同的环境下完成训练，克服了智能家居环境的异构性问题；类别增量学习方法可以持续学习数据分布变化，克服了智能家居环境的高可变性问题。

(3) 面向智能家居环境的物联网设备识别原型系统。基于前文研究成果，本文设计并实现了一套面向智能家居环境的物联网设备识别原型系统。该原型系统考虑的是真实的智能家居环境，因此增添了非物联网设备流量过滤模块，同时结合数据分析领域主流的实践方案，将前述研究成果集成到工程项目中，例如使用了 ClickHouse 数据库作为存储流量特征数据的工具。最终使用构建的数据集进行实验，验证了系统具有良好的设备识别能力。

以上三个研究点从智能家居设备识别的痛点入手，层层递进，为智能家居设备的厂商和类型识别任务分别提出解决方法。研究点一实现了对智能家居流量的设备类型识别，以类型识别的结果为基础，研究点二为每种设备类型独立训练分类器，实现了设备厂商识别。两个研究点分别借助迁移学习和增量学习的技术克服了智能家居环境的高可变性和异构性问题。研究点三则实现了一套灵活性较强的物联网设备识别原型系统，该系统为经过 NAT 后的每条流识别所属设备的厂商与类型信息，解决了传统的物联网设备识别方法难以应用于智能家居环境的问题。

1.4 论文组织结构

围绕第1.3节中阐述的研究内容，本论文共分为六章。论文的组织结构如图1-4所示，该图简要地概述了本论文每章节的主要内容与解决的问题。每一章的具体内容介绍如下：



图1-4 论文组织结构
Figure 1-4 Paper Organization.

第一章为绪论，它阐明了本课题的研究内容与研究意义。在研究背景中，论文阐述了物联网设备的规模日趋扩张的现状，介绍了智能家居环境下的物联网设备，即智能家居设备的含义和特点，指出了其安全隐患；在研究意义中，论文说明了智能家居设备识别对于保障网络安全的重要意义；在研究内容与创新点中，论文指出了在智能家居环境下进行设备识别时所存在的特殊问题，并概述了围绕上述问题而开展的工作内容，最后说明了工作的创新成分。

第二章是物联网设备识别方向的研究工作综述。智能家居设备本质上是物联网设备，因此本章概述了物联网设备识别的国内外研究现状。论文根据研究思路，将物联网设备识别的研究工作划分为基于机器学习技术进行分类的方法，和基于指纹提取与匹配的方法。论文先是简要说明了物联网设备识别与传统计算机网络设备识别的异同之处，随后对基于指纹匹配识别物联网设备的研究现状进行论述与总结。因为基于指纹匹配的方法大多需要通过主动探测的方式获取设备的响应报文，这不适用于隐藏在家庭网关之后的智能家居设备，因此论文随后详细介绍了基于被动流量分析和机器学习分类的设备识别方法。根据使用数据信息的来源不同，基于机器学习的方法可以划分成：基于有效载荷、基于网络协议栈和基于流量统计特征这三条技术路线。最后论文总结了上述研究工作的不足，即它们无法在具有高可变性、高异构性的智能家居环境下有效地应用。

第三章是基于流量时序特征的设备类型识别及迁移能力研究。在本章的引言部分，论文以智能家居环境的特殊问题为论据，论述设计具有良好迁移能力的识别模型的必要性，这可以极大地减少为异构的智能家居环境部署模型时，频繁地从头训练所带来的资源浪费。然后，论文详述了基于流量时序特征的物联网设备类型识别方法，该方法借助 Bi-LSTM 和 1D-CNN 的混合神经网络学习不同类型设备的网络行为模式，模型被拆解成三个模块逐一介绍，分别是：数据预处理模块，特征提取模块，分类及迁移模块。在实验评估部分，论文详述了模型的识别能力和迁移能力。在对比实验部分，论文将本章方法与另外两个公开发表的方法做对比，展示了本章方法相对优秀的迁移能力。最后对本章内容做了小结。

第四章是基于自动数据标注与类别增量学习的设备厂商识别。在本章的引言部分，论文阐述了因为智能家居环境的高可变性与异构性，导致传统的物联网设备识别方法难以应用于智能家居设备，进而引出了本章提出的两个方法的作用。随后，论文给出了基于域名的数据标注算法的研究动机与具体实现，它在 NAT 之后的网络环境中标注了一部分流量的厂商标签，并为分类模型提供标记数据。为了实现对大部分流量的持续性准确识别，论文又提出了基于网络协议栈特征的设备厂商增量识别方法，该方法从网络协议的各层字段值中提取特征，挖掘不同厂商在同类型设备的网络实现上的差异，对无法被自动标注的流量实现了准确识别。此外，基于类别增量学习技术，该方法具有在线上持续学习新厂商的能力，可以应对智能家居环境下高可变性的问题。两个方法相结合，使分类模型的训练过程与特定的实验室环境相解耦，解决了智能家居环境下的异构性问题。在实验评估部分，论文详述了两个方法各自的实验效果。最后对本章内容做了小结。

第五章是面向智能家居环境的物联网设备识别原型系统，它为智能家居环境下的物联网设备识别问题提供工程解决方案。本章首先介绍了真实的智能家居环境是物联网设备与非物联网设备共存的环境，因此，为原型系统设计实现了过滤非物联网设备流量的功能模块。随后，论文介绍了原型系统的总体设计方案，与各个模块的具体实现。系统实现后，论文使用模拟真实智能家居环境的流量数据集进行系统测试，展示了系统的识别效果。最后对本章内容做了小结。

第六章是总结和展望，它对本论文的工作进行了总结，反思了不足之处，并对未来的工作进行展望。

第2章 相关研究工作综述

智能家居设备属于物联网设备，物联网设备识别的工作来源于传统计算机网络设备识别，因此设备识别的问题实际上由来已久，但是由于物联网设备的特殊性，传统的网络设备与物联网设备在识别目标上有所不同，导致大部分方法无法直接迁移应用到物联网环境。如表2-1所示，传统的网络设备识别的目标主要是：操作系统及其版本、应用程序及其版本和设备身份，这主要是由于终端设备（PC、移动终端、服务器等）和网络基础设施设备（路由器、交换机等）相对于物联网设备来说属于重资产，其上运行的操作系统和应用程序的网络通信行为对于整个网络稳定性的影响远大于单个物联网设备，同样的原因，识别出这些设备独一无二的身份以方便认证和管理也是必要的。相对的，物联网设备的识别目标则是：生产厂商、类型、型号等信息，这是因为物联网设备的操作系统大多是各厂商基于Linux/BSD等开源操作系统进行自研的^[11]，例如小米的Xiaomi Vela操作系统^[12]，因此同厂商同类型的设备往往复用同样的系统代码，进而表现出相似的网络通信行为，也就会存在相似的安全问题，所以对物联网设备来说，厂商与类型的识别更有实际意义。

表2-1 物联网设备识别和传统设备识别对比

Table 2-1 Comparison between IoT device identification and traditional device identification.

	识别目标	有关工作
传统网络设备识别	操作系统、应用程序、设备身份	[13,14]
物联网设备识别	厂商、类型、型号	[15-42]

出于识别目标的不同，接下来将主要梳理适用于物联网设备识别的有关工作，诸如Kohno等^[13]，Vanhoeff等^[14]这类对传统网络设备身份进行识别的工作将不再进行讨论。此外，关于本论文重点研究解决的问题，即智能家居环境下的物联网设备识别，下述研究工作中如果有涉及将会重点介绍。

按照识别方法进行分类，物联网设备识别的工作可以分为两大类，即：基于指纹匹配的方式进行识别；基于机器学习的方式进行分类或聚类，如图2-1所示。其中，基于指纹匹配的方式根据指纹的构造方式不同可以分为基于经验构造指纹库的方式和基于指纹自动化生成的方式；基于机器学习的方式可以根据学习数据的来源分为基于流量统计特征的识别、基于网络协议栈特征的识别和基于有效载荷的识别。

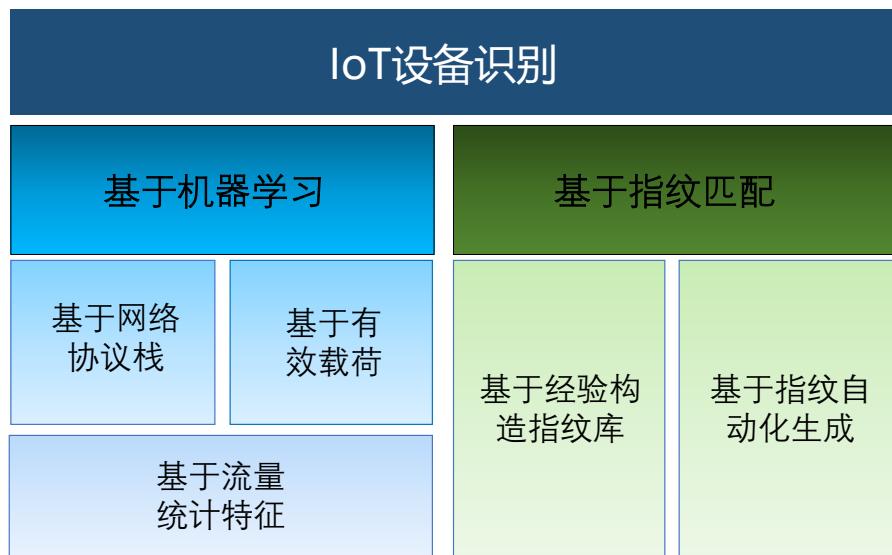


图 2-1 物联网设备识别方法分类
Figure 2-1 Classification of IoT Identification Methods.

2.1 基于指纹匹配的识别

2.1.1 基于经验构造指纹库的方式

基于指纹匹配的物联网设备识别的思想是生成并匹配能够表示设备的厂商、类型等信息的指纹，所以解决问题的关键在于设备指纹如何构造，早期的设备指纹通常根据先验知识人工构造而成，然后提取待识别设备的指纹与现有的指纹库进行比对，完成匹配。

该方式已经被广泛应用于工业界。代表工作有 Nmap^[15]，Zmap^[16] 等端口扫描工具和 Shodan^[17]，Censys 等网络搜索引擎，它们维护着庞大的设备指纹数据库用于匹配。以 Nmap 为例，它通过主动扫描 IP 地址空间，获取目标设备响应的数据包，提取协议报文中的关键字段值，并采用正则表达式匹配的方式和设备指纹库进行比对，实现设备识别，例如可根据不同操作系统在 TCP/IP 协议栈实现上的不同来区分 Windows 和 Linux。而 Shodan 等搜索引擎则主要提取协议报文里的内容文本标签，比如 HTTP 协议的 User-Agent 字段、SSH 的 banner 等，根据报文中的文本信息来识别设备。

2.1.2 基于指纹自动化生成的方式

随着物联网设备规模的不断扩大，依靠先验知识构造指纹库耗费的人力成本过高，且更新维护指纹库越发困难，基于经验构造指纹库的方式便不再流行，转而出现了很多指纹自动化生成的研究工作，具体地说，该工作通过自然语言处理和数据挖掘算法来处理设备的内容标签信息，提取厂商、类型、型号等命名实体，进而生成设备指纹。

Li^[18] 提出了一种设备识别框架，主要用于工业控制设备的识别。该框架对

IPv4 地址空间进行扫描，基于设备响应报文的应用层数据构造设备指纹，作者使用该框架扫描发现了 130 万个网络摄像头。类似的，Feng^[19] 等人对工业控制行业中主流的 17 种工控协议进行深入分析，并提出了一种有效的发现机制，该机制能够从数据报文中提取表征协议内容的特征字段并构造指纹，用于在整个 IPv4 地址空间中搜索在线的工控设备。

物联网设备的型号往往会被硬编码到设备的固件中，因此在很多通信情况下网络报文可能会包含设备的型号信息。邹宇驰^[20] 等人的工作着重于提升对型号这类细粒度设备属性的识别能力，他们发明了一种基于搜索的物联网设备识别框架，该框架利用物联网设备协议标语中的产品属性信息，通过爬虫技术自动化地构建物联网设备指纹信息库，进而实现对设备细粒度的识别。通过实验，该框架对视频监控和工控设备的型号识别准确率均超过 90%。

上述构建物联网设备指纹信息库的方式，指纹规模增长缓慢，难以用于实际需求。Feng^[21] 等人实现了一种基于采集规则的物联网设备识别引擎（Acquisitional Rule-based Engine, ARE）。ARE 提出了一种规则发现技术，它使用数据挖掘中的 Apriori 算法自动地发现报文中的内容文本标签与设备类别之间的关联关系，形成相关语料与设备类别的映射规则，实现了对物联网设备的厂商、类型、型号等细粒度的识别，实验结果表明 ARE 达到了 95.7% 的精确率和 94.9% 的召回率。

ARE 基本上已经满足了高效、准确地扫描和识别大规模物联网设备的需要，但仍有可以改进的地方。针对部分物联网设备在响应数据中缺失厂商或类型关键字，ARE 无法识别的情况，Wang^[22] 等人提出了一种新的物联网识别引擎（IoT Tracker），它通过利用同一厂商或类型的物联网设备之间响应数据的相似性来识别物联网设备。与现有方法相比，IoT Tracker 增加了 40.76% 的可识别设备。

上述工作均是通过主动探测的方式获得设备指纹，Guo 等人^[23] 则通过被动流量分析的方式进行指纹匹配。具体地说，他们提出了三种方法来发现流量中的物联网设备：流量中的服务器 IP 地址；DNS 查询中的服务器名称；TLS 握手的自签名证书，它们都利用了设备制造商运行的服务器的知识来识别设备厂商。该方法被应用于真实世界的流量进行实验，实验结果确认了家庭层面的物联网设备的大幅增长。

2.1.3 研究工作小结

基于指纹匹配的物联网设备识别的研究工作总结如表 2-2。在该方法中，不管是基于经验构造指纹库的方式还是基于指纹自动化生成的方式，常规做法都是通过主动探测的方法获取目标物联网设备的响应数据包，再从中提取设备指纹进行匹配，这种方式更加适合具有公网 IP 并暴露在互联网上的物联网设备，但不适用于隐藏在 NAT 之后的智能家居设备，因为 NAT 之后的程序如果试图主动和 NAT 之前的设备建立连接，必须借助打洞技术。另一方面，如果是通过被动流量分析的方式提取设备指纹，则需要报文的应用层未加密，因为指纹信息

基本都提取自应用层的内容文本标签，但这不符合实际上越来越流行的流量加密的趋势，且涉及到用户隐私等问题。

表 2.2 基于指纹匹配的物联网设备识别研究现状

Table 2-2 Summary of research status for identification of IoT based on fingerprint matching

代表论文	研究方法	贡献	局限性
经验构造指纹库 Nmap, Zmap Shodan, Censys Li,et al.(2017)	主动探测, 正则匹配, 扫描发现 IP 地址 内容文本标签比对 空间的网络设备		设备指纹库维护困难
指纹自动化生成 Feng,et al.(2018) Wang et al.(2019) Guo,et al.(2020)	自然语言处理, 数据挖掘	大大提升了指纹匹配 的识别准确率和效率	主动探测无法应用于智能家居设备

借助机器学习的技术，可以解决加密流量环境下设备识别的问题，论文也是使用机器学习技术进行被动流量分析来解决问题的，因此下面将详述基于机器学习的物联网设备识别的相关研究工作。

2.2 基于机器学习的识别

2.2.1 基于流量统计特征的识别

流量统计特征也被称为流量的元数据，因为它不需要访问数据包的载荷就可以得到，不会涉及到私自使用用户数据而引发的道德问题，所以一直是很多工作的重点研究对象。值得一提的是很多研究混合使用了流量统计特征和网络协议的关键字段值，为方便起见这些工作也被视为基于流量统计特征的识别工作而被列在此节，只有大部分或全部特征取自协议字段值时，才会认为是基于网络协议栈特征的识别工作。

Sivanathan 等人^[24]的研究是较早且较典型的基于机器学习的物联网设备识别的工作。他们开发了一个物联网设备分类框架，该框架使用多种信令网络协议的行为特征和从流量中派生的统计特征：如设备的活动周期、网络端口号、DNS 和 NTP 查询模式、TLS 密码套件等，训练一个基于多阶段的机器学习分类算法，证实了其根据网络活动识别特定物联网设备的能力。论文中使用了 28 种不同的物联网设备来组成智能环境，这些设备涵盖摄像头、灯、插头、运动传感器、电器和健康监视器，他们从该智能环境里收集和处理流量，为期六个月，其中一部分作为公开数据集发布供社区使用，该数据集也是后续很多基于机器学习的物联网设备识别的研究工作所广泛使用的数据集之一。

该工作是典型的基于被动流量分析和机器学习的物联网设备识别的工作，其实验准确率达到了令人满意的 99%。但是该论文的方法无法应用于 NAT 之后的网络环境，因为论文中使用了很多流间聚合特征，比如统计一个设备在一个小时的时间段内的 DNS 查询和 NTP 查询情况，而 NAT 设备的一个关键影响就是修

改了流量的源 IP 地址和端口号，使得 NAT 之后的观测者无法分辨哪些流来自同一个物联网设备，也就无法统计一个设备在一个时间段内的流量行为，所以论文的方法不能适用于智能家居环境。

与 Sivanathan 等^[24]相似的工作还有：Meidan 等人^[25]提出的物联网设备识别方法 Profilot，该方法首先训练二分类模型对非物联网设备（如智能手机、PC 机）的流量进行过滤，以减少第二阶段识别过程中的负样本数量，该方法提高了对物联网设备 + 非物联网设备组成的智能环境的识别准确率；Bai^[26]等人利用网络流量的时间依赖性，提取设备之间的特征和不变的依赖关系，训练一个 LSTM-CNN 的级联模型，实现物联网设备识别。

上面的工作是用被动流量分析的方法进行物联网设备识别，而 Radhakrishnan^[27]等人提出的 GTID 技术则是通过主动探测的方式解决问题的。GTID 会向目标设备发送探测请求包，然后使用目标设备响应的数据包的时间间隔序列作为该类型设备的特征，并将其送入神经网络分类器中进行训练。该工作可行是由于不同类型的物联网设备在硬件结构上的差异，比如设备的 CPU 和网卡的不同会导致设备处理请求数据包的时间存在差异。显然该方法同样不适用于 NAT 之后的网络环境，因为 GTID 无法主动与被 NAT 隐藏的设备建立连接，其次 NAT 设备本身也会干扰包的时间间隔。

Marchal^[28]等人发明了一个无监督学习的方法来对同一类型的设备进行聚类。该方法使用的特征来自物联网设备发送的周期性流量，创新点在于不需要人工标记训练数据，论文中使用了 33 个设备来验证该方法的实验效果，但是该方法仍然不适用 NAT 之后的网络环境。

Antônio J.^[29]等人专注于分析流量统计特征对物联网设备识别的贡献程度，因此他们仅使用了一秒时间窗口内的数据包长度的统计特征来训练随机森林模型，而没有使用端口号、协议类型等信息，最终论文实现了在 21 个物联网设备上 96% 的识别准确率。该论文证明了只要采用合适的流量分割方法和特征提取方式，仅通过流量统计特征就可以实现比较准确的物联网设备识别。

与 Antônio J. 等人的工作相似的是 Msadek 等人^[30]的工作，论文提出了一种扩展滑动窗口技术来从流量中提取协议类型、端口号间隔、包长序列等统计特征，并实验了多种机器学习模型，最终在更小的数据集上实现了 18.5% 的显著平均精度提高。

上述工作都难以应用于 NAT 之后的网络环境，基于此问题，近期出现了一些针对 NAT 之后的网络环境进行物联网设备识别的工作。

Dong 等人的^[31]是率先针对智能家居环境下 NAT 和 VPN 之后的流量进行物联网设备识别的工作之一，论文发现，当将连续数据包分组到流量窗口中时，这种时间关系可以用 RNN 建模。他们提取了描述这种时间关系的特征，包括协议类型、包间时间间隔等，并使用 LSTM 在两个数据集上进行了评估。实验结果表明，在存在非物联网设备的环境里，双向 LSTM 模型可以在网关配置 NAPT 和 VPN 的情况下分别实现 95.3% 和 80.9% 的准确率。

值得一提的是，Dong 等人的工作是对 NAPT 的研究，NAPT 是当前最常见的 NAT 形式，它在修改 IP 地址的同时还会修改传输层端口号，使用 NAPT 的网关拥有一个转换表，该表记录着地址和端口的映射，以便将入站数据包路由到正确的内部设备，同一时刻该映射是唯一的。由此可知，NAPT 不会造成流与流之间的混淆，NAPT 之后，标识一条流的五元组（源 IP，源端口，目的 IP，目的端口，协议）依旧存在，只是出站数据包的源 IP 和源端口、入站数据包的目的 IP 和目的端口被替换。Dong 等人在 NAPT 之前捕获数据包作为训练集，并通过仅在单条流中提取特征，和避免使用 IP、端口号信息的方式，来模拟 NAPT 之后的环境。本文的研究工作采用相同的手段来模拟 NAT 之后的网络环境。

Pashamokhtari 等人^[32]则致力于站在 ISP 的视角解决物联网设备识别的问题。他们开发了一个多类分类器，从 NAT 之后的 IPFIX 记录推断家庭网络中是否存在某些物联网设备类型，他们开发了一个信任指标，以跟踪一段时间内检测到的设备的网络活动，实验结果表明 IPFIX 在检测设备类型时平均准确率为 96%。与之类似的，Meidan^[33]等人使用 NAT 之后的 NetFlow 数据训练 LGBM 模型，从单条流的 NetFlow 记录里进行物联网设备识别。使用 IPFIX 或 NetFlow 的好处是对 ISP 来说这些数据是轻而易举可以得到的。

基于流量统计特征进行物联网设备识别在很多实验数据集上都表现出了良好的识别效果，这本质上是由于不同的物联网设备在网络通信行为模式上具有差异性。

2.2.2 基于网络协议栈特征的识别

对物联网设备来说，不同生产厂商往往使用自研的嵌入式操作系统，而且不同类型的设备往往因为其功能而对网络通信代码进行定制，这为基于网络协议栈特征的识别工作提供了基础。

