电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

硕士学位论文

MASTER THESIS

论文题目 融合主机和网络的车联网入侵检测系统设计与实现

学科专业 计算机科学与技术

学　　号 201821080855

作者姓名 刘健男

指导老师 罗建超　副教授

分类号 _____ 密级 _____

UDC 注 1 _____

学 位 论 文

融合主机和网络的车联网入侵检测系统设计与实现

（题名和副题名）

刘健男

（作者姓名）

指导老师 罗建超 副教授

电子科技大学 成都

（姓名、职称、单位名称）

申请学位级别 硕士 学科专业 计算机科学与技术

提交论文日期 2021.3.18 论文答辩日期 2021.5.25

学位授予单位和日期 电子科技大学 2021 年 6 月

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

Design and Implementation of Intrusion Detection System for Connected Cars Integrating Host-level and Network-level Detection

A Master Thesis Submitted to
University of Electronic Science and Technology of China

Discipline:	**Computer Science and Technology**
Author:	**Jiannan Liu**
Supervisor:	**Prof. Jianchao Luo**
School:	**School of Computer Science and Engineering**

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，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，也不包含为获得电子科技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谢意。

作者签名：刘健男　　日期：2021 年 5 月 31 日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学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、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名：刘健男　　导师签名：[illegible]

日期：2021 年 5 月 31 日

摘　要

近年来，无人驾驶，智能汽车，智慧交通等概念爆发式涌现，汽车已不是以前相对封闭，相对独立的个体结构。为了智能化，汽车需要更加频繁的与外部环境交互，这种交互以车与人，车与车，车与智能交通设备之间通过无线网络或移动网络的通信为基础，从而形成车联网概念。车联网带来了智能便利的同时，也埋下了信息安全隐患，大部分车载终端缺乏有效手段阻止网络攻击行为，如窃取个人隐私，远程控制汽车等会对生命与财产产生严重威胁的攻击。入侵检测是有效的网络安全防护手段之一，本文为解决汽车终端车联网安全问题，提出一种基于神经网络算法的融合主机级别和网络级别的入侵检测系统。

本文对设计与实现融合主机与网络的车联网入侵检测系统进行如下工作：

1. 网络级入侵检测模型设计。车联网可分为基于 TCP/IP 协议与外部通信的网络和基于 CAN 总线的车内各个电子控制单元互相通信的网络。本文总结和分析常见的针对两种网络的攻击方式，并基于人工神经网络算法，如卷积神经网络与自编码网络技术，从数据采集，数据预处理，检测引擎三个方面展开设计基于网络的入侵检测模型。

2. 主机级入侵检测模型设计。车载终端运行着以 linux 为代表的嵌入式操作系统，在此基础上，进程会执行一系列系统调用，这种程序与操作系统之间的交互过程，可作为主机级别重要的入侵检测数据源，本文以进程的系统调用序列为数据源，对系统调用序列使用词嵌入技术，并设计基于卷积神经网络入侵检测引擎，形成主机级别的入侵检测模型。

3. 车联网入侵检测系统实现。在嵌入式车载终端上实现融合主机与网络的入侵检测系统，主要包括数据采集模块，预处理模块，检测引擎模块三部分，并将其部署在 linux 车载终端上。模型训练采用离线训练的方式，在服务器端完成训练并将模型传入车载 linux 终端上。最后，本文对提出的入侵检测系统进行测试和验证。

关键词：车联网，入侵检测，神经网络

ABSTRACT

In recent years the concepts of driverless cars, intelligent vehicles and intelligent transportation have sprung up. A vehicle has changed from a traditional relatively closed embedded system into the new one that interacts with the external environment more frequently and besides the communication between vehicles and people, it contains the exchange between vehicles and vehicles, even vehicles and infrastructures through wireless network or mobile network. While the Internet of vehicles brings people convenience, it also brings latent dangers about network security. Most of vehicle terminals lack effective cyber security defenses. Attacks against cars, such as stealing personal information and remote control, will pose a serious threat to personal life and property. An intrusion detection system is an effective way of cyber defense. To solve the security problem of this network, this paper proposes an intrusion detection system based on neural network integrating host-level and network-level intrusion detection.

The main contributions of this work are summarized as follows:

1. Design of network-level intrusion detection. The Internet of vehicles can be divided into the network based on TCP / IP protocol for external communication and the one based on CAN bus in which each electronic control unit of the vehicles communicates with each other. Based on convolutional neural network and auto-encoders, a complete network-level intrusion detection system is designed from three parts: data acquisition, data preprocessing and detection engine.

2. Design of host-level intrusion detection. On the vehicle terminal based on Linux system, the system call executed by the process reflects the interactive behavior between the program and the operating system. With the system call sequence of the process as the data source and the word embedding technology for preprocessing, the host-level intrusion detection engine based on convolutional neural network is completely designed.

3. Implementation of intrusion detection system for Internet of vehicles. The intrusion detection system includes data acquisition module, preprocessing module, training module, detection engine module. They are deployed on the Linux-based vehicle terminal. Besides, the model training part adopts the way of offline training and the server completes the training part and transmits the detection model to the vehicle terminal. To evaluate the system, we carry it out on public intrusion detection datasets, the experi-

mental results show that the system can effectively protect information security of vehicle terminals.

Keywords: internet of vehicles，intrusion detection，neural network

目 录

第一章 绪 论

1.1 研究工作的背景与意义

车联网随着无人驾驶技术与人工智能技术，逐渐走进了人们的生活，依靠着5G 通信技术飞速发展，汽车变得更智能，车联网应用价值也逐渐提高。吸引了大量的风险投资的同时，车联网相关内容也成为研究的热点。车联网无缝隙地连接着车与智能交通，车与车，车与云服务，构成人类未来生活的重要一环。

网络带来开放与便利的同时，也会带来信息安全问题，频繁的网络攻击在互联网爆发时期是常见的现象，比特币勒索事件是近年来比较严重的网络安全事例，造成数以万计的财产损失。在车联网的发展过程中也不例外，特斯拉是智能汽车研发的先驱，不久前特斯拉 ModelS 系统在黑客竞赛中被轻易攻破，远程控制智能汽车成为一件“轻松”的事。国内也有不少针对智能汽车举办的网络攻击比赛，这些因素潜在影响着人们对智能汽车的信任，严重阻碍了智能汽车的发展。工信部相关负责人曾表示车联网的安全性并没有随着车联网快速发展而发展起来，车联网安全处于初始阶段，主要原因是产业链相关企业存在投入不够，车联网信息安全意识不足，缺乏技术积累，其中智能汽车内 85% 的重要部件存在可以被攻破的漏洞，80% 以上的车联网平台存在数据明文不加密，没有有效的身份识别等严重的安全隐患，近 60% 的车联网企业，网络安全响应的自动化能力不强，缺乏安全监测能力。

对汽车的攻击方式总体可分为接触式攻击、远程攻击以及远程应用软件攻击三类，其中物理接触式攻击的主要方式是通过诊断调试接口来展开，调试诊断接口往往被设计在方向盘附近，是访问汽车内部网络 CAN 总线的接口，现代汽车内部有很多电子控制单元，连接这些单元的网络就是 CAN 总线，CAN 总线（如图1-1所示）分为高速总线和低速总线，高速总线连接着如刹车单元、加速单元、测速仪表、安全气囊控制等行驶功能性电子控制单元；而低速总线则连接像门窗、座椅、照明灯等对实时性不敏感的电子控制单元。CAN 总线协议没有考虑通信安全因素，有广播机制但没有身份验证机制，一旦被攻破，远程控制汽车就变得简单了。远程攻击包括借助无线网络和移动网络，包括 4G，5G，WIFI，蓝牙，车辆钥匙遥控等等展开，应用软件的攻击包括将各种有危害的 APP 软件安装到车载终端上展开，如可以窃取个人隐私的 APP。另外，通过无线网络或者应用软件等以车载终端为媒介进而入侵 CAN 总线也是一种方式。

网络入侵往往是隐蔽的行为，攻击者尽量伪装自己，在不被发现识破的情况

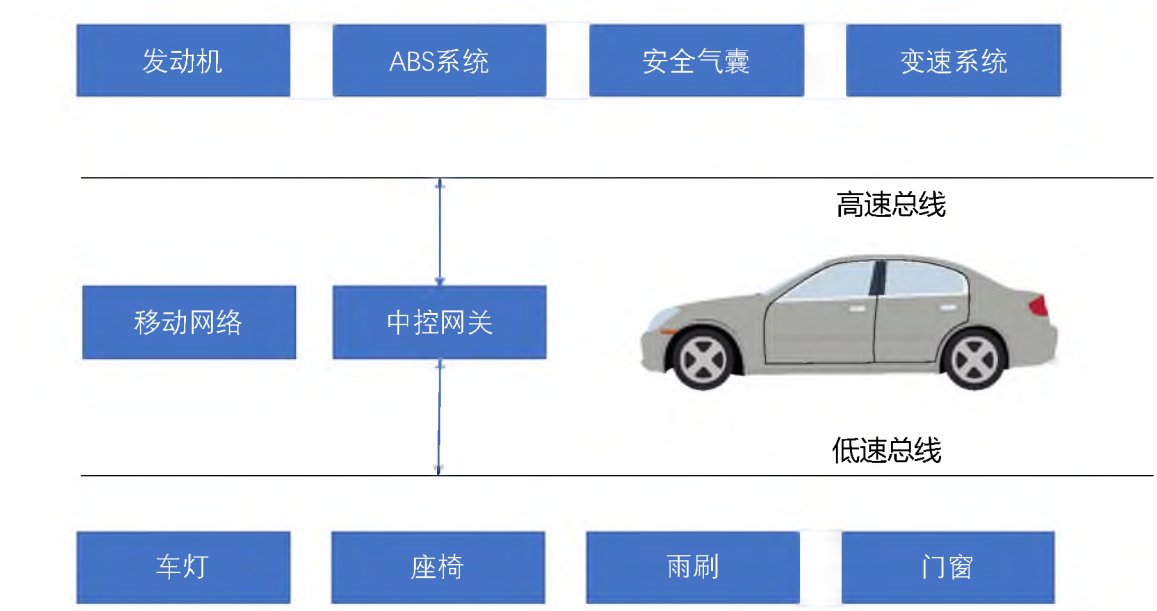


图 1-1 汽车网络结构

下，进行入侵破坏。安全从业者设计了很多入侵检测系统，以期望及时发现并阻止这些破坏行为。因此，实现车联网入侵检测系统是保障车联网安全的重要手段之一，尽早发现针对汽车入侵的行为，可以有效阻止攻击者更进一步行动以降低损失。本文聚焦车联网中的入侵检测系统，考虑多方面的信息源，包括：车联网基于以 TCP/IP 协议为代表的通信，这主要涉及汽车通过以 4G、5G 为代表的移动网络与外部环境包括云服务端在内的通信行为；车内电子控制单元通过以 CAN 总线为代表的通信，电子控制单元提高了汽车内部自动化程度，CAN 消息帧可以直接的控制和监测汽车的各个部件，因此，车联网重要的防线之一是对 CAN 总线的防护，由于 CAN 总线协议设计时，没有重视信息安全问题，其通信安全受到外部环境的严重威胁；现代化的车载终端上往往运行着操作系统，以进行合理的资源分配，嵌入式 linux 是常见的操作系统代表之一，针对此操作系统的入侵，往往会在系统的进程上留下“足迹”，进程信息也可以作为重要的入侵检测数据源之一。

本文使用人工神经网络算法构建入侵检测系统，神经网络有很强的非线性拟合能力，是在很多领域取得突破性进展的算法。神经网络算法在面对复杂的数据类型，比如图像，语音，视频，文本等方面，能够从复杂的数据中提取关键特征，有很强的处理能力。入侵检测数据源，包括各种网络报文，操作系统信息，应用神经网络算法可以有效地提取这些数据源关于入侵行为的特征，进而检测是否发生入侵行为，以达到有效保护车联网的目的。

1.2 国内外研究历史与现状

最初的入侵和入侵检测的概念，可以追溯到上世纪 80 年代，入侵行为是通过非法手段取得未授权的权限，对操作系统的信息进行非法访问，篡改，严重影响系统正常运行以及正常用户的使用。入侵检测是通过采样分析计算机中的数据包括网络数据和主机上数据，以发现其中的违规和异常行为。同时期，以审计为基础的通用的入侵检测模型框架也被提出。从此，入侵检测成为计算机网络安全中的重要一环。

入侵检测技术的研究主要集中在规则审计与机器学习。以规则审计为代表的入侵检测系统 snort，是一款开源的网络入侵检测系统。snort 使用模块化的设计便于使用者可以灵活使用各个模块包括协议分析，内容搜索和各种预处理器的功能，snort 维护庞大的规则库，便于检测如扫描蠕虫等各种类型的攻击，但面对新兴的攻击，如 0-day 漏洞，难以及时更新规则库。AIDE 是开源的主机级别入侵检测系统，通过维护文件数据库，验证文件完整性，以达到检测入侵行为的目的。

传统机器学习的技术在入侵检测上有着广泛的应用，研究者研究了一系列关于机器学习与数据挖掘在车联网入侵检测上的应用，数据来源包括 CAN 总线协议以及常见的 TCP/IP 协议数据，使用的方法包括从支持向量机，模糊逻辑与关联规则，信息熵算法，贝叶斯网络，聚类，决策树，马尔科夫链到一些集成学习方法[1–8]。文献 [9] 采用两层结构，使用朴素贝叶斯结合 KNN 邻近节点算法来设计模型。文献 [10] 提出支持向量机结合朴素贝叶斯算法设计入侵检测。文献 [11] 提出聚类算法结合遗传算法设计入侵检测。文献 [12] 提出结合随机森林与神经网络算法设计入侵检测。文献 [13] 在软件和硬件层为嵌入式设备，设计了以机器学习为基础的入侵检测算法以达到降低能耗的目的。

深度学习由于其良好的特征提取能力，在语音，图像，文本领域取得广泛的应用。深度学习可以从原始数据中直接学习到深层次的特征，相对于传统的机器学习来说，减轻了繁琐的特征工程相关内容，不需要依赖人工经验式地寻找特征。因其强大的学习能力，国内外已有学者尝试将深度学习技术甚至结合统计机器学习，共同应用于网络安全领域中。其中有将自编码网络结合集成学习的方法应用到异常检测中，依据自编码网络的均方误差作为分类的依据，并作出轻量级可在线训练的入侵检测框架[14]。也有将长短期记忆网络和卷积神经网络应用到入侵检测系统上[15]，还有将对抗生成网络算法应用到入侵检测系统上，训练一种可以判别用户正常行为的模型[16]。文献 [17] 结合监督学习方法和异常检测的方法设计对汽车车内总线的入侵检测模型，文献 [18] 使用二维卷积神经网络在 KDD 数据集上进行测试以证明卷积结构特征提取能力，文献 [19] 将神经网络与遗传算法结合

来构造入侵检测模型，文献 [20] 针对异常检测的问题提出改进的生成对抗网络算法，来检测异常数据，文献 [21] 提通的网络入侵检测框架，以解决数据稀缺和数据不平衡的挑战。该框架新颖性侧重于将深度对抗性学习与统计学习结合起来，并利用基于学习的数据增强。文献 [22–24] 设计使用 lstm 网络设计主机级别入侵检测模型，文献 [25] 提出 LSTM 和 SVDD 结合方法进行入侵检测检测。文献 [26] 使用集成学习结合神经网络的方式设计入侵检测，文献 [27] 介绍一系列使用深度学习的方法在入侵检测上的应用。文献 [28–31] 详细介绍深度学习在入侵检测中的应用。文献 [32] 将神经网络算法应用到车联网入侵检测中并提出车载终端入侵检测框架。目前虽然许多研究者使用机器学习的方法在数据集上做研究，但这些研究并没有实际建立一个入侵检测系统，而是仅研究这些算法在一些网络安全数据集上的性能。

1.3 本文的主要贡献与创新

论文在网络入侵行为严重影响车联网发展的现状上，选择设计与实现车载终端的入侵检测系统。本课题分析入侵检测技术的历史和发展，分析攻击原理以及检测攻击的数据源，围绕神经网络算法提出结合网络级别和主机级别的入侵检测模型，基于设计的模型在车载终端上实现入侵检测系统，并模拟真实攻击环境验证该系统，同时优化完善存在问题。

本文提出融合主机与网络的入侵检测系统，该系统基于神经网络算法实现的。对网络级别的入侵检来说，数据采集与特征提取模块负责获取原始的网络流量包并且通过协议解析工具来获取网络流量特征，将这些特征送入自编码网络获得编码特征，随后将这些原始特征和自编码网络获取的特征拼接并送入神经网络分类层进行分类。对主机级别的入侵检测而言，数据采集与特征提取模块负责收集与记录每个进程的系统调用号序列，并且使用自然语言处理中词嵌入的方法作为特征提取方法，来捕获这些系统调用号的语义层相似性，保持序列的关系，在神经网络决策引擎中加入一维卷积神经网络层，并加入不同尺寸的卷积核，提取到系统调用号序列的关键信息，从而更好的进行决策。

本文贡献如下：

1. 本文设计了一种融合了网络级入侵检测和主机级入侵检测并基于神经网络算法的车联网入侵检测系统，该系统可以在两个维度做到主动的防御入侵行为。同时，在车载终端嵌入式平台实现了这一套入侵检测系统，包括数据采集模块，数据预处理模块，检测模块并且模拟真实攻击环境进行了测试。

2. 在网络级别入侵检测，对 TCP/IP 协议和 CAN 总线协议数据依据特点采用

不同的设计方法，其中对 TCP/IP 协议数据，本文设计了自编码网络结合流量特征作为预处理部分，结合神经网络决策。对 CAN 总线协议，本文采用字符级别独热编码预处理以及卷积神经网络算法进行设计。

3. 在主机级别入侵检测，本文使用进程系统调用序列作为原始数据，并且采用词嵌入的方法作为预处理，捕获不同系统调用之间的语义联系，之后以卷积神经网络为核心设计分类器。

1.4 本论文的结构安排

本文工作的介绍顺序如下：

第一章为绪论。主要介绍本课题的研究背景与意义，以及国内外研究现状，明确主要工作与章节安排。

第二章为相关理论与工作基础。该章介绍了车联网的体系结构，TCP/IP 协议与针对车内 CAN 总线协议的内容进行说明。其次，介绍了入侵检测相关技术，包括入侵检测技术的概念，通用框架，分类，以及 KaliOS 系统上本文所用到的渗透测试工具。最后则介绍了本文使用到的神经网络算法相关概念，ARM 提供的神经网络加速库 ARM NN，自编码网络的概念，卷积神经网络的特点，基于神经网络的词嵌入的方法等等，为后续的入侵模型设计与实现提供了坚实的理论基础。

第三章为车载终端网络级别入侵检测系统设计。该章详细介绍了网络级别入侵检测的数据源，数据处理的方法与流程，以及模型的设计与实验评估部分，包括对 TCP/IP 协议以及 CAN 总线协议的入侵检测系统设计。

第四章为车载终端主机级别入侵检测系统设计。该章详细介绍了在主机级别的入侵检测模型的设计，首先介绍数据源，即主机上进程行为所留下的与操作系统交互的信息，其次介绍数据处理方法，最后描述卷积网络的模型设计与评估。

第五章为车载终端入侵检测系统实现。该章介绍根据上两章设计，在车载终端上实现融合主机与网络的入侵检测系统，包括介绍数据采集部分，数据预处理部分，模型检测部分以及包括在模拟攻击环境下进行的实验验证。

第六章为总结与展望。总结本文在车联网入侵检测方面的研究以及设计与实现。分析了论文中还存在不足的地方，研究和工程中需完善的地方，以及以后的研究方向等。

第二章 关键技术研究与相关理论

本章将会对车联网入侵检测系统的基础技术与理论进行介绍。首先介绍车联网相关基础知识，包括车联网架构和层次划分，以及 TCP/IP 协议和 CAN 总线协议，其次介绍入侵检测的相关概念，包括通用框架，分类以及渗透测试平台 Kali OS，其次介绍神经网络算法包括前馈神经网络结构的相关概念，自编码网络结构，以及卷积神经网络，最后介绍词嵌入的概念与方法。