Mittinen^[34]等人致力于研究接入局域网的物联网设备的识别，因此也只适用于 NAT 之前的环境。IoT Sentinel 部署于网关上，被动地获取每个物联网设备启动时的流量，提取各层网络协议的特征，诸如传输层协议类型、应用层协议类型、IP 选项字段、目的 IP 地址等 23 个特征，然后为每一种设备类型训练一个随机森林模型来识别当前设备是否是该模型所对应的设备类型。如果有超过一个模型判断同一个设备的样本为正样本，IoT Sentinel 会使用特征的编辑距离来确定设备的类型。该方法只能识别能够采集得到设备刚刚启动时流量的设备类型，这个条件为 IoT Sentinel 的可用性带来较大的限制。

Shaikh 等人^[35]从网络流量中提取一系列协议特征，包括数据包数量、不同目的 IP 地址数量、TCP 标志位（如 SYN、FIN、FIN-ACK）数量、UDP 数据包数量等，并采用多种机器学习方法进行了对比实验，评估不同分类模型对识别性能的影响。实验结果表明随机森林与 Gradient Boosting 的训练方法具有最高的精准率与召回率。

Sun 等人^[36]则专注于研究物联网设备在 TLS 流量上的可区分性。他们对

SSL/TLS 加密的物联网流量进行了深入分析，并确定了各种有价值的数据特征，以准确表征物联网设备通信，在此基础上，他们开发了基于深度学习的物联网设备指纹分类模型，实现了仅通过单个网络流达到 99% 的准确率。

2.2.3 基于有效载荷的识别

无论是基于流量统计特征的识别，还是基于网络协议栈特征的识别，都需要人工精心设计特征提取方法，而基于有效载荷的识别往往更加直观，甚至在借助端到端的深度学习模型的情况下，可以直接使用有效载荷得到设备类型。

任春林^[37]等人基于物联网设备的 WEB 管理页面通常包含设备信息的基本事实，利用信息增益模型提取特定类型设备的特征，论文提出了正样本反馈增强的 PU 学习方法 (FE-PU)，从网络空间中过滤特定类型的物联网设备，实验结果的召回率较人工方法提升 10%。

Li 等人^[38] 使用自然语言处理技术对 HTTP 和 HTTPS 协议的文本数据进行预处理，并采用机器学习方法训练分类模型以识别网络空间中的视频监控设备，实验达到了 99% 的精确率和 96% 的召回率。

Yang 等人^[39] 通过主动探测的方式获取设备的响应数据包，并分别从网络层、传输层和应用层提取特征，特别是应用层，他们在分析了 20 种流行的应用层协议后，从响应数据包中提取单词、词序和语义信息，作为设备特征，随后基于神经网络算法生成物联网设备指纹，最终论文使用这些设备指纹在互联网上发现了 1530 万个联网设备。

Yu 等人^[40] 则专注于从 Wi-Fi 网络识别移动和物联网设备，论文提出了 OWL，这是一种面向网络管理员和普通用户的新型设备识别机制。OWL 首先从被动接收的广播和组播数据包中提取网络流量特征，学习嵌入表示以将特征建模为六个独立且互补的视图。然后，论文提出了一个新的多视图宽深度学习 (MvWDL) 框架，该框架在泛化性能和标签视图交互性能方面进行了优化。同时，论文设计了恶意设备检测机制评估多视图分类器中视图之间的不一致，用以识别异常设备。论文在真实 WI-FI 环境下采集的流量中进行实验，获得了 90.98% 的整体识别率。

Yin 等人^[41] 则不再手动提取特征，而是直接截取数据包载荷作为灰度图进行训练。他们提出了 CBBI，一方面，CBBI 使用混合神经网络模型 Conv-BiLSTM 从网络流量中自动学习具有代表性的时空特征，例如内部组织结构在网络通信流量中的位置关系，数据包的时间顺序以及网络流的持续时间。另一方面，CBBI 包含数据增强模块 FGAN，解决了深度学习中数据不平衡的问题，提高了模型的准确性。最后，利用公共数据集和实验室数据集对 CBBI 进行评估。

Hu 等人^[42] 将开集识别的方法应用于物联网设备识别的领域，他们针对未知流量的识别问题，使用构建的 EVT 模型来重新校准流量被分为未知类的概率，使模型具备识别出训练时未知设备流量的能力。

2.2.4 研究工作小结

基于机器学习的物联网设备识别的研究工作总结如表 2-3。基于流量统计特征的物联网设备识别建模了不同类型的物联网设备在网络通信行为模式上的异构性，不需要访问数据包的有效载荷就能获得优良的实验结果，不会引发隐私问题，因此一直是很多工作的重点研究对象；基于网络协议栈特征的识别挖掘不同厂商在网络协议栈实现上的差异以实现设备识别；基于有效载荷的识别可以利用更全面的信息，在相同数据分布的情况下，基于有效载荷的识别效果往往最好，因为它使用了更多的信息，并且在借助端到端的深度学习模型的情况下，省略了人工提取特征的工作，不过盲目使用有效载荷可能会造成模型过拟合问题，因为经调研发现，部分物联网设备可能会在通信过程中传输设备 ID 等设备身份信息，例如小米旗下的 yeelight 智能灯具通信使用的 mihome 协议格式中存在设备 ID 字段^[43]。

表 2-3 基于机器学习的物联网设备识别

Table 2-3 Summary of research status for identification of IoT based on machine learning

	代表论文	研究方法	贡献	局限性
基于流量统计特征	Median,et al.(2017)			
	Sivanathan,et al.(2018)	NAT 之前提取流间聚合特征	使用流量元数据建模	NAT 限制了特征提取的能力
	Marchal,et al.(2019)	NAT 之后提取单流包长特征	挖掘设备网络行为特征	
	Pinheiro,et al.(2019)			
	Msadek,et al.(2019) Dong,et al.(2020)	随机森林等分类模型		
基于网络协议栈特征	Mittinen,et al.(2017)			
	Shaikh,et al.(2018)	提取 TCP/IP 协议栈特征	挖掘设备在协议栈实现上的差异	离线学习的模型无法适应高可变的智能家居环境
	Sun,et al.(2019)	随机森林、神经网络等分类模型		
基于有效载荷	Yang,et al.(2019)			
	Yu,et al.(2020)	CNN、LSTM 等人工神经网络	不需要人工提取特征的工作	孤立模型不具备迁移学习的能力
	Yin,et al.(2021)			
	Hu,et al.(2021)			

NAT 为物联网设备识别的工作带来困难，例如无法使用流间聚合特征、无法利用局域网流量等，最近 Dong 等人^[31]的工作似乎解决了这个问题，但是现有的方法仍然难以应用于智能家居环境，这是因为现有的机器学习模型多是离线训练且不具备迁移学习能力，无法适应具有高可变性和异构性的智能家居环境，关于这点将在 2.3 节中详述。

2.3 本章小结

基于指纹匹配的物联网设备识别的常规做法是通过主动探测的方法获取目标物联网设备的响应数据包，再进一步提取指纹进行匹配，这种方式不适用于隐藏在 NAT 之后的智能家居设备。另一方面，如果通过被动流量分析的方式提取设备指纹，则要求报文不加密，这是因为指纹信息基本都提取自应用层的内容文本标签，但这不符合实际上越来越流行的流量加密的趋势，且涉及到用户隐私问题。

基于被动流量分析和机器学习的技术可以解决上述两个问题，本论文也是基于被动流量分析和机器学习的工作，所以本章重点介绍了它们。

对于现有的解决 NAT 之后的物联网设备识别的研究，看似等价于在同时满足这两个约束条件下进行设备识别：1. 在单条流内实现数据包级别的分类，而不使用跨多条流才能提取的信息，因为 NAT 之后无法分辨流与流是否属于同一设备；2. 不使用 IP、端口号，校验和等作为特征，因为 NAT 会修改它们。类似^[31-33]的工作似乎已经很好地解决了这个问题，但是他们的工作仍然难以应用于实际场景，原因有以下两点：

首先，由于设备成本低廉，智能家居环境具有高度可变性，一段时间内一个家庭所包含的物联网设备可能被频繁更替而造成数据分布变化，然而现有的基于机器学习的识别工作全部使用离线学习的模型，此时必须再次采集数据、重新训练模型，才能避免模型的识别准确率下降，最近 Kolcun 等人^[44]的工作也证实了这一点，然而频繁地重新训练模型带来的资源浪费过大。

其次，现有的基于机器学习的识别工作都是不具备迁移能力的孤立模型。即：论文的研究者往往在一组特定的设备上进行实验，并在该环境中同时训练和测试模型，此时模型可以得到很好的识别效果。但是如果测试环境包含训练时所没有的设备，那么模型的准确率将得不到保障。事实上由于智能家居的异构性，现实中的任意两个智能家居环境都可能存在较大差异，那么此时在一个环境中训练好的模型将很难应用于另一个环境。另一方面，试图从数据集上解决该问题也是不现实的，因为不同于 ImageNet^[45] 这类推动计算机视觉领域发展的大型有标签数据集，现实中很难收集到海量的物联网流量数据并且对它进行标注，以驱动模型学习具备普适性的智能家居识别能力。因此，选择具有一定迁移能力的分类方法，使已训练好的模型能够快速地被应用到其他环境，才是更加经济的做法，也更有利物联网设备识别方法的推广。

由于上述问题，现有的基于被动流量分析和机器学习技术的物联网设备识别方法，难以应用于智能家居设备识别工作。

第3章 基于流量时序特征的设备类型识别及迁移能力研究

本章针对智能家居设备识别中的设备类型识别任务，提出了基于流量时序特征的设备类型识别方法，并实验验证了模型具有较强的迁移能力，使得模型可以在保持较高识别精度的同时以极低的成本完成训练。

本章的引言部分介绍了对智能家居设备类型识别任务，模型具有良好迁移能力的必要性；然后介绍了基于流量时序特征进行物联网设备类型识别的研究方法，该方法使用数据包长度和方向信息表征不同类型设备的网络行为模式，在三个公开数据集上的实验结果显示其具有优良的识别能力；随后在迁移实验部分，论文验证了该方法在不同的物联网设备环境下，其识别能力具有较强的可迁移性，并选取了两个公开发表的方法，对其设备识别能力和迁移能力进行了对比。特别是通过与基于有效载荷灰度值的识别方法进行对比，证实了载荷中可能存在的设备厂商相关信息等会限制模型的迁移能力的假设。最后，对本章内容做了小结。

3.1 引言

正如第二章所述，现有的物联网设备识别的研究均是在一组特定的物联网设备上进行实验，即在完全相同的设备环境下进行训练和测试，此时训练集与测试集满足数据独立同分布的基本假设，能够取得较好的实验结果。然而，这样的方法很难应用于智能家居设备识别，因为智能家居环境具有高可变性和异构性：高可变性是指同一个智能家居环境，随着时间推移其中设备频繁发生更替，导致在旧数据集上训练的模型因无法适应新环境而造成识别能力下降；异构性是指现实中任意两个智能家居环境包含的物联网设备情况大不相同，此时在其中一个环境中训练好的模型将很难应用于另一个环境。

为了克服上述问题，针对智能家居设备类型识别的任务，论文认为有必要设计具有良好的可迁移性的识别方法，使得模型在一个环境下训练好后，能够以极低的代价快速迁移至另一个环境，减少因反复从头训练物联网设备识别模型所带来的资源浪费。

本章的研究模型如图 3-1 所示。整个模型工作在具有 NAT 功能的家庭网关之后，将经过 NAT 的每条流识别出其所属设备的类型，本文中的流被定义为具有相同五元组（源 IP，源端口，目的 IP，目的端口，协议）的双向流量。首先，流量经过数据预处理模块，该模块首先过滤难以被识别的信令流量，然后提取包长和方向信息组合成流量时序特征数据，该特征具有与厂商无关的特点，这是方法具有较强迁移能力的基础。预处理后的每条流被表示成一个特征矩阵；随后，特征矩阵被送入特征提取模块，该模块由多层双向 LSTM (Bi-LSTM) 和一维卷积神经网络层 (1D-CNN) 组成，二者分别用于捕捉流内数据包的长期依赖

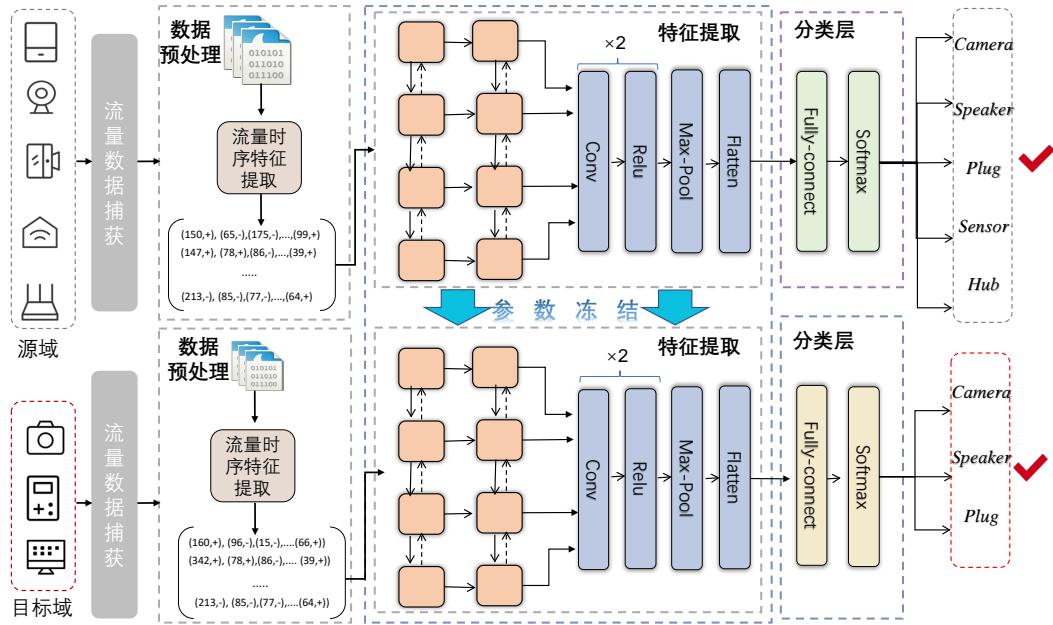


图 3-1 研究内容结构图

Figure 3-1 Research content framework diagram.

特征和局部特征，增强模型对流量时序特征的表征能力；最后，特征提取模块的输出被送入以 softmax 作为激活函数的全连接层进行分类，识别设备类型；对迁移场景而言，Bi-LSTM 和 1D-CNN 共同组成的特征提取模块作为模型的重用模块，其网络结构与参数保持不变，全连接层作为迁移学习层，将在新的数据环境下被重新训练。

3.2 基于流量时序特征的物联网设备类型识别

3.2.1 数据预处理模块

与笔记本电脑和智能手机等同时运行着多种复杂功能应用的非物联网设备不同，物联网设备往往只运行着服务于特定用途的应用程序，这导致：

(1) 不同类型的物联网设备具有不同的网络行为模式。一个简单的例子是摄像机设备在传输视频流数据的过程中会在较短的时间内传输大量数据包，而传感器设备产生相同数目的数据包则需要更长时间。因此，基于网络行为模式可以用于区分设备类型；

(2) 即使被不同的厂商生产，相同类型的设备仍具有相似的网络行为模式。原因在于网络行为模式是由设备本身的用途决定的，而与厂商的具体实现无关。例如为了保障传输视频流的效率，小米的网络摄像机和三星的网络摄像机在传输的过程中会产生大量长度相似的数据包。因此，基于网络行为模式的识别方式在不同的厂商设备上具有一定普适性，这决定了模型的迁移能力。

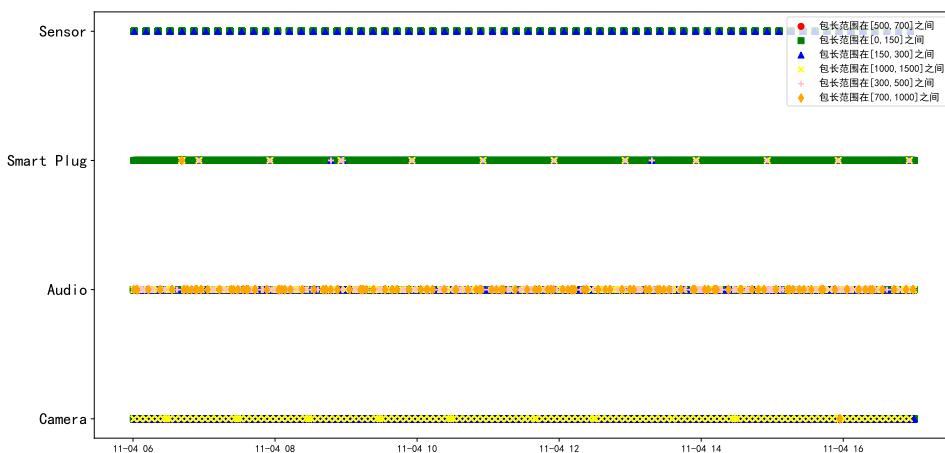
权衡了模型的识别能力与迁移能力后，本章提取流量时序特征数据表示不同类型设备的网络行为模式。流量时序特征由：数据包长度信息，数据包传输方

向信息，和天然存在的数据包时间顺序信息共同构成，这是不需要检查有效载荷内容就可以轻松获得的流量元数据。为应对 NAT 之后的网络环境，本章仅从单条流内提取这两个信息来完成智能家居设备的类型识别。除包长和包方向信息外，帧到达时间同样被常用于表示网络行为模式。但是帧到达时间很容易受当时网络环境影响，例如网络拥塞将会增大数据包与数据包之间的时间间隔，为方法的可迁移性考虑，本章没有选择使用数据包的时间信息。

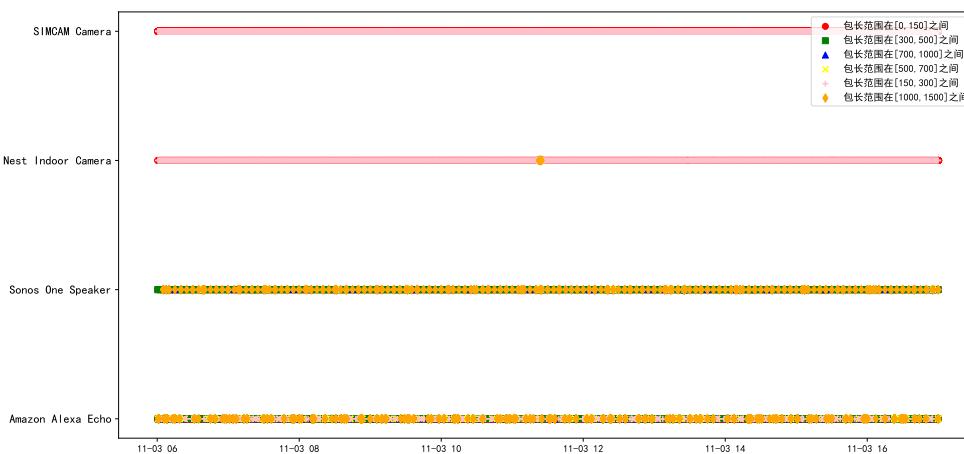
论文接下来在加拿大网络安全研究院^[46]发布的数据集上进行实验，初步验证基于流量时序特征区分设备类型的可行性。该数据集是本论文实验的主要数据集之一，其详细情况将在第 3.3.1 节进行介绍。

图 3-2 不同类型的物联网设备的包长分布统计图。

Figure 3-2 Statistical chart of package length distribution of different types of IoT devices.



(a) 不同类型的物联网设备的包长分布具有区分性



(b) 不同厂商的相同类型设备的包长分布具有相似性

如图 3-2a 所示是数据集中四种不同类型的智能家居设备（摄像机，语音助手，智能插座，传感器）在相同时间窗口内发送和接收数据包的包长信息统计图，图中的每种类型由两个不同的设备组成。该图说明了不同类型的设备具有明显不同的流量时序特征，例如，摄像机设备的包长普遍大于另外三种设备；传感器设备的数据传输行为具有间歇的周期性等。如图 3-2b 所示是摄像头类型的两个设备和语音助手类型的两个设备，各自在相同时间窗口内发送和接收数据包的包长信息统计图。图中 SIMCAM Camera 与 Nest Indoor Camera 均为摄像机设备，而 Sonos One Speaker 与 Amazon Alexa Echo 均为语音助手设备。由图可知，即使设备来自不同的厂商，相同类型的设备仍具有相似的流量时序特征。这说明了流量时序特征是一种与厂商无关的特征，因此无论设备是否来自相同厂商，流量时序特征均可以表征不同类型设备的网络行为模式，具有一定普适性。

如图 3-1 所示，数据预处理模块从原始流量中派生出上述流量时序特征数据，作为神经网络模型的输入。数据预处理模块的工作过程如下：

首先，经过对公开数据集分析后发现，物联网设备一般会比非物联网设备更加频繁地发送信令流量，包括 DNS 查询、NTP 查询、ICMP 等。这些信令流量作为公共服务的实现，不能直接表征物联网设备的应用特点。此外，信令流量有着相对固定的数据包长度和交互模式，例如 DNS 查询的流里仅有请求-响应两个数据包。如果把大量信令流量也作为识别物联网设备类型的样本，将严重干扰模型的学习能力。所以，预处理模块首先根据协议类型过滤信令流量，仅保留承载应用层载荷的网络流量。

然后，预处理模块提取每个数据包的两个元数据：数据包长度 $pktLen$ 与方向 dir ，对 dir ，当数据包进入设备的方向时取 1，反之取 0。所以，每条流的第 i 个数据包 P_i 可以表示成两个元素的特征向量 $P_i = (pktLen_i, dir_i)$ 。对每条流，提取前 N 个数据包的特征向量，将它们依次拼接在一起。所以，每条流 F 可以表示成二维的特征矩阵 $((pktLen_1, dir_1), \dots, (pktLen_N, dir_N))^T$ 。在过滤大量较短的信令流量后，大部分流的长度均在 100 个数据包以上，所以实验中 N 取 128。

论文没有使用数据包长度的正负值表示对应数据包的方向，这主要是因为在 RNN 中经常使用的 ReLU 函数在输入值为负数时，梯度为 0，会出现梯度消失的问题；并且数据包方向反映了设备与服务器交互信息的主动程度，在表征物联网设备的网络行为模式上与数据包长具备相当的重要程度，因此论文将包方向作为一个与包长并列的元素，共同送入 LSTM 的一个单元进行训练。

如前文所述，智能家居环境的一大特点是面向 NAT 之后的网络环境，现在讨论本节的特征提取方法在 NAT 之后的网络环境中的可行性。经过 NAT 后，流与流之间的关系变得不可分辨，但是设备与服务器的每个连接都拥有一个不变的连接五元组 $\langle sip, sport, dip, dport, protocol \rangle$ ，在同一五元组下的数据包与同一 TCP/UDP 会话相关联，自然也就来自或前往同一设备。即，NAT 后一个设备发出的流依旧存在且可区分，只是出站数据包的源 IP 和端口号被隐藏。本节所提取的特征均提取自单条流内，不涉及流间聚合特征，且没有使用被 NAT 设备修

改或隐藏的任何信息，因此天然地适应 NAT 之后的网络环境。

3.2.2 特征提取模块

特征提取模块旨在学习和表征流量时序特征数据所蕴含的设备类型信息。本文的特征提取模块由两层双向 LSTM (Bi-LSTM) 层和两层一维 CNN (1D-CNN) 层共同组成，如图 3-3 所示。

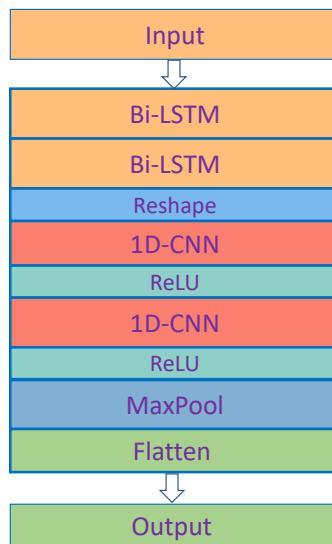


图 3-3 特征提取模块网络结构图

Figure 3-3 Feature extraction module network structure diagram.

长短时记忆网络 (Long Short-Term Memory, LSTM) 是一种递归神经网络 (RNN)，被广泛应用于序列数据建模任务。LSTM 通过在传递信息的过程中显式地选择哪些信息可以通过时间跨度较长的传递来解决传统 RNN 中梯度消失和梯度爆炸的问题。

LSTM 和传统 RNN 的区别在于，LSTM 有三种门结构，分别是遗忘门、输入门和输出门。遗忘门用来控制哪些信息需要丢弃，输入门用来控制哪些信息需要更新，输出门用来控制输出哪些信息。这种门结构使得 LSTM 可以更好地控制信息的流动，从而更好地处理长序列数据。LSTM 的数学原理如下。假设 t 时刻输入为 x_t ，隐藏状态为 h_t ，单元状态为 c_t ，则可以通过以下公式来计算 LSTM

的输出：

$$\begin{aligned}
 f_t &= \sigma(W_f[x_t, h_{t-1}] + b_f) \\
 i_t &= \sigma(W_i[x_t, h_{t-1}] + b_i) \\
 \tilde{c} * t &= \tanh(W_c[x_t, h * t - 1] + b_c) \\
 c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c} * t \\
 o_t &= \sigma(W_o[x_t, h * t - 1] + b_o) \\
 h_t &= o_t \odot \tanh(c_t)
 \end{aligned} \tag{3-1}$$

其中， σ 是 sigmoid 函数， \odot 是按元素乘法， f_t 是遗忘门， i_t 是输入门， \tilde{c}_t 是新的单元状态候选值， c_t 是单元状态， o_t 是输出门， h_t 是 LSTM 的输出。 W_f, W_i, W_c, W_o 和 b_f, b_i, b_c, b_o 分别是 LSTM 的参数，可以通过训练得到。

Bi-LSTM 是 LSTM 的变种。Bi-LSTM 在 LSTM 的基础上增加了一个反向的 LSTM，可以同时处理当前时间步的上下文和未来时间步的上下文。Bi-LSTM 的输入序列被分成两部分，一部分按照时间顺序输入到正向 LSTM 中，另一部分按照时间倒序输入到反向 LSTM 中。最后，正向和反向 LSTM 的输出被拼接在一起作为整个 Bi-LSTM 的输出。

第二层 Bi-LSTM 的输出为 N 个时间步的隐藏状态向量 $H = (h_1, h_2, \dots, h_N)$ ，将全部向量按时间步顺序拼接后得到的嵌入表示作为输入数据，送入 1D-CNN 层进一步提取特征。

卷积神经网络（Convolutional Neural Network, CNN）是一种被广泛应用于图像识别领域的深度学习模型。CNN 的数学原理基于卷积和池化操作，其中卷积操作将卷积核与输入数据进行逐元素相乘并求和，以便检测输入数据中的局部特征。卷积操作通常使用 ReLU 等激活函数来提高非线性能力。池化操作通过将特征图的局部区域汇聚为一个单独的特征来降低特征图的尺寸，在保留重要信息的同时避免过拟合和减少计算成本。

下面是 CNN 中卷积和池化操作的数学公式：

$$\begin{aligned}
 (f * g)(n) &= \sum_{m=-\infty}^{\infty} f(m)g(n-m) \\
 y_{i,j,k} &= \max_{p,q}(x_{i+p,j+q,k})
 \end{aligned} \tag{3-2}$$

其中， f 是输入数据， g 是卷积核， $*$ 表示卷积操作， n 表示输出数据的索引； x 是输入数据， y 是池化后的输出数据， p 和 q 表示池化窗口的大小。

相对于其他网络模型，CNN 的特点是通过卷积和池化等操作，实现对数据的特征提取和降维处理。因此 CNN 具有更好的分类性能和鲁棒性，可以适用于不同类型的图像分类任务。

一维卷积神经网络（1D-CNN）和二维卷积神经网络（2D-CNN）都是卷积神经网络的变种。其中，1D-CNN 的卷积核只有一个维度，通常用于提取序列数据中的时序特征。网络流量可以看成是一种按时间顺序组织的一维包长序列或

字节流，因此 1D-CNN 常被用于加密流量分类领域。例如，Wang 等人^[47]提出的基于一维卷积神经网络的端到端加密流量分类框架，在所有四个实验的 12 个评价指标中，有 11 个超过了当时最先进的方法，这表明了一维卷积神经网络强大的序列特征提取能力。在两层卷积操作之后，通过一层最大池化操作来压缩特征图，减小 1D-CNN 的输出维度，只保留对分类更重要的特征。

此外，特征提取模块没有使用批量归一化（Batch Normalization）等技术，这主要是因为 LSTM 输出的数值通常较为平稳，不易出现梯度爆炸和消失的问题，而且 1D-CNN 的层数较少，也不需要额外的正则化方法来稳定模型。

综上，如图 3-3 所示，Bi-LSTM 捕获流量的长期时序特征，1D-CNN 通过卷积核捕获小段流量的局部特征，二者结合更好地从全局与局部两个角度表征不同类型设备的流量行为模式。此外，1D-CNN 和池化操作还减小了 Bi-LSTM 输出的维度，提高了模型的泛化性，有利于模型迁移。

为了确定特征提取模块的超参数。在加拿大网络安全研究院^[46]发布的公开数据集上进行超参数选择实验。将数据集以 7: 2: 1 的比例划分为训练集，验证集和测试集，采用十倍交叉验证法选择模型超参数。如表 3-1 所示即为特征提取模块的超参数实验范围及其最终采取的值。

表 3-1 特征提取模块超参数设置
Table 3-1 Feature extraction module hyperparameter setting.

超参数	调整范围	最终值
优化器	Adam, Adamax, RMSProp, SGD	Adam
学习率	[0.001, 0.01]	0.002
迭代批次	[20, 100]	20, 60
批量大小	[16, 256]	128
Bi-LSTM 隐藏层神经元个数	[16, 128]	64
Bi-LSTM 层数	[1, 4]	2
1D-CNN Filter 大小	[2, 16]	2, 4
1D-CNN Stride 大小	[2, 16]	1, 2
1D-CNN 通道数	[4, 64]	16, 32
池化层 Pool 大小	[2, 16]	4
池化方式	Average, Max	Max
激活函数	Tanh, ReLU, ELU	ReLU

3.2.3 分类及迁移模块

特征提取模块的输出为多个卷积通道的输出压平（Flatten）后得到的特征向量，将其作为分类模块的输入。分类模块仅由一层全连接神经网络层组成，使用 softmax 激活函数。该层旨在学习输入的特征向量在设备类型空间上的差异性，从而将特征向量映射到分类概率上。

分类模块也是被设计用于迁移学习任务的模块。迁移学习是指在不同任务

之间共享知识以提高学习性能的机器学习方法。假设存在源任务和目标任务，其中源任务的数据和模型可以用来帮助学习目标任务。迁移学习的目标是通过从源任务中获得的知识来提高目标任务的学习性能。迁移学习可以分为四种类型：基于实例、基于特征、基于参数和基于关系迁移。在这些方法中，基于特征的迁移学习是最常用的方法，它通过使用源任务和目标任务共享的特征来完成知识迁移。

迁移学习的数学定义如下。给定一个源域 D_s 和一个目标域 D_t ，迁移学习的目标是将源域中学到的知识迁移到目标域中。假设源域和目标域中的数据分别表示为 $X_s = x_1, x_2, \dots, x_{n_s}$ 和 $X_t = x_1, x_2, \dots, x_{n_t}$ ，相应的标签表示为 $Y_s = y_1, y_2, \dots, y_{n_s}$ 和 $Y_t = y_1, y_2, \dots, y_{n_t}$ ，其中 n_s 和 n_t 分别表示源域和目标域的数据数量。迁移学习的目标是在目标域中学习一个分类器 $f_t(\cdot)$ ，使其能够对目标域中的数据进行正确分类。使用源域中的数据和模型可以帮助学习目标域中的分类器，其中源域和目标域之间的关系可以表示为 $p(x_s) \neq p(x_t)$ ，即源域和目标域之间的数据分布不同。

模型具有较强的迁移能力意味着只需要使用很少的标记数据微调部分参数，就可以获得一个与从头训练效果相当的模型，这极大地减少了部署物联网设备识别模型所带来的资源浪费，是解决智能家居环境异构性问题的方式之一。在迁移学习任务中，全连接层是唯一需要改变网络结构和参数的部分。这是由于在不同的智能家居环境下，相同类型的设备在流量时序特征上具有相似性，即使它们可能来自不同的厂商。特征提取模块经过 RNN 与 CNN 的共同训练后具备了提取相对通用特征的能力，这些特征使得模型具备了较强的迁移能力，而在更高的全连接层中，使用这些特征执行特定的任务，完成对新环境的适应。关于模型迁移能力的实验将在第 3.3.3 节中介绍。

3.3 实验评估

3.3.1 实验设置

关于实验环境。本节实验环境为 PC 机，操作系统为 Windows 10 家庭版 19045.2604；处理器型号为 Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz；GPU 为两块 NVIDIA GeForce GTX 1050Ti，GPU 内存为 20GB；机带 RAM 16.0 GB (15.9 GB 可用)，速度为 2400MHz；磁盘 1 型号为 NVMe SAMSUNG MZVLW128，容量为 119GB，类型为 SSD，磁盘 2 型号为 ST1000LM035-1RK172，容量为 932GB，类型为 HDD；编程语言为 Python 3.7，使用 Pytorch 1.12.0 及相关第三方库完成深度学习模型的训练与测试。

关于数据集。论文选用了三个公开数据集进行实验。

(1) UNSW-TMC2018，Sivanathan^[24]等人发布的数据集。作者团队于 2016 年 9 月 23 日至 2016 年 10 月 12 日在自行搭建的实验环境下采集的原始流量，实验环境包括 22 台各种类型的物联网设备和若干台非物联网设备。设备被连接到

家庭网关的 LAN 和 WLAN 口，作者使用 tcpdump 在 LAN/WLAN 口捕捉数据包。

(2) CUHK-AsiaCCS2020, Dong^[31] 等人发布的数据集。作者团队在 10 个物联网设备和 4 个非物联网设备组成的实验环境下采集原始流量，通过自动和手动的方式触发设备活动，并使用网络分析工具 tshark 同时监控网关的 wlan0 口和 eth0 口。此外，作者还发布了网关配置 VPN 时的流量数据。

(3) CICIoT-DataSet2022-Idle，加拿大网络安全研究院（Canadian Institute for Cybersecurity, CIC）^[46] 发布的较为新颖和全面的物联网设备流量数据集。数据集共包括 Power、Idle、Interactions、Scenarios、Active、Attacks 这六种类型的流量数据，分别表示了物联网设备处于上电、闲置、交互、场景、活跃、攻击这六种状态。论文选用了 Idle 类型的流量数据进行实验，这是因为智能家居环境下的物联网设备大多数时间均处于该状态。

上述三个公开数据集均于网关的局域网网络接口上采集数据，在这种情况下需要过滤局域网协议流量、随机化会被 NAT 修改的 IP 和端口号等方式来模拟 NAT 之后的网络环境，下述实验中所使用的数据均经过了这样的预处理，这也是相关研究中主要采用的方式。这样做好处在于设备 MAC 等原始信息得到了保留，可以方便地获得真实标签（ground truth）。此外，在本章和第四章的实验中，数据集中非物联网设备的流量也被过滤。三个公开数据集各有特点，且互相间隔了较长的时间，论文通过在三个数据集上进行实验，验证了论文提出的方法具有良好的通用性。

关于分类标签。在综合考虑了三个数据集各自所包含的设备情况，并调研了近年来消费者市场中流行的智能家居设备情况后，本章将待识别的物联网设备类型定义为以下六类：智能摄像头（Smart Camera）；智能语音助手（Smart Speaker）；智能电灯（Smart Lamp）；传感器（Sensor）；智能插座（Smart Plug）；智能网关（Smart Hub）。上述六类不能囊括全部物联网设备，例如数据集 UNSW-TMC2018 中存在的“PIX-STAR Photo-frame”电子相框不属于上述任一类型，但是与之类似的设备在另外两个公开数据集中并没有出现，判定其为小众设备，因此将其排除在识别范围之外是合理的。

在排除了不属于上述六种类型的物联网设备后，三个数据集均出现了数据分布不均衡的现象，例如在数据集 UNSW-TMC2018 中，智能摄像机设备的流量占比约为 20%，而智能网关设备的流量占比仅为 2.79%，因此本章对包含多数样本的类别仅选用了其中一部分设备进行实验。在实验中使用的三个公开数据集所包含的设备情况如表 3-2 所示。

关于评价指标。因为物联网设备类型识别的任务本质上是分类任务，因此本章采用了准确率（Accuracy），精确率（Precision），召回率（Recall）和 F1 分数（F1-score）等指标来衡量识别模型的效果。假设标签共有 n 类，那么通过将真实标签与识别标签进行对比，可以得到第 i 类的真阳性 (TP_i)、假阳性 (FP_i)、真阴性 (TN_i) 和假阴性 (FN_i)。四个指标均取宏观平均（macro avg）的计算形式，它

表 3-2 实验中使用的数据集情况**Table 3-2 Data Sets used in the experiment.**

类别	设备 MAC	设备名
智能摄像头 (Smart Camera)	70:ee:50:18:34:43	Netatmo Welcome
	00:16:6c:ab:6b:88	Samsung SmartCam
	f4:f2:6d:93:51:f1	TP-Link Day Night Cloud camera
智能语音助手 (Smart Speaker)	44:65:0d:56:cc:d3	Amazon Echo
	18:b7:9e:02:20:44	Smart Speaker
智能电灯 (Smart Lamp)	d0:73:d5:01:83:08	Light Bulbs LiFX Smart Bulb
传感器 (Sensor)	ec:1a:59:83:28:11	Belkin wemo motion sensor
	18:b4:30:25:be:e4	NEST Protect smoke alarm
	70:ee:50:03:b8:ac	Netatmo weather station
	00:24:e4:20:28:c6	Withings Aura smart sleep sensor
智能插座 (Smart Plug)	ec:1a:59:79:f4:89	Belkin Wemo switch
	50:c7:bf:00:56:39	TP-Link Smart plug
	74:c6:3b:29:d7:1d	iHome Powerplug
智能网关 (Smart Hub)	d0:52:a8:00:67:5e	Samsung Smart Things

(a) 数据集 UNSW-TMC2018 的使用情况

类别	设备 MAC	设备名
智能摄像头 (Smart Camera)	78:11:dc:cf:c8:f1	xiaobai camera
	b0:59:47:34:16:f	360 camera
智能语音助手 (Smart Speaker)	88:71:e5:ed:be:c7	Amazon Echo Dot
	f4:f5:d8:db:61:84	google home
智能插座 (Smart Plug)	30:20:10:fb:7c:05	tplink plug
	b4:e6:2d:08:63:0c	orvibo plug
	78:0f:77:1b:00:8c	broadlink plug
智能网关 (Smart Hub)	78:11:dc:e1:f0:6b	xiaomi hub

(b) 数据集 CUHK-AsiaCCS2020 的使用情况

类别	设备 MAC	设备名
智能摄像头 (Smart Camera)	70:EE:50:68:0E:32	Netatmo Camera
	44:BB:3B:00:39:07	Nest Indoor Camera
	10:2C:6B:1B:43:BE	SIMCAM 1S (AMPAKTec)
智能语音助手 (Smart Speaker)	1C:FE:2B:98:16:DD	Amazon Alexa Echo Dot
	1C:12:B0:9B:0C:EC	Amazon Alexa Echo Spot
智能电灯 (Smart Lamp)	d4:a6:51:30:64:b7	HeimVision SmartLife Lamp
	50:02:91:b1:68:0c	Globe Lamp ESP_B1680C
传感器 (Sensor)	f0:b4:d2:f9:60:95	D-Link DCHS-161 Water Sensor
	70:ee:50:6b:a8:1a	Netatmo Weather Station
智能插座 (Smart Plug)	b8:f0:09:03:9a:af	Gosund ESP_039AAF Socket
	b8:5f:98:d0:76:e6	Amazon Plug
	d4:a6:51:76:06:64	Teckin Plug 1
智能网关 (Smart Hub)	00:17:88:60:d6:4f	Philips Hue Bridge
	8c:85:80:6c:b6:47	Eufy HomeBase 2

(c) 数据集 CICIoT-DataSet2020-Idle 的使用情况

们的计算公式分别为：

$$Accuracy = \frac{\sum_{i=1}^n (TP_i + TN_i)}{\sum_{i=1}^n (TP_i + FP_i + TN_i + FN_i)} \quad (3-1)$$

$$Precision = \frac{1}{n} \sum_{i=1}^n \frac{TP_i}{TP_i + FP_i} \quad (3-2)$$

$$Recall = \frac{1}{n} \sum_{i=1}^n \frac{TP_i}{TP_i + FN_i} \quad (3-3)$$

$$F1 = \frac{1}{n} \sum_{i=1}^n \frac{2 * Precision * Recall}{Precision + Recall} \quad (3-4)$$

关于对比实验。本节选取了两个公开发表的方法进行对比实验，分别是 Yin 等人^[41]的工作和 Dong 等人^[31]的工作。将在第 3.3.4 节详细介绍。

在对比实验之后，将本章提出的方法与以 Yin 等人^[41]为代表的基于有效载荷识别物联网设备的方法做了重点比较，并为二者在迁移能力方面的差异性进行探索性实验并试图提供解释。

3.3.2 设备类型识别实验结果

在三个公开数据集上分别独立地进行实验，最终模型的分类效果如表 3-3 所示，由实验结果可知模型在三个公开数据集上共 12 个评价指标的值均在 90% 以上。模型的分类效果呈现出准确率较高而其他三个指标较低的现象，经分析发现这是因为实验数据集存在一定程度数据分布不均衡的现象，导致实验中出现了小样本类别的样本被误分成大样本类别的情况。如图 3-4 所示是分类模型在三个数据集上的混淆矩阵。由图可知，三个数据集上均出现了智能电灯（Smart Lamp），智能插座（Smart Plug）等少样本类别的数据被误分成智能摄像头（Smart Camera）等大样本类别的情况，导致存在较高的混淆矩阵中的假阳性率。

表 3-3 设备类型识别结果

Table 3-3 IoT Device type identification results.