2.1 车联网

2.1.1 车联网简介

车联网是物联网的延伸，包括车内网络，车际网和车载移动终端等组成，这些节点按照通信协议相互进行信息的交换，以实现智能汽车，智能交通，智慧城市等。车联网的架构按功能可划分为三部分（如图2-1），第一部分是车联网应用层，这部分主要包括智能交通和远程诊断监控车载娱乐，道路事故处理系统等应用层次的功能。第二部分是车联网的网络层这主要包括以 4G，5G 为代表的移动网络和 WIFI，蓝牙为代表的无线网络，也起到车内网络和车外网络相连的作用。第三部分则是车联网感知层，这部分主要是包括连接各个外部感知传感器和车内电子控制单元的网络，主要涉及到汽车内部的相关部件。

车联网也可以分为外部网络和内部网络。外部网络以无线通信技术为基础把车载终端与外界网络如基站，道路智能设施连接起来，实现车和车，车和基站，车辆和智能设施之间的信息交换，通信协议包括 TCP/IP，蓝牙，WIFI，NFC 以及 RFID 等等。而车内网络则是通过总线技术比如 CAN 总线协议建立标准化的车内通信局部网络，实现车内传感器和电子控制单元之间有机协调合作，监控车辆内部状态与诊断车内故障。

2.1.2 TCP/IP 协议概述

TCP/IP 是指一种在许多不同的网络间实现信息传输的协议簇，包括以传输层 TCP 协议和网络层 IP 协议为代表的许多协议。TCP/IP 协议制定了一个网络通信与数据交换的标准，以保证数据在传输中的及时性与完整性。TCP/IP 协议包括了应用层、传输层、网络层和数据链路层在内的四层结构体系（如表2-1）。

该协议栈负责在消息传输过程中，对每层报文附加相应层的报文头，该头部包含了相关协议的说明，比如源地址与目的地址等等。下面将介绍 TCP 与 IP 的协

图 2-1 车联网架构划分

议报文结构。

表 2-1 TCP/IP 协议 4 层模型

TCP/IP 模型	功能	协议
应用层	应用服务	HTTP，FTP，TELNET，SMTP 等
传输层	端到端接口	TCP，UDP 等
网络层	路由	IP，ICMP，BGP，IGMP 等
链路层	帧以及校验	PPP，ARP

2.1.2.1 TCP 报文

TCP 协议的报文结构如表2-2所示，其中源端口号和目的端口号作用是标识报文的返回地址和指明报文接收计算机上的应用程序地址。序号被用来说明端发送的数据大小，而确认序号则被用来通知数据成功接收。首部长度占 4 位，用来说明首部长度。随后就是 6 位标志位，按顺序依次为：URG 紧急、ACK 应答、PSH 推、RST 复位、SYN 同步、FIN 释放。位窗口大小，用来表示接受 TCP 数据段的大小，流量控制是通过这一字段完成的。校验和，起校验报文的作用。此外，TCP 协议通过三次握手建立连接，四次挥手结束连接。

表 2-2 TCP 报文首部结构

0 15	31
源端口号	目的端口号
序号	
确认序号	
首部长度 一些标志位	窗口大小
校验和	紧急指针
可选项	
数据	

2.1.2.2 IP 报文

IP 协议报文结构如2-3所示，版本目前有两个分别为：ipv4 为 4，ipv6 为 6。标识、标志、分偏移，起到帮助 IPv4 主机发送的报文的作用，TTL，该字段规定报文可经过的路由器上限，协议则代表上层协议类型，分为 TCP 和 UDP。

表 2-3 IP 报文首部结构

0 15			31	
版本	首部长度	服务类型	总长度	
标识			标志	片偏移
TTL		协议	首部校验和	
源地址				
目的地址				
可选项				
数据				

2.1.3 CAN 总线协议概述

2.1.3.1 车内 CAN 架构

CAN[16] 是控制器局域网络的简称，由德国公司研发，是国际上应用最广泛的现场总线之一，CAN 总线协议是串行协议，由于其数据结构简单，抗干扰能力强，广泛应用于车内电子控制单元之间的通信中，在国外成为汽车计算机控制系统和嵌入式工业控制局域网的标准总线。CAN 总线通过报文帧以广播的方式传输信息，在车内网络中每个电子控制单元都有固定的 ID，每个电子控制单元在收到报文帧时，可以比较 ID 来确定是否接收报文。智能汽车内部包含大量的电子控制单元，这些电子控制单元分布于汽车的各个位置，当他们需要协作时则通过 CAN 总线协议进行通信，车内 CAN 总线主要包含高速 CAN 总线和低速 CAN，它们分别适用于不同的电子控制单元进行收发数据（如图2-2），其中高速 CAN 总线又分为动力

CAN 总线和车盘 CAN 总线，动力总线主要负责动力系统各个电子控制单元之间的通信，而车盘总线则负责变速箱，刹车系统等之间的通信。OBD-II 是车载诊断系统，主要负责诊断和监控汽车内部运行状态。中控网关与外部网络通过车载终端设备相连，车载终端设备上往往运行嵌入式 linux 系统或 android 系统。

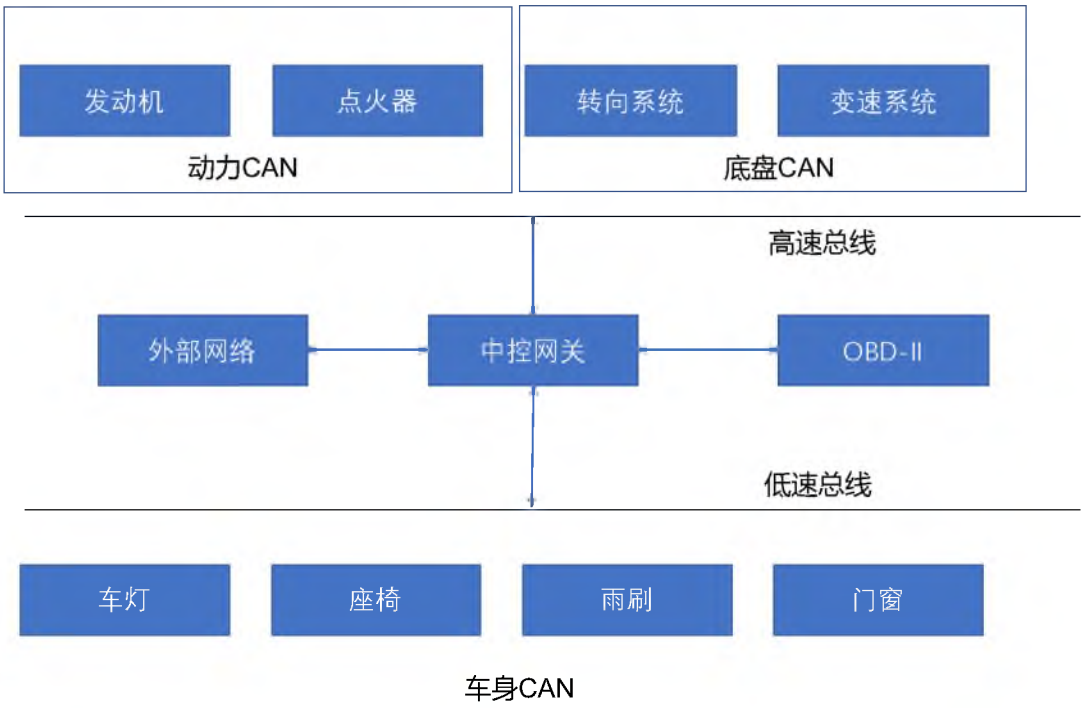


图 2-2 车内 CAN 总线结构

CAN 总线协议，传输距离远，抗干扰能力强，实时性强，具有很多特点：

（1）CAN 总线具有仲裁功能，在 CAN 总线上多个电子控制单元组成局部网络，每个电子控制单元具有唯一的 ID 标识，都可以发送消息，多个电子控制单元同时发送消息时，依据 ID 优先级来判断发送权利的归属，除此之外，每个电子控制单元也可以依据 ID 来确定是否要接收消息。

（2）CAN 总线具有错误处理功能，在 CAN 总线上的所有节点都可以检测错误，并且在检测到错误后，会通知其他节点该错误节点的 ID，如果发生故障的节点正在发送数据，则 CAN 总线会强行结束消息的发送。

（3）CAN 总线具有故障封闭功能，CAN 总线可以判断节点发生故障的类型由于外部暂时干扰，还是内部持续性错误，当总线上有持续性错误的节点时，CAN 总线会将该节点隔离出去，以免干扰其他节点的正常通信。

CAN 总线与其他网络协议相比，设计简单，并没有考虑复杂的安全问题，随着车联网发展，CAN 总线的安全功能显得不够用：

（1）广播机制：CAN 总线的任何节点都可以广播发送，并且接受所有 CAN

帧，这使得伪装的节点可以自由的窃取或仿冒其他电子控制单元的信息，为不法分子提供可乘之机。

（2）明文传输：CAN 总线出于对实时性的考虑，没有任何的加密通信机制，因此 CAN 帧是明文的，没有获取的难度，可以被轻易的获取与分析。

（3）缺少认证机制：CAN 总线上，任何节点无法区分仿冒节点，因为其没有身份认证机制，很容易被攻击者仿冒发送虚构的 CAN 帧，以达到特定的目的。

（4）仲裁机制：CAN 总线以 ID 作为优先级判断的唯一标准，如果攻击者发送大量仿冒的高优先级 CAN 帧，可能会耗尽 CAN 总线带宽，进而扰乱其他节点的功能。

（5）诊断接口：诊断接口最初的目的是检测汽车的故障，但由于没有合理的安全机制，容易被不法分子利用，一旦取得诊断接口权限，汽车内大量的电子控制单元均可以被控制和利用。

针对 CAN 总线的攻击有：重放攻击，DoS 攻击，模糊攻击，特殊报文注入，仿冒攻击等等。

2.1.3.2 CAN 总线协议

CAN 总线协议模型分为物理层、链路层、网络层和应用层。其中物理层由两种总线 CANH 和 CANL 组成，链路层有 CAN 帧构成，网络层以 CANTP 协议为代表，负责分组与重组多个 CAN 帧。应用层则由各个汽车厂家自行定义，主要功能有车内诊断，传感器控制等。一个标准的 CAN 帧格式如图2-3，仲裁段又称 ID 段代表帧的优先级同时又成为不同车内电子控制单元的标识。

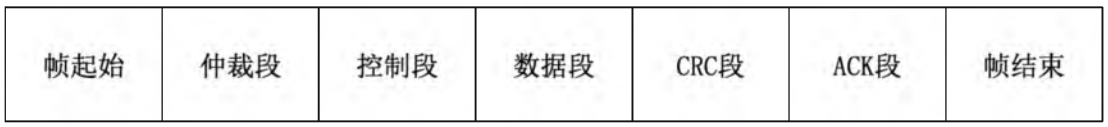


图 2-3 CAN 总线帧结构

CAN 帧分为两种基本格式，标准帧和扩展帧，标准帧的标识符 ID 为 11 位，而扩展帧的标识符为 29 位。CAN 帧按应用类别又可以分为数据帧，远程帧，错误帧，过载帧，间隔帧。数据帧用来节点之间收发数据；远程帧命令对方节点发送数据帧；错误帧是某节点发现帧错误通知其他节点的帧；过载帧是接收节点向发送节点通知接收能力的帧；间隔帧用于将数据帧、远程帧与前面帧隔离。

2.2 入侵检测技术

2.2.1 入侵检测通用框架

入侵检测的主要目的是识别对计算机设备，计算机网络，操作系统等攻击，干扰其正常运行或窃取隐私信息的行为。CIDF 是著名的通用入侵检测框架（如图2-4），其中外部数据，代表入侵检测的数据源，可以是主机进程数据也可以是网络数据包。CIDF 包括 4 部分，事件产生器，负责收集原始数据，并对数据做一些初步处理，将这些原始的数据分别送到事件数据库中存储与事件分析器中进行分析，事件分析器作为决策核心，可以依据规则进行判断，也可以借助异常检测的方法进行识别，如果有发现入侵行为，则送入事件数据库和响应单元进行进一步处理。响应单元则负责采取一些措施，来阻止入侵行为，如断开相应连接等等。事件数据库负责存放检测的数据。

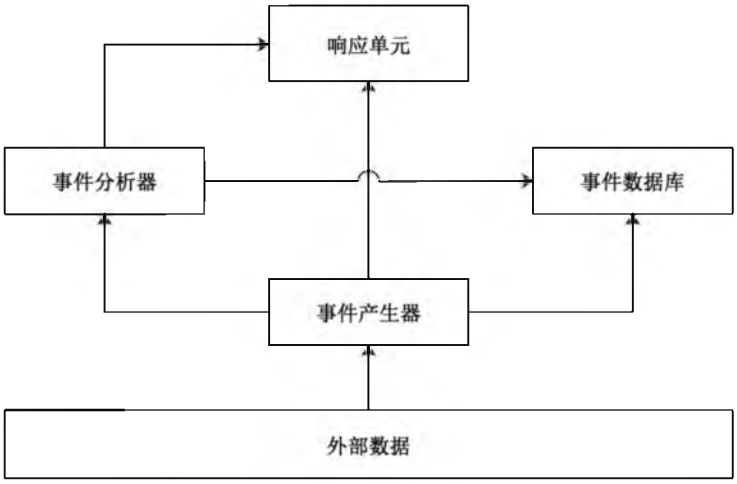


图 2-4 CIDF 框架

2.2.2 入侵检测分类

入侵检测系统可以根据分类标准的不同（如图2-5），进行多种划分：

依据入侵检测的数据源划分可以分成基于主机的入侵检测系统和基于网络的入侵检测系统。基于主机的入侵检测系统主要是对主机的数据进行监控和分析来发现入侵行为，数据的主要来源在与主机审计日志，进程信息等。基于主机的入侵检测系统优点在于可以直接根据进程行为来判断是否有威胁，检测的准确率高，不足的是对计算机的资源消耗较大。基于网络的入侵检测系统主要通过对网络的数据流进行分析。通过分析网络数据包的包头和内容提取出流量特征进而进行分析是否为入侵行为，这种系统的优点在于资源消耗较少，但缺点在于误报率比较

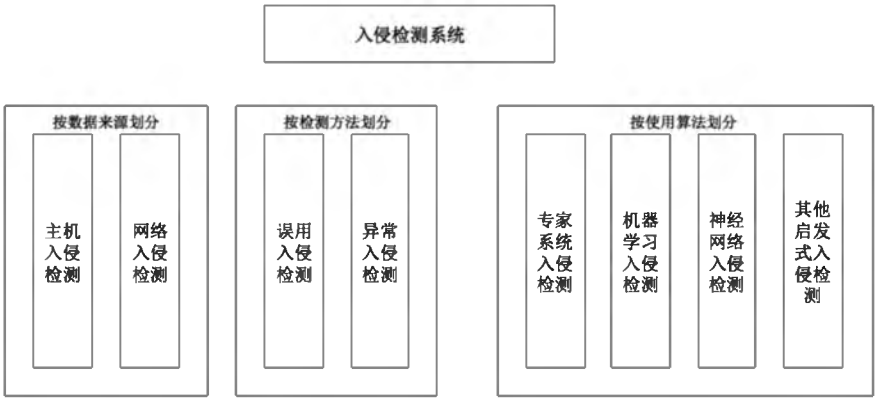


图 2-5 入侵检测系统划分

高。

按照不同的检测方法分为误用检测和异常检测。其中误用检测主要是将已知的攻击方式和手段以专家系统的方式维护成规则库，用规则库来判断行为正常与否。因此，对于已知的攻击方式有较高的准确率，但对于未知的攻击则不能进行有效的检测。而异常检测则是将用户的正常行为提取特征存储到特征数据库中，之后将用户的行为与数据库之中正常行为相比较，如果发现两者偏差较大，则判断存在攻击行为。异常检测的优点在于能够及时的发现未知攻击。缺点也显而易见就是由于正常用户行为的改变而发生警报，误报率偏高，需要进行周期性的训练。

根据入侵检测算法，又可分为基于专家系统的入侵检测，基于统计机器学习的入侵检测，基于深度学习的入侵检测，基于其他启发式算法的入侵检测。基于专家系统的入侵检测，是指通过先验知识维护攻击规则库，将输入的数据与规则库中的规则相比较。snort 系统是这一类入侵检测系统的代表，这类入侵检测系统的特点是对于已知的攻击准确率很高，但对于那些没有记录在规则库中的攻击则无能为力，并且还需要维护和更新规则库。基于统计机器学习的入侵检测，将统计机器学习的一些现有算法应用到入侵检测，涉及到决策树，聚类，信息熵，隐式马尔可夫链，混合高斯模型，这类算法有较强的可解释性，但是依赖人工特征提取。基于深度学习的入侵检测，将在其他领域表现良好的神经网络应用入侵检测中，神经网络的拟合能力强且有很好的特征提取能力，可解释性稍差。基于其他启发式算法的入侵检测，主要包括遗传算法，粒子群算法，免疫算法等一些启发式算法。

2.2.3 Kali 渗透测试平台

Kali OS 是专为安全从业人员提供的 linux 发行版，Kali 预装了大量的渗透测试工具，包括从基础的扫描嗅探到社会工程类工具。这些工具包括：嗅探欺骗工具，无线与密码攻击工具，漏洞探测利用工具，信息搜集工具，数字取证，逆向工程等等。下面将介绍本文使用的工具。

Hydra：Hydra 是密码破解工具，内置多种协议包括邮件，数据库，SMB，VNC，FTP，SSH 等等，其方法是使用暴力破解的方法或采用字典攻击的方法针对各种形式的登录进行账号密码的破解。

Metasploit：Metasploit 是一款著名且开源的黑客漏洞检测与利用工具。它包括各种各样的针对当代系统中的漏洞与相应的利用工具，使用者可以轻易的获取与开发针对计算机系统或相关服务的漏洞利用工具。其中的 Meterpreter 是其框架的一个扩展工具，起初作为 shellcode 后续其功能越来越完善。在攻击成功后，Meterpreter 在目标机的内存中展开，返回一个连接目标主机的网络通道，可以悄悄连接目标机的 Meterpreter shell。Meterpreter shell 提供了很多渗透功能，比如对操作系统上用户的控制，对文件系统的修改，甚至捕获用户操作如对鼠标与键盘的监控。另外，Meterpreter 为了躲避一些以日志文件为检测源的入侵检测系统，以内存模式进行暗中操作，因此很难在日志中留下痕迹，也很难被入侵检测系统发现。

nmap：nmap 是开源的网络探测和安全审核的工具。具备快速扫描大型网络的功能，并且提供多种协议以及多种方式的报文发送方法，包括使用原始 IP 报文，特殊报文等等，nmap 可以扫描，嗅探主机所开放的端口，并提供其服务版本与相应的漏洞，此外，还可以测试防火墙情况等。

2.3 人工神经网络

人工神经网络算法自上世纪 40-50 年代提出，起源于模拟人脑神经元形成。它的目的是形成有记忆与联想学习功能的智能系统。1958 年，出现了以感知机为代表的神经网络模型，具有初步学习机制，引发神经网络热潮。但在 1969 年，这种感知机被证明不能解决线性不可分的问题，与此同时基于逻辑符号处理方法以其优越性得到研究者青睐，把神经网络技术挤出研究热点，神经网络的研究进入寒冬期。1982 年 Hopfield 网络模型理论可以解决非线性问题，从而再次激发神经网络的研究，1986 年反向传播算法提出，又解决了神经网络中权值调整的问题，自此神经网络进入蓬勃发展阶段，后来的研究者们又提出递归神经网络，卷积神经网络，对抗生成网络，图神经网络等等，这些神经网络在图像，视频，语音，文本

等领域取得显著的成就。

本文重点介绍后续模型应用的三种神经网络前馈神经网络，自编码神经网络，以及卷积神经网络。

2.3.1 前馈神经网络

2.3.1.1 神经网络结构

人工神经网络的基本单元是神经元，神经元的结构（如图2-6）包括与输入 $x_1x_2x_3$ 对应的权重 $w_1w_2w_3$ 和偏置 b，和激活函数 f，神经元的输出 y，其数学表达式 (2-1)：