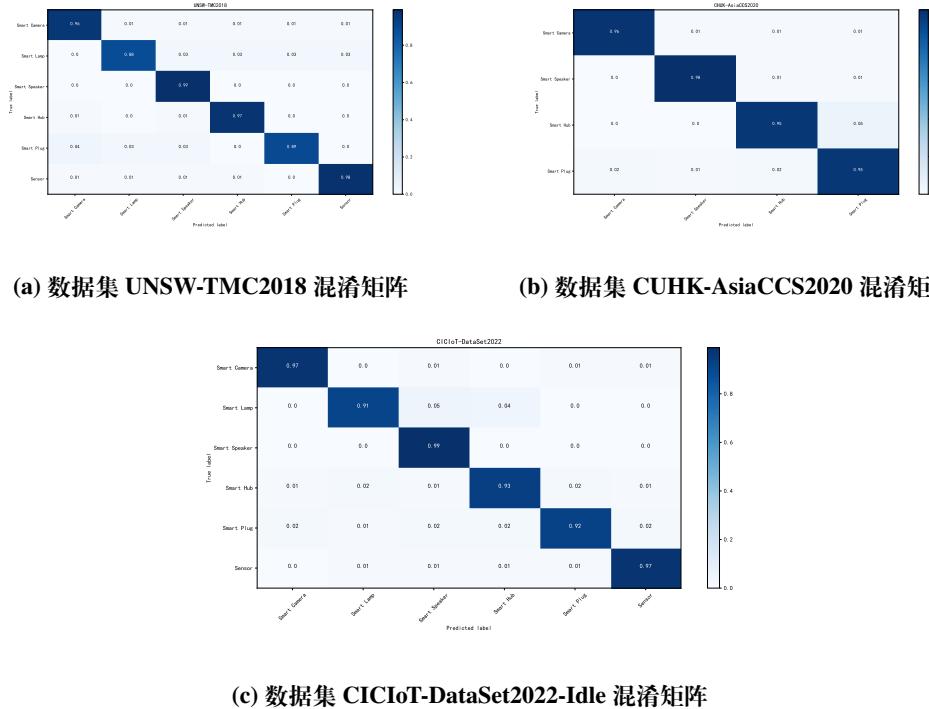
	UNSW-TMC2018	CUHK-AsiaCCS2020	CICIoT-Dataset2022-Idle
accuracy	96.68%	96.77%	97.63%
precision	90.38%	92.11%	91.38%
recall	94.46%	95.96%	94.83%
f1-score	91.48%	93.89%	92.47%

3.3.3 迁移实验结果

本节将三个公开数据集分别视为三个不同的智能家居环境。由表 3-2 可知，数据集 CUHK-AsiaCCS2020 仅包含四种类型的设备，不适合作为迁移场景的源域数据集，因此本节选用了如下三个迁移场景来评估模型的迁移能力：

图 3-4 设备类型识别混淆矩阵。

Figure 3-4 Device type identification confusion matrix.



- (1) 使用数据集 UNSW-TMC2018 作为源域数据集, 使用数据集 CUHK-AsiaCCS2020 作为目标域数据集;
- (2) 使用数据集 CICIoT-DataSet2022-Idle 作为源域数据集, 使用数据集 CUHK-AsiaCCS2020 作为目标域数据集;
- (3) 使用数据集 UNSW-TMC2018 作为源域数据集, 使用数据集 CICIoT-DataSet2022-Idle 作为目标域数据集。

在源域数据集上训练模型, 随后固定特征提取模块的网络结构与权重, 在目标域数据集上重新训练全连接层, 当达到迭代次数, 或者在验证集上的损失 (loss) 超过 1000 个批数据没有下降时, 则认为模型收敛, 结束训练并使用目标域测试集进行测试。

如表 3-4所示为三种迁移场景下的目标域分类结果。由实验结果可知, 三种迁移场景下, 模型在目标域数据集上的识别准确率均保持在 90% 以上。如图 3-5所示为三种迁移场景下的目标域混淆矩阵。由图可知, 模型迁移后, 在目标域上对大样本类别 (智能摄像头, 语音助手) 的识别能力较强, 但对少样本类别 (智能电灯, 智能网关) 等的识别精度有所下降, 这是由于源域数据集的分布不均衡导致模型对少样本类别识别能力较弱的结果。

如图 3-6所示是上述三个迁移场景下, 使用目标域数据集重新训练全连接层时, 验证集的准确率-损失折线图。对比表 3-1可知 (表中 epoch 行, 20 为训练 CUHK-AsiaCCS2020 时的迭代次数, 60 为训练另外两个数据集的迭代次数), 迁移后的模型能够在 epoch 次数远小于从头训练的情况下完成收敛, 大大减少了从

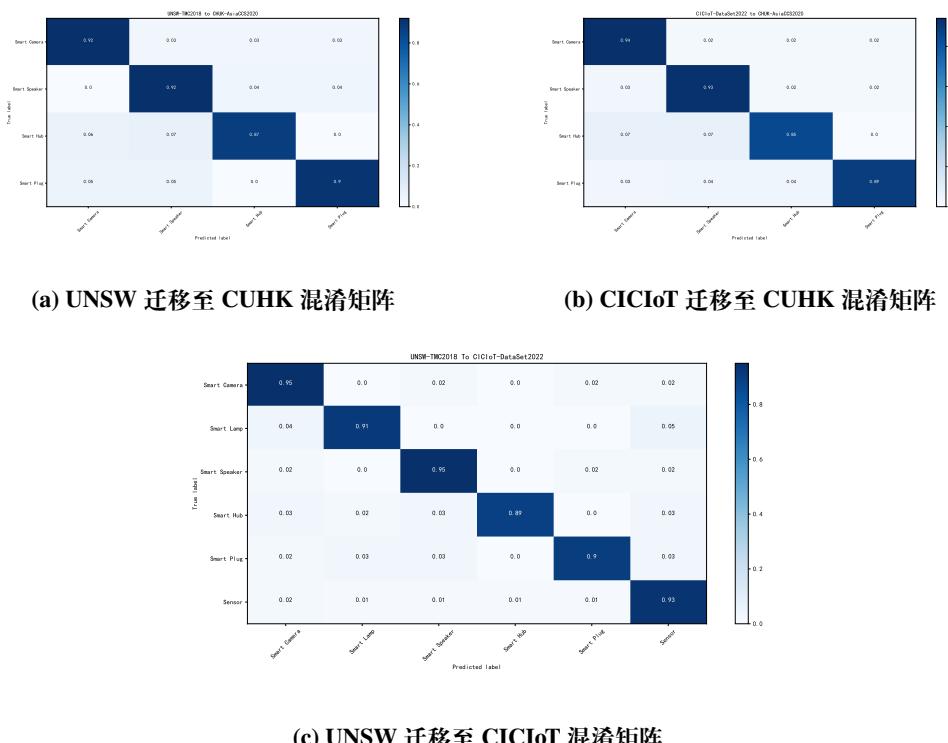
表 3-4 迁移实验结果

Table 3-4 Results of transfer experiments.

迁移场景		评价指标			
		accuracy	precision	recall	f1-score
源域数据集	UNSW-TMC2018	91.50%	80.86%	90.33%	84.59%
目标域数据集	CHUK-AsiaCCS2020				
源域数据集	CICIoT-DataSet2022-IDLE	92.53%	85.13%	90.33%	87.46%
目标域数据集	CHUK-AsiaCCS2020				
源域数据集	UNSW-TMC2018	93.71%	91.06%	92.18%	91.58%
目标域数据集	CICIoT-DataSet2022-IDLE				

图 3-5 迁移实验混淆矩阵。

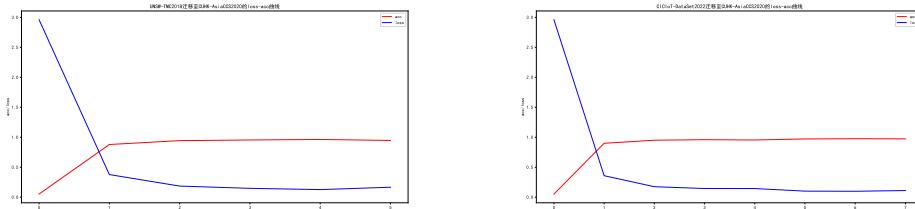
Figure 3-5 Confusion Matrix of transfer experiments .



头训练所带来的资源消耗。

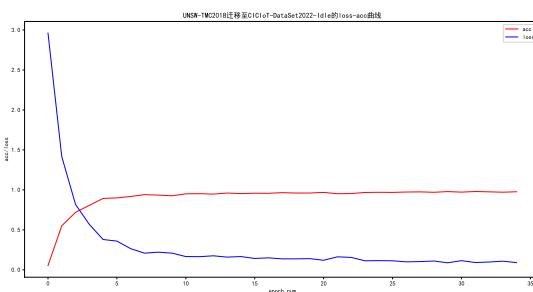
图 3-6 迁移场景下验证集的 loss-acc 曲线。

Figure 3-6 Loss-acc curve for the validation set in the transfer scenario.



(a) UNSW 迁移至 CUHK 的 loss-acc 曲线

(b) CICIoT 迁移至 CUHK 的 loss-acc 曲线



(c) UNSW 迁移至 CICIoT 的 loss-acc 曲线

由实验结果可知，本章提出的方法具有良好的迁移能力。将模型由源域迁移至目标域后，仅通过重新训练全连接层就可以快速收敛，并达到与从头训练神经网络相近的分类效果，这证明了基于流量时序特征的识别方法在厂商不同的物联网设备上具有一定普适性。

3.3.4 对比实验

本节选取了两个公开发表的方法进行对比实验。

对比实验一是 Yin 等人^[41]提出的物联网设备识别方法 CBI，该方法通过直接截取数据包载荷作为灰度图，训练组合 CNN 和 Bi-LSTM 的级联神经网络模型。在该论文中，作者使用了数据集 UNSW-TMC2018 进行实验，以每个设备作为标签，分类准确率达到 99.83%。

对比实验二是 Dong 等人的^[31]工作，他们是率先针对智能家居环境下 NAT 之后的流量进行物联网设备识别的工作之一。论文提取了协议类型、包间时间间隔等信息，并使用 LSTM 在两个数据集上进行了评估。实验结果表明，在存在非物联网设备的环境里，Bi-LSTM 模型可以在配置 NAT 的情况下实现 95.3% 的准确率。

两个对比实验均取自物联网设备识别领域较为新颖的研究工作，相关论文分别发表于 2021 年和 2020 年，但是他们都没有考虑分类模型迁移性的问题。

3.3.4.1 识别能力对比

如图3-7所示是本章提出的方法与两个对比方法在识别物联网设备类型上的分类效果对比图。由该图可知，在识别物联网设备类型能力上，基于流量时序特征的分类方法弱于Yin等人提出的基于有效载荷的方法，与Dong等人的方法基本持平。

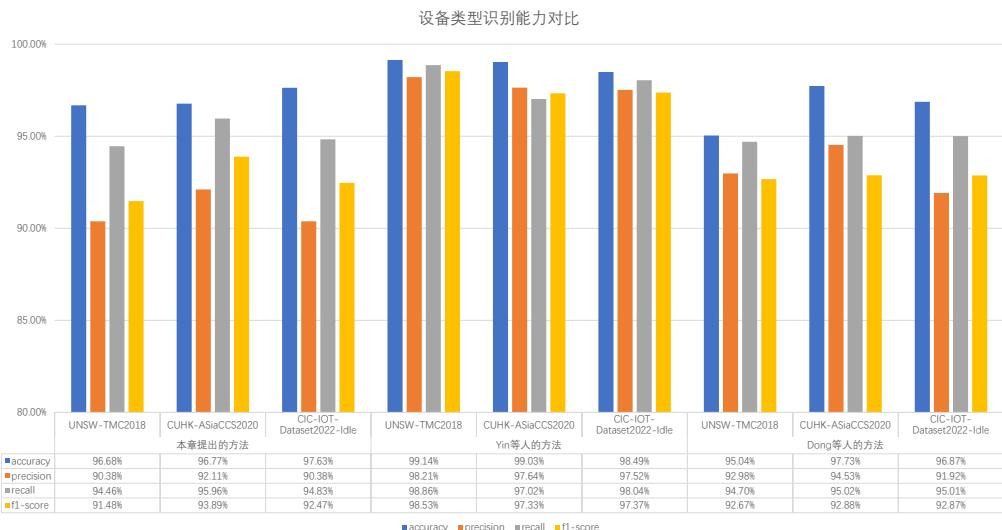


图3-7 物联网设备类型识别结果对比

Figure 3-7 Comparison of IoT device type recognition results.

3.3.4.2 迁移能力对比

尽管Yin等人的方法与Dong等人的方法没有直接设计模型的迁移模块，但仍可以按照“上层网络提取通用特征，底层网络特化具体分类任务”的基本思想，将他们模型的最后一层全连接层看作迁移模块层。因此在对比实验中，他们的模型在迁移场景中同样只重新训练最后一层的权重，以评估其迁移能力。

关于迁移实验，采取与第3.3.3节中完全一致的三个迁移场景进行实验，实验结果如图3-8所示。将三个迁移实验的四个指标分别取算术平均值，结果对比如表3-5所示。由实验结果可知，Yin等人的方法与Dong等人在模型迁移后，识别效果出现了大幅度下降，而本章提出的基于流量时序特征的物联网设备类型识别方法在模型迁移后仍能保持90%左右的识别准确率。

3.3.5 基于有效载荷模型的迁移能力解释实验

Dong^[31]等人的方法迁移能力较差，这很可能是因为他们使用了每个数据包的帧到达时间信息。帧到达时间因网络环境不同而具有较大差别，这造成了模型过分拟合源域数据集的情况，这个结果是直观。事实上在他们的原论文中，使用

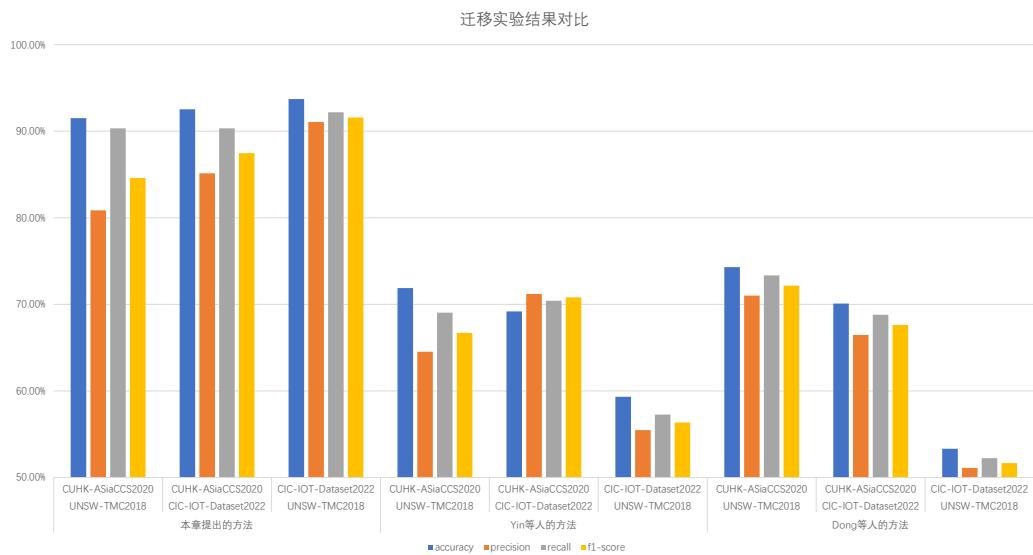


图 3-8 迁移实验结果对比

Figure 3-8 Comparison of Transfer experiment results.

表 3-5 迁移实验结果平均值

Table 3-5 Transfer experimental average results.

Methods	average accuracy	average precision	average recall	average f1-score
本章提出的方法	92.58%	85.68%	90.95%	87.88%
Yin 等人的方法	66.80%	63.74%	65.58%	64.63%
Dong 等人的方法	65.91%	62.86%	64.80%	63.81%

提取的特征数据训练随机森林模型，发现帧达到时间这一特征值的重要程度为12.8%，位列第三，仅次于目标端口号和包长。可见使用帧到达时间识别设备类型严重影响了其方法的迁移能力。

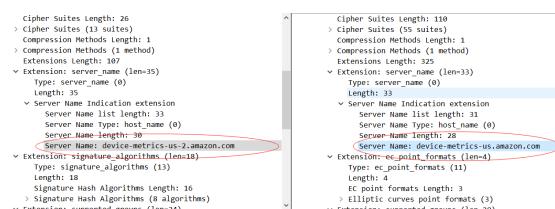
然而Yin^[41]等人为代表的基于有效载荷的方法，其迁移能力较差的可解释性不足。为此，本节进行了简单的实验以试图给出解释。

本节认为，基于有效载荷的方法迁移能力较差的原因在于：载荷中包含了与设备厂商有关的信息被模型学习用于设备分类，尽管这些信息在源域数据集上有利于模型区分设备，但是当模型被迁移到厂商情况不同的目标域数据集上时，这些信息将干扰模型的识别能力。

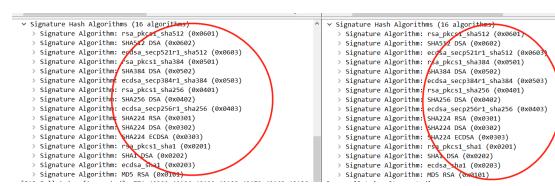
举例说明。如图3-9所示是数据集UNSW-TMC2018中的Amazon Echo（语音助手）与数据集CICIoT-DataSet2022-Idle中的Amazon Plug（智能插座）各自的载荷中值相似或相同字段的举例。可知，尽管他们是不同类型的设备，但因为它们同属于Amazon公司，造成它们在载荷中具有一定的相似性。此时将已经能良好识别Amazon Echo的模型迁移到识别Amazon Plug上的任务时，这些相似字段会对模型的识别能力造成干扰，提升将Amazon Plug误分成语音助手的风险。同理，本是属于相同类型的设备，可能因为各自的生产厂商不同，造成有效载荷的很多字段具有较大差异，而降低模型的识别能力。

图3-9 相同厂商不同类型设备的载荷字段比较举例

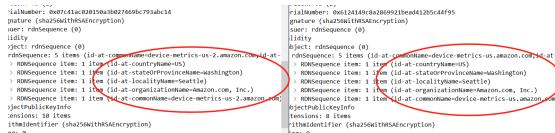
Figure 3-9 Example of comparison of load fields of different types from same vendor.



(a) TLS Client Hello 中的 SNI 字段



(b) TLS Client Hello 中的 Hash 算法



(c) TLS Certificate 中的 RDN 序列

更泛化地说，基于有效载荷识别设备类型的方法，可能在做到准确识别设备类型的同时，也从载荷中学习到了设备厂商名、设备ID等限制模型迁移能力的

信息，这将导致模型过分依赖源域数据集中的类型无关信息，阻碍了模型的可迁移性。更通俗地说，在 UNSW-TMC2018 上训练完成的模型，与其说能够识别“语音助手类型的设备”，倒不如说只能够识别“数据集中的 Amazon 语音助手设备”。

表 3-6 基于有效载荷方法进行迁移实验的数据集

Table 3-6 Dataset based on payload method transfer experiments.

设备类型	源域数据集	目标域数据集
智能摄像头（Smart Camera）	Netatmo Camera	Netatmo Welcome
智能语音助手（Smart Speaker）	Amazon Alexa Echo Dot	Amazon Echo
智能电灯（Smart Lamp）	HeimVision SmartLife Lamp Globe Lamp ESP_B1680	Light Bulbs LiFX Smart Bulb
传感器（Sensor）	D-Link DCHS-161 Water Sensor Netatmo Weather Station Eufy HomeBase 2	Belkin wemo motion sensor NEST Protect smoke alarm Withings Aura smart sleep sensor
智能插座（Smart Plug）	Gosund ESP_039AAF Socket Gosund ESP_032979 Plug Teckin Plug	Belkin Wemo switch TP-Link Smart plug iHome Powerplug

为了验证上述猜想，本节从公开数据集中选择设备，重新组织了两个数据集，如表 3-6 所示。两个数据集的特点在于：源域数据集与目的域数据集在智能摄像头（Smart Camera）和智能语音助手（Smart Speaker）两类中使用相同厂商和产品线的设备；在传感器（Sensor）和智能插座（Smart Plug）两类中使用属于不同厂商的设备。

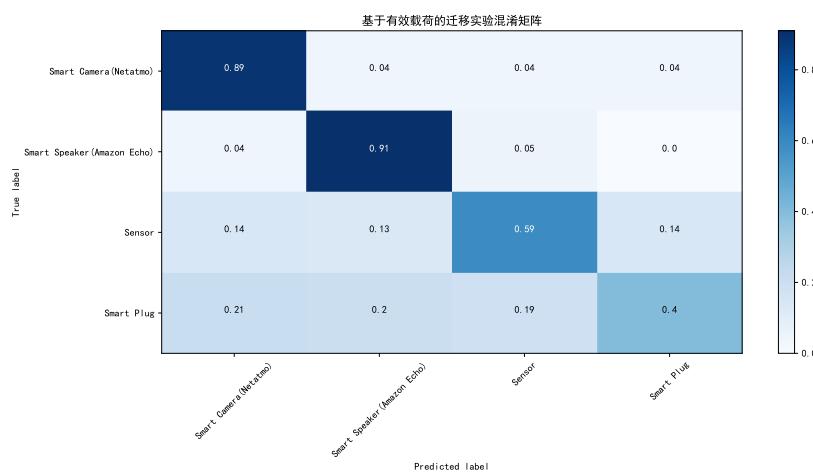


图 3-10 基于有效载荷的模型迁移实验
Figure 3-10 Payload-based model Transfer experiments.

使用第 3.3.4.2 节中同样的方法验证 Yin 等人方法的迁移能力。模型迁移后，

在目标域上的识别结果如图 3-10 所示。可见对同类型的设备，目标域与源域属于相同厂商的情况下，其迁移效果远好于属于不同厂商的设备。该实验结果一定程度上证明了“载荷中可能存在的与设备厂商有关信息会限制模型迁移能力”的假设。

3.4 本章小结

本章提出了基于流量时序特征的设备类型识别方法，该方法旨在解决经过 NAT 的智能家居设备流量的类型识别问题。方法从单条流中提取数据包长度与方向这两个流量元数据作为流量时序特征，使用 Bi-LSTM 与 1D-CNN 结合的混合神经网络学习不同类型设备的网络行为模式。在公开数据集上的实验表明提出的方法具有超过 90% 的识别准确率。尽管与部分没有对网络环境进行假定的研究相比，实验指标似乎有待提高，但是与同样针对 NAT 环境下进行物联网设备识别的工作相比，该方法没有显著劣势。更重要的是，本章通过实验证明了方法在不同的智能家居环境下具有较强的可迁移性，这是因为方法使用的流量时序特征在不同厂商的设备上具有普适性，而与基于有效载荷的识别方法进行比对发现，载荷中可能存在的设备厂商信息会造成模型过分依赖源域数据，严重限制模型的迁移能力。本章提出的方法针对设备类型识别的任务，为智能家居环境特有的异构性问题提供了解决办法，即：提高模型的迁移能力，降低为不同的智能家居环境反复从头训练识别模型所带来的资源浪费。同时，本章提出的方法对于智能家居环境的高可变性问题也有同样的意义，即当数据环境发生显著变化时，只需要重新训练最后一层全连接神经网络就能使模型在新环境下保持较高的识别精度。

第4章 基于自动数据标注与类别增量学习的设备厂商识别

本章从智能家居环境具有高可变性与异构性这两个突出问题入手，提出了基于域名的自动化数据标注和基于类别增量学习的设备厂商识别这两种方法，它们都适用于 NAT 之后的网络环境。

本章首先介绍了基于域名的自动化数据标注，它基于物联网设备定期与其厂商运营的服务器通信这一基本事实，实现了模型自适应地在智能家居环境中进行训练，将模型的训练过程与特定的智能家居环境相解耦。由于数据标注方法只能识别小部分流量，本章随后介绍了基于有监督机器学习的设备厂商识别方法，它基于不同厂商的物联网设备在网络协议栈实现上的区别进行分类，在单条流内提取特征，实现了识别无法被标注的大部分流量的设备厂商。针对现实中智能家居环境高度可变的问题，引入了类别增量学习的方法改进模型的训练形式，使模型可以在家庭中加入新厂商设备时快速再训练，以适应新的数据环境。为了验证方法的有效性，本章随后在公开数据集上进行实验，并选取了三种离线学习的方法进行对比。最后对本章内容做了小结。

4.1 引言

智能家居环境具有高可变性和异构性，这为现有的物联网设备识别方法应用于智能家居环境带来了挑战。针对这一问题，第三章提出的基于流量时序特征的设备类型识别方法，通过提高模型的迁移能力，减少模型在异构的智能家居环境下从头训练所带来的资源浪费，解决了智能家居环境下的设备类型识别问题。针对物联网设备识别的另一任务——设备厂商识别，论文认为可以通过设计合适的自动数据标注算法，将模型的训练过程从单一的实验室环境推向真实的智能家居环境。为此，论文实现了一种适用于 NAT 之后网络环境的基于域名的数据标注算法，使得模型有能力在真正待识别的环境下同时完成训练和测试。

需要特别说明的是，论文提出的自动化数据标注算法仅能为小部分满足特定条件的流量标注上设备厂商标签，而大部分流量仍是无法被标注的。即，该方法仅能回答“在这台 NAT 设备之后存在哪些厂商的物联网设备”的问题，但是无法解决“当前这条 TCP 流来自哪个厂商的物联网设备”的问题。为了解决后者，论文从已经被标注的流量中提取特征，训练机器学习分类器，识别其他不能被标注的流量。特征描述的是不同厂商在实现相同类型设备时的网络协议栈差异，这些差异存在于全部基于 TCP/IP 的流量中。此外，不同类型的设备本身在网络协议栈上可能存在天然的区别，为了避免设备类型的差异对模型识别厂商能力造成干扰，论文借鉴“控制变量”的思想，基于设备类型的识别结果，为每种设备类型的流量独立训练分类器。

最后，针对智能家居环境高度可变的特性，论文实验了基于类别增量学习的

随机森林，使得在已经达到良好识别效果的智能家居环境下加入新厂商设备时，模型可以快速再训练以识别新厂商设备。

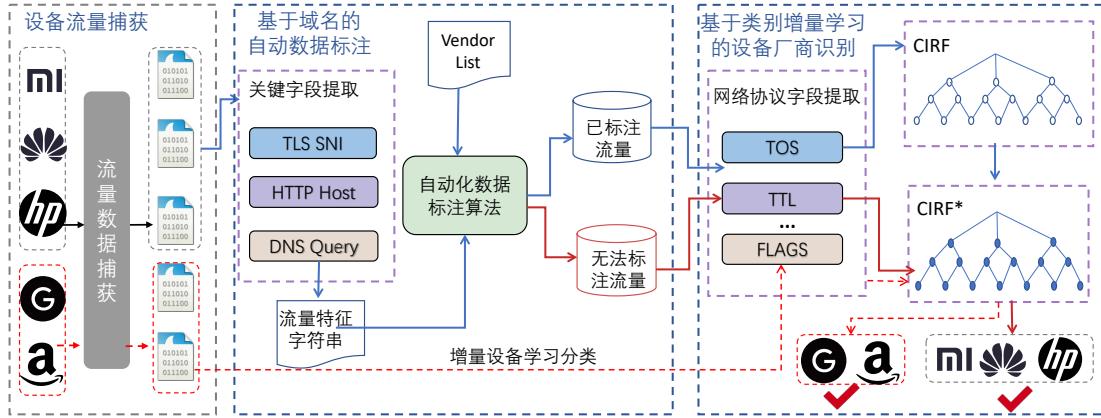


图 4-1 研究内容结构图

Figure 4-1 Research content structure chart.

本章的研究模型如图 4-1 所示。整个模型工作在 NAT 之后的网络环境，也就是现实中智能家居环境的家庭网关外面。首先，基于域名的数据标注算法实现了将 NAT 之后的一部分流量识别为特定厂商，然后执行特征提取算法，从这些流量中提取能够表征设备厂商的网络协议栈特征，并将样本数据送入对应设备类型的分类器中进行训练。分类器是一种基于类别增量学习的随机森林模型，这使得环境中加入新厂商设备时模型能够快速收敛，以更好的性能持续地识别新厂商。

4.2 基于域名的自动数据标注

4.2.1 基于域名的数据标注的研究动机

本节验证了基于域名标注一部分流量所属设备厂商的基本原理：即大多数物联网设备会不定期地与其厂商运营的服务器交换数据，可以从这些流量中提取域名信息进行厂商识别。关于这点之前已有研究实验验证过^[23,24]。但是在之前的工作里，域名主要来源于 DNS 流量，并且他们的方法仅能通过域名判断存在哪些厂商的物联网设备，无法将识别结果推广到不包含域名信息的流量上。本章通过将基于域名的方法作为一种自动化数据标注方法，提取网络协议栈特征训练机器学习分类器，解决了这个问题。

论文提出的数据标注算法主要从 DNS 流量，HTTP 流量，TLS 握手流量中提取服务器域名信息，在这些流量中，域名分别存在于 DNS 请求数据包的 Queries

字段，HTTP 请求数据包的 Host 字段，Client Hello 握手数据包的 Server Name Indication (SNI) 字段中。需要补充说明的是：因为数据标注是在 NAT 之后的网络环境中进行的，所以诸如 mDNS 这样的局域网协议不在算法的考虑范围之内。

论文接下来在 Sivanathan 等^[24]发布的数据集上进行实验，初步验证标注算法的可行性。如图 4-2 所示是该数据集中的四个主要厂商的设备，随着时间推移在流量中暴露域名信息的情况。四个厂商 Samsung, TP-Link, Withings, Amazon 涵盖了智能摄像头，智能传感器，语音助手等设备，流量的统计时间从 9 月 22 日 21 点到次日 21 点。由图可知，尽管流行着各种各样的加密算法，但主流物联网厂商的设备在通信过程中暴露域名信息是一种常态，这主要是由于域名的特殊作用。具体地说，一台物理服务器上经常为多个不同的域名提供服务，并为每个域名使用不同的证书，因此客户端需要在建立连接的第一个数据包里就告知服务器请求的是哪个域名，才能确保服务器为客户端提供正确的服务。

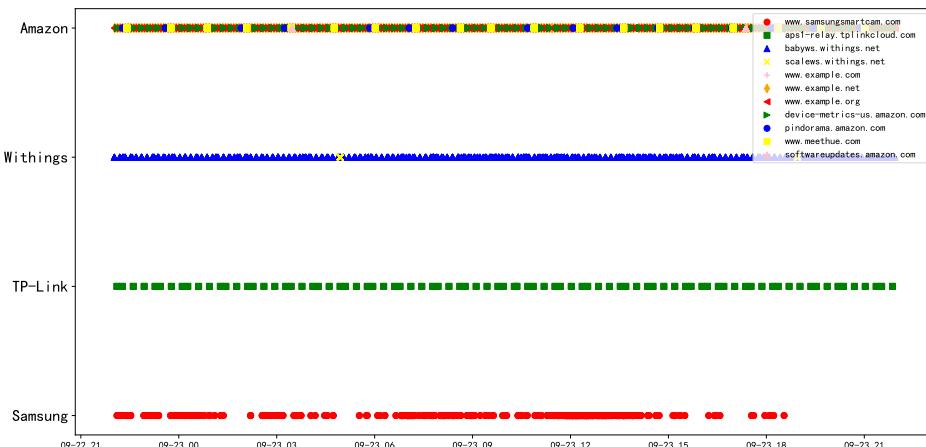


图 4-2 物联网设备在通信流量中暴露域名信息

Figure 4-2 IoT devices expose domain names in traffic.

如图 4-3 所示是根据四个厂商的设备流量中包含的域名信息而制作的词云，流量统计时间从 9 月 22 日到 10 月 6 日共计两周。图中域名体积越大代表其在该厂商通信流量中出现的次数越多，域名标红的部分代表其可以通过众所周知的厂商名进行匹配。由图可知，在排除 NTP 等公共服务的域名之后，大部分域名都能够用设备厂商的名字进行匹配。

综上可知，主流厂商的物联网设备会在通信流量中持续暴露包含自己厂商名的域名信息，可以利用这部分信息对流量实现数据标注。

4.2.2 基于域名的自动数据标注算法

上一节阐述了物联网设备会不定期与其厂商运营的服务器通信，从而在流量中暴露域名信息的基本事实，本节给出基于域名的自动化数据标注算法的实



图 4-3 物联网设备流量中的域名词云图

Figure 4-3 Word-Cloud of domain names in IoT devices traffic.

现。

算法 1 基于域名的自动化数据标注算法

输入: NAT 后的物联网设备流量 $traffic$, 待匹配厂商名列表 $vendorNameList$

输出: 被标注的流量及其厂商标签 $(flow, vendorName)$

```

1: procedure DOMAIN2VENDOR( $traffic$ ,  $vendorNameList$ )
2:   for  $flow$  in  $traffic$  do
3:      $domainName \leftarrow [DNSQuery, HTTPHost, TLSNI] of flow$ 
4:      $filter(domainName)$             $\triangleright$  Filter Public Service Domain Name
5:     for  $vendorName$  in  $vendorNameList$  do
6:       if  $match(vendorName, domainName)$  then
7:         return  $flow, vendorName$ 
8:       end if
9:     end for
10:   end for
11: end procedure

```

算法旨在从经过 NAT 后的物联网设备流量中挖掘域名信息，并根据域名匹配可能的设备厂商，从而给匹配成功的流（五元组定义的双向流量）打上标签。如算法 1 所述，其中 $filter()$ 方法用于过滤公有云服务的域名，例如 `aws.amazon.com` 或 `alibabacloud.com`，以防止物联网设备在访问这样的公有云服务时被错误地标记为云服务提供商的厂商名。 $match()$ 方法执行具体的匹配算法，就本论文实现的方法而言，执行的是子串匹配算法，即根据厂商名 $vendorName$ 生成其全拼、缩写、全部大写、全部小写、别名的形式，逐一与流量中提取的 $domainName$ 进

行子串匹配，匹配成功则将当前流标注为厂商 *vendorName*。另一方面，*match()* 方法也可以通过 WHOIS^[48] 服务来实现，WHOIS 是一种查询协议，可以用于查询域名的注册信息，例如查询.gstaticac.com 形式的域名可以得到其注册公司 Google。

在匹配成功之后，如果 *domainName* 提取自 HTTP 协议的 Host 字段或 TLS 的 SNI 字段，那么仅标注当前流；但是如果 *domainName* 提取自 DNS 请求的 Query 字段，算法还需要将当前 DNS 与后续传输应用数据的 TCP 或者 UDP 流量进行关联，以最大限度地标注尽可能多的流量。经过对数据集的实验发现，超过 92% 的 DNS 请求与后续基于 TCP/UDP 的应用数据流量共存于 7s 的时间窗口内，其中“后续基于 TCP/UDP 的应用数据流量”具体指的是目的 IP 属于前面对应 DNS 请求的解析结果的流量。因此，在当前 DNS 匹配成功后，算法将继续监听该 DNS 请求的响应数据包，记录其解析得到的 IP 地址，并在后续 7s 时间窗口内，将目的 IP 属于解析得到的 IP 地址的 TCP/UDP 流标注为相应的厂商。数据标注算法执行过程示意如图 4-4 所示。

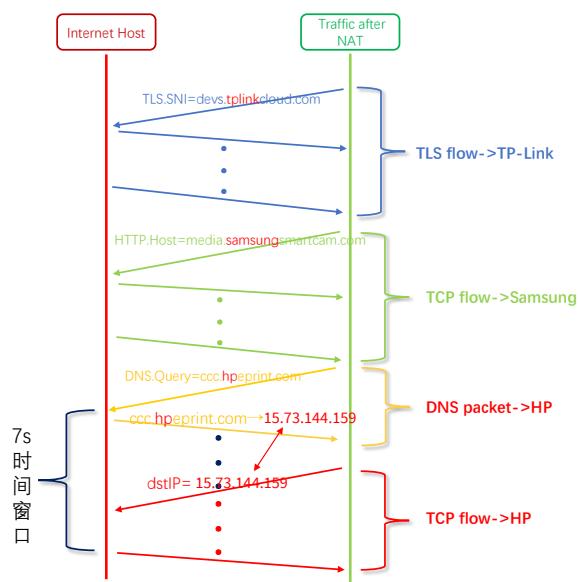


图 4-4 基于域名的自动数据标注示意图

Figure 4-4 Schematic diagram of automatic data annotation based on domain name.

算法 1 实现了自动化地将 NAT 之后的智能家居设备流量中包含厂商域名的部分进行标注，而不需要用户提前掌握该网络环境的任何信息。算法的执行结果为后续机器学习模型从中学习厂商的网络协议栈特征提供了数据基础。关于算法 1 的实验结果将在第 4.4.2 节中给出。

4.3 基于网络协议栈特征的设备厂商增量识别

4.3.1 网络协议栈特征提取

尽管 RFC 文档对 TCP/IP 协议栈的实现方式进行了规范，但不同的设备在具体实现上往往存在差异，从而产生具有差异性的网络数据包。导致不同设备在网络协议栈的实现上不同的原因主要有两个：一是设备类型不同，此时网络协议栈需要为设备的应用场景进行定制，例如摄像头设备的网络数据吞吐能力应该大于传感器设备；二是不同设备厂商的实现细节存在差异，例如小米公司基于 NuttX 内核开发的 Xiaomi Vela 操作系统^[12]，它提供了一种轻量级的 TCP/IP 协议栈实现，目标是实现设备网络功能的低功耗。

为了准确实现设备厂商识别，需要在设备类型相同的情况下，挖掘不同厂商在网络协议栈实现上的差异。本节通过在同类型设备流量的网络层和传输层中提取字段值特征，并结合包长特征来挖掘不同的厂商在网络协议栈实现上的差异，从而完成设备厂商识别。

表 4.1 不同厂商设备的网络协议栈字段的实现情况

Table 4-1 Implementation of network protocol stack fields of devices from different vendor.

	厂商	TTL	ServerPort	Win	MSS
智能摄像机	Samsung	64,56	80	939,1728	1460
	TP-Link	64,50,47	53,80	2920,71	1460
	Withings	64,44	80,443	2920,3456	1460
智能语音助手	Netatmo	64,47	443,80	363,235	1456
	Amazon	64,56	53,443	639,822	1380
	Invoxia	48,64	443,80	23168,15544	1460

网络层主要负责在不同的网络之间进行数据传输和路由选择。网络层的 TOS (Type of Service) 字段用于指定传输数据的服务质量要求，从而实现网络资源的最优化利用；网络层的 TTL (Time-to-Live) 字段用来限制 IP 数据报在网络中传输的跳数，防止数据包在网络中不断地循环，导致网络资源的浪费和网络拥塞；网络层的 Flags 字段用于控制 IP 数据报的分片和重组，帮助路由器和主机在分片和重组 IP 数据报时保持一致性；网络层的 Protocol ID 字段用于标识网络层所包含的上层协议类型，在论文的方法中只取 0 或 1 表示 TCP 或 UDP。

传输层主要实现了数据包的端到端传输。传输层的端口号 (Port) 标识了计算机上的应用程序，除去用于标准协议的 1024 个保留端口，其余端口号可以被用于标识自定义的应用程序，不同的物联网厂商可能为自家的服务器选用不同范围的端口号来和自家设备进行通信；TCP 协议的窗口 (Window) 用于 TCP 协议的流量控制与拥塞控制，它反映了设备本身接收处理数据的能力；TCP 协议的最大报文段长度 (Maximum Segment Size, MSS) 指的是 TCP 协议在发送数据时，每个 TCP 报文段的最大允许长度，它是由设备与通信的服务器两端通过协商来确定的。TCP 协议的选项字段用于扩展 TCP 协议的功能，通常用于优化

TCP 协议的性能或提供额外的安全性保证，不同的 TCP 实现支持不同的选项类型和长度。

此外，不同厂商的物联网设备通常会产生具有特定长度和方向的数据包，这些特征是由设备厂商在设计和开发网络功能的过程中决定的，因此流内的数据包长度与方向分布情况也被本节所使用。

如表 4-1 所示展示了智能摄像机类型下的四个厂商和语音助手类型下的两个厂商在上述网络协议栈字段的部分实现情况，因为大部分字段值的数据呈现稀疏分布，因此该表格仅展示了每个字段出现次数最多的若干个值。由表格可知，即使是同类型的设备，不同厂商在这些字段上的实现差异仍是明显的。

综上所述，本文最终选取的网络协议栈特征如表 4-2 所示。

表 4-2 网络协议特征
Table 4-2 Network protocol characteristics.

特征		描述
	TOS	区分服务类型
网络层	TTL	生存时间
	Flags	是否分片
	Protocol ID	传输层协议
	ServerPort	服务器端口号
传输层	Win	窗口大小
	MSS	最大报文段长度
	OptKinds	TCP 选项 Kind 序列
	PktLen	包长
	Dir	方向

对每个数据包提取上述特征值：对于网络层和传输层的协议栈字段值均转换成对应的十进制形式；如果传输层协议是 UDP，则属于 TCP 协议的字段值全部置为 0；对于 TCP 选项的 Kind 序列只提取前 4 个值。因此每条流的第 i 个数据包 P_i 可以表示成长度为 13 的特征向量：

$$P_i = (tos_i, ttl_i, flags_i, protocolID_i, serPost_i, win_i, mss_i, optKinds_i, pkt_i, dir_i) \quad (4-1)$$

对每条流，提取其前 n 个数据包的特征并依次拼接在一起，实验中 n 取 20，若一条流中的数据包个数不足 n 则以 0 值填充。因此对每条流 F 可以表示成长度为 260 的特征向量：

$$F = \{P_1, \dots, P_n\} \quad (4-2)$$

本节所提取的特征均提取自单条流内，不涉及流间聚合特征，且没有使用被 NAT 设备修改或隐藏的任何信息，因此天然地适应 NAT 之后的网络环境。

4.3.2 基于类别增量学习的设备厂商识别

当现实生活中的数据环境随着时间发生变化时，预先部署的分类模型将无法识别训练时未知的类型，甚至可能干扰对已知类型的识别，此时为了适应数据类型的变化，传统的基于离线学习的方法必须从头开始重新训练整个模型，这对具有高可变性的智能家居环境而言意味着大量的计算资源和内存资源被浪费。

类别增量学习（Class Incremental Learning）是一种机器学习领域的研究方向，其目标是在已有的模型基础上，对新类别的样本进行快速、准确的分类。类别增量学习的主要用途是处理现实生活中不断增加的数据集和新出现的分类任务。相比于重新训练整个模型或从头开始训练新的分类器，类别增量学习可以更好地利用已有的数据和模型，同时避免在增加新类别时出现的灾难性遗忘（catastrophic forgetting）问题。

在现有研究中，基于随机森林的类别增量学习是比较流行的方法，该方法通过对原有随机森林模型的扩展，将新类别的特征融合到模型中，实现新类别的在线识别。随机森林模型本身具有良好的可扩展性和鲁棒性，加之其对高维度数据有很好的适应性，因此该方法在流量分类等领域得到了广泛的应用。

本节以 Hu^[49] 等人于 2018 年提出的 CIRF (Class Incremental Random Forests) 为研究对象，验证这个已在其他领域得到良好效果的类别增量学习方法被应用到智能家居环境时的可行性。

CIRF 采用一种基于分离轴定理（separation axis theorem）的新型分裂策略来增加内部节点，通过该策略，父节点的变化不会导致信息丢失，允许在不重建子树的情况下插入新节点。同时使用基尼指数或信息增益来划分决策树的叶子节点，使树在类别增量的场景下合理生长。

4.4 实验评估

4.4.1 实验设置

关于实验环境。实验环境为 PC 机，操作系统为 Windows 10 家庭版 19045.2604；处理器型号为 Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz；机带 RAM 16.0 GB (15.9 GB 可用)，速度为 2400MHz；磁盘 1 型号为 NVMe SAMSUNG MZVLW128，容量为 119GB，类型为 SSD，磁盘 2 型号为 ST1000LM035-1RK172，容量为 932GB，类型为 HDD；编程语言为 Python 3.7，使用 scikit-learn 及相关三方库实现机器学习算法。

关于数据集。本节仍然使用第三章中所介绍的三个公开数据集进行实验，数据集介绍详见第 3.3.1 节。在本节中，对自动数据标注算法的实验在数据集的全部设备上进行，而对设备厂商增量识别的实验则根据标注算法的执行结果在部分设备上进行。

关于评价指标。为了对基于域名的自动数据标注算法的标注效果进行合理的评估，本节使用标注率（Labeling Rate），标注成功率（Labeling Success