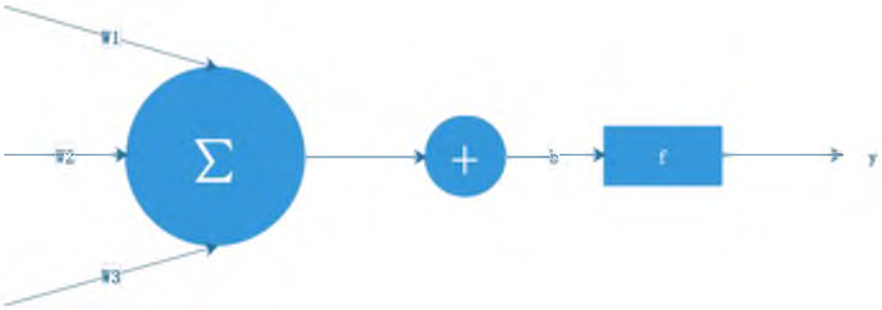


图 2-6 神经元结构

$$y = f(w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + b) \tag{2-1}$$

相当于进行一次仿射变换并外加通过激活函数，其中激活函数向神经网络中引入非线性因素，借助激活函数，神经网络可以拟合各种曲线。引入激活函数首先需要考虑其非线性的特点，使用线性激活函数，则无论使用多少层网络等价于使用一层神经元的网络结构，其证明可以通过对上式 (2-1) 展开得到。其次考虑输出范围，当激活函数的输出受限制时，加快后续提到的训练算法的收敛速度。

常见的激活函数有 Sigmoid 激活函数，Softmax 函数，Relu 激活函数等等。Sigmoid 函数，输出在 (0，1) 之间，它可以将一个实数映射到 (0，1) 的范围内。在二分类问题中受到广泛应用。该函数对神经元的输出加以限制，使之处于 (0，1) 范围之内，公式：

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{2-2}$$

ReLu 函数的公式如2-3，当输入 $x < 0$ 时，输出为 0，当 $x > 0$ 时，输出为 x。

该激活函数由于使用了简单的阈值化处理，求导方便，不会有梯度问题，模型收敛很快，训练速度也很快，总之，计算效率很高。

$$ReLu(x) = \max(0x) \tag{2-3}$$

softmax 用于多分类过程中，它将多个神经元的输出，映射到（0,1）区间内，并且让他们的总和为 1，因此可以看成概率来理解，从而来进行多分类。

$$softmax = \frac{e^i}{\sum_j e^j} \tag{2-4}$$

一个人工神经网络等于神经元加上其中的连接部分，即神经网络是由不同层级的神经元连接而成的 (如图2-7)，前馈神经网络结构可以分为三部分输入层，隐藏层和输出层。输入层负责接收外部输入，隐藏层负责处理进一步处理数据，输出层则负责输出信号，其中隐藏层可以由多层神经元构成。

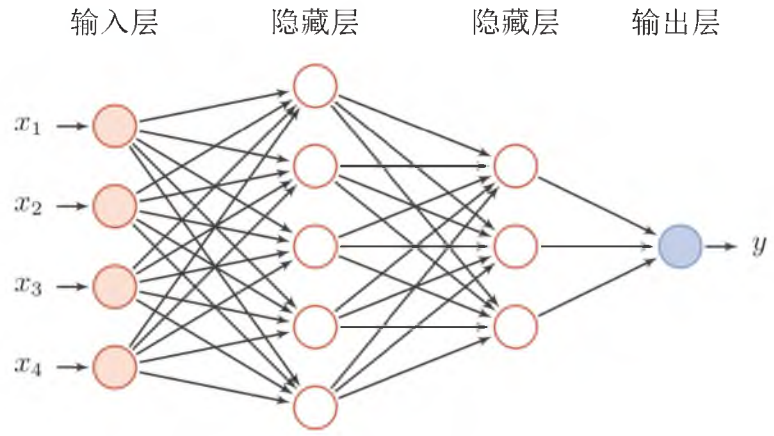


图 2-7 前馈神经网络

2.3.1.2 损失函数

损失函数在机器学习中有重要作用，损失函数用来衡量真实值与机器学习模型输出之间的差距，并且可以通过将损失函数最小化的方式，来优化机器学习模型相关的参数。因此损失函数对于模型训练，参数优化均有重要意义。

损失函数在常见的机器学习问题分类与回归中，选取不同，选择合适的损失函数可以加快模型训练与收敛速度，并且训练出参数更合理的模型。常见的分类问题的损失函数为交叉熵：

$$loss = -\sum_{i=1}^{N} p(x_i) \log q(x_i) \tag{2-5}$$

其中 x_i 为实际输入，p 和 q 分别为真实数据的分布和模型拟合出的数据分布。

其中分类类问题中，会遇到样本类别不平衡等问题，这种问题在网络安全领域常常发生，尤其是新型攻击的样本数量偏少，而 focal 函数通过修改交叉熵，通过增加类别权重以及样本难度权重调因子，来提高训练模型准确度。

$$Focal(p_t) = -\alpha_t (1 - p_t)^{\gamma} log(p_t) \tag{2-6}$$

常见的回归问题的损失函数如均方误差：

$$loss = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \tag{2-7}$$

m 表示 m 个样本的，$loss$ 为 m 个样本的 $loss$ 均值，y_i 和 $\hat{y}_i$ 分别对应真实的样本输出和拟合过后的样本输出。

2.3.1.3 正向传播算法

正向传播是指数据从神经网络的输入层，层层递进，传输到输出层并计算得到损失函数的过程，用于模型计算算损失函数以及推理，一个简单的三层结构的前馈神经网络，正向传播的公式如下：

$$H = f(W_1 \cdot X + b_1) \tag{2-8}$$

$$Y = f(W_2 \cdot H + b_2) \tag{2-9}$$

其中 W_1 代表中间层神经元权重矩阵，b_1 代表中间层神经元的偏置，H 代表中间层的输出，$W_2 b_2$ 则代表输出层的权重与偏置，Y 则代表模型输出。结合2-5或2-4便可计算损失函数。

2.3.1.4 反向传播算法

神经网络训练过程，是为了减少损失函数，也就是减少输出层的误差，往往使用以梯度下降法为代表的优化算法，梯度下降法借助损失函数逐层求导并对连接权重进行负梯度方向进行更新，在损失函数是凸函数的情况下，依据凸分析相关理论，局部极小值点即全局最小值点。由输出层逐步向前以链式求导法则逐层更新参数的过程称为反向传播，公式：

$$W_2 = W_2 - \eta \frac{\partial loss}{W_2} \tag{2-10}$$

$$b_2 = b_2 - \eta \frac{\partial loss}{b_2} \tag{2-11}$$

$$W_1 = W_1 - \eta \frac{\partial loss}{H} \frac{\partial H}{W_1} \tag{2-12}$$

$$b_1 = b_1 - \eta \frac{\partial loss}{H} \frac{\partial H}{b_1} \tag{2-13}$$

η 为模型参数的学习率。

2.3.2 自编码神经网络

自编码神经网络是一种无监督的神经网络技术，简单的自编码神经网络是由三层结构组成的，基本和前馈神经网络一样（图2-8）。不同的是自编码网络目的在于重构输入，捕获输入数据潜在的特征。自编码网络学习输入 X 到自身的映射 X，分为编码器和解码器两部分，采用的损失函数往往为均方误差如式2-7。

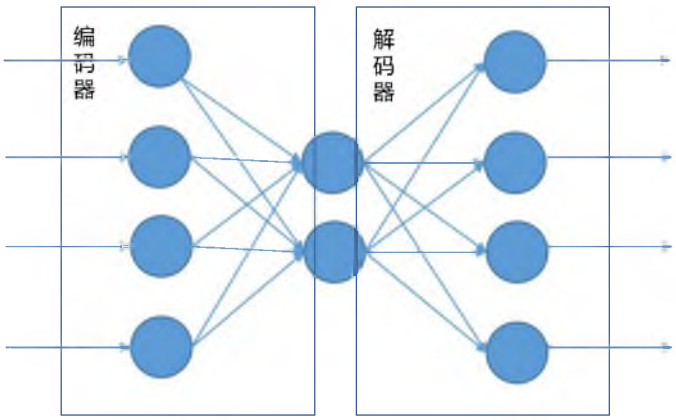


图 2-8 自编码神经网络

编码器，将输入 X 映射到隐藏层 H，从图中可以看到，隐藏层维度上 (神经元的个数) 在减少，编码器的表达式如下：

$$H = f(WX + b) \tag{2-14}$$

解码器，将隐藏层 H 在映射为 $\hat{X}$，以期望 $\hat{X}$ 与 X 之间的误差尽可能小，的表达式如下：

$$\hat{X} = f(WH + b) \tag{2-15}$$

在优化阶段，同样采取反向传播算法，以式2-7为损失函数进行以梯度下降法为代表的训练。

经过一段时间的训练后，损失函数误差变得很小，也说明中间层输出 H 已经包含绝大部分输入的特征，具备重构输入的能力。在之后的应用中，只需要取自编码的编码器部分，即可完成对原始输入的特征提取。

2.3.3 卷积神经网络

2.3.3.1 卷积结构介绍

卷积神经网络是含有卷积结构的神经网络，其中卷积结构是仿照生物视觉构建的，因此最初的卷积结构被应用在图像识别领域。卷积结构相较于全连接网络，具有局部连接与权值共享的特点，这是通过卷积核实现的。卷积神经网络，有强大的学习能力，且运算与存储资源消耗较少，已逐渐成为深度学习中被广泛采纳的一种结构。一个典型的卷积结构包括卷积层和池化层，卷积运算提取特征，池化层起到采样的作用，最后这些特征会被送入全连接神经网络结构处理 (如图2-9)。

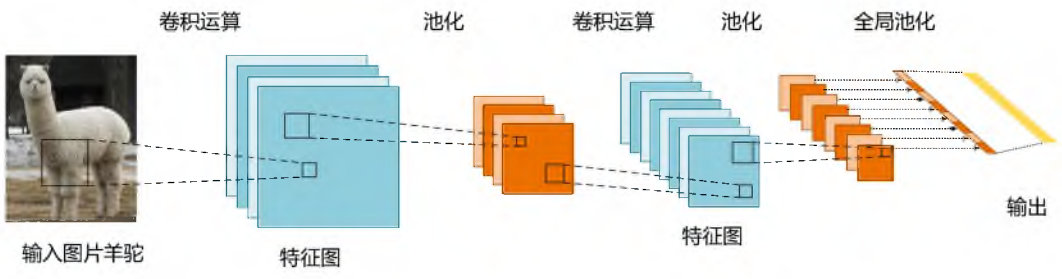


图 2-9 一个二维卷积运算的例子

卷积层数学表达式为：

$$y = f(W * X + b) \tag{2-16}$$

其中 X 为原始图片，$*$ 为卷积运算，b 为偏置，f 为激活函数，y 为输出特征图。

卷积运算的过程如图2-10，输入矩阵中每个原始的位置和对应的卷积核位置相乘，之后求和，作为特征图的输出，随后移动输入矩阵，重复运算。池化过程，则是将卷积核相乘替换为取其中的最大值或平均值起到下采样的作用。

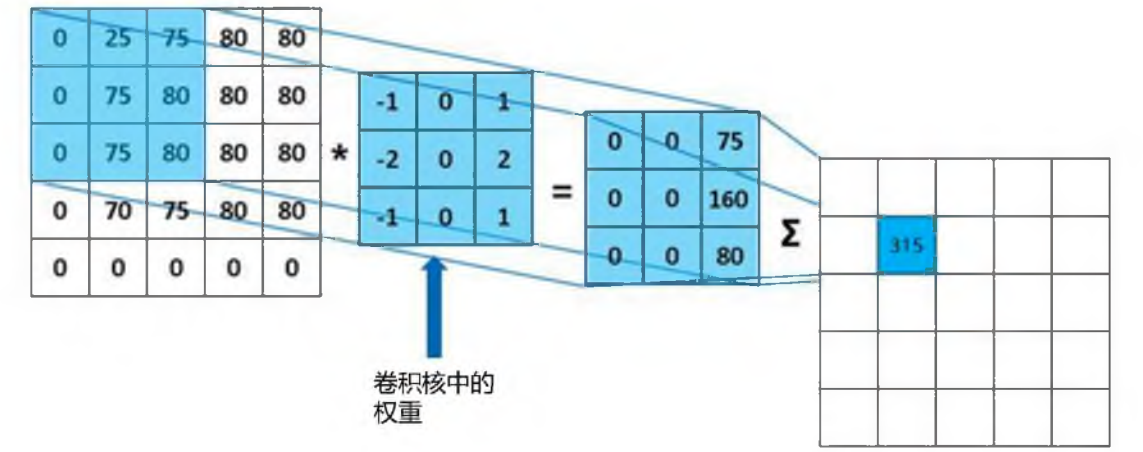


图 2-10 一个二维卷积处理图片的例子

卷积层的输出特征图的尺寸由卷积运算的超参数决定，其数学公式：

$$o = \left\lfloor \frac{n + 2p - f}{s} \right\rfloor + 1 \tag{2-17}$$

设输入图片的尺寸为 $n \times n$，卷积核的尺寸为 $f \times f$，填充为 p，步长为 s，输出特征图的尺寸为 o，其中 $\lfloor \rfloor$ 为下取整符号。

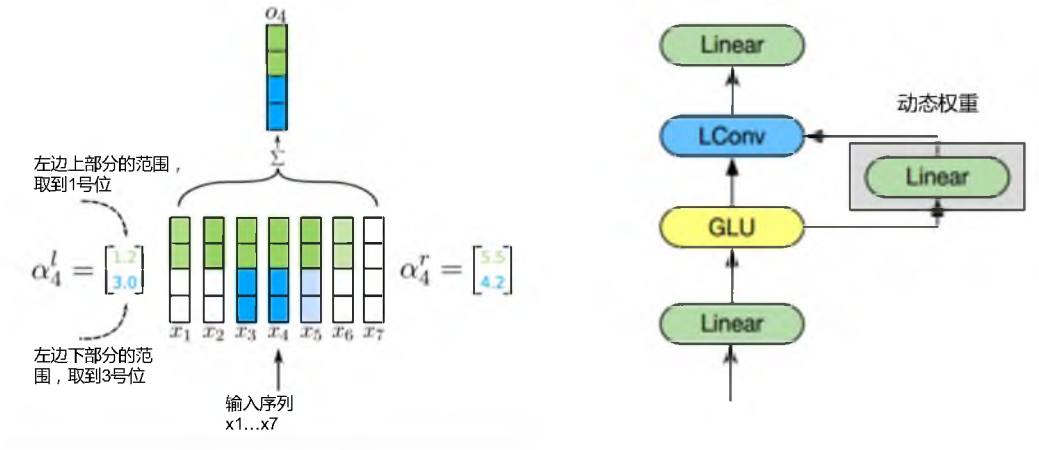
2.3.3.2 一维卷积网络

一维卷积结构，适用于序列数据的处理，常见于对时序信号的处理，如语音或文本序列的处理中，在单一维度上进行卷积运算，相较于二维卷积结构，结构简单，运算更快。其运算公式与式2-16类似，不同点是只在单一维度上进行移动，卷积核大小固定。

目前，在文本领域提出新的模型，多是结合自注意力模型，这些模型相较于卷积来讲，具有全局范围 (相比于固定的卷积核大小而言) 和动态权重 (相较于卷积核权重固定) 的优点，但是计算开销大。

之后，针对上述注意力结构的优点，有许多对一维卷积结构的改进，比如时间自适应的一维卷积结构 (如图2-11(a))，将固定的卷积核大小映射为自适应的与时间有关的范围，这样做就改变了卷积固定步长的移动，改善了局部感知范围。另一个改进，轻量级动态权重一维卷积结构 (如图2-11(b))，通过在卷积结构外构造

一层线性网络，使得训练完成后的模型可以产生动态权重，这一点也类似于自注意力。总之，一维卷积结构，由于其训练与运算速度快，在时间序列数据处理中占有重要地位。



(a) (b)

图 2-11 卷积网络结构改进 (a) 时间自适应卷积网络；(b) 动态权重卷积网络

2.3.4 嵌入式神经网络加速库

ARM NN 是 arm 公司推出的开源神经网络运算加速库，用于基于 arm 处理器的高性能平台上进行人工神经网络相关的训练与推理，以及相关程序的构建。包括神经网络算法在内的机器学习需要大量算力与存储带宽，这是嵌入式设备面临的最大挑战。ARM NN 适配了目前主流的机器学习框架，例如 tensorflow，caffe 等等，并且做了硬件抽象层以适配 ARM 各种底层处理器包括 CPU，GPU，APU 等等，神经网络算法需要两个阶段，训练与推理，目前包括本论文使用的方法在内，模型训练一般在服务器中进行，但模型推理则转移到边缘网络，这是 ARM NN 主要解决的一个问题。

ARM NN 隐藏了底层硬件的复杂性，提供了各种神经网络框架接口，自动将各种神经网络框架，转换为优化过后的图表，并使用 Compute Library 库面向目标硬件进行优化，也提供了低级别神经网络函数对 Cortex-M 平台系列进行优化。

2.4 词嵌入方法

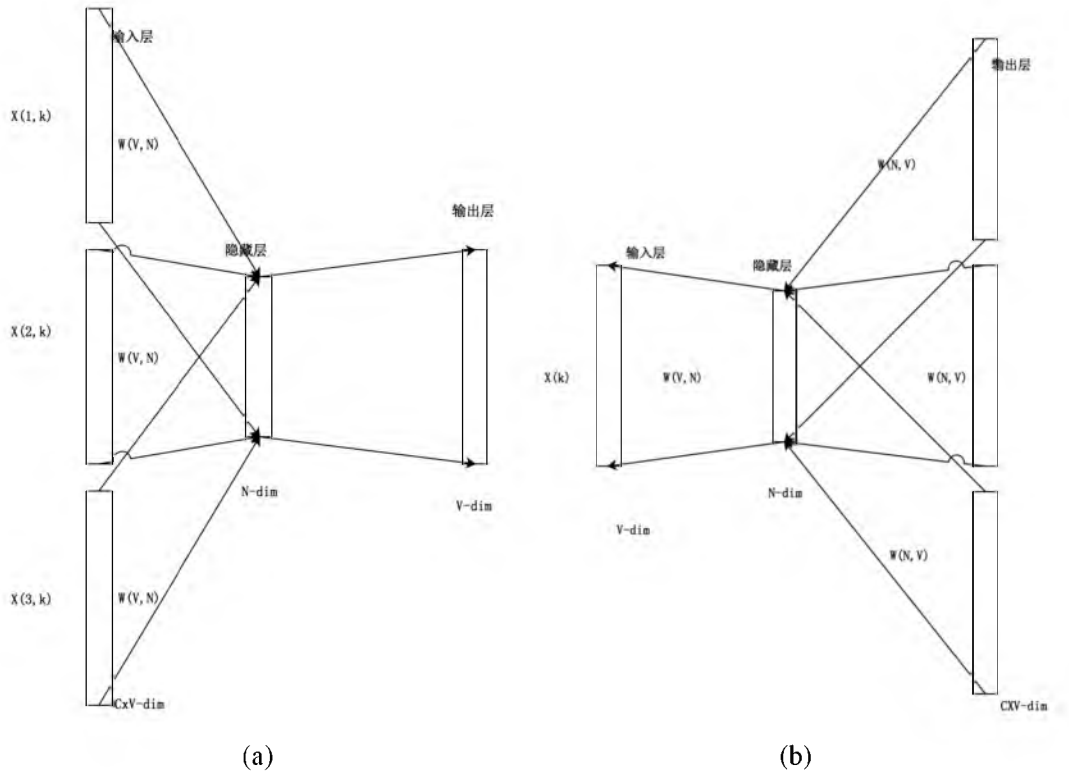
词嵌入方法来自于自然语言处理，一般自然语言的词包含几千甚至上万，如果使用独热编码，则维度相当大，无法直接计算，不利于训练，词嵌入可将词的特征映射到较低的维度，使用模型参数更少，训练更快。词嵌入用特征描述取代