Rate) , 标注成功数据包数目 (Labeling Success Number) 三个指标来衡量标注算法的效果。对第 i 个厂商, $LabelingRate_i$ 表示被标注的数据包数目占厂商 i 的全部数据包数目的比例, 它描述了算法的标注能力; $LabelingSuccessRate_i$ 表示被标注的数据包中事实上确实属于厂商 i 的比例, 它描述了算法的准确率; $LabelingSuccessNumber_i$ 表示被正确标注的数据包的数目。而对设备厂商增量识别实验则采取第三章中使用的准确率, 精确率, 召回率和 F1 分数的宏平均形式来衡量。

关于对比实验。为了验证将类别增量学习算法应用于智能家居设备识别领域时, 它与传统离线学习算法之间的效果差异, 同时也为了验证网络协议栈特征识别设备厂商的可行性, 本节还将 CIRF 与传统离线学习中的随机森林 (RF)、梯度提升决策树 (GBDT)、K 临近 (KNN) 进行实验结果对比。

4.4.2 自动数据标注

基于域名的数据标注算法在三个公开数据集上的执行结果分别如表4-3到4-5 所示。经统计, 数据标注算法在三个公开数据集上正确标注数据包的比例分别是 36.87%, 42.15%, 31.07%。分析实验结果可知:

算法在共计 60 个设备中的 45 个设备上实现了 20% 到 90% 的标注率, 此时算法能够为训练分类模型积累足够多的样本。标注率大小波动的原因有两点: 一是不同设备包含域名信息的流量占比不同, 例如 CUHK-AsiaCCS2020 数据集中的米兔故事机的 HTTP 协议流量和 TLS 协议流量占比超过 90%, 二者包含的 Host 和 SNI 字段使大部分数据包都能够被标注, 因此算法对其标注能力较强; 二是数据集本身存在分布不均衡的现象, 例如语音助手设备和智能摄像机设备产生的数据包基数大, 导致标注率相对较小。

算法在 iHome 等规模较小的厂商上标注效果较差, 分析数据集后发现原因在于其通信流量中的域名均来自第三方平台, 而没有能与厂商名匹配的流量。以 UNSW-TMC2018 数据集中的 iHome 公司为例: iHome 厂商设备流量中的域名以 api.evrythng.com, mqtt.evrythng.com 为主, 经 WHOIS 查询后得知这些域名属于 EVRYTHNG 公司^[50], EVRYTHNG 公司是一个物联网平台公司, 为硬件企业提供将设备连接到互联网的方法, 帮助企业对其产品进行数字化。iHome 公司选择了 EVRYTHNG 作为其智能家居产品的物联网云平台^[51]。这意味着 iHome 生产的设备仅与 EVRYTHNG 运营的服务器进行交互, 导致基于域名的数据标注算法失效。由此可知, 算法的标注效果与厂商的服务运营方式密切相关。

对于能够被标注的厂商, 标注准确率普遍较高, 只有在 Amazon 和 Google 这两家公司上标注准确率稍差, 经过手动检查发现其原因在于其他厂商的设备访问了域名包含.amazon, .google 的第三方服务, 这些域名没有被 $filter()$ 过滤, 导致部分流量被误标记为属于 Amazon 或 Google 公司。

根据数据的标注情况, 分别从三个公开数据集中选择 12 个, 6 个, 12 个设备进行后续实验, 这些设备的 $LabelingSuccessNumber$ 值最小为 4164, 更小的

表 4-3 标注算法在 UNSW-TMC2018 的执行结果**Table 4-3 Implementation results of labeling algorithm in UNSW-TMC2018.**

Type	Vendor	Labeling Rate	Labeling Success Rate	Labeling Success Number
Smart Speaker	Invoxia	34.44%	100.00%	10947
	Amazon	14.45%	96.16%	21377
Smart Camera	Dropcam	0.05%	100.00%	98
	Netatmo	46.21%	100.00%	87433
	TP-Link	25.04%	100.00%	42164
	Samsung	13.87%	100.00%	14019
	Withings	24.19%	100.00%	30972
	Insteon	27.48%	100.00%	52082
	TP-Link	47.38%	100.00%	6418
Smart Plug	Belkin	45.55%	100.00%	4729
	iHome	0.00%	0.00%	0
Smart Hub	Samsung	0.58%	100.00%	491
Smart Lamp	LIFX	11.40%	100.00%	4896
	Belkin	41.07%	100.00%	9757
Sensor	Netatmo	44.39%	100.00%	7981
	Withings	20.22%	100.00%	9827
	NEST	4.90%	100.00%	103
	Blipcare	0.00%	0.00%	0

表 4-4 标注算法在 CUHK-AsiaCCS2020 的执行结果**Table 4-4 Implementation results of labeling algorithm in CUHK-AsiaCCS2020.**

Type	Vendor	Labeling Rate	Labeling Success Rate	Labeling Success Number
Smart Speaker	Amazon	29.44%	98.99%	24607
	Google	23.66%	94.57%	41279
	tmall	49.01%	100.00%	21263
	XiaoMi	90.80%	100.00%	70043
Smart Hub	XiaoMi	41.13%	100.00%	5318
Smart Camera	360	34.01%	100.00%	42614
	XiaoMi	16.43%	100.00%	11931
Smart Plug	TPLink	21.13%	100.00%	929
	orvibo	0.00%	0.00%	0
	broadlink	0.00%	0.00%	0

表 4-5 标注算法在 CICIoT-DataSet2022-Idle 的执行结果**Table 4-5 Implementation results of labeling algorithm in CICIoT-DataSet2022-Idle.**

Type	Vendor	Labeling Rate	Labeling Success Rate	Labeling Success Number
Smart Speaker	Amazon	50.56%	97.18%	107618
	Sonos	37.94%	100.00%	53960
	Google NEST	28.81%	100.00%	80935
Smart Camera	SIMCAM	0.00%	0.00%	0
	Amcrest	20.32%	100.00%	8223
	HeimVision	0.00%	0.00%	0
	Google NEST	0.38%	100.00%	991
	Netatmo	35.50%	100.00%	35879
	Arlo	88.71%	100.00%	101539
	Sichuan AI-Link	0.00%	0.00%	0
	D-Link	23.06%	100.00%	12026
Smart Plug	Teckin	0.00%	0.00%	0
	Amazon	25.59%	100.00%	8080
	Gosund	0.00%	0.00%	0
Smart Hub	Philips	27.15%	100.00%	12308
	Eufy	26.92%	100.00%	45009
Smart Lamp	HeimVision	0.00%	0.00%	0
Sensor	Netatmo	27.37%	100.00%	7536
	D-Link	43.40%	100.00%	4164

值被视为无法为训练分类器提供足够多的样本，不适合论文提出的数据标注方法。

4.4.3 设备厂商增量识别

基于网络协议栈特征的设备厂商识别需要在相同类型设备的流量上进行，因此分别为三个数据集上的每个设备类型独立训练分类器 CIRF，以 8:2 的比例划分训练集和测试集进行评估，最终每个数据集的评价指标为该数据集上各个 CIRF 测试结果的算术平均值。

对 CIRF，首先评估在加入单一新类场景下的学习能力。在此实验中，一个类被视为新类，其余类被视为已知类，初始模型是用已知类的全部数据训练的，而新类数据则用于增量学习。轮流选取一个类作为新类重复地进行实验，最后将重复实验的实验结果取算术平均值。实验结果如表 4-6。对三个数据集均取上述重复实验中的其中一次作为分类结果展示其混淆矩阵如图 4-5 到图 4-7 所示。由实验结果可知，基于网络协议栈特征的设备厂商识别方法适用于各种类型设备下的厂商识别任务，且 CIRF 在学习新厂商类别数据后识别能力较为稳定。

表 4-6 单个新类增量识别结果
Table 4-6 Incremental recognition result of single new class.

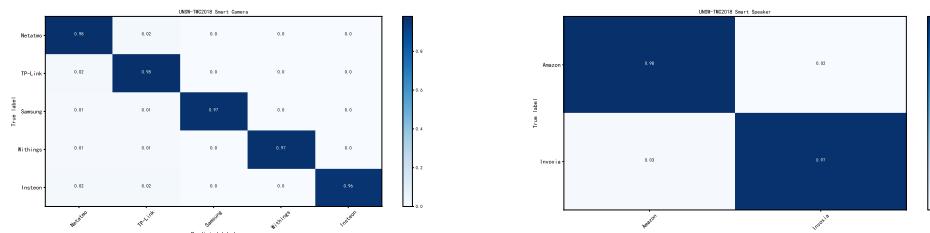
		accuracy	precision	recall	f1-score
Initial	UNSW-TMC2018	97.01%	96.84%	96.93%	96.88%
	CUHK-AsiaCCS2020	97.20%	97.04%	96.71%	96.97%
	CIC-IOT-Dataset2022-IDLE	92.38%	91.71%	92.23%	91.89%
Incremental	UNSW-TMC2018	96.93%	96.15%	96.75%	96.45%
	CUHK-AsiaCCS2020	96.97%	95.98%	96.74%	96.36%
	CIC-IOT-Dataset2022-IDLE	92.17%	91.52%	91.02%	91.27%

然后评估 CIRF 持续学习新类的能力。对三个数据集，实验分别随机生成类别序列，初始时以两个类作为已知类训练模型，随后将未知类逐一送入模型训练并测试，对每个类别依次记录测试效果。实验结果如图 4-8 所示。由实验结果可知，随着新类别厂商数据的持续送入，CIRF 的识别效果会出现上下波动，但没有发生大幅变化，最终模型的识别效果会趋于稳定，且与单一新类场景下相差不大，说明了 CIRF 对设备厂商识别任务具有强大的持续学习能力。

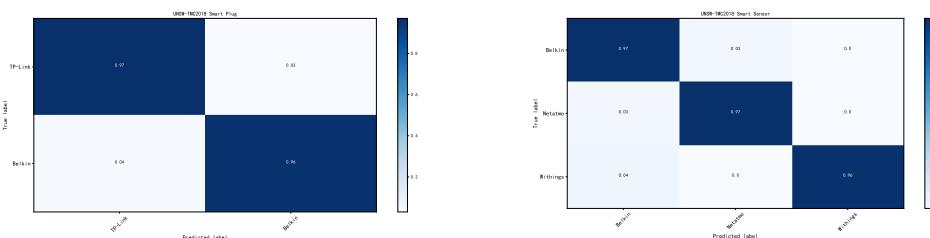
最后，为评估 CIRF 与传统离线学习算法的实验效果差异，同时也是为了验证提取的网络协议栈特征具备区分不同设备厂商之间差异的能力。随机选择两个类作为已知类，将未知类的训练集逐一送入模型进行训练，训练完成后使用全部测试集进行测试。将全部类别的训练集用于训练随机森林（RF）、梯度提升决策树（GBDT）、K 临近（KNN）算法。最终实验结果如图 4-9 所示。由实验结果可知，CIRF 的分类性能略差于传统 RF，与 GBDT 相当，明显好于 KNN，可以满足持续学习智能家居环境变化的需求。

图 4-5 数据集 UNSW-TMC2018 上单个新类增量识别混淆矩阵.

Figure 4-5 Single New Class Incremental Identification on DataSet UNSW-TMC2018.



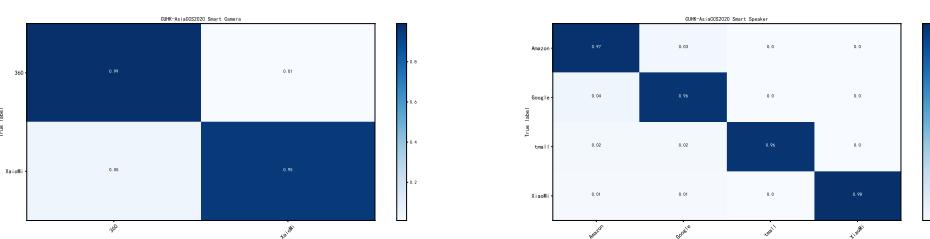
(a) 数据集 UNSW-TMC2018 上智能摄像机类型 设备混淆矩阵 (b) 数据集 UNSW-TMC2018 上语音助手类型设备混淆矩阵



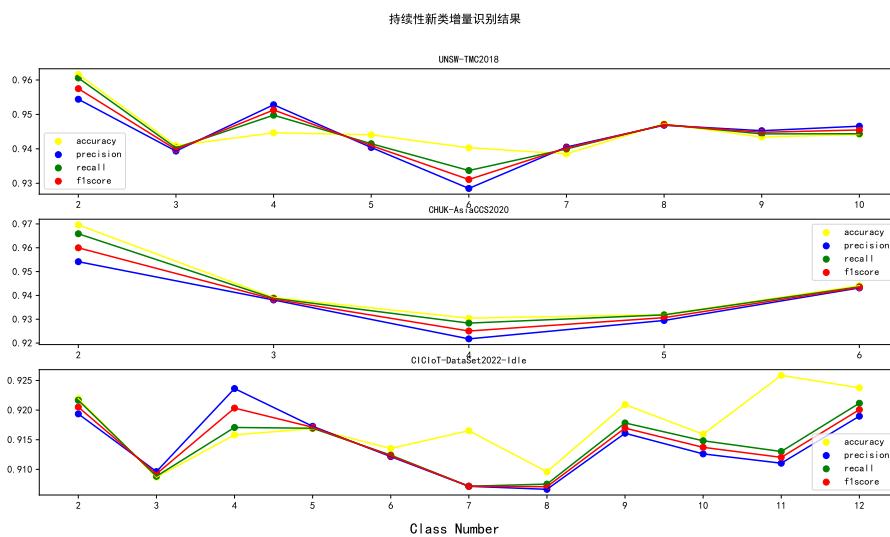
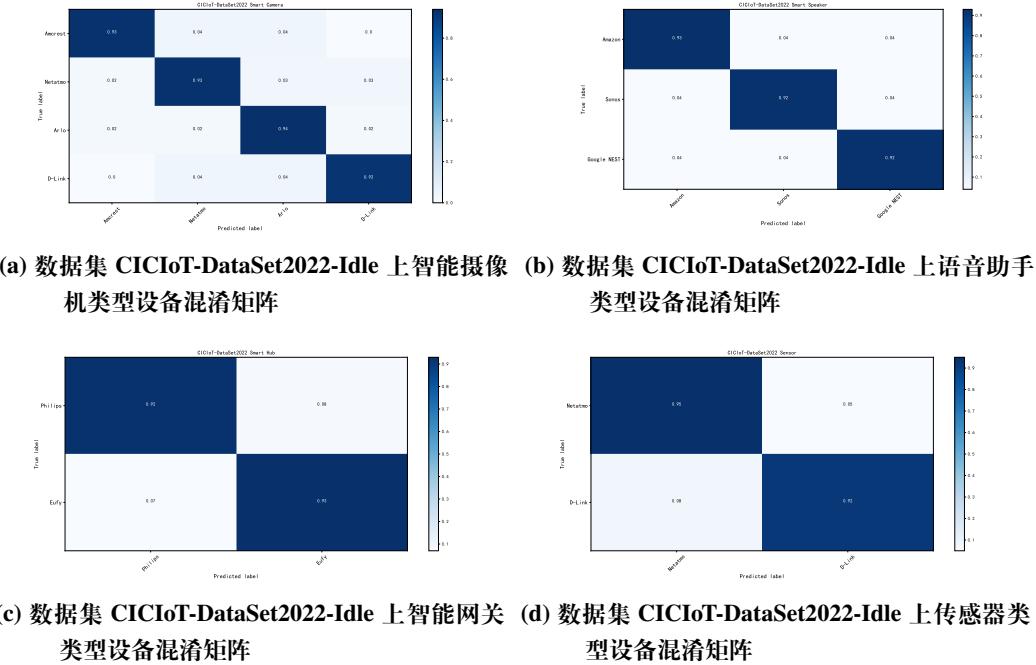
(c) 数据集 UNSW-TMC2018 上智能插座类型设备混淆矩阵 (d) 数据集 UNSW-TMC2018 上传感器类型设备混淆矩阵

图 4-6 数据集 CUHK-AsiaCCS2020 上单个新类增量识别混淆矩阵.

Figure 4-6 Single New Class Incremental Identification on DataSet CUHK-AsiaCCS2020.



(a) 数据集 CUHK-AsiaCCS2020 上智能摄像机类型设备混淆矩阵 (b) 数据集 CUHK-AsiaCCS2020 上语音助手类型设备混淆矩阵

图 4-7 数据集 CICIoT-DataSet2022-Idle 上单个新类增量识别混淆矩阵.**Figure 4-7 Single New Class Incremental Identification on CICIoT-DataSet2022-Idle.****图 4-8 持续性识别新类实验效果图****Figure 4-8 Effect Chart of Continuous Recognition New Class Experiment.**

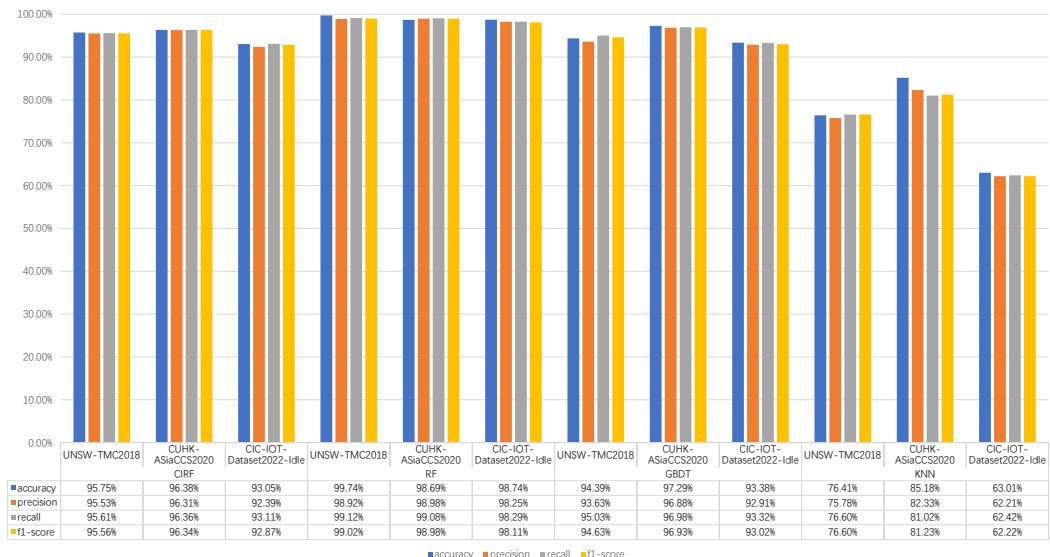


图 4-9 与传统离线学习方法对比

Figure 4-9 Comparison with traditional offline learning methods.

4.5 本章小结

本章提出了基于域名的自动化数据标注和基于类别增量学习的设备厂商识别这两种方法，共同解决了经过 NAT 的智能家居设备流量的厂商识别问题。基于域名的自动化数据标注算法通过提取并匹配包含厂商名字符串的域名信息，自动化地标注一部分数据，回答了“存在哪些厂商的物联网设备”的问题，并为有监督学习的训练提供了数据基础；基于类别增量学习的设备厂商识别首先提取流量的网络协议栈特征，该特征挖掘了不同厂商在同类型设备网络功能实现上的差异，随后实验了一种典型的类别增量学习算法——CIRF，证实了增量学习在物联网设备识别领域的可行性，解决了无法被标注流量的厂商识别问题。上述两种方法相结合，基于域名的自动数据标注使分类器可以自适应地在各不相同的环境下完成训练，解决了智能家居环境的异构性问题；类别增量学习方法的引入可以持续学习数据分布变化，解决了智能家居环境的高可变性问题。

第5章 面向智能家居环境的物联网设备识别原型系统

前两章以物联网设备识别在智能家居环境下的特殊问题为导向，分别完成了设备类型识别与设备厂商识别这两大任务。本章将基于前两章的研究成果，设计并实现一套面向智能家居环境的物联网设备识别原型系统。相对于前两章，本章考虑的是更加符合真实家居情况的网络环境，该环境中物联网设备与非物联网设备共存。为此，系统设计添加了过滤非物联网设备流量的模块，并将第三、四章的实验代码落实到工程实践中来。本章首先介绍了原型系统的总体设计架构。随后逐个介绍了系统的功能模块，包括：数据采集层，存储层，业务层，接口层，表示层共5个模块，多个功能模块被有机地组合起来。系统实现后，使用构建的模拟智能家居环境的流量数据集进行系统测试，展示了系统的性能。最后对本章内容做了小结。

5.1 引言

非物联网设备是指智能手机、平板电脑、PC等消费级终端设备。在真实的智能家居环境中，物联网设备流量与非物联网设备流量共存，大量的非物联网设备流量会给物联网设备识别模型造成严重干扰。以本论文之前提出的方法为例说明过滤非物联网设备流量的必要性：非物联网设备上多种多样的应用程序可能具有更加复杂的网络行为模式，混杂这些流量将严重干扰模型学习物联网设备的流量时序特征的能力，例如手机上视频播放软件的行为模式可能与智能摄像头设备相似。非物联网设备上多种多样的应用程序会产生大量包含域名信息的DNS、HTTP、TLS等流量，这会严重干扰基于域名信息的数据标注算法。例如小米的手机或平板发出的流量可能会被误标注成属于小米的物联网设备。因此，设计高效准确的非物联网设备流量过滤模块是实现物联网设备识别的第一步，也是关键的一步。

在过滤非物联网设备流量后，本章先后利用第三章和第四章的方法完成设备类型识别和设备厂商识别的任务。两个识别任务具有顺序关系，相同类型设备的流量将被送入同一个分类器中进行厂商识别。此外，基于第三章的方法，设备类型识别模型需要预先进行离线训练，但第三章的实验部分已证明了该模型具有较强的迁移能力，因此在迁移识别的场景下不再需要从头训练，只需要对旧模型的全连接层进行微调。基于第四章的方法，设备厂商识别模型不需要预先进行训练，而是被部署到线上后使用基于域名标注的数据自动化地完成训练。

5.2 系统总体设计

本节以绪论中的图1-3为实现目标，在深入调研流量分类领域的工程实现方案后，为原型系统设计技术架构如图5-1所示。

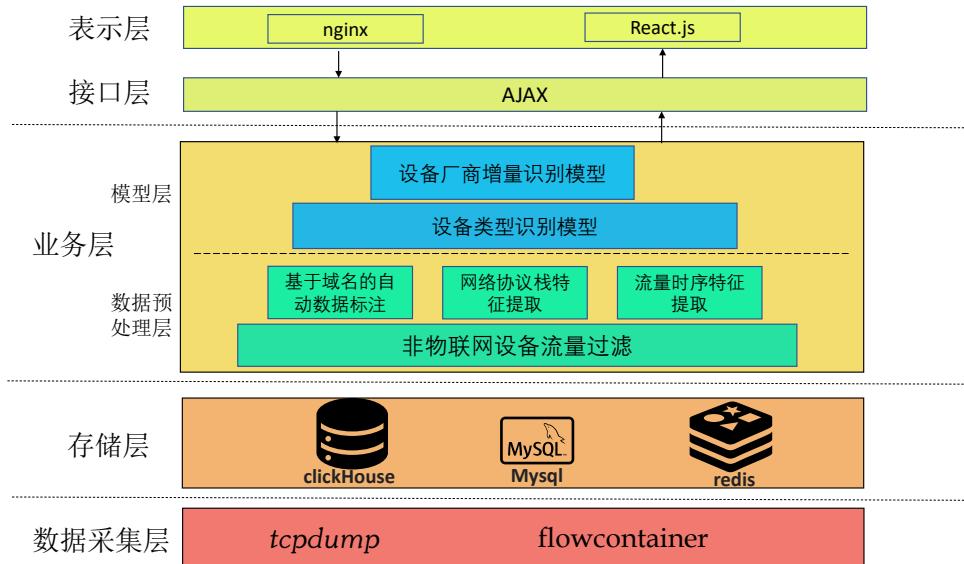


图 5-1 原型系统技术架构图

Figure 5-1 Prototype system technical architecture diagram.