符号描述，是一种提取特征的手段，便于进行语言的对比泛化知识迁移，提取到词语词之间语义特征，语义上有关联的词语在特征空间的距离上较近。一个极具影响力的经典的词嵌入方法是 2013 年提出的 Word2Vec。Word2Vec 建立了一种神经网络结构，该结构的输入与输出均为经过编码后的词。该算法给出了两种训练模型，CBOW 和 Skip-gram。随后随着深度学习技术的发展，建立在庞大语料库之上的深度学习的词嵌入模型，ELMo，Bert 等深刻的改变了自然语言处理。

Word2Vec 的网络结构很简单，包括一个输入层、一个隐藏层、一个输出层。经过训练之后，使用输入层和隐藏层之间的连接权重矩阵 $W_{V\times N}$ 表示单词之间的关系。实现单词维度从 V 降到 N 的词嵌入映射。

CBOW 用上下文词来预测中间词 (如图2-12(a))，将一个词所在的上下文中的词作为输入，输出层为语料库中各个词是输入中缺失词的概率分布，即词表中每个词在该上下文中的概率。隐藏层则是由 N 个神经元构成的。模型的设计思路为在一个上下文中，预期预测特定词的意思。在大量的语料训练中，得到从输入层到隐含层的权重模型，作为词嵌入的映射关系。

而 Skip-Gram 的做法与之相反 (如图2-12(b))，将上述网络反转，将该词本身作为输入，同时将该词可能的上下文作为输出。



(a) (b)

图 2-12 Word2Vec 网络结构 (a)CBOW；(b)Skip-Gram

Word2Vec 在许多下游任务中表现优异，但仍有一些缺陷，如 Word2Vec 在训练时只考虑局部的句子信息，词序关系也没有考虑。其后另一种相似的词嵌入方法 GloVe 算法，修正了前面算法的一些缺点，比如包括了全局统计信息和局部语境信息。

2.5 本章小结

本章首先介绍了车联网的概念其中包括车联网的架构，车联网所用到的 TCP/IP 协议与 CAN 总线协议，其次，介绍了入侵检测的通用框架，包括事件产生，事件分析，事件响应与事件存储。然后，介绍了人工神经网络算法，包括神经网络算法的训练，自编码神经网络与卷积神经网络以及 ARM 神经网络加速库，最后介绍了词嵌入方法。本章为下面两章的模型设计提供了坚实的理论基础。

第三章 车载终端网络级别入侵检测模型设计

本章将围绕车外网与车内网即以 TCP/IP 协议为基础的网络和以 CAN 总线为基础的网络，使用神经网络算法分别设计适合各自系统的入侵检测模型。从数据源的说明，到数据预处理的方法，以及模型设计与试验评估这三方面进行本章的展开。

3.1 TCP/IP 网络入侵检测模型设计

3.1.1 数据来源及攻击方式

针对网络级别入侵检测本章采用 NSL-KDD 数据集进行研究，并以此为基础进行模型设计以及模型验证。NSL-KDD 数据集是 KDDCUP99 数据集的改进版本，主要去除冗余数据和不平衡数据等缺陷。KDDCUP99 是为了数据挖掘比赛设计的，在九星期内模拟局部网络环境和一些攻击行为，最后将这些流量收集并进行标记，做成训练数据集和测试数据集，其中测试数据集中为了考验算法的泛化能力，将一些在训练数据集中未出现过的攻击方式的数据加入其中，这样提高了该数据集的识别难度。

在 NSL-KDD 中包含 5 中类型的数据，正常类型和 4 种攻击类型（如表3-1）：嗅探扫描（PROBE），拒绝服务（DOS），远程非法访问（R2L），本地非法访问（U2R）。

PROBE，探测扫描是指对服务器的端口进行逐一的扫描，以探测系统信息，以及端口开放程度，著名的扫描工具有 nmap。

DOS，是指入侵者使用正常的手段，频繁请求服务器资源，资源耗尽后服务器无法提供正常的服务的攻击方式。

R2L，采取远程检测的方式尝试获取本地登录权限，比如远程猜测 SSH 密码。

U2R，获取本地登录权限后，使用普通权限登录之后进一步获取系统权限，比如利用缓冲区溢出的漏洞进行提权。

在 NSL-KDD 中的每条记录由 41 个特征组成（如表3-3），NSL-KDD 记录的数据分布情况见表3-2。这些特征可分为三类：TCP 连接基本特征，该特征集合主要涉及 TCP 相关内容，包括传输的协议类型，连接时间等等；TCP 连接的内容特征，该特征是针对 TCP 的内容部分进行抽取，以发现可疑行为，如是否为 root 用户，是否尝试登陆，是否对文件系统进行操作等等，以达到检测 U2R 和 R2L 之类的攻击；基于时间的网络流量统计特征，网络攻击往往与时间有关联，有的攻击持续时

表 3-1 攻击方式

类别	具体攻击形式
拒绝服务（DOS）	Back，Land，Neptune，Pod，Smurf，Teardrop，Mailbomb，Processtable，Udpstorm，Apache2，Worm
嗅探（Probe）	Satan，IPsweep，Nmap，Portsweep，Mscan，Saint
远程非法访问（R2L）	Guess_password，Ftp_write，Imap，Phf，Multihop，Warezmaster，Xlock，Xsnoop，Snmpguess，Snmpgetattack，Httptunnel，Sendmail
本地非法提权（U2R）	Buffer_overflow，Loadmodule，Rootkit，Perl，Sqlattack，Xterm，Ps

间长，有的攻击持续时间短，统计一定时间内的流量特征，可以有效地检测入侵行为，这些信息包括在局部网络内对相同主机的流量信息和相同服务的流量信息。

表 3-2 NSLKDD 数据集

数据集	总数	正常	PROBE	DOS	U2R	R2L
训练集	125973	67343	11656	45927	52	995
测试集	22544	9711	2421	7458	67	2887

3.1.2 数据处理

每条记录中要处理的特征数据类型可分为离散型和连续型两种（如图3-1），其中离散型数据往往是协议，服务，或标志位的类型，比如 protocol_type，service，flag，除此之外还有一些布尔类型的数据可以不做任何处理，其他的数据都是连续型数据。

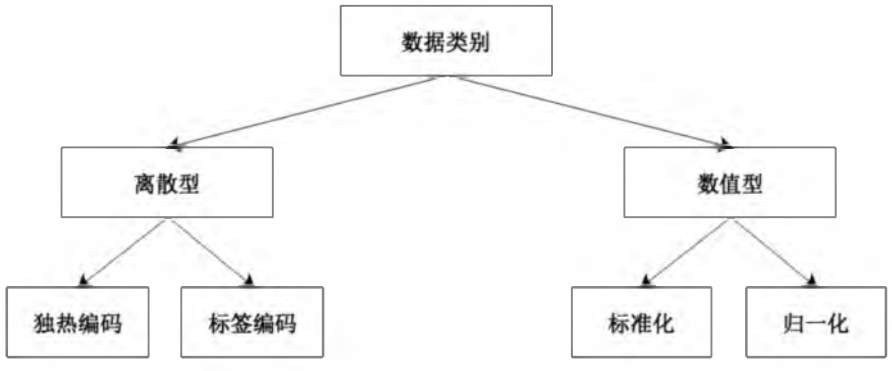


图 3-1 数据类型以及相应处理方式的分类

表 3-3 NSLKDD 数据特征

特征类型	特征列表
基础特征	duration，protocol_type，service，flag，src_bytes，dst_bytes，land，wrong_fragment，urgent
基于时间的流量特征	count，srv_count，serror_rate，srv_serror_rate，rerror_rate，srv_rerror_rate，same_srv_rate，diff_srv_rate，srv_diff_host_rate
基于主机的流量特征	dst_host_count，dst_host_serror_rate，dst_host_rerror_rate，dst_host_same_srv_rate，dst_host_diff_srv_rate，dst_host_srv_count，dst_host_srv_serror_rate，dst_host_srv_rerror_rate，dst_host_srv_diff_host_rate，dst_host_same_src_port_rate
内容特征	hot，num_failed_logins，logged_in，num_compromised，root_shell，su_attempted，num_root，num_file_creations，num_shells，num_access_files，num_outbound_cmds，is_host_login，is_guest_login

3.1.2.1 离散型数据

对于离散型数据，其中的取值之间没有大小关系的，即没有偏序关系的，采用独热编码的方式处理，取值之间含有大小含义的，则做标签编码。独热编码，其编码方法为用 N 位取值为 0 或 1 状态位对 N 个状态进行编码，每个状态对应一个状态位。这样的编码方式解决了分类器处理非数值数据的问题，并且使得特征之间距离计算更加合理。标签编码，即按顺序进行自然数编码，进而可以将数值映射到 0 至 1 范围内。本文做预处理时仅使用到独热编码。

以 NSL-KDD 数据集中 Protocol_type 特征作为说明，该特征的取值只有三种情况，icmp、tcp、udp，是一种离散型的特征，通过独热编码将其数字化，可以送入后续的模型进行处理。首先对特征维度进行扩充，对此特征扩充为 3 位，tcp(1,0,0)，udp(0,1,0)，icmp(0,0,1)。独热编码，将离散特征取值从单一维度的点扩展到了欧式空间，这样做在计算距离时更加合理。但同样这么做会增加维度，如果数据维度过高同样不利于处理，这种情况下可使用一些降维的手段如 PCA，T-SNE 等等。

3.1.2.2 连续型数据

对于连续型数据，采用归一化或标准化处理方式，归一化是将数据取值范围从原始的 (min, max) 映射为 $(0, 1)$，这么做的优势在于避免不同特征之间取值范围差异过大，从而导致对模型的影响力不同。标准化，通过变换特征的分布情况，使得每个特征的均值变为 0，标准差变为 1。这样做，进行优化算法训练时，处理起

来更加快速。

3.1.2.3 特征处理流程

特征处理的流程（如图3-2）检查每一个特征是连续的还是离散的，如果是连续的，则做归一化操作，如果是离散的，则判断该特征是否有偏序关系，如果有则做标签编码，合适的情况下可以进一步映射到 [01] 区间，如果无，则做独热编码处理，随后进行下一个特征处理，如果特征全部处理完成，便可以作为后续构建神经网络模型的输入。

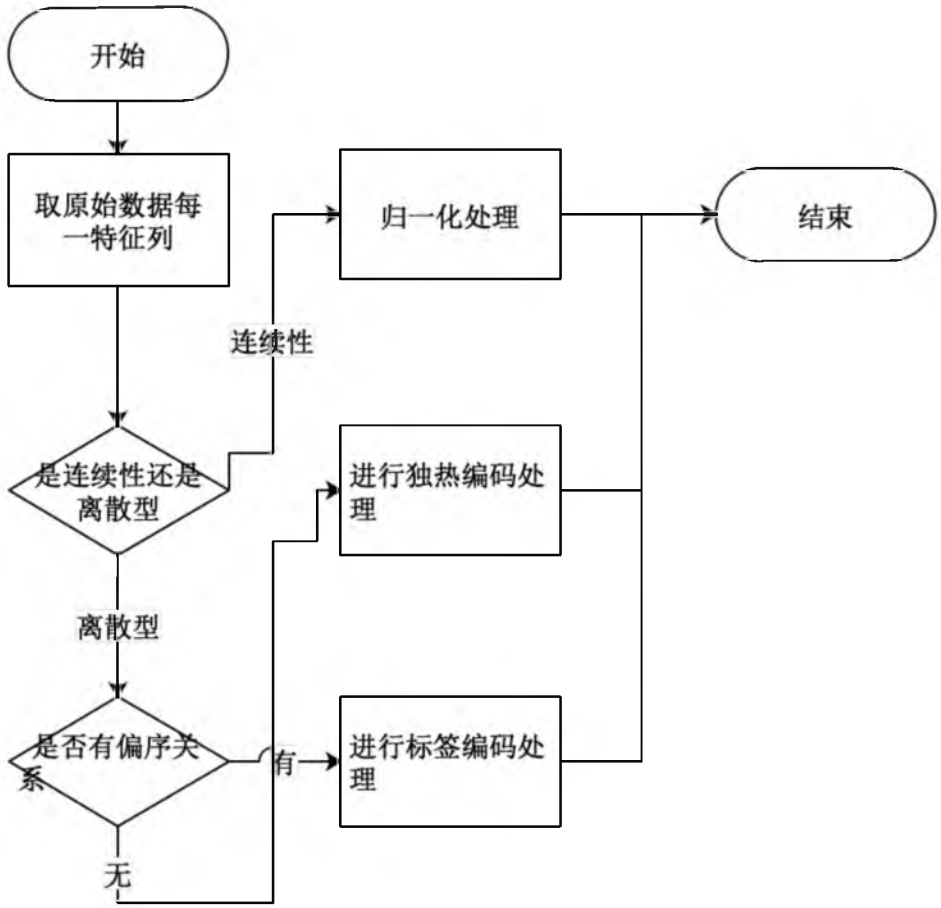


图 3-2 特征处理流程

3.1.3 模型设计与评估

3.1.3.1 模型设计

本文采用自编码网络结合前馈神经网络的模型设计入侵检测算法，结构见图3-3。经过前小节数据预处理后，得到可以进行训练的原始数据，首先，构建自编码网络，自编码网络的输入为原始数据，输出为重构的输入数据，损失函数为

均方误差函数，构建完成后用所有的数据进行训练，目的在于获取重构输入的映射，进而可以获取新的潜在特征。训练完成后，取自编码网络编码器部分，即图中输入部分和红框框中的部分，将这部分结构，拼接到后面将要构筑的前馈神经网络中去。

在构建后续前馈神经网络分类器，首先获取自编码器的编码器部分，将这部分结构拼接到神经网络的输入中，即构建原始数据拼接自编码特征数据，这种结构充分利用了原始特征和自编码提取的特征，做到深层次的特征提取。在模型的最后一层使用 softmax 分类层，进行分类，类别有正常，DOS，PROBE，R2L，U2R，模型构建完成后，损失函数可选取为交叉熵函数或 focal 函数，focal 函数是交叉熵函数的改进版，可以缓解数据样本不平衡的问题。

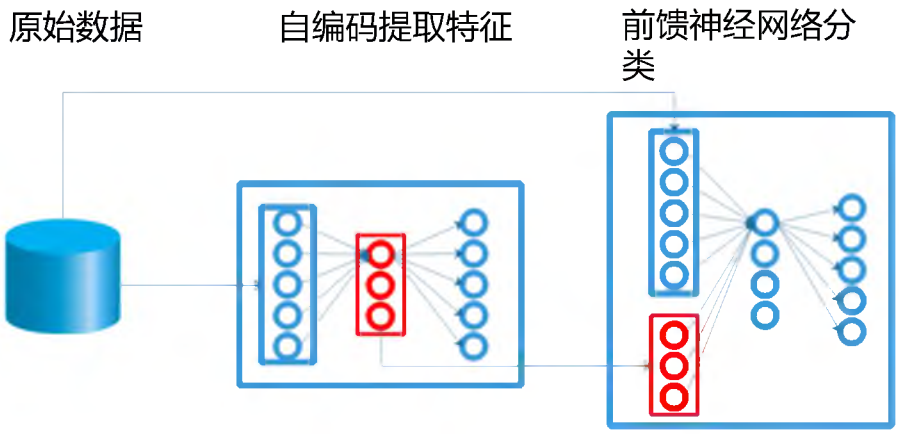


图 3-3 自编码结合前馈神经网络模型

为了适应车载终端嵌入式设备，本文力求采用简单神经网络结构，其中自编码网络采用的是三层全连接网络结构，激活函数选择为 ReLu 函数，并以均方误差为目标函数进行训练，同样的，前馈神经网络采用三层全连接网络结构，输入层和中间层的激活函数为 ReLu，最后一层以 softmax 函数为激活函数作为分类器，以交叉熵或 focal 函数作为目标函数进行训练。对于其他超参数的选择，因具体训练数据而定。训练流程如图3-4，首先在训练自编码神经网络，在满足误差后，取得其中自编码器，固定权重后，构建分类神经网络，进行训练，直到满足误差要求。训练算法如算法3-1，其中 TrainData 代表训练集数据，AE，NN 代表自编码网络和前馈神经网络的权重等参数。

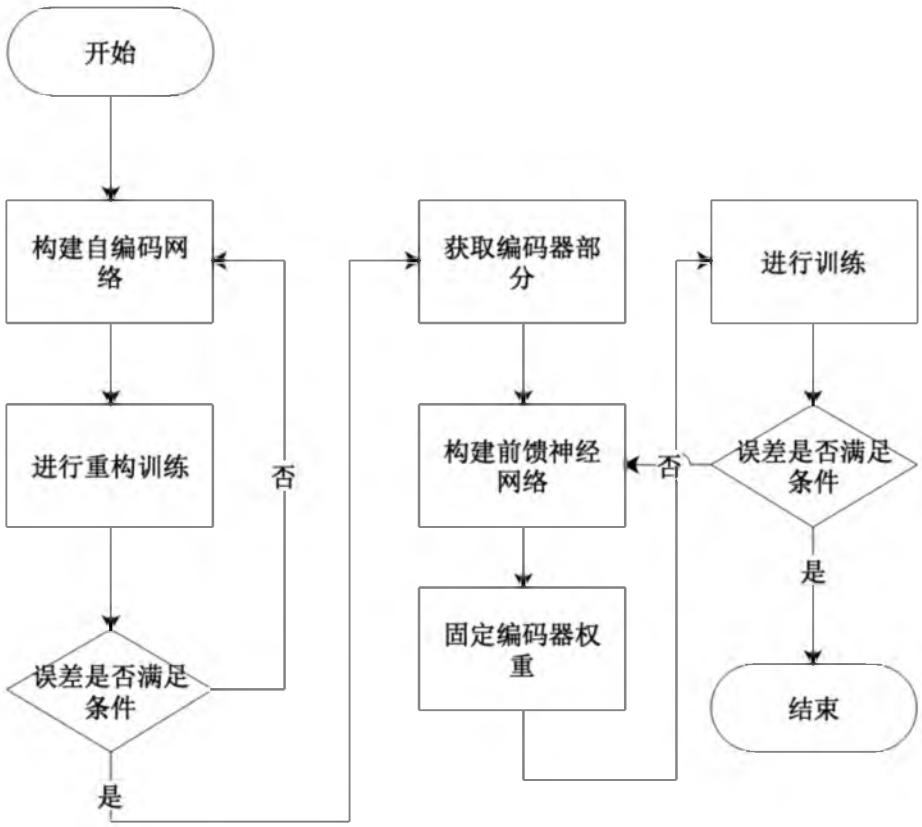


图 3-4 训练流程

3.1.3.2 评价指标

为了评估模型性能，本文在 NSL-KDD 数据集进行测试本模型，对一个二分类问题，将实例分成正类或者负类。实际中分类时，会出现四种情况：

(1) 实例为正，预测正，即为真阳 (TP)。

(2) 实例是正，预测负，即为假阴 (FN)。

(3) 实例为负，预测正，即为假阳 (FP)。

(4) 实例是负，预测负，即为真阴 (TN)。

Accuracy，Precision，Recall，F1-Score 是以上述 4 种情况为基础的二级评价指标。

Accuracy 可以衡量整体的分类正确程度，但在类别不均衡的情况下，不能有效的反应分类算法的能力，因此准确率仅仅能作为评价指标之一，还需要别的指标辅助。Accuracy 表达式为：

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{3-1}$$

算法 3-1 训练算法

```
Data: TrainData
Result: AE，NN
initialization;
while epoch<epochs do
    for x_i y_i in TrainData do
        loss ← loss_func(x_i y_i AE);
        grads ← grad_func(lossAE);
        AE ← AE − η · grads;
        total_loss ← total_loss + loss;
    end
    if total_loss/epoch < threshold then
        break;
    end
end
Encode ← part(AE);
total_loss ← 0;
while epoch < epochs do
    for x_i y_i in TrainData do
        loss ← loss_func(x_i y_i NNEncode);
        grads ← grad_func(lossNN);
        NN ← NN − η · grads;
        total_loss ← total_loss + loss;
    end
    if total_loss/epoch < threshold then
        break;
    end
end
```

Precision 代表预测为正例的那些数据里预测正确的数据个数所占比例。

$$\text{precision} = \frac{TP}{TP + FP} \tag{3-2}$$

Recall 表示真实为正例的那些数据里预测正确的数据个数所占比例。

$$\text{recall} = \frac{TP}{TP + FN} \tag{3-3}$$

Precision 和 Recall 是此消彼长的，在一些场景下要兼顾两者，F1-score 可以达到这一目的。F1-score 是两者的调和平均数，调和平均数的特点是只有两者分数都高，结果才会高，单方低会将结果拉向低值。

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3-4}$$

3.1.3.3 模型评估

本文采用深度学习框架 tensorflow 依据上述模型设计过程构建模型，自编码器和分类神经网络均为三层结构，对于中间层超参数的选择，其中自编码结构神经元分为 32 和 64 两种，前馈神经网络的神经元从 32 一直到 512。NSL-KDD 的实验数据如表3-4所示，另外使用支持向量机（SVM），K 邻近（KNN）方法作为对照试验，由 F1 分数可看出神经网络算法分类性能较好。

表 3-4 实验结果

Architecture	AE	Accuracy	Precision	Recall	F1-score
NN-32	32	0.9946	0.7708	0.7067	0.7228
NN-64		0.9954	0.9347	0.7595	0.7951
NN-128		0.9957	0.9583	0.7915	0.8261
NN-256		0.9954	0.9502	0.8254	0.8461
NN-512		0.9949	0.9012	0.8113	0.8288
NN-32	64	0.9933	0.7583	0.6966	0.7125
NN-64		0.9954	0.8989	0.7615	0.7881
NN-128		0.9959	0.9657	0.7995	0.8372
NN-256		0.9959	0.9550	0.8488	0.8775
NN-512		0.9951	0.9951	0.8217	0.8374
SVM	/	0.9682	0.4131	0.3442	0.3547
KNN		0.9977	0.8873	0.8459	0.8464

3.2 CAN 总线入侵检测模型设计

3.2.1 数据来源及攻击方式

OTIDS 数据集 [33]，是韩国一所大学发布的针对汽车 CAN 总线入侵检测数据集，该数据集是在攻击汽车的同时，通过接入汽车 OBD 接口，并且以记录日志的方式收集完成的。该数据集包括 4 种数据类型，包括拒绝服务，模糊攻击，重放攻击，正常数据。

拒绝服务攻击，攻击者控制节点（某个 ECU 或者仿冒 ECU），在较短的时间内高频注入高优先级的 CAN 帧，以达到扰乱其他节点通信，耗尽带宽的作用。（图3-5）。

模糊攻击，与模糊测试类似，都是通过向 CAN 总线系统提供非预期的 CAN

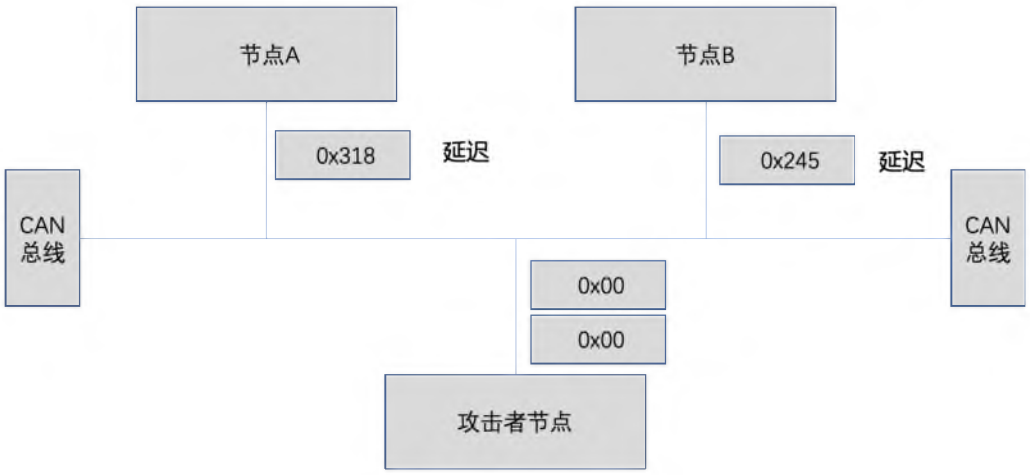


图 3-5 CAN 拒绝服务攻击

帧，来干扰系统正常进行，攻击者会按一定的规律生成有特定规则的 CAN 帧序列，以一定频率发送到 CAN 总线中，并观测总线及汽车变化，以推测出各个 ECU 的相关信息。(图3-6)。

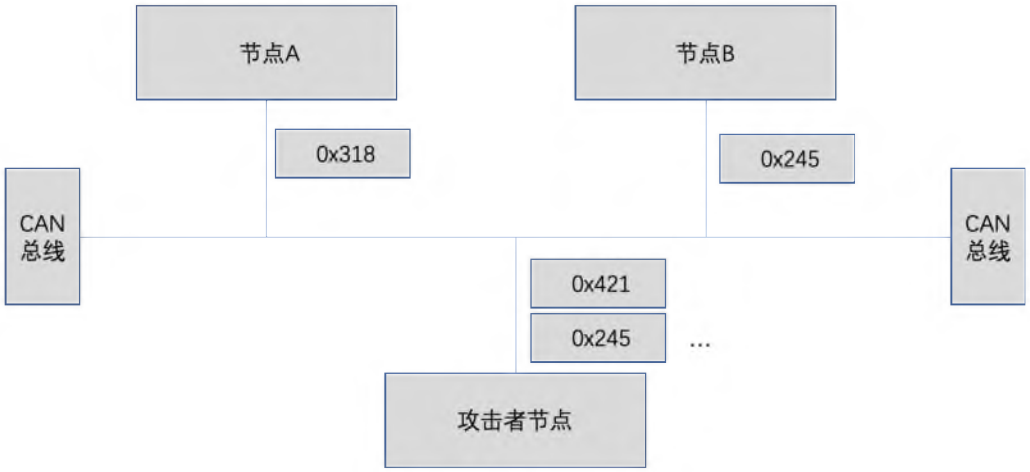


图 3-6 Fuzzy 攻击

重放攻击，攻击者控制 ECU 之后，重复发送某个特定含义的 CAN 帧，以达到扰乱 CAN 总线和探测 CAN 总线信息的目的。重放攻击是有效地针对汽车总线协议破解的攻击手段，攻击者在不了解汽车 CAN 总线协议内容时，可以通过重放攻击观察汽车功能表现，以达到分析破解汽车 CAN 协议的目的。(图3-7)。

3.2.2 数据处理

每一条数据是以 CAN 帧的方式记录的，消息帧主要包括 ID 段，控制段和数据段 (3-8)。在处理 CAN 数据过程中，仅使用 ID 段作为入侵检测的原始数据，这样做可以有效减少编码时间，适合实时检测。

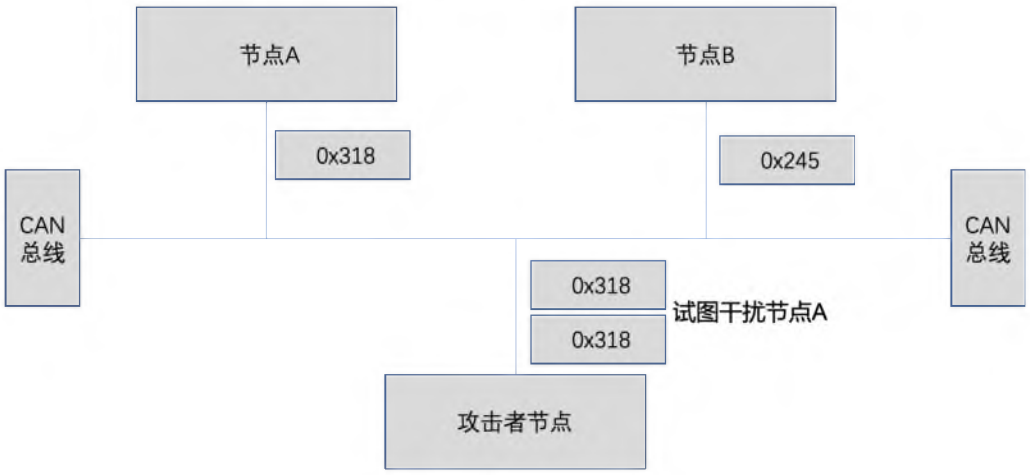


图 3-7 重放攻击

仲裁段	控制段	数据段
11位（0x000-0x7FF）	4位	8位

CAN ID

图 3-8 CAN 帧

对 ID 段进行预处理，其过程如图3-9所示，收集的 ID 段的数据是 3 个十六进制的，车辆中电子控制单元远没有达到可以使用每一个 ID 的数量，并且各个厂家对电子控制单元的 ID 分配不一致，将每个 ID 作为整体进行独热编码不合适。于是，本文采取类似字符级别 CNN 的预处理方法，即对 ID 中的每一位进行独热编码，从原始的 ID 数据序列取得编码过后的 ID 序列，每一个 ID，都用三个独热编码向量来表示，而非采用每一个 ID 用一个独热编码表示，这样做极大的减少编码后的空间。

3.2.3 模型设计与评估

本文采用一维卷积神经网络结构，来设计 CAN 总线入侵检测模型。一维卷积可以应用于时间序列分析，同样也可以用于分析具有固定长度周期的信号数据。

3.2.3.1 模型设计

本文利用深度学习框架 Tensorflow-Keras，搭建一维卷积结构和一维下采样（池化）结构，构造一种简单的网络模型，结构如图3-10所示。其中一维卷积层，卷积核的个数为 64，卷积核的大为 4；一维池化层，下采样的方式为取特征最大值，下采样的尺寸为 4；最后用 softmax 分类函数作为分类器，以确定 CAN ID 数

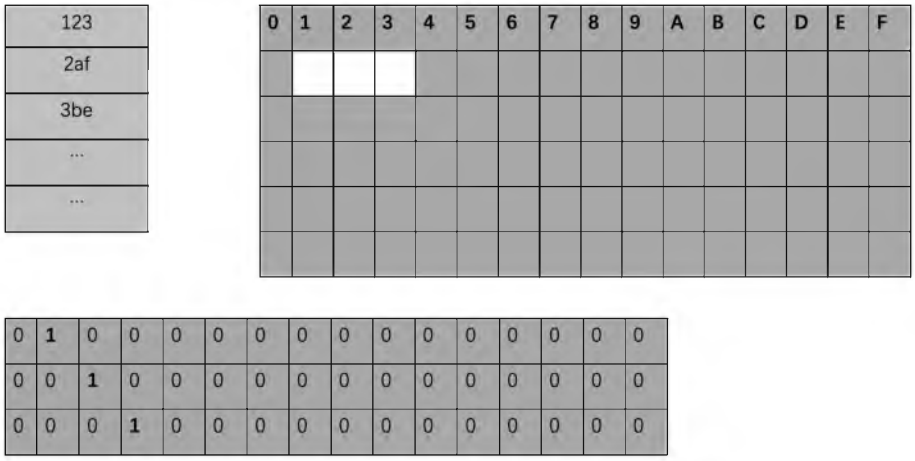


图 3-9 字符级独热编码

据序列是否含有入侵行为。

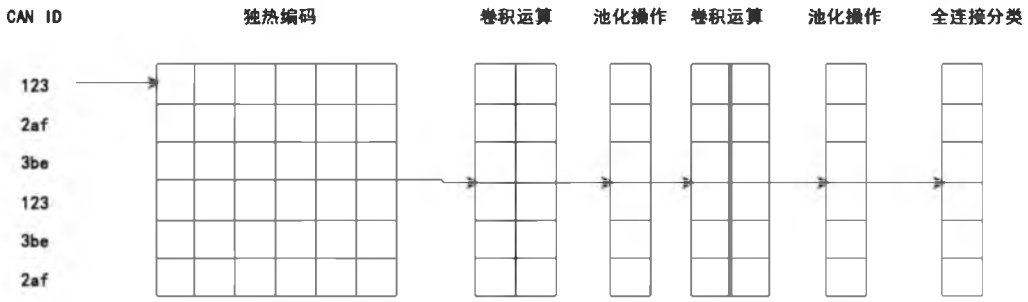


图 3-10 CAN 入侵检测模结构

该模型设计主要包括三个步骤：第一步数据预处理，主要是将 CAN 总线消息帧 ID 序列，进行字符级别的独热编码。第二步是特征提取，将编码完成的数据送入卷积神经网络，进行特征提取，在此采用的是一维卷积神经网络，相较于二维卷积结构，在更少的参数下提取序列时间维度的特征。第三步是采用全连接结构和 softmax 激活函数构造分类器。整个网络由三个隐藏层组成。每个隐藏层包含一个卷积层和一个池层。每个隐藏层的卷积核数是不同的，将原始特征映射到高维空间，从而提高了学习特征的能力。

其整体的训练算法如3-2，其中 CAN_ID_SEQ 代 CAN ID 序列，CNN 代表卷积网络结构：

算法 3-2 CAN 训练算法

Data: CAN_ID_SEQ

Result: CNN

initialization;
for *&x_i in CAN_ID_SEQ* **do**
    $x_i \leftarrow one\_hot(x_i)$;
    $x_i \leftarrow con\_cat(x_i)$;
end
while *epoch<epochs* **do**
    for $x_i y_i$ *in CAN_ID_SEQ* **do**
        $loss \leftarrow loss\_func(x_i y_i CNN)$;
        $grads \leftarrow grad\_func(lossCNN)$;
        $CNN \leftarrow CNN - \eta \cdot grads$;
        $total\_loss \leftarrow total\_loss + loss$;
    end
    if $total\_loss/epoch < threshold$ **then**
        break;
    end
end

3.2.3.2 模型评估

在测试数据上，我们用训练好的卷积网络模型进行预测。表3-5显示了对每个攻击数据检测的精度，召回率和 F1 分数，由于该数据集的样本平衡性很好，因此实验结果的数据上表现很好。

表 3-5 CAN 入侵检测数据集实验结果

算法	Accuracy	Precision	Recall	F1-score
CNN	0.99	0.99	0.99	0.99
SVM	0.99	0.99	0.99	0.99
KNN	0.99	0.99	0.99	0.99

3.3 本章小结

本章是本文的核心之一，具体介绍了基于 TCP/IP 协议和 CAN 总线协议的入侵检测的数据特点，处理方式以及模型设计与评估，其中基于 TCP/IP 协议的模型设计部分采用自编码网络结合前馈神经网络的方式，基于 CAN 总线的模型设计部分采用一维卷积结构。另外针对 CAN 总线数据特点，使用字符级别的独热编码处理，降低了数据处理的维度。最后，在入侵检测数据集对两种模型进行评估。

第四章 车载终端主机级别入侵检测模型设计

本章将会介绍主机级入侵检测设计，包括数据来源及攻击方式，数据预处理方法以及模型设计。其中根据数据的特点，本文将会借用词嵌入作为数据预处理方法，以更好的提取数据特征，随后将使用卷积神经网络的结构设计模型。

4.1 数据来源及攻击方式

本文采用系统调用序列作为主机级入侵检测的数据源，系统调用序列是基于异常检测方法的重要数据源。操作系统上的程序在运行时会表现出稳定性，即大部分程序在正常执行任务的过程中，所产生的与操作系统的交互具有一定的规律性，但一旦遭遇攻击行为，这种规律性会被打破，即会在程序执行的过程中留下痕迹。因此程序的系统调用序列可以反映程序运行的情况，也可以被入侵检测系统所监视以查看是否有入侵行为，入侵行为体现在异常的系统调用序列。

linux 系统的安全机制保证对系统资源的访问必须通过系统调用来实现，图4-1显示应用程序通过系统调用来与操作系统内核交互的过程，而一些非法攻击者访问系统资源来实现其各种入侵目的比如盗取个人信息。远程控制之类的。在此举缓冲区溢出的案例来说明，当缓冲区溢出时，程序可能跳转到指定的恶意代码处运行，恶意代码完成对目标攻击以后，往往不会恢复到正常程序运行的起始位置，而是直接退出，由于进程和恶意代码对系统资源的访问都要借助于系统调用实现，因此会在系统调用上留下足迹。

ADFA 数据集是基于系统调用序列的数据集，是澳大利亚某院校发布的一套主机级入侵检测系统的数据集合。发布之后便被应用于各种入侵检测的实验中，该数据集包括 Linux（ADFA-LD）和 Windows(ADFA-WD) 两个平台的数据，而本文针对车载 linux 终端设计入侵检测系统，故只采用 ADFA-LD 数据集。该数据集包含了针对 linux 最常见的服务，比如提供文件共享，数据库服务，远程连接，web 服务的攻击。在 Linux 的每一个系统调用均对应着一个唯一的 ID，开启审计功能在系统运行期间，所有被调用到的系统调用均被记录在审计日志中。该数据集是基于这一原理收集的。该数据集包含六种攻击方式（如表4-1）：FTP 暴力破解，SSH 暴力破解，添加超级用户，webshell 攻击，Meterpreter 和 java 版 Meterpreter 攻击。这些攻击方式足以代表中级水平的黑客行为，利用了包括从嗅探，密码猜测到漏洞利用与社会工程。

图 4-1 程序与内核交互

表 4-1 攻击方式

攻击方式	样本数量
密码破解-FTP	162
密码破解-SSH	148
添加超级用户	91
java-Meterpreter	125
Meterpreter	75
C100 Webshell	118

密码爆破：该攻击类型代表在开放服务器上远程进行密码破解时，服务器中的进行产型异常的系统调用序列的过程。由于 FTP 和 SSH 服务为典型的服务经常会受到暴力破解攻击，因此该攻击具有代表性。

Meterpreter 和 Java-Meterpreter：Meterpreter 是增强的功能命令 shell，主机被攻破后，是漏洞利用的一种常见方法，攻击者可以通过 meterpreter 在主机上执行各种命令。

添加新的超级用户：在 Linux 系统添加超级用户是常见的提权行为。若黑客成功添加超级用户则会产生严重的后果，本数据来源于客户端投毒进而添加超级用户的行为。

webshell：利用 web 服务器的某个漏洞进行远程提权操作，该操作在主机的进程中会有所反映。

4.2 数据处理

ADFA-LD 数据集每一条数据是系统调用序列，在此用词嵌入的处理方式，将每个编号映射为词向量，这样做避免了独热编码带来的维度变高的问题，还可以捕获各个系统调用之间的语义关系，比如 open 和 close 系统调用在新的词向量空间的距离会近一些。本文采用 keras 提供的以神经网络为基础的嵌入层进行词嵌入工作（如图4-2），结合后续卷积神经网络直接形成端到端的神经网络模型。

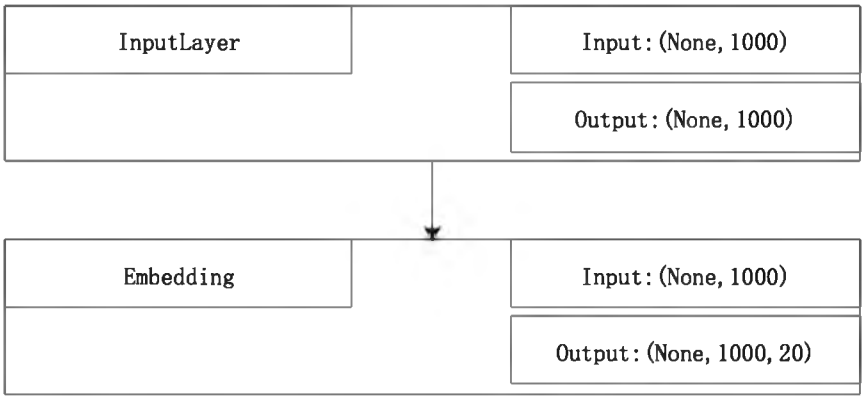


图 4-2 嵌入层结构

图4-2显示该嵌入层将最多长度为 1000 的系统调用序列映射到 20 维度的空间中，该嵌入层的本质是一个特殊的全连接神经网络结构，因此可以写成矩阵运算的形式（如式4-1），其中左侧代表系统调用号的独热编码结果，中间的矩阵代表全连接结构的权重参数，这里不需要偏置参数，由此可以看出，每行神经网络的参数，代表了某个系统调用的词向量。因此在训练更新参数时采用部分更新，只更新系统调用所使用到的词向量。

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \\ w_{41} & w_{42} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} \\ w_{31} & w_{32} \\ w_{21} & w_{22} \end{bmatrix} \tag{4-1}$$