系统按照实现功能依次被划分为数据采集层，存储层，业务层，接口层，表示层共 5 个模块。数据采集层实现了流量捕获与相关特征提取。存储层实现了数据持久化。数据预处理层实现了非物联网设备流量过滤、基于域名的自动数据标注，并生成对应分类模型输入的待识别特征数据。模型层维护了具备识别能力的机器学习模型：对于设备类型识别模型，它具备较强的可迁移性；对于设备厂商识别模型，它具备增量学习的能力。数据预处理层与模型层共同完成了原型系统的主要业务。表示层使用前端框架 React 构建用户界面，展示系统识别结果并与用户交互。表示层与业务层之间通过 Ajax 技术进行通信，它通过 HTTP 协议访问后端服务提供的 API，使表示层在不刷新页面的情况下动态地更新数据，因此被称作接口层。各个模块的具体实现将在第 5.3 节中详述。

5.3 系统模块实现

5.3.1 数据采集层

对经过家庭网关之后的设备流量，系统使用 `tcpdump` 持续抓取指定网卡的数据包，并生成 `pcap` 文件。使用 `flowcontainer` 库^[52] 解析与处理 `pcap` 文件，生成待持久化的数据。`flowcontainer` 是一款基于 `tshark` 的数据包解析器，它通过使用多线程等技术大大提高了数据解析速度。系统使用它从 `pcap` 文件中提取的信息包括：提取每条流的源 IP 地址、源端口号、目的 IP 地址、目的端口号、协议号，并以此五元组的值作为一条流的 key；提取流中每个数据包的数据包长度与方向信息，用于识别设备类型；提取流中每个数据包的网络协议栈特征，用于识别设备厂商；提取流中的域名信息字符串（如有），用于自动数据标注；提取流中的

User-Agent 字段值（如有），用于初步过滤非物联网设备流量；通过数据包长信息计算流的统计特征，如平均数、中位数、最大包长、最小包长等，用于过滤非物联网设备流量；此外，提取流开始时间，结束时间，帧到达时间戳等信息，虽然它们没有被直接用于设备识别，但是可以方便用户查询数据库维护系统。上述信息被写入存储层的数据库中。

5.3.2 存储层

存储层选用了 ClickHouse 数据库持久化从 flowcontainer 中提取的流量特征数据。ClickHouse 是一个开源的列式数据库管理系统，它主要被用于快速、可扩展的数据分析和查询业务，并且被设计为高性能、高可用性和高并发的数据库系统，现今已被广泛应用于大数据分析领域。ClickHouse 中存储流量特征数据的表结构示意如表 5-1 所示，受限于篇幅，该表仅展示了部分字段值的定义及说明。

表 5-1 ClickHouse 数据库流表结构

Table 5-1 Clickhouse database flow table structure.

字段	类型	说明
CliIP	String	客户端 IP 地址
SerIP	String	服务端 IP 地址
CliPort	Int64	客户端端口号
SerPort	Int64	服务端端口号
Protocol	String	应用层协议
PayloadLenArray	Array(int)	数据包长度序列
DirArray	Array(int)	数据包方向序列
TosArray	Array(Tuple(int,int))	流中 Tos 值序列
TTLArray	Array(Tuple(int,int))	流中 TTL 值序列
...
MinPayloadLen	Int32	最小包长
MaxPayloadLen	Int32	最大包长
...
DNSQuery	String	DNS 查询域名，若无为空
HTTPHost	String	HTTP 的 Host 字段，若无为空
TLSSNI	String	TLS 的 SNI 字段，若无为空
Vendor	String	设备厂商识别结果
Type	String	设备类型识别结果

存储层还选用了 Mysql 数据库记录其他业务数据表。包括：记录着典型桌面浏览器或移动设备应用程序的 User-Agent 字段值的表 user_agents；记录着事先维护的待匹配厂商名及其产品系列相关信息的表 vendors；记录着设备识别历史结果的表 IoT_identification_results。这些数据表由人工手动维护或随系统的识别结果而变化，写入频率远远小于流量特征数据，因此使用独立的存储引擎和单独一套接口维护它们。

此外，存储层添加了一层缓存层，它使用 Redis 实现，主要目标是在高并发场景下加速数据的读取速度，降低持久化数据库的负载，提升系统的并发处理能力。

5.3.3 业务层

业务层从存储层读取数据，运行原型系统的核心业务，即物联网设备识别任务。其中，业务层的“流量时序特征提取”模块，“基于域名的数据标注”模块，“网络协议栈特征提取”模块，“设备类型识别模型”和“设备厂商增量识别模型”的基本原理均已在第三、四章进行了详细说明，这里不再赘述。原型系统在实现时将它们的功能进行了组合。现从数据流向的角度解释上述各个模块的关系，即：

(1) 物联网设备流量首先送入“流量时序特征提取”模块，为每条流生成设备类型识别所需要的特征数据，并送入“设备类型识别模型”进行识别。

(2) “基于域名的数据标注”模块对部分流量进行数据标注，将标注成功的流量标签回写入 clickHouse 数据库。“网络协议栈特征提取”模块为每条流生成厂商识别所需要的特征数据。“设备厂商增量识别模型”为每种类型的设备分别维护一个 CIRF 分类器。当某个厂商被标注的数据规模达到系统设定的阈值后，其数据被送入对应类型的 CIRF 分类器中进行训练。当所有已知厂商的数据被训练完成，剩余没有标签的数据将被送入模型进行识别。

(1) 和 (2) 各自所代表的设备类型识别与设备厂商识别顺序执行，先后识别出一条流所属设备的类型和厂商信息。此外，“设备类型识别模型”需要离线训练完成再部署到线上，而“设备厂商增量识别模型”是线上自动训练的。

除上述模块外，本节重点介绍新添加的“非物联网设备流量过滤”模块。原型系统中使用了如下过程来过滤非物联网设备流量，分别为：根据协议类型过滤，根据协议字段值过滤，根据包长特征过滤。如图 5-2 所示。过滤过程详述如下：

(1) 首先，系统根据协议类型可以直接判断部分流量属于物联网设备。例如当流量属于 MQTT, RTSP, XMPP 等标准物联网协议时，系统认为这部分流量来自于物联网设备，它们将被直接送出过滤模块用于设备识别。为了实现这一功能，系统维护了一个物联网行业常见的协议格式文件，目前该文件包含最常见的 15 种物联网网络协议；

(2) 当流量属于 HTTP 协议时，系统会检查 User-Agent 字段是否是典型桌面浏览器或移动设备 APP 的形式，若是则认为流量属于非物联网设备流量，直接过滤。为了实现这一功能，系统维护了一个常见的桌面浏览器或移动 APP 的 User-Agent 值列表，近 100 个；

(3) 前两种方式只能过滤出小部分满足特定条件的物联网或非物联网设备流量，对于绝大部分流量，模块系统采用机器学习方法，使用基于包长统计特征的随机森林模型进行二分类，来判断流量是否属于物联网设备。

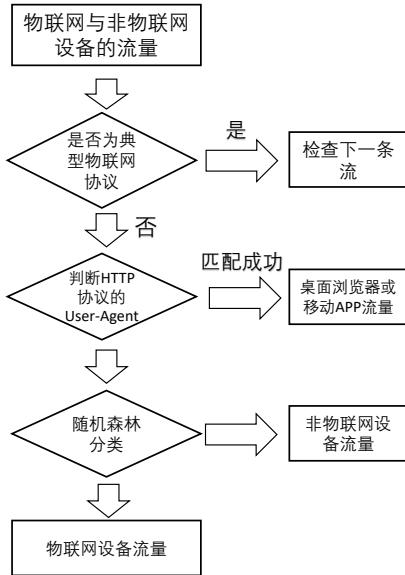


图 5-2 非物联网设备流量过滤模块执行流程图

Figure 5-2 Non-IoT device traffic filtering module flow chart.

关于使用包长信息区分物联网设备与非物联网设备的工作，之前已有 pinheiro^[29] 等人的研究作为基础。其原理在于物联网设备与非物联网设备的流量模式具有很大不同，例如：与非物联网设备相比，物联网设备在单条流内的数据包往往更少；与非物联网设备相比，物联网设备在单条流内的唯一数据包长度数量更少。造成这些现象的原因可能是物联网设备通常功能比较单一，其数据包个数与包长分布的复杂程度因此受限。

基于前人的工作，系统实现了基于包长统计特征识别物联网设备流量的分类器。首先，在一条流中提取如下特征，包括数据包数量、总字节数、唯一数据包长度的数量、数据包长度的最大值、最小值、中位数、方差和偏度。系统根据数据包的传输方向，将一条流拆分成传入流量，传出流量与双向流量，系统在设备的三种流量上分别提取上述特征，拼接在一起，因此由一条原始流得到一个长度为 24 的特征向量。系统选用随机森林算法进行训练和测试，随机森林算法具有强大的表征能力和鲁棒性，在流量分类领域的工程实践中被广泛应用。

基于上述三个步骤，非物联网设备流量过滤模块的最终效果将在第 5.4.2 节中给出。

5.3.4 接口层和表示层

业务层，存储层和数据采集层组成了原型系统的大后端。接口层与表示层负责将后端数据进行展示。

接口层使用 Ajax (Asynchronous JavaScript and XML) 技术向后端发送 HTTP 请求，获取业务端的识别结果及相关信息。Ajax 是一种用于创建交互式 Web 应用的技术。它利用 JavaScript 与服务器进行异步通信，从而在页面不需要重新加

载的情况下更新部分内容，提高用户的交互体验。

表示层使用前端框架 React.js^[53] 实现，用于构建用户界面。它通过组件化的方式来管理 UI，并且使用虚拟 DOM (virtual DOM) 技术来提高性能。此外，系统使用 nginx 来管理 React.js 应用程序，作为一个反向代理服务器，nginx 将所有来自客户端的请求转发给 React.js 应用程序的服务器，并将得到的结果返回给浏览器，从而管理 React.js 应用程序。通过表示层实现的前端页面有两个，在流量视角页面，以表格的形式展示每条流的识别结果，用户可以根据 IP 地址、时间范围等规则搜索特定条件的流量；在设备视角页面，以饼状图的形式展现当前流量中包含的厂商及设备类型占比情况，它是定期刷新的。

5.4 系统测试

5.4.1 测试环境

关于系统运行环境。原型系统被部署在服务器上进行实验，服务器操作系统为 Ubuntu Server 系统；CPU 型号为 Intel(R) Xeon(R) Silver 4110CPU@2.10GHz；深度学习模型使用的 GPU 型号为 Tesla V100，16 GB 内存，1.53 GHz；服务器内存最大为 128GB。实现方面，使用 Python 3.7 实现数据采集层和业务层的代码逻辑；使用 Pytorch 1.12.0 及 scikit-learn 实现机器学习模型；使用 React 0.14.3 构建表示层。

表 5-2 系统测试数据集

Table 5-2 System test data set.

	类别	设备 MAC	设备名
物联网设备	智能摄像头	70:ee:50:18:34:43	Netatmo Welcome
		00:16:6c:ab:6b:88	Samsung SmartCam
		f4:f2:6d:93:51:f1	TP-Link Day Night Cloud camera
	智能语音助手	44:65:0d:56:cc:d3	Amazon Echo
		18:b7:9e:02:20:44	Invoxia Triby Speaker
	智能电灯	d4:a6:51:30:64:b7	HeimVision SmartLife Lamp
		d0:73:d5:01:83:08	Light Bulbs LiFX Smart Bulb
	传感器	ec:1a:59:83:28:11	Belkin wemo motion sensor
		f0:b4:d2:f9:60:95	D-Link DCHS-161 Water Sensor
		70:ee:50:03:b8:ac	Netatmo weather station
		00:24:e4:20:28:c6	Withings Aura smart sleep sensor
	智能插座	ec:1a:59:79:f4:89	Belkin Wemo switch
		50:c7:bf:00:56:39	TP-Link Smart plug
	智能网关	b8:5f:98:d0:76:e6	Amazon Plug
		8c:85:80:6c:b6:47	Eufy HomeBase
		00:17:88:60:d6:4f	Philips Hue Bridge
非物联网设备	智能手机	40:f3:08:ff:1e:da	Android Phone
		a4:50:46:06:80:43	Xiaomi mobile
		d0:a6:37:df:a1:e1	Iphone
	个人电脑	ac:bc:32:d4:6f:2f	MacBook
		74:2f:68:81:69:42	Laptop

关于实验数据集。本章仍然使用前述实验中的公开数据集进行测试。但是第三章和第四章使用的数据集为纯净的物联网设备流量数据集，为了尽可能模

拟真实的智能家居环境，本章混杂了非物联网设备与物联网设备流量构造了一个系统测试数据集，该数据集中的非物联网设备流量来自 UNSW-TMC2018。数据集的设备详细情况如表 5-2 所示。非物联网设备与物联网设备流量的占比约为 4:6。

关于评价指标，本章仍取准确率（Accuracy），精确率（Precision），召回率（Recall）和 F1 分数（F1-score）的宏平均（macro）形式。

5.4.2 非物联网设备流量过滤模块测试

“非物联网设备流量过滤”模块的实验工作没有被包含在第三、四章的研究内容中，因此在本节单独给出。使用系统测试数据集验证模块的识别效果，以 7:3 的比例划分训练集和测试集，分类结果混淆矩阵如图 5-3 所示。准确率，精确率，召回率和 F1 分数的值分别为 99.11%，98.87%，98.31%，98.59%。由实验结果可知，该模块能够以较高的精度完成现实环境中对非物联网设备流量过滤的任务。

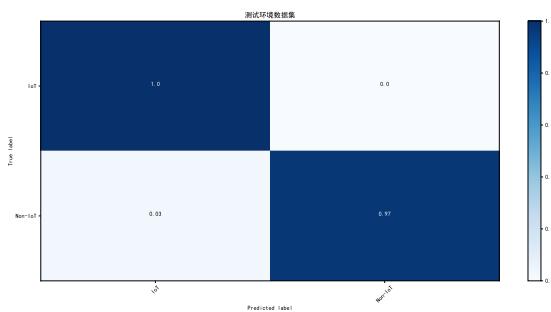


图 5-3 非物联网设备流量过滤实验混淆矩阵

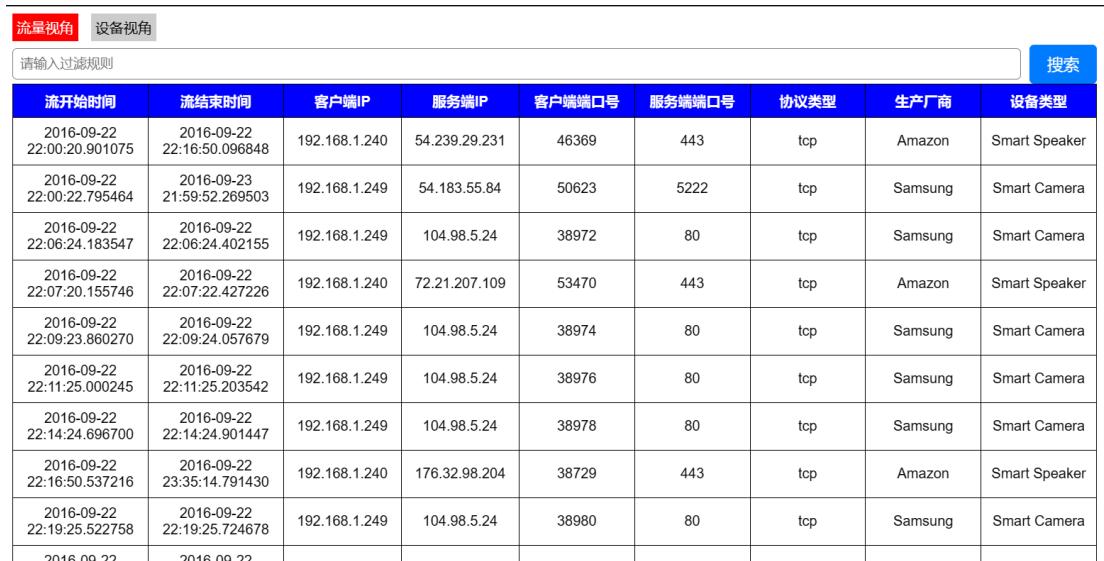
Figure 5-3 Confusion Matrix for Non-IoT Device Traffic Filtering.

5.4.3 集成测试

原型系统中，设备类型识别模型需要预先进行离线训练，因此将系统测试数据集以 8:2 的比例划分训练集与测试集，在系统测试时使用测试集验证设备类型识别效果；设备厂商识别模型不需要预先进行训练，而是被部署到线上后自动化地使用基于域名标注的数据完成训练，待训练完成后，使用没有被标注的数据用于模型识别，因此集成测试中全体未标注的数据被看作测试集，用以验证设备厂商识别效果。

使用原型系统处理系统测试数据集，前端运行结果如图 5-4 和图 5-5 所示。图 5-4 以流量视角展示了每条流的识别结果，表格中每一行即为五元组定义的流，后两个字段为该流的识别结果。图 5-5 以设备视角展示了流量中包含的厂商及设备类型占比情况。

原型系统在构造的系统测试数据集上的实验结果如表 5-3 和表 5-4 所示。由实验结果可知，即使原始数据集中混杂了大量非物联网设备的流量，原型系统



The screenshot shows a table with 10 columns: 流开始时间 (Flow Start Time), 流结束时间 (Flow End Time), 客户端IP (Client IP), 服务端IP (Server IP), 客户端端口号 (Client Port), 服务端端口号 (Server Port), 协议类型 (Protocol Type), 生产厂商 (Manufacturer), and 设备类型 (Device Type). The data consists of 10 rows of network flow logs. The last row is a header row with the same column names.

流开始时间	流结束时间	客户端IP	服务端IP	客户端端口号	服务端端口号	协议类型	生产厂商	设备类型
2016-09-22 22:00:20.901075	2016-09-22 22:16:50.096848	192.168.1.240	54.239.29.231	46369	443	tcp	Amazon	Smart Speaker
2016-09-22 22:00:22.795464	2016-09-23 21:59:52.269503	192.168.1.249	54.183.55.84	50623	5222	tcp	Samsung	Smart Camera
2016-09-22 22:06:24.183547	2016-09-22 22:06:24.402155	192.168.1.249	104.98.5.24	38972	80	tcp	Samsung	Smart Camera
2016-09-22 22:07:20.155746	2016-09-22 22:07:22.427226	192.168.1.240	72.21.207.109	53470	443	tcp	Amazon	Smart Speaker
2016-09-22 22:09:23.860270	2016-09-22 22:09:24.057679	192.168.1.249	104.98.5.24	38974	80	tcp	Samsung	Smart Camera
2016-09-22 22:11:25.000245	2016-09-22 22:11:25.203542	192.168.1.249	104.98.5.24	38976	80	tcp	Samsung	Smart Camera
2016-09-22 22:14:24.696700	2016-09-22 22:14:24.901447	192.168.1.249	104.98.5.24	38978	80	tcp	Samsung	Smart Camera
2016-09-22 22:16:50.537216	2016-09-22 23:35:14.791430	192.168.1.240	176.32.98.204	38729	443	tcp	Amazon	Smart Speaker
2016-09-22 22:19:25.522758	2016-09-22 22:19:25.724678	192.168.1.249	104.98.5.24	38980	80	tcp	Samsung	Smart Camera
2016-09-22	2016-09-22							

图 5-4 原型系统的流量视角观察设备识别效果截图

Figure 5-4 Screenshot of the prototype system's traffic view to observe the IoT recognition effect.