除此之外，还可以将 Word2Vec 或其他结构的向量赋值到 tensorflow-keras 嵌入层中并且冻结期参数，这样可以是为了做到静态的词向量，而本文所采用的则

是直接使用 keras 提供的 embedding_layer 结合下游任务实现端到端的模型设计。

4.3 模型设计与评估

4.3.1 模型设计

本文设计的模型使用词嵌入层结合一维卷积神经网络的方法，词嵌入可以捕获语义特征，将原始的系统调用号映射到特征向量中。随后将这些向量送到一维卷积层中，进一步提取时间维度上的特征。使用不同大小的卷积核可以从不同的范围提取特征。在此之后，通过池层对它们进行下采样，最后送到一个全连接的神经网络中进行分类。其中，词嵌入层，采用使用 Keras 提供的词嵌入层，实现是基于神经网络，可以在任务训练的同时学习到映射关系。

本文设计的模型结构如图4-3，首先将收集到的不同进程的系统调用序列，经过嵌入层处理得到词向量，相较于之前的模型采用独热编码进行处理，这样做会降低输入数据的维度，随后在时间维度上进行一维卷积运算，得到特征序列，通过不同的卷积核处理可以得到不同的卷积序列，这样做可以捕获到不同范围的上下文关系，将这些特征序列拼接起来，一同送入池化层进行池化操作，最后送入前馈神经网络进行分类操作。

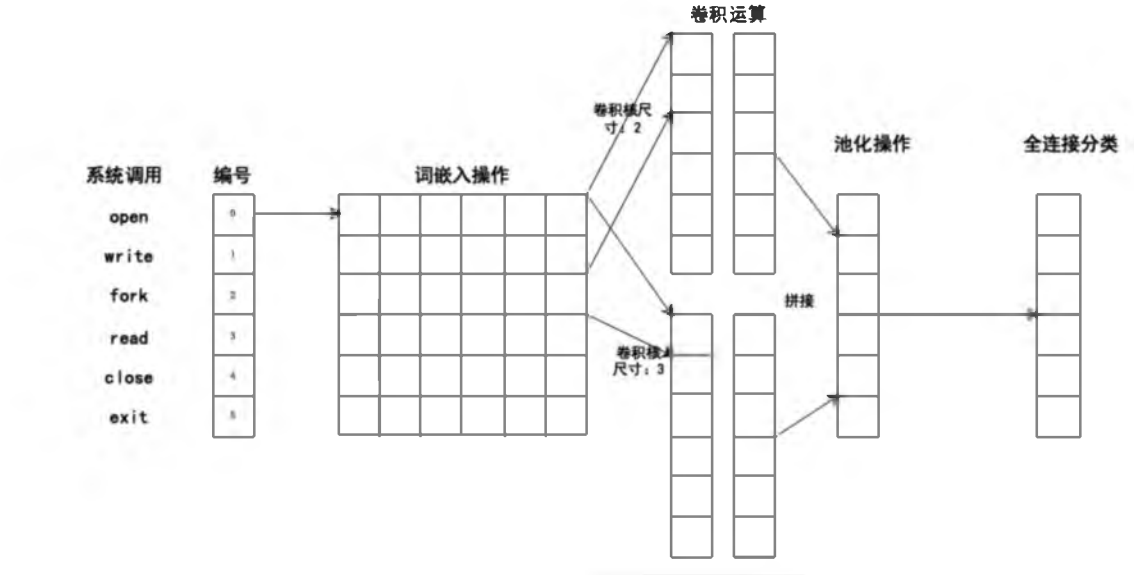


图 4-3 主机级别模型设计

其训练过程如算法4-1所示，其中 *EMBEDDING* 为神经网络构建的嵌入层相关参数，*CNN* 代表上述神经网络模型中的权重和偏置，输入为系统调用序列与标注，输出为训练完成的嵌入层和卷积网络。

算法 4-1 系统调用模型训练算法

Data: syscalls

Result: EMBEDDING，CNN

initialization;
while *epoch*<*epochs* **do**
  for $x_i y_i$ *in syscalls* **do**
    $x_i \leftarrow func\_embedding(x_i EMBEDDING)$;
    $loss \leftarrow loss\_func(x_i y_i CNN)$;
    $grads \leftarrow grad\_func(lossCNN)$;
    $CNN \leftarrow CNN - \eta \cdot grads$;
    $EMBEDDING \leftarrow EMBEDDING - \eta \cdot grads$;
    $total\_loss \leftarrow total\_loss + loss$;
  end
  if $total\_loss/epoch < threshold$ **then**
    break;
  end
end

4.3.2 模型评估

本文采用深度学习框架 tensorflow 构建模型，在 ADFA-LD 数据集中进行实验，实验数据如表4-2所示，对于卷积核尺寸的选择从 3 到 5 进行实验，此外还进行多尺度卷积核的实验。ADFA-LD 数据集中攻击样本与正常样本数量极不平衡，且攻击样本过少，选择 focal 函数作为损失函数，在本次测试中进行二分类的测试即区分正常与异常数据更有意义。

表 4-2 ADFA-LD 数据实验结果

算法	Accuracy	Precision	Recall	F1-score
CNN 卷积核尺寸-3	0.8959	0.7020	0.8699	0.7500
CNN 卷积核尺寸-4	0.8643	0.6755	0.8908	0.7188
CNN 卷积核尺寸-5	0.8898	0.7008	0.9010	0.7514
CNN 卷积核尺寸-3，4	0.8860	0.6982	0.9075	0.7485
CNN 卷积核尺寸-4，5	0.9281	0.7528	0.8139	0.7790
SVM	0.9264	0.7480	0.6670	0.6975
KNN	0.8822	0.6838	0.8564	0.7285

4.4 本章小结

本章详细介绍了主机级别入侵检测的模型，包括入侵检测数据源以及常见的攻击方式，暴力破解，添加超级用户，漏洞利用，以及 Meterpreter 远程控制载荷。系统调用序列因能反映程序和系统的交互行为，被广泛采纳为入侵检测的重要数据源，本次设计以系统调用序列为数据来源，以词嵌入为预处理方法，结合一维卷积神经网络构建检测模型，并在 ADFA-LD 数据集上进行多次实验，以验证该模型的可行性。

第五章 车载终端入侵检测系统设计与实现

本章将介绍车联网终端入侵检测系统的设计与实现部分，首先，介绍车载终端入侵检测系统框架，其次，按模块介绍数据采集模块，数据预处理模块，模型检测模块的设计与实现。最后，在真实车载嵌入式终端平台上进行部署和功能性测试。

5.1 车联网入侵检测系统框架

融合网络和主机级别的车联网入侵检测系统的整体架构，按照平台与功能可以划分为如图5-1所示的部分，主要涉及服务器端平台和车载终端入侵检测的平台。

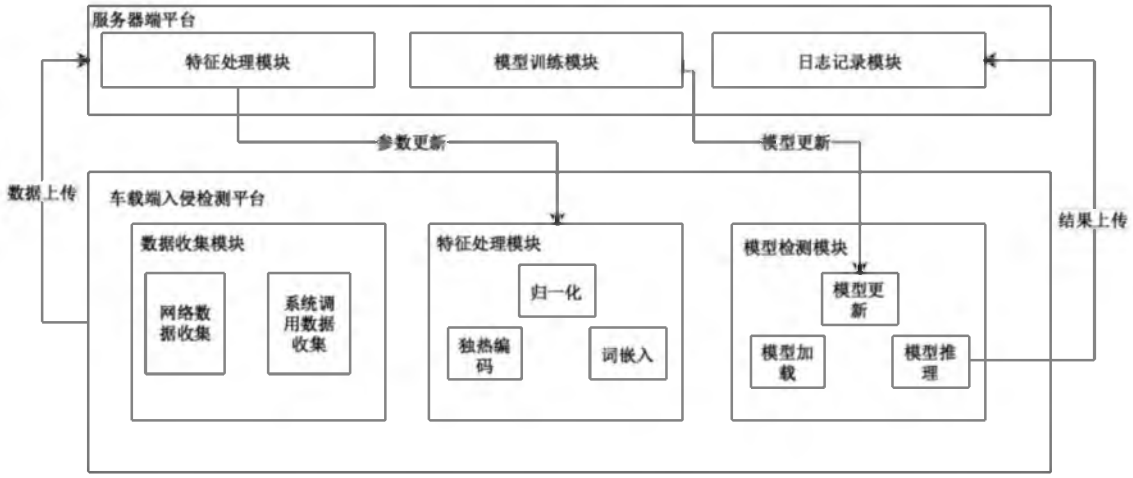


图 5-1 车联网入侵检测系统功能架构

其中车载终端入侵检测平台是本文的重点，从左到右依次为数据收集，特征处理，模型检测。数据采集模块，利用数据采集工具如 TCPDUMP 和 AUDIT，采集相应的网络流数据和主机进程的系统调用序列，经过初步格式处理之后，可将数据送入特征处理模块进行处理，特征处理模块使用归一化，独热编码，词嵌入等方法将数据处理成可被神经网络模型接收的形式，送入模型检测模块进行模型推理，模型推理的结果将显示目前主机是否遭受入侵行为，如果发现入侵行为则首先存储到本地文件中，按照特定周期上传至服务器端，以供后续分析。服务器端为了模型训练直接接收车载端搜集的数据作为输入，并完成与车载端一样的特征处理后，送入模型训练模块进行训练，特征处理模块的参数更新，是为了加速车载端的处理速度，在服务器计算出数据特征参数，直接用于在车载端特征处理模块中，车载端不需要重复计算。在模型训练模块训练完成时 (模型设计详见三，

四章)，可以将模型更新部署到车载终端模型检测模块中。日志记录模块将收集各个车载终端的检测结果作为记录，日志的主要信息为车载终端编号，发生时间和攻击方式，此模块只做数据存储，不做后续处理，但日志内容可供后续的观察，如判断攻击范围，采取对应防御手段等等。

本文所设计的车载终端入侵检测框架，包括数据采集模块，特征处理模块，模型检测模块。

数据采集模块，负责采网络数据包和记录进程执行的系统调用序列，当记录完成时，将原始数据进行格式处理后送入特征处理模块做进一步处理。

特征处理模块，负责将数据进行独热编码，归一化，词嵌入处理，以便数据能够被后续的模型所接收。

模型检测模块，负责根据采集到的数据进行入侵检测的判断，其中模型来自服务端训练完成的神经网络模型，在终端上采用 ARMNN 神经网络加速库进行推理。同时该模块还负责更新模型。

5.2 模块设计与实现

5.2.1 数据采集模块

数据采集模块是入侵检测系统的基础模块，该模块负责从三个数据源包括 TCP/IP 协议网络，车内 CAN 总线，以及进程信息中获取数据，该模块集成 tcpdump，candump，audit 等工具负责对数据源进行采集，该模块负责管理这些工具的使用周期，包括配置，启动，执行时间间隔，异常重启，正常结束等等，同时还负责将上述采集过程得到的原始数据进行初步处理，包括将 tcpdump 所得到的 pcap 格式的数据报文转换为 NSL-KDD 格式，将 candump 处理得到的数据报文提取 CANID，将 audit 处理得到的文本日志中，进行字符处理，搜索相关进程执行的系统调用形成序列的形式。

该模块提供两个功能，采集工具管理，采集数据初步处理。

（1）采集工具管理，负责采集工具的配置与启动，异常重启等。

采集工具管理功能负责管理采集工具的生命周期（如图5-2），是为了监控各个工具的正常运行，以及合理的分配采集时间。以 tcpdump 为例说明首先配置 tcpdump 参数，然后启动 tcpdump 抓包，如遇到 tcpdump 异常退出的情况下，一种情况，物理连接中断，则监控 tcpdump 的进程会以间隔一定时间重新启动 tcpdump。

（2）采集数据格式处理，负责对原始数据的格式进行处理，以便后续处理。

采集数据格式处理（如图5-3），包括将 tcpdump 所得到的 pcap 格式的数据报文转换为 NSL-KDD 格式；将 candump 处理得到的数据报文提取 ID 字段；提取

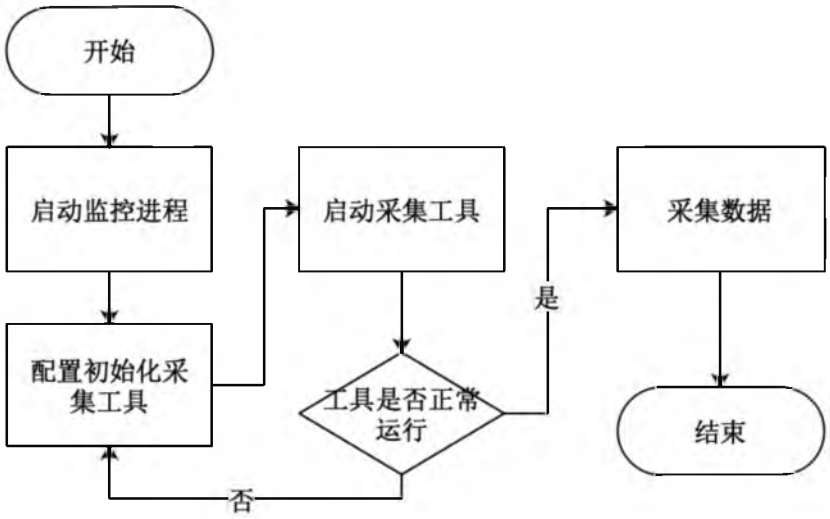


图 5-2 采集工具管理流程

audit 采集得到的日志系统调用信息，处理成序列形式。在此格式处理并不涉及到异常点或者缺失值处理，而是主要涉及一些字符串的拼接与统计操作。

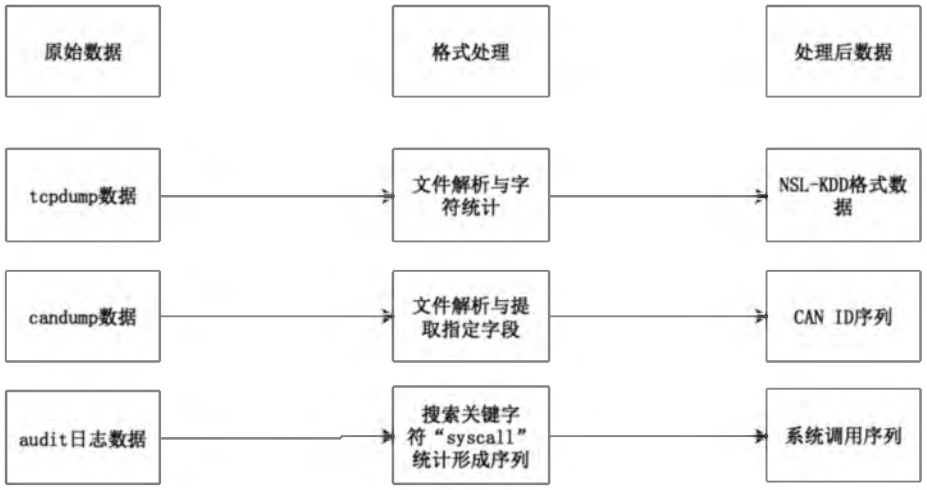


图 5-3 采集数据格式处理

对于网络数据的采集，协议解析过程由 tcpdump 和 candump 完成，其作用是从网络中捕获数据包文件，此后经过格式处理，得到特征处理模块所需要的数据。

对于主机系统调用序列的采集，开启 auditd，audit 会在内核中持续不断地记录系统进程信息，生成审计日志，这些日志包括很多内容，对于本文来讲，值得注意的是各个进程执行的系统调用信息，为此需要在文本中，提取相同进程的系统调用信息。

该模块的整体设计分为 4 个类实现（如图5-4），其中有一个接口类，Collection_Tool 负责制定，上述三类各种工具的接口方法，包括初始化方法 init 和采集

方法 sample，其中初始化的作用是配置采集工具，包括采集的时间，采集生成数据的格式，采集后存放数据的位置等等。其中采集方法接受一个布尔型参数，表示开始或停止数据收集。

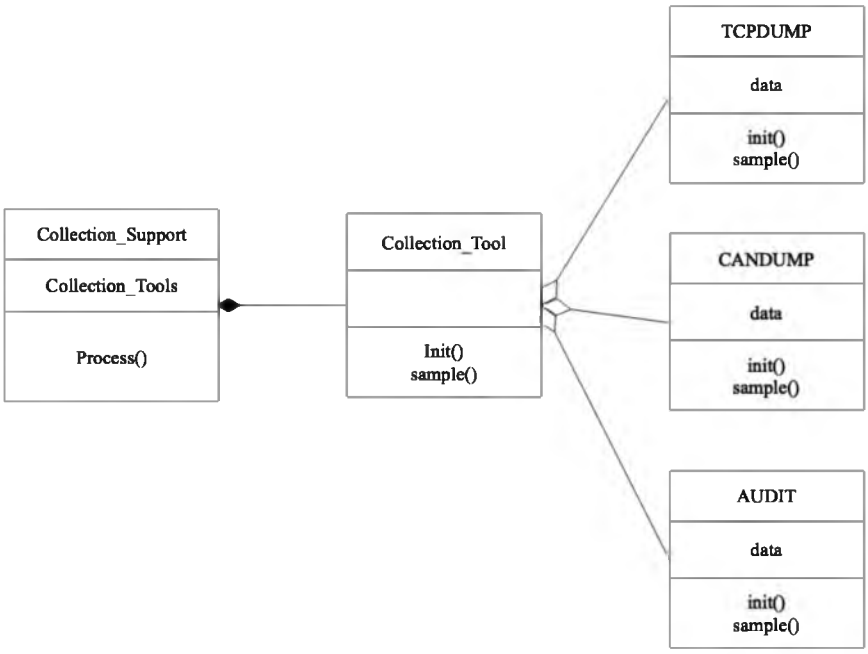


图 5-4 数据采集类设计

实现该接口的一共有三个类分别对应着上述三种工具，其中 TCPDUMP 类封装使用 tcpdump 工具进行 tcp/ip 网络数据的采集，CANDUMP 封装使用 candump 工具进行 can 总线数据的采集，AUDIT 封装使用 auditd 工具进行系统调用序列的采集。除此之外，Collection_Support 类，将整个数据采集过程封装起来，可以做到同时指定某个采集工具进行采集行为。该模块采集的数据存储在系统的文件中，以供后续模块进行处理。至此介绍了数据采集模块的设计框架，以下小节将具体介绍各个工具在模块中的应用方式。

5.2.1.1 TCP/IP 数据采集

在上述数据采集模块框架的基础上，本文利用 tcpdump 进行 TCP/IP 网络协议的数据采集。tcpdump 是网络数据包截获与分析的工具。tcpdump，需要超级用户权限，并且配置网卡至混杂模式，该模式可以监听终端所在网络的全部流量信息。因此，该工具可以分析网络上针对部署主机以及其他计算机的攻击行为。

本文利用 tcpdump 进行 TCP/IP 数据包截获，截获过程，生成 pcap 文件，进而进行数据解析与分析。由于 tcpip 协议是四层协议协议数据是从上层到下层封装，然后发送对于协议的解析需要从下至上进行重组，首先对链路层数据的协进行分

析解析，链路层数据主要包括 IP，ICMP 等，对于 IP 协议去掉报文首部重组后送入传输层进行协议解析，解析器是否为 TCP 或 UDP，之后对于应用层数据由于种类繁多且大都有加密的方式不再对内容进行解析。在此过程中各层级的协议特征被保留下来。

在 tcpdump 完成流量的解析和处理工作后，得到 pcap 文件，该文件是原始数据报文，随后使用 Bro 工具 (流量分析工具)，编写 bro 脚本，仿照 NSL-KDD 数据集所规定的特征，进行统计获取指定的特征。这些特征可分为基于时间窗口的特征和基于流量窗口的特征。基于时间窗口的特征又可分为一定时间内统计连接同主机和同服务的会话两种，基于流量的特征，便于检测低频攻击，是统计一定数量的会话中相关的特征。

tcpdump 采集流程如图5-5，首先将网卡配置成混杂模式，然后配置 tcpdump 和 bro 的解析规则，随后规定使用 tcpdump 在一定时间抓取一定数量的网络包生成 pcap 文件，用 bro 进行解析与统计，从 pcap 中统计特征，随后生成 NSL-KDD 格式的数据，以便于后续模块的使用。

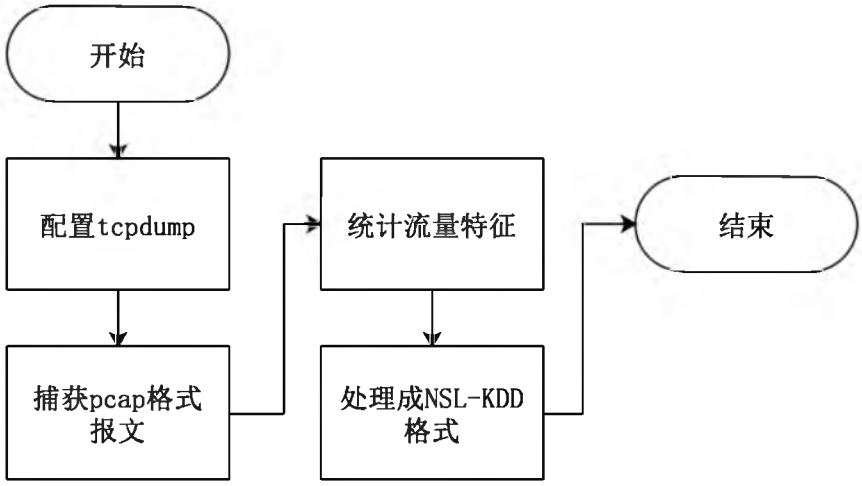


图 5-5 tcpdump 采集流程

本文采用工具库方式实现该模块，主要为了快速实现原型系统，在后续的优化过程中，会将其逐步替换为使用程序库的形式，来进行数据采集模块。

5.2.1.2 CAN 数据采集

本文利用 candump 工具进行 CAN 帧的捕获，candump 是基于 linux 内核提供的 CANSocket 开发的 CANutils 的工具之一，主要有过滤，捕获，存储 CAN 帧的功能。can-utils 其他的工具:cansend，具有构造并发送 can 帧的作用，往往可以用于对指定电子控制单元进行 can 帧测试。cangen，随机的产生 can 帧流量，可以作

为模糊测试的工具。cansequence，以序列的形式发送 can 帧。cansniffer，自动比较 can 帧内容的不同点。

本文 can 总线入侵检测模型设计过程中，CAN 总线基本帧格式，定义了 12 个不同的字段，然而并不是所有字段都适合做入侵检测，为减轻车载终端的负担，只采用 CAN_ID 字段作为入侵检测依据，CAN_ID，这样做可以有效的检测重放攻击和泛洪攻击，对异常的报文也有很好的检测功能。

由于仅仅使用了 CAN 帧中 CAN_ID 这个字段，在使用 candump 捕获大量 can 帧后，还需要截取相应的 CAN_ID 字段。其中 CAN 帧结构见第二章图2-3，截取完成的数据将送入预处理模块进行处理。CAN 数据采集的整体流程如图5-6。

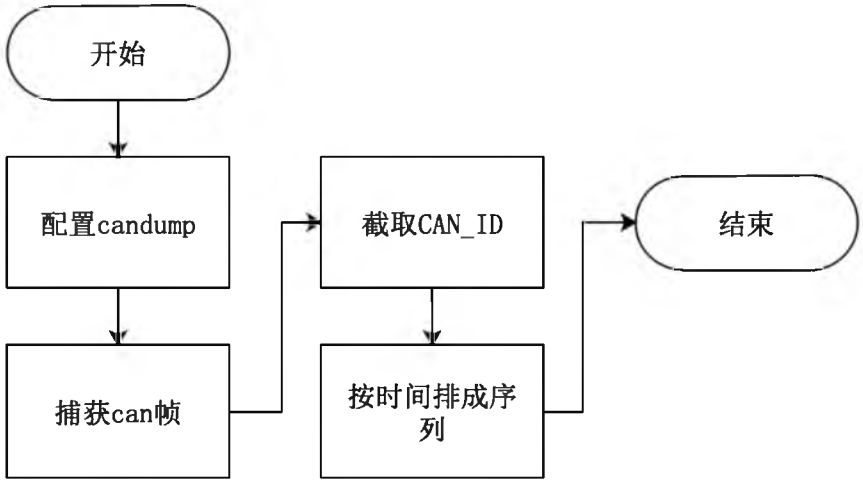


图 5-6 candump 采集流程

首先对 candump 规则进行配置，包括是否过滤指定的 can 帧等，candump 完成了数据报文的捕获以及协议的解析过程，随后生成原始的 can 帧文件，接着程序会截取 ID 字段，形成 can_id 序列，以供后续与处理程序处理。

5.2.1.3 系统调用数据采集

本小节介绍在上述数据采集模块框架下利用 audit 收集系统调用序列的过程。audit 是 linux 安全体系的重要组成部分，在内核中记录系统中发生的各种动作和事件，比如进程执行的系统调用，文件修改，执行的子程序，系统登入登出等等。audit 的功能有：观测文件访问信息，监视系统调用号，记录用户执行的命令，监视网络访问情况。在此更加关注系统调用序列的信息，本文会在 audit 日志中提取相应的系统调用序列。

本文所采用的方式是以 rotate 方式执行 audit，在此模式中 audit 会生成多份审计日志，并且可以通过配置 num_logs 来规定最大日志文件数量，当达到最大个数

时，会循环日志文件。

开启审计功能在系统运行期间，所有被调用到的系统调用均被记录在审计日志中。后续的格式处理将获取到审计日志内容按照进程号进行分组提执行的系统调用号，处理成系统调用序列的格式，然后传入数据预处理模块。syscall 采集流程如图5-7。

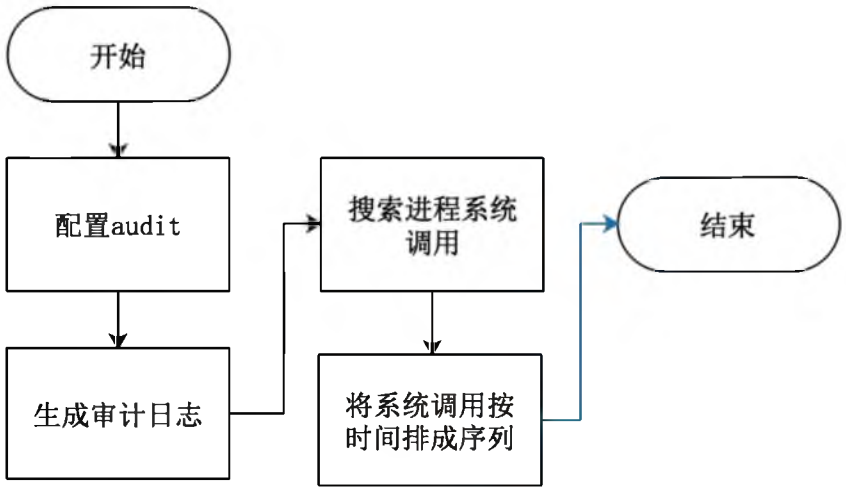


图 5-7 audit 采集系统调用序列

首先启动 auditd 进程，随后等待 auditd 进程生成审计日志，生成日志以后，在审计日志中检索相同 pid 即同一个进程运行时，一定时间内所执行系统调用序列序列，即首先确定 pid 然后搜索该 pid 所执行的全部系统调用序列，以这种序列的方式存入文件，随后重复这一过程，直至程序收到退出指令，这些采集到的系统调用序列将会被送入数据预处理模块进行处理。

5.2.2 特征处理模块

特征处理模块主要实现三个算法：归一化，独热编码，词嵌入。

归一化是指将连续性数值特征映射到 $(0,1)$ 区间内，归一化需要接收两个参数，即数据的最大值和最小值，进行归一化。

$$x = \frac{x - min}{max - min} \tag{5-1}$$

独热编码对离散的特征，依据离散特征的种类扩展到多维度向量，其中需要接收样本数据类型的个数作为参数。

词嵌入，则是将系统调用序列映射为词向量序列，需要接收嵌入层的权重，即向量映射向量关系作为参数，在此的设计并不生成词嵌入映射关系，而是仅仅将

训练好的映射关系存储下来，进行映射。

5.2.2.1 特征处理设计

本文设计五个类来完成这一模块 (如图5-8)，其中每种算法分别设计一个类，这些类包括 Onehot，Normalization，Embedding 分别存有各自算法的配置信息和处理方法，此外本文还设计了算法类，用于自由配置各种数据需要的处理算法，这借鉴设计模式中的门面模式。该算法类进行处理时，输入和输出均为 DataTable 类型的参数，DataTable 类是为了方便处理而设计的数据存储类，其设计思路参考 Python 的 DataFrame 结构，其中实现了可以按行和列遍历的矩阵，也提供按索引访问的方法和读取数据的方法。

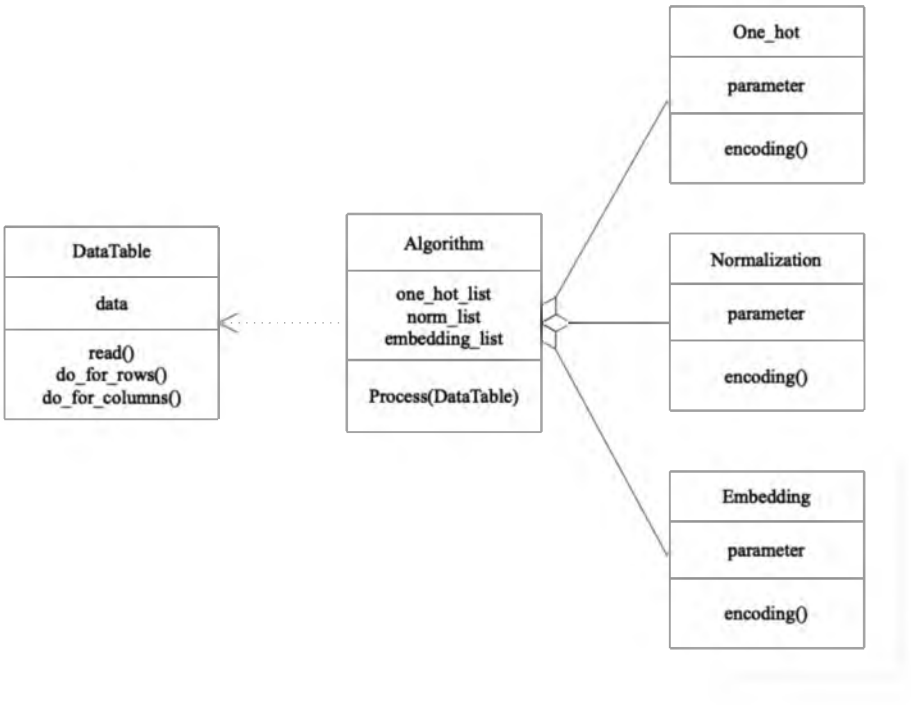


图 5-8 特征处理模块类图

该模块的执行流程（图5-9）。首先来自数据收集模块的数据，读入 DataTable，然后配置算法类，并将需要进行处理的特征列名称连同 DataTable 实例一起传入算法类实例中，执行处理的方法，结束后将数据送入检测模块进行处理。

其中对采集的 TCP/IP 的数据，主要采用归一化和独热编码的方式进行处理，对采集的 CAN 总线上的 CAN 帧 ID 片段，进行字符级别独热编码进行处理，对于系统调用序列，采用词嵌入的处理方式，其中词嵌入的权重矩阵来源于服务器端训练好的词嵌入层。

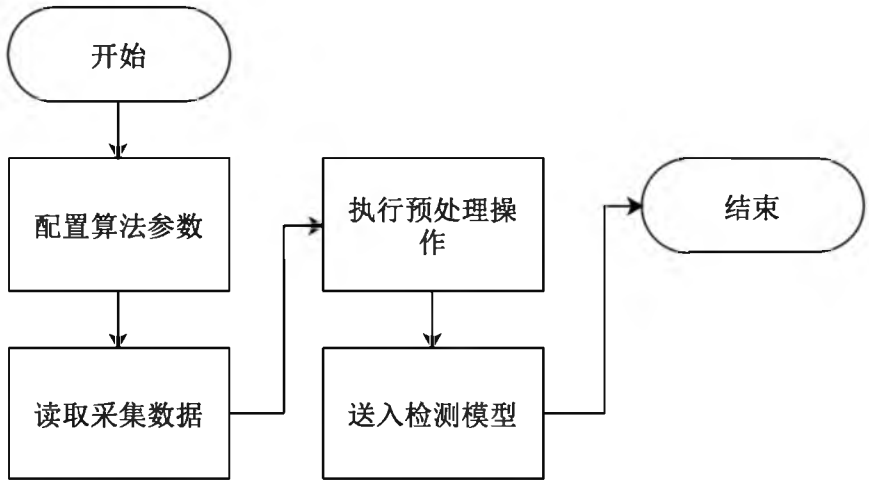


图 5-9 预处理模块流程

5.2.2.2 性能优化

本文为减少特征处理对车载主机的负担，做了以下优化：服务器端预先处理，车载端收集到的数据送入服务器端进行归一化，独热编码，词嵌入等处理，将获取到的算法参数传入车载主机，这样在车载端进行上述算法时，仅仅需要完成最后的映射这一步骤。以词嵌入过程为例进行说明，服务器端，将收集到的系统调用序列进行词嵌入训练，得到各个系统调用的词向量，随后将这些词向量组送入车载主机，而车载主机端，不需要在进行词向量的生成工作，而是直接将系统调用号映射为对应的词向量，时间复杂度为为常数级别。在真实环境的运行过程中，特征处理模块在一定时间内要处理的数据，相较于之前在数据集中训练的数据要少很多，处理时，CPU 占用率极低并不会构成性能瓶颈，此外还可以控制数据采集频率以减轻预处理的负担。

5.2.3 模型检测模块

该模块是入侵检测的核心，该模块从服务器端获取三种入侵检测模型，并且使用 ARMNN 框架将模型加载到指定的设备上，绑定模型的输入和输出节点。随后从预处理模块中获取 DataTable 的样本数据，送入各个模型的输入节点，随后利用 ARMNN 加速库进行神经网络的推断，从各个模型的输出节点获取输出的结果，其结果便是正常和各种入侵行为的概率。如果入侵行为的概率过高，则判定为发现了入侵行为，并写入日志，目前日志设计较为简单，主要包括两个方面：攻击事件和攻击方式。由于涉及到 ARMNN 库的内容，本文为了内聚性，将所有的设计 ARMNN 库的内容封装到一个 ARMMModel 类中，所以此处设计仅有一个类，类

图省略。

该模块提供三个功能，加载模型，执行推理，更新模型。

（1）加载模型，负责将模型加载到内存中。

在加载模型中，借助于 ARMNN 的 CreateNetworkFromBinaryFile 接口将模型从 tensorflow-keras 模型文件中加载到内存中，随后绑定模型中神经网络输入和输出的节点，然后选择特定设备（CPU，GPU，APU）的硬件加速优化，最后将模型加载到特定的设备上。程序执行流程如图5-10。当模型加载完成后，便可以进行模型的推理了。

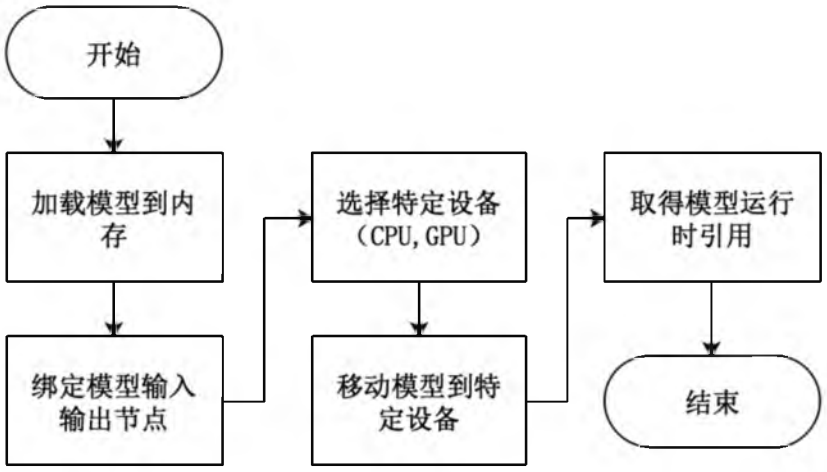


图 5-10 加载模型流程

（2）执行推理，将数据送入模型中，经过计算后获取输出结果。

执行推理，则通过 ARMNN 的 EnqueueWorkload 接口完成的，首先预先分配一块内存，作为输入和输出的缓冲区，将这块内存绑定在上一步骤中的绑定的输入输出节点中。随后，将输入数据拷贝到该内存块中，利用 ARMNN 提供的神经网络运算加速库进行推理运算，随后将得到的结果，即个种攻击类型的概率进行判断，如果超过某一阈值则写入日志文件中，执行的流程如图5-11所示。

在此部分中三个模型分别有三个输出，采用三个进程分别进行三种模型的推理过程，结果会生成三种日志，最后会将这些日志合并，如果需要则上传至服务器端进行分析。

（3）更新模型，从服务器端获取最新模型。

更新模型功能可以方便的进行模型替换，在本原型系统中，便于调试，车载终端可以主动请求服务器端进行模型更新，若服务器没有最新的模型，便可以不更新以节省流量，采用版本号的形式来实现这一流程，流程如图5-12所示。

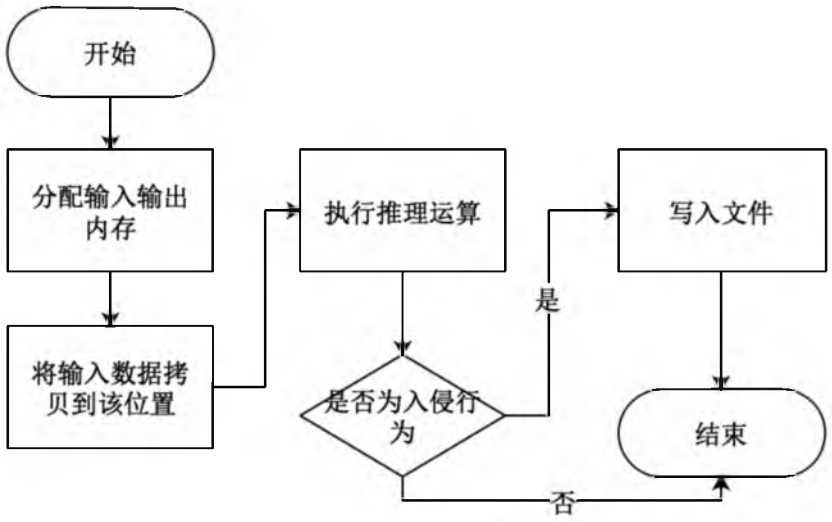


图 5-11 执行推理流程

首先向服务器请求版本号，然后与本地版本号比对，如果是最新版本则不做操作，否则向服务器请求下载模型，待到下载完成后，进行完整性校验，若此时，正在执行模型推理的过程，则待到本次推理完成后，重新启动模型检测模块，重新加载新的模型。

5.3 系统测试与验证

在完成了系统的模型设计，总体设计以及各个子模块的详细设计与实现以后，需要对系统各部分功能进行验证与测试。本小节对前面所实现的车载终端入侵检测系统进行功能性测试。首先介绍实验中中需要用到的工具并搭建相应的测试环境，然后设计实验，最后对各个模块进行实验验证。

5.3.1 实验环境

该实验环境主要包括:

（1）主机服务器一台运行 Ubuntu 18.04，带 RTX2080 GPU。

（2）车载终端 NVIDIA Jetson TX2，运行 ARM Linux。

（3）CAN 总线收发器。

5.3.2 实验设计

测试平台使用 Kali OS，Kali OS 是用于渗透测试的 linux 版本，提供了大量的渗透测试工具。本次测试主要分为三类，针对 TCP/IP 网络的攻击测试，主要使用 nmap 工具和 Hydra 工具进行模拟，针对 CAN 总线网络的攻击主要使用 canutils 等