仍然较好地完成了设备识别的任务，并且在大部分类型和厂商上的识别效果与前文在纯净物联网设备流量上的实验效果相当。

表 5-3 原型系统设备类型识别结果

Table 5-3 Prototype system device type identification results.

	precision	recall	f1-score
Smart Camera	97.61%	96.55%	97.08%
Smart Lamp	91.36%	87.72%	89.50%
Smart Speaker	98.47%	96.83%	97.64%
Smart Hub	79.11%	98.42%	87.71%
Smart Plug	85.88%	93.26%	89.42%
Sensor	95.86%	96.71%	96.28%

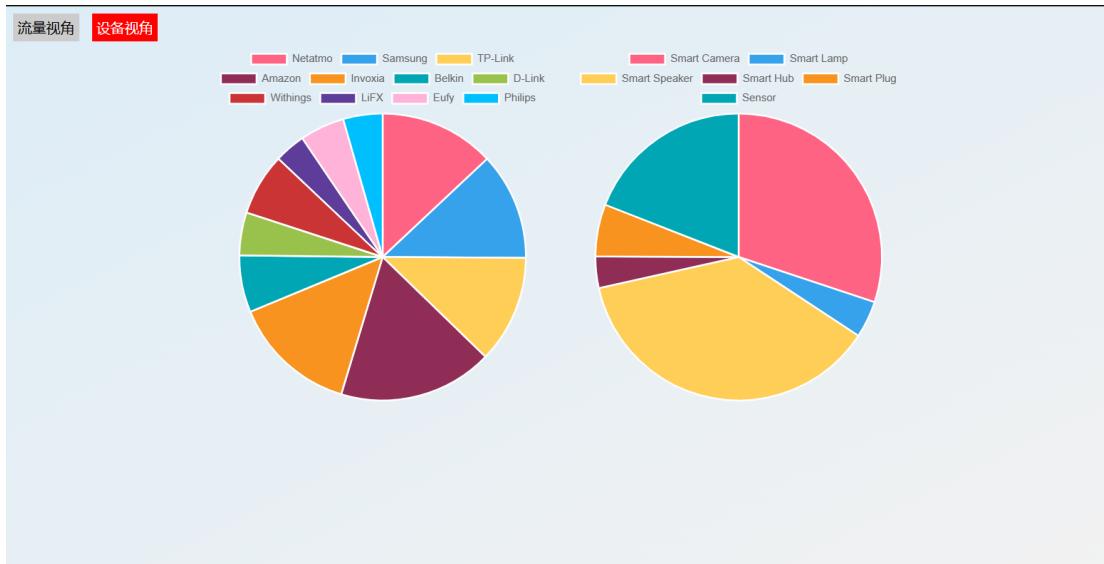


图 5-5 原型系统的设备视角观察设备识别效果截图

Figure 5-5 Screenshot of the prototype system's device view to observe the IoT recognition effect.

表 5-4 原型系统厂商识别结果

Table 5-4 Prototype system's device vendor identification results.

Type	Vendor	98.72%	97.70%	98.21%
Netatmo	98.16%	96.51%	97.32%	
Smart Camera	Samsung	98.48%	97.38%	97.92%
	TP-Link	95.41%	97.44%	96.42%
Smart Speaker	Amazon	96.38%	95.90%	96.14%
	Invoxia	98.69%	94.60%	96.61%
Smart Lamp	LiFX	95.04%	95.70%	95.37%
	Belkin	94.38%	96.38%	95.37%
Sensor	Netatmo	93.83%	95.39%	94.60%
	D-Link	96.00%	94.74%	95.36%
	Withings	97.99%	97.32%	97.66%
Smart Plug	Belkin	96.50%	96.15%	96.33%
	TP-Link	94.40%	97.44%	95.90%
	Amazon	95.35%	97.55%	96.44%
Smart Hub	Eufy	96.74%	96.24%	96.49%
	Philips	97.03%	96.85%	96.94%

5.5 本章小结

本章考虑的是真实的智能家居环境下的物联网设备识别，这是一个非物联网设备与物联网设备共存的复杂环境。为此，本章设计并实现了面向智能家居环境的物联网设备识别原型系统，系统基于第三章与第四章的研究结果，并添加了过滤非物联网设备流量的新模块，结合流行的数据分析网站前后端技术栈，

为智能家居环境下的物联网设备识别提供工程解决方案。系统的设备类型识别模型具有较强的迁移能力，意味着模型能够以极低的代价在异构的智能家居环境之间复用；系统的设备厂商识别模块具有线上自动训练的特点，这解决了智能家居环境的异构性和高可变性问题。值得一提的是，尽管系统的设备类型识别结果看起来是模糊或粗粒度的，但这对于保障模型的迁移能力来说是必要的，之前的研究中以设备本身作为分类目标，很容易造成模型对源域数据集的过拟合，阻碍模型的迁移能力，这在第三章的实验中已经得到验证。因此，本章提出的原型系统在解决智能家居环境特殊问题的前提下，以提高模型泛化性和扩展性的方式完成了物联网设备识别的任务。

第6章 总结与展望

物联网设备识别的问题由来已久，但是鲜有专门针对智能家居环境的特殊问题来优化物联网设备识别方法的工作。本文以该问题为导向，研究了基于流量分类的智能家居设备识别技术，实现了一套灵活性和扩展性更强的智能家居设备识别解决方案，完成了智能家居环境下的设备类型识别与设备厂商识别的任务。

6.1 本文工作总结

智能家居环境存在的特殊问题导致现有的物联网设备识别方法很难被有效地应用。具体而言：智能家居环境的设备被 NAT 所隐藏，具有隐蔽性，因此基于主动探测的方法不适用于该环境；智能家居设备在一段时间内被频繁更替，具有高可变性，因此以往的在一组特定的设备上进行离线训练的模型不能适用；现实中任意两个家居的物联网设备情况可能大不相同，具有异构性，因此迁移能力较差的孤立模型将完全不能应用于新环境，这大大限制了设备识别模型的推广能力。

以解决智能家居环境的特殊问题为导向，结合物联网设备流量的实际情况，本文综合使用迁移学习思想和增量学习技术，分别完成了设备类型识别与设备厂商识别的任务，最终完成了面向智能家居环境的物联网设备识别原型系统。

(1) 基于流量时序特征的设备类型识别及迁移能力研究。该工作解决了智能家居环境下的设备类型识别问题。首先，从流量元数据中提取流量时序特征，使用 Bi-LSTM 和 1D-CNN 的混合神经网络挖掘不同类型设备的网络行为模式，其中 Bi-LSTM 通过门结构选择记忆长段流量的全局特征，1D-CNN 通过卷积核捕获小段流量的局部特征，最终实现设备类型识别。然后，实验证明了该方法在不同的智能家居设备环境下具有较强的可迁移性，即仅通过训练最后一层全连接神经网络便能使模型在新环境下保持较高的识别精度，大大降低了模型迁移到异构的智能家居环境下所带来的资源浪费。该工作在完成设备类型识别任务的基础上，通过设计具有较强迁移能力的识别模型，解决了智能家居环境的高可变性和异构性问题。

(2) 基于自动数据标注与类别增量学习的设备厂商识别。该工作解决了智能家居环境下的设备厂商识别问题。首先，基于大部分物联网设备会不定期与其厂商运营的服务器通信这一基本事实，实现了基于域名的自动数据标注算法，该算法通过提取流量中的域名信息与厂商名进行匹配，可以标注一部分流量，作为训练分类器的数据基础。随后，基于不同厂商对同类型的设备在网络协议栈实现上的不同，在同类型设备的流量中提取特征训练机器学习分类器，识别无法被标注的流量。此外，为了应对智能家居环境高可变的问题，引入了一种基于类别

增量学习的随机森林方法——CIRF，使得分类器可以适应持续加入新厂商设备的数据环境。基于域名的自动数据标注使分类模型可以自适应地在各不相同的环境下完成训练，解决了智能家居环境的异构性问题；类别增量学习方法的引入使分类器可以持续学习数据分布变化，解决了智能家居环境的高可变性问题。

(3) 面向智能家居环境的物联网设备识别原型系统。基于前文研究成果，本文设计并实现了一套面向智能家居环境的物联网设备识别原型系统。该原型系统考虑的是真实的智能家居环境，因此增添了非物联网设备流量过滤模块，并同时将前述实验代码落实到工程项目中来，例如，使用了 ClickHouse 数据库作为存储流量特征数据的工具。最终使用构建的模拟真实智能家居环境的数据集进行实验，验证了系统具有良好的识别能力。

6.2 未来展望

本文对面向智能家居环境的物联网设备识别问题进行了深入分析，基于前述工作，论文认为未来的工作可以在如下几点进行展开：

(1) 智能家居设备类型识别的研究工作中，分类标签共有六类，这是基于三个公开数据集的设备情况而自行定义的，它们不能囊括多种多样的物联网设备，因此设备类型可以被进一步扩展和细化，优化识别效果。

(2) 基于域名的数据标注算法较为依赖物联网设备所通信服务器的情况。经过在三个公开数据集上共 60 个设备的实验发现，有 10 个设备的流量中完全没有发现与其厂商名有关的域名信息，有 5 个设备的被标注数据包数目不过千，因此这 15 个设备的厂商将无法被后续的机器学习方法识别。针对该问题，未来应该致力于改进算法的匹配方式，提高算法的标注能力，例如借助 WHOIS 服务来帮助算法使用不包含厂商名字符串的域名信息，从而实现对这部分流量的标注。

(3) 物联网设备流量数据集普遍存在数据分布不均衡的现象，例如同一个采集时间段里，单个智能摄像机设备的流量远多于单个传感器设备。因为数据分布不均衡，设备类型识别的实验中出现了小样本类别的样本被误分成大样本类别的情况，造成评价指标中的准确率较高而精度较低。未来应该设计合适的方法来解决小样本识别问题，对智能家居环境的设备识别问题颇为关键。

参考文献

- [1] Union I T. Internet of things: Iot day special: volume 7 [M]. LexInnova Technologies, LLC, 2005.
- [2] Lueth K L, et al. State of the iot 2018: Number of iot devices now at 7b—market accelerating [J]. IoT Analytics, 2018, 8.
- [3] Schiefer M. Smart home definition and security threats [C]//2015 ninth international conference on IT security incident management & IT forensics. IEEE, 2015: 114-118.
- [4] MiSecurity. Cyber-security-baseline-for-consumer-internet-of-things [J/OL]. GitHub repository, 20. <https://github.com/charlespwd/project-title>.
- [5] Han T, Han B, Zhang L, et al. Coexistence study for wifi and zigbee under smart home scenarios [C]//2012 3rd IEEE International Conference on Network Infrastructure and Digital Content. IEEE, 2012: 669-674.
- [6] NVD-CVE-2018-1150. Nvd-cve-2018-1150 [EB/OL]. 2018. <https://nvd.nist.gov/vuln/detail/CVE-2018-1150>.
- [7] Alharbi R, Aspinall D. An iot analysis framework: An investigation of iot smart cameras' vulnerabilities [J]. 2018.
- [8] Kolias C, Kambourakis G, Stavrou A, et al. Ddos in the iot: Mirai and other botnets [J]. Computer, 2017, 50(7): 80-84.
- [9] Egevang K, Francis P. The ip network address translator (nat) [R]. 1994.
- [10] Lu N, Zhang G, Lu J. Concept drift detection via competence models [J]. Artificial Intelligence, 2014, 209: 11-28.
- [11] Gaur P, Tahiliani M P. Operating systems for iot devices: A critical survey [C]//2015 IEEE region 10 symposium. IEEE, 2015: 33-36.
- [12] Xiaomi. Xiaomi vela [EB/OL]. 2022. <https://iot.mi.com/vela>.
- [13] Kohno T, Broido A, Claffy K C. Remote physical device fingerprinting [J]. IEEE Transactions on Dependable and Secure Computing, 2005, 2(2): 93-108.
- [14] Vanhoef M, Matte C, Cunche M, et al. Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms [C]//Proceedings of the 11th ACM on Asia conference on computer and communications security. 2016: 413-424.
- [15] Lyon G F. Nmap network scanning: The official nmap project guide to network discovery and security scanning [M]. Insecure, 2009.
- [16] Durumeric Z, Wustrow E, Halderman J A. Zmap: Fast internet-wide scanning and its security applications. [C]//USENIX Security Symposium: volume 8. 2013: 47-53.
- [17] Shodan. The search engine for internet-connected devices. [EB/OL]. 2022. <https://www.shodan.io/>.
- [18] Li Q, Feng X, Zhao L, et al. A framework for searching internet-wide devices [J]. IEEE Network, 2017, 31(6): 101-107.
- [19] Feng X, Li Q, Wang H, et al. Characterizing industrial control system devices on the internet [C]//2016 IEEE 24th International Conference on Network Protocols (ICNP). IEEE, 2016: 1-10.
- [20] 邹宇驰, 刘松, 于楠, 等. 基于搜索的物联网设备识别框架 [J]. Journal of Cyber Security 信息学报, 2018, 3(4).

- [21] Feng X, Li Q, Wang H, et al. Acquisitional rule-based engine for discovering internet-of-things devices [C]//27th {USENIX} Security Symposium ({USENIX} Security 18). 2018: 327-341.
- [22] Wang X, Wang Y, Feng X, et al. Iottracker: an enhanced engine for discovering internet-of-thing devices [C]//2019 IEEE 20th International Symposium on” A World of Wireless, Mobile and Multimedia Networks”(WoWMoM). IEEE, 2019: 1-9.
- [23] Guo H, Heidemann J. Detecting iot devices in the internet [J]. IEEE/ACM Transactions on Networking, 2020, 28(5): 2323-2336.
- [24] Sivanathan A, Gharakheili H H, Loi F, et al. Classifying iot devices in smart environments using network traffic characteristics [J]. IEEE Transactions on Mobile Computing, 2018, 18 (8): 1745-1759.
- [25] Meidan Y, Bohadana M, Shabtai A, et al. Profiliot: A machine learning approach for iot device identification based on network traffic analysis [C]//Proceedings of the symposium on applied computing. 2017: 506-509.
- [26] Bai L, Yao L, Kanhere S S, et al. Automatic device classification from network traffic streams of internet of things [C]//2018 IEEE 43rd conference on local computer networks (LCN). IEEE, 2018: 1-9.
- [27] Radhakrishnan S V, Uluagac A S, Beyah R. Gtid: A technique for physical device and device type fingerprinting [J]. IEEE Transactions on Dependable and Secure Computing, 2014, 12 (5): 519-532.
- [28] Marchal S, Miettinen M, Nguyen T D, et al. Audi: Toward autonomous iot device-type identification using periodic communication [J]. IEEE Journal on Selected Areas in Communications, 2019, 37(6): 1402-1412.
- [29] Pinheiro A J, Bezerra J d M, Burgardt C A, et al. Identifying iot devices and events based on packet length from encrypted traffic [J]. Computer Communications, 2019, 144: 8-17.
- [30] Msadek N, Soua R, Engel T. Iot device fingerprinting: Machine learning based encrypted traffic analysis [C]//2019 IEEE wireless communications and networking conference (WCNC). IEEE, 2019: 1-8.
- [31] Dong S, Li Z, Tang D, et al. Your smart home can't keep a secret: Towards automated fingerprinting of iot traffic [C]//Proceedings of the 15th ACM Asia Conference on Computer and Communications Security. 2020: 47-59.
- [32] Pashamokhtari A, Okui N, Miyake Y, et al. Inferring connected iot devices from ipfix records in residential isp networks [C]//2021 IEEE 46th Conference on Local Computer Networks (LCN). IEEE, 2021: 57-64.
- [33] Meidan Y, Sachidananda V, Peng H, et al. A novel approach for detecting vulnerable iot devices connected behind a home nat [J]. Computers & Security, 2020, 97: 101968.
- [34] Miettinen M, Marchal S, Hafeez I, et al. Iot sentinel: Automated device-type identification for security enforcement in iot [C]//2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017: 2177-2184.
- [35] Shaikh F, Bou-Harb E, Crichigno J, et al. A machine learning model for classifying unsolicited iot devices by observing network telescopes [C]//2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC). IEEE, 2018: 938-943.
- [36] Sun J, Sun K, Shenefiel C. Automated iot device fingerprinting through encrypted stream classification [C]//Security and Privacy in Communication Networks: 15th EAI International Conference, SecureComm 2019, Orlando, FL, USA, October 23-25, 2019, Proceedings, Part I. Springer, 2019: 147-167.

- [37] 任春林, 谷雨, 崔杰, 等. 基于 WEB 信息的特定类型物联网终端识别方法 [J]. 通信技术, 2017, 50(5): 1003-1009.
- [38] Li Q, Feng X, Wang H, et al. Automatically discovering surveillance devices in the cyberspace [C]/Proceedings of the 8th ACM on Multimedia Systems Conference. 2017: 331-342.
- [39] Yang K, Li Q, Sun L. Towards automatic fingerprinting of iot devices in the cyberspace [J]. Computer networks, 2019, 148: 318-327.
- [40] Yu L, Luo B, Ma J, et al. You are what you broadcast: Identification of mobile and iot devices from (public) wifi. [C]/USENIX Security Symposium. 2020: 55-72.
- [41] Yin F, Yang L, Ma J, et al. Identifying iot devices based on spatial and temporal features from network traffic [J]. Security and Communication Networks, 2021, 2021: 1-16.
- [42] Hu X, Li H, Shi Z, et al. A robust iot device identification method with unknown traffic detection [C]/Wireless Algorithms, Systems, and Applications: 16th International Conference, WASA 2021, Nanjing, China, June 25–27, 2021, Proceedings, Part I 16. Springer, 2021: 190-202.
- [43] javenleeCH. Xiaomi's mihome binary protocol [J/OL]. Gitee repository, 20. <https://gitee.com/javenleeCH/mihome-binary-protocol/tree/master>.
- [44] Kolcun R, Popescu D A, Safronov V, et al. Revisiting iot device identification [J]. arXiv preprint arXiv:2107.07818, 2021.
- [45] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database [C]// 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009: 248-255.
- [46] Dadkhah S, Mahdikhani H, Danso P K, et al. Towards the development of a realistic multi-dimensional iot profiling dataset [C]/2022 19th Annual International Conference on Privacy, Security & Trust (PST). IEEE, 2022: 1-11.
- [47] Wang W, Zhu M, Wang J, et al. End-to-end encrypted traffic classification with one-dimensional convolution neural networks [C]/2017 IEEE international conference on intelligence and security informatics (ISI). IEEE, 2017: 43-48.
- [48] WHOIS. Icann:whois. [EB/OL]. 1983. <https://whois.icann.org/en>.
- [49] Hu C, Chen Y, Hu L, et al. A novel random forests based class incremental learning method for activity recognition [J]. Pattern Recognition, 2018, 78: 277-290.
- [50] evrythng. Evrythng. [EB/OL]. 2023. <https://developers.evrythng.com/>.
- [51] yahoo. Evrythng selected by ihome as iot cloud to power its smart home consumer electronics products. [EB/OL]. 2015. <https://finance.yahoo.com/news/evrythng-selected-ihome-iot-cloud-120000737.html>.
- [52] JiangMinghao. jmhicoding/flowcontainer [EB/OL]. 2022. <https://github.com/jmhIcoding/flowcontainer>.
- [53] React. React –a javascript library for building user interfaces [EB/OL]. 2023. <https://reactjs.org/>.

致 谢

在我即将完成毕业论文的时刻，我要向所有曾经给予我支持和帮助的人表示由衷的感谢。

我要特别感谢我的导师舒敏老师。在我研究生阶段的学习和科研生涯中，舒敏老师一直是我最重要的指导者和支持者。她在我研究工作中的细心指导和严谨治学的态度，给我带来了很多启发和帮助。我将永远感激舒敏老师对我的关心和帮助，这是我研究生生活中最宝贵的财富。我要感谢熊刚老师、苟高鹏老师、侯承尚老师。熊刚老师在我研究生的学习生涯中，一直以来都是我心目中的学术楷模，他严谨治学的态度和优秀的学术成果一直激励着我前进。苟高鹏老师不仅在学术研究上给予了我很多指导和帮助，还在我的生活和心理方面给予了我很多关心和照顾，在我完成这篇毕业论文的过程中，苟高鹏老师一直给予我指导和帮助，让我能够顺利地完成这篇论文。侯承尚老师给了我很多宝贵的建议和指导，让我能够更好地开展研究工作，帮助我在学术研究的道路上不断前行。我感激他们对我在学术和生活方面的关心和支持，没有他们的帮助，我无法完成这篇毕业论文。最后，我还要感谢李镇老师、刘畅老师，他们在我的学习生涯中给予我很多的支持和指导，感谢王清老师，她对待学生工作一丝不苟，为我营造了一个良好的日常学习环境。在此我向所有老师呈献最高的敬意和感谢。

其次，我要感谢组内的师兄师姐们，感谢朱博士，管中师兄，蒋明昊师兄，崔天宇师兄，蔡玮师兄，感谢杨琛师兄，李思佳师姐，他们为我的研究生生活提供了巨大的帮助和支持，在我还是一个对网络行为学领域一无所知的小白时，师兄师姐们总是耐心地解答我的问题，给予我很多宝贵的建议和意见，鼓励和支持我完成学业。

最后，我要感谢与我同窗三年的郭敬宇同学，顾哲媛同学，赵晨同学，张西源同学，徐林峰同学，周思源同学，夏耀华同学，黄一洋同学，时光荏苒，感谢三年前的今天和你们相遇，感谢长达三年的陪伴，你们为我提供了很多帮助与支持，督促我进步。

最后，再次衷心地感谢所有帮助我、支持我的老师、同学和朋友们。

2023 年 6 月

作者简历及攻读学位期间发表的学术论文与其他相关学术成果

作者简历：

胡伟业，河南省濮阳市人，中国科学院大学信息工程研究所硕士研究生。

2016年09月——2020年06月，在华北电力大学计算机系获得学士学位。

2020年09月——2023年06月，在中国科学院信息工程研究所攻读硕士学位。

工作经历：无

参加的研究项目及获奖情况：

- (1) 2021年6月-2021年7月，参与加密隧道移动应用流量采集项目，负责多个主流加密隧道环境的搭建与流量采集工作。
- (2) 2021年10月-2022年1月，参与某品牌路由器设备型号识别研究项目。基于被动流量分析与机器学习方法给出不同型号路由器的特征，生成某品牌路由器的网络行为日志。