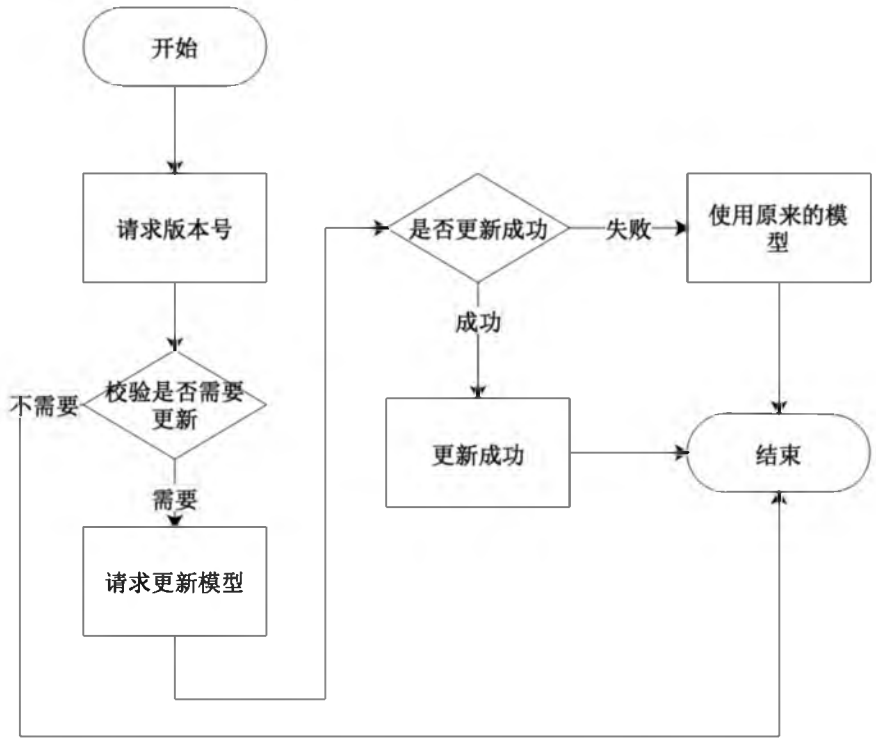


图 5-12 更新模型流程

can 分析工具进行模拟，针对主机的攻击使用 metasploit，Veil，Meterpreter 工具进行模拟。

使用 nmap 工具主要进行端口的嗅探以及发动拒绝服务攻击，而 Hydra 工具则进行 ssh 以及 ftp 的密码暴力破解等攻击。

对 CAN 总线攻击，需要整车环境提供各个 ECU 的 CAN 帧数据，由于成本控制需要，采用生成合法的 CAN 帧序列并且掺入攻击行为的 CAN 帧序列来模拟整车环境的方法完成，以此方式模拟对 can 攻击行为，由于开发板中集成 CAN 总线控制器，可以进行回环测试，如需要进行真实的 CAN 帧收发还需要外接 CAN 总线收发器。没有实车环境，在此以回环的方式进行实验，包括以 linux 回环测试的方式发送大量的 CANID 为 0x00 的报文，进行拒绝服务攻击，以及生成大量随机 ID 的 Fuzzy 攻击。

针对主机的攻击，采用 veil 或 metasploit 生成 meterpreter 工具部署到车载终端，然后进行远程连接。

表 5-1 模块测试

攻击类型	测试内容
暴力破解	检测 Hydra-ssh 检测 Hydra-ftp
木马程序	检测 C 版本 meterpreter 检测 java 版本 meterpreter
嗅探	检测 nmap 扫描
拒绝服务	检测 nmap dos 攻击
CAN 总线攻击	检测 can dos 攻击 检测 can fuzzy 攻击

5.3.3 性能测试

5.3.3.1 分类性能测试

在前面三、四章节已经在公开的入侵检测数据集上对模型的分类能力做过评估。在此进一步验证模型在实际环境中的表现，表5-1列出本次测试的攻击方式。测试环境为一台运行 Kali OS 的主机，与运行着 armlinux 开发板，其上运行如 ssh 和 ftp 等少量服务。

在执行上述测试的同时，首先将攻击时采集到的数据存储下来并进行人工标注，随后执行检测程序，将程序执行的结果与人工标注的数据进行对比。采集 10000 条 tcp/ip 网络数据和 10000 条 CAN 帧数据，并且在攻击时选择特定的端口和协议，以方便进行快速人工标注和批处理。由于 CAN 总线是以模拟攻击直接发送第三章所述的数据集的部分报文在此不重复说明。采集 300 条系统调用序列数据，并且在攻击时，记录攻击进程号，以便进行人工标注。混淆矩阵为表5-2，表5-3，表5-4。其中网络数据大致按比例 1：1 进行采样，含有 4643 条攻击数据，CAN 帧数据按 1：1 进行混合，攻击数据定为 5000 条，系统调用为 93 条攻击数据。

表 5-2 TCP/IP 数据混淆矩阵

预测数据 真实数据	攻击流量包	正常流量包
攻击流量包	4638	5
正常流量包	38	5319

分析上述混淆矩阵，从网络流量包可以看出，识别到 4638 条攻击数据，漏报 5 条数据，另外将 38 条正常数据识别成为攻击数据。从 CAN 帧数据可以看出识别到 4980 条攻击数据，漏报 20 条，误报 4 条。从系统调用序列数据可以看出，可以正确识别出 91 条数据，误报 5 条，漏报 2 条。通过进一步分析，漏报和误报主要

表 5-3 CAN 帧数据混淆矩阵

预测数据 真实数据	攻击流量包	正常流量包
攻击流量包	4980	20
正常流量包	4	4996

表 5-4 系统调用数据混淆矩阵

预测数据 真实数据	异常系统调用序列	正常系统调用序列
异常系统调用序列	91	2
正常系统调用序列	5	202

在于非攻击场景和攻击场景转换时产生的数据。本次分类的准确率较高一方面在于攻击类型的特征明显，另一方面在于训练模型时采集的数据样本较为均衡，没有出现如第三张第四章所测试的数据集部分样本不均衡的问题。

5.3.3.2 响应性能测试

这部分评估在加载了入侵检测系统后其操作系统的性能是否会受到明显影响。在实验中，选取系统响应时间变化作为主要指标进行对比。本文通过记录未开启入侵检测系统时，和开启入侵检测系统时，执行 bash 脚本所需时间，以衡量性能。其中 bash 脚本内含多条常规的命令，使用 time 记录执行时间。

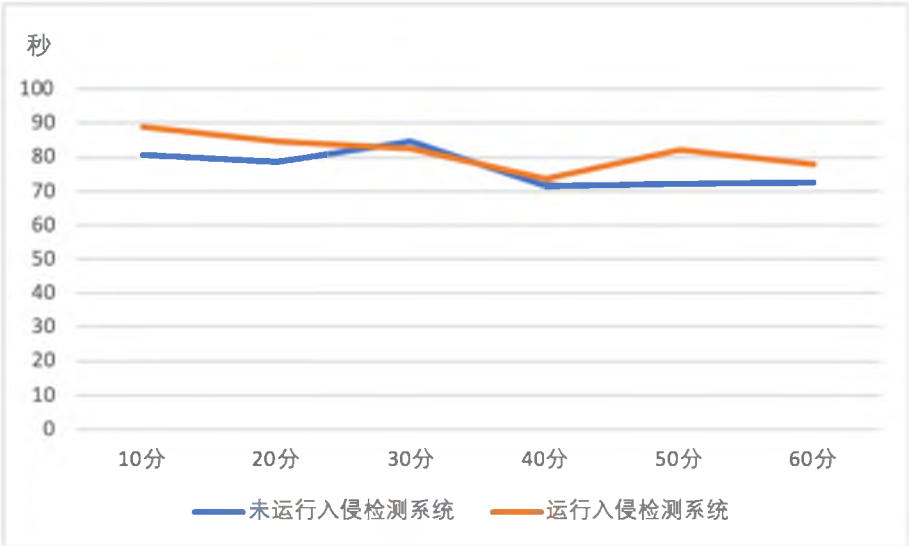


图 5-13 运行入侵检测前后程序执行时间

实验结果如图5-13，其中横轴为系统运行时间，纵轴为上述测试程序的执行时间，通过对比发现，加载入侵检测系统后，系统响应时间略微增加，大致在 10%

上下浮动。

5.3.4 功能测试

采用事后检验的方法，在模拟上述攻击的同时，侧重对系统的数据采集模块，特征处理模块，模型检测模块的进行验证，以验证其功能的完整性。（表5-5）

（1）数据采集模块

该部分的测试主要涉及采集 TCP/IP 协议数据，采集 CAN 总线数据，采集主机日志数据的实验流程，随后查看所收集到的数据，符合预期。

（2）数据预处理模块

对采集到的三种来源的数据，进行按第三章，第四章给出的方案进行预处理，获得预处理结果，符合预期。

（3）模型检测模块

在攻击环境中进行检测，测试的目的是验证系统检测流程的完整性，模型分类与识别性能在第三章，第四章均已验证，用上述设计的实验方式进行攻击行为，验证检测到攻击的流程。

表 5-5 模块测试

测试模块	测试内容	期望输出	实际情况
数据采集模块	采集 TCP/IP 协议数据	采集数据文件	与期望一致
	采集 CAN 协议数据	采集数据文件	与期望一致
	采集进程系统调用序列	采集数据文件	与期望一致
特征处理模块	处理 TCP/IP 协议数据	预处理数据	与期望一致
	处理 CAN 协议数据	预处理数据	与期望一致
	处理进程系统调用序列数据	预处理数据	与期望一致
模型检测模块	检测 nmap 扫描	识别	与期望一致
	检测 nmap dos 攻击	识别	与期望一致
	检测 Hydra 攻击	识别	与期望一致
	检测 can dos 攻击	识别	与期望一致
	检测 can fuzzy 攻击	识别	与期望一致
	检测 meterpreter 攻击	识别	与期望一致

5.4 本章小结

本章介绍了车载终端入侵检测系统框架设计与实现，介绍了车联网入侵检测系统框架，重点介绍了数据采集模块，特征处理模块和模型检测模块的设计与实现，另外对车载终端系统进行了真实环境下模拟进行攻击的测试与验证。融合网络和主机车联网入侵检测系统包括从数据采集模块，特征处理模块到模型检测模

块，最后通过日志模块记录入侵数据并上传服务器记录。数据采集模块，主要介绍了使用相关工具 TCPDUMP，CANDUMP，AUDITD 等进行数据采集的流程。特征处理模块，主要介绍了三种特征处理算法，独热编码，归一化，词嵌入以及实现的流程。模型检测模块，则介绍模型检测的工作流程。此外模拟真实环境对车载终端进行攻击验证入侵检测系统系统。

第六章 总结与展望

6.1 全文总结

随着移动网络，车联网技术的发展，汽车越来越智能，越来越多的车载应用连接到外部网络，同时，内部网络中电子控制单元也越来越多。而网络带来便利的同时，也会带来安全隐患，智能汽车的安全性是重中之重，其安全性可分为功能安全和信息安全，一些不法分析可以通过攻击车外网络进而攻破车内 CAN 总线网络，不仅仅可以窃取个人隐私，还可以控制车辆系统。设计车联网入侵检测系统，对保护人们财产与生命安全极为重要。目前，汽车信息安全研究正处于初级阶段，论文针对车载终端车联网入侵检测系统进行研究，构造多数据源维度的入侵检测系统以达到对车辆信息安全进行全面防护，融合主机级别的入侵检测和网络级别的入侵检测，可以从多个维度对入侵行为进行识别，更全面对车辆网络进行防护。与此同时本文采用人工神经网络设计模型，人工神经网络模型在很多领域取得突破性进展，从侧面说明该技术的优势，应用神经网络技术可以进一步提高入侵检测识别的能力。本文简要介绍车联网相关概念包括 CAN 总线网络和 TCP/IP 网络，其次介绍了入侵检测相关技术，包括常见的入侵检测方法以及渗透测试平台，再次，介绍了神经网络的相关概念，包括基本模型和一些特殊结构以及前沿的研究，与此同时又介绍了词嵌入方法，并选择 Word2vec 模型和 bert 模型为代表进行介绍。在此相关理论的基础上，设计了网络级别入侵检测模型和主机级别入侵检测模型，其中网络级别入侵检测模型，根据数据来源分为两类，第一类是 TCP/IP 协议的网络入侵检测模型，该模型并依据 NSL-KDD 数据格式作为输入格式，采用自编码网络结合前馈神经网络分类器进行设计，可以检测常见的扫描，拒绝服务，远程提权的一些攻击。第二类是 CAN 总线协议网络入侵检测模型，该模型用 CAN 帧的 ID 字段序列作为输入，使用字符级别卷积网络进行模型设计。对于主机级别的入侵检测，系统调用序列被广泛使用的入侵检测数据源，本模型使用进程系统调用序列作为输入，采用词嵌入的方法，将系统调用序列映射为词向量，这样做在降低维度的同时，可以捕获到系统调用序列之间的语义关系。最后一章详细介绍本文所实现车联网入侵检测系统架构以及各个模块的相互关系，其中包括数据采集，特征处理，模型检测等，最后本文在车载嵌入式终端构建了这以入侵检测系统，并用 Kali OS 提供的各种工具进行渗透攻击，验证了这一系统的可行性与功能完整性。

6.2 后续工作展望

本文基于神经网络算法实现了融合主机与网络的车联网入侵检测系统，该系统能够检测 TCP/IP 网络和 CAN 总线常见的攻击行为，也可以根据主机进程信息发现入侵行为。可以为汽车信息安全从业人员提供一个初步的设计框架与系统原型。随着车联网的发展，车载终端不在是一个个独立的个体，而会形成一个分布式的网络，入侵检测系统，需要以分布式的方式部署到各个终端上，以便获得更完整的数据以及更强的运算能力；此外，随着网络安全的发展，新型攻击也会层出不穷，而新型攻击往往具有小样本甚至零样本的情况，这样的问题亟待解决。最后，由于汽车厂商对汽车数据往往采取封闭的形式，制作一个合适汽车信息安全研究的数据集也显得很重要。

致 谢

在攻读硕士学位期间，首先衷心感谢教研室肖老师，两位罗老师，陈老师，李老师，乔老师等全体老师，感谢他们一直以来提供的无私的帮助与指导，再由衷的感谢教研室可爱的同为研三的小伙伴们，张文，雷品源，金宙贤，郭子伦，张玲慧同学，以及王龙，何俊鹏，李瑞坤，易佳佳等研二同学，以及谭钦云等研一的小学弟，感谢他们在学习与科研中的强力支持，感谢在为辰信安一起工作过的同事们，感谢他们的包容与支持。感谢叶丹老师和韩孟洁老师，感谢她们在生活上给予的强力保障。最后感谢我的父母，感谢红鱼和悦然以及感谢缠姐。

最后借缠姐一首诗，“雪压神州三万里，依然北极共星辰，冬云不待东风破，腐鼠寒鸦枉噪尘。”马上毕业了，心情自然是愉悦的，同时多少危机是发生在最“万无一失”的时候呢，阙疑之后，更须慎言其余，慎行其余。

参考文献

[1] A. L. Buczak, E. Guven. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection[J]. IEEE Communications Surveys and Tutorials, 2016, 18(2): 1153-1176

[2] 侯怀臻. 基于大数据技术的分布式入侵检测系统设计 [D]. 山东大学, 2019

[3] 董琛. 车辆 can 总线入侵检测系统的研究与实现 [D]. 北京交通大学, 2019

[4] 靳亚洽. 基于机器学习的 ddos 实时网络入侵检测系统关键技术的研究 [D]. 华南理工大学, 2019

[5] 曾凡. 网联汽车入侵检测系统的研究与实现 [D]. 电子科技大学, 2018

[6] 苟玲. 基于经验模态分解的网络流量检测与分析方法 [D]. 电子科技大学, 2018

[7] 孔令智. 基于网络异常的入侵检测算法研究 [D]. 北京交通大学, 2018

[8] 闫　涵. 基于 k-means 的入侵检测方法研究 [D]. 哈尔滨工业大学, 2017

[9] H. H. Pajouh, R. Javidan, R. Khayami, et al. A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks[J]. IEEE Transactions on Emerging Topics in Computing, 2019, 7(2): 314-323

[10] A. A. Halimaa, K. Sundarakantham. Machine learning based intrusion detection system[J]. Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019, 2019, 916-920

[11] J. V. Anand Sukumar, I. Pranav, M. M. Neetish, et al. Network Intrusion Detection Using Improved Genetic k-means Algorithm[J]. 2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018, 2018, 2441-2446

[12] E. Min, J. Long, Q. Liu, et al. TR-IDS: Anomaly-Based Intrusion Detection through Text-Convolutional Neural Network and Random Forest[J]. Security and Communication Networks, 2018, 2018: 4943509

[13] E. Viegas, A. O. Santin, A. Franca, et al. Towards an energy-efficient anomaly-based intrusion detection engine for embedded systems[J]. IEEE Transactions on Computers, 2017, 66(1): 163-177

[14] Y. Mirsky, T. Doitshman, Y. Elovici, et al. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection[J]. 2018 16th Annual Conference on Privacy, Security and Trust, PST 2018, 2018, 18-21

[15] W. Wang, Y. Sheng, J. Wang, et al. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection[J]. IEEE Access, 2017, 6(February 2018): 1792-1806

[16] E. Seo, H. M. Song, H. K. Kim. GIDS: GAN based Intrusion Detection System for In-Vehicle Network[J]. 2018 16th Annual Conference on Privacy, Security and Trust, PST 2018, 2018

[17] K. Karray, J. L. Danger, S. Guilley, et al. Prediction-based intrusion detection system for in-vehicle networks using supervised learning and outlier-detection[M]. Springer International Publishing, 2019, 109-128

[18] R. U. Khan, X. Zhang, M. Alazab, et al. An Improved Convolutional Neural Network Model for Intrusion Detection in Networks[J]. 2019 Cybersecurity and Cyberforensics Conference (CCC), 2019, 74-77

[19] Chiba. Intelligent approach to build a Deep Neural Network based IDS for cloud environment using combination of machine learning algorithms[J]. Computers & Security, 2019, 86: 291-317

[20] C. P. Ngo, A. A. Winarto, C. K. K. Li, et al. Fence GAN: Towards Better Anomaly Detection[J]. , 2019, 1-13

[21] H. Zhang, X. Yu, P. Ren, et al. Deep Adversarial Learning in Intrusion Detection: A Data Augmentation Enhanced Framework[J]. , 2019, 1-10

[22] G. Kim, H. Yi, J. Lee, et al. LSTM-Based System-Call Language Modeling and Robust Ensemble Method for Designing Host-Based Intrusion Detection Systems[J]. , 2016, 1-12

[23] 卢嘉中. 基于全网流量与日志深度分析的 apt 攻击建模与检测技术 [D]. 电子科技大学, 2019

[24] 吕少华. 基于循环神经网络的攻击行为预测研究 [D]. 北京交通大学, 2019

[25] N. Thanh Van, T. N. Thinh, L. T. Sach. a Combination of Temporal Sequence Learning and Data Description for Anomalybased Nids[J]. International Journal of Network Security & Its Applications, 2019, 11(03): 89-100

[26] 莫坤. 基于 stacking 技术的入侵检测系统的设计与实现 [D]. 北京邮电大学, 2019

[27] R. Vinayakumar, M. Alazab, K. P. Soman, et al. Deep Learning Approach for Intelligent Intrusion Detection System[J]. IEEE Access, 2019, 7: 41525-41550

[28] S. Naseer, Y. Saleem, S. Khalid, et al. Enhanced network anomaly detection based on deep neural networks[J]. IEEE Access, 2018, 6: 48231-48246

[29] 张宝安. 基于深度学习的入侵检测研究与实现 [D]. 北京邮电大学, 2019

[30] 许聪源. 基于深度学习的网络入侵检测方法研究 [D]. 浙江大学, 2019

[31] 刘煜. 基于集成学习与半监督学习的网络入侵检测方法的研究 [D]. 华南理工大学, 2019

[32] 李宁宁. 基于机器学习的车联网入侵检测技术的研究与实现 [D]. 电子科技大学, 2019

[33] H. Lee, S. H. Jeong, H. K. Kim. OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame[J]. Proceedings - 2017 15th Annual Conference on Privacy, Security and Trust, PST 2017, 2018, 57-66

攻读硕士学位期间取得的成果

[1] J. Liu, K. Xiao, L. Luo, et al. An intrusion detection system integrating network-level intrusion detection and host-level intrusion detection[C]. 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS), 2020, 122-129