|  |
| --- |
| **学士学位论文** |
| **机器学习应用到威胁程序行为识别的研究** |
| |  |  | | --- | --- | | 学 号： | **20131003261** | | 姓 名： | **任飞** | | 学 科 专 业： | **软件工程** | | 指 导 教 师： | **张剑波 副教授** | | 培 养 单 位： | **信息工程学院** | |
| 二○一七年六月 |

中国地质大学（武汉）学士学位论文原创性声明

本人郑重声明：本人所呈交的学士学位论文《基于浮动车轨迹数据的城市空间画像及可视化方法研究》，是本人在指导老师的指导下，在中国地质大学（武汉）攻读学士学位期间独立进行研究工作所取得的成果。论文中除已注明部分外不包含他人已发表或撰写过的研究成果，对论文的完成提供过帮助的有关人员已在文中说明并致以谢意。

本人所呈交的学士学位论文没有违反学术道德和学术规范，没有侵权行为，并愿意承担由此而产生的法律责任和法律后果。

学位论文作者签名：

日 期： 年 月 日

摘要

随着互联网的高速发展，沟通交流便利的同时，各种计算机恶意软件带来的麻烦也随之而来。尽管病毒检测查杀技术也在不断的升级和改进，但总体上呈现出一种道高一尺，魔高一丈的趋势。传统的静态特征匹配思路虽然在效率上很高，庞大的病毒库使绝大多数恶意软件无所遁形。但对于新出现的恶意软件抵御力非常差。

得益于大数据技术，数据科学技术的的发展，应用机器学习到计算机安全领域逐渐火热起来。本文主要研究了基于机器学习方式对应用程序行为进行识别，从而区分开恶意程序和正常程序。本文的工作主要分为以下几个部分：

⑴ 在恶意软件分析方面，本文采用词袋模型对应用软件行为进行表示。应用程序运行时，统计其运行期间各个内核API的调用次数。将得到的API调用频率字典最后转换为向量作为数据数据。结果表明，该表示方法有一定效果，在聚类分析和分类预测上都有一定成果。

⑵ 在软件行为分类方面，本文K-means聚类算法对行为进行聚类分析。K-means聚类算法作为常用的一类聚类算法，拥有算法效率高，实现简单的特点。本文基于词袋模型，定义向量夹角大小作为衡量行为相似度标准。

⑶ 在软件行为分类方面，本文使用两种方式对应用程序行为进行分类，分别采用Softmax模型和CNN卷积神经网络的方式。Softmax作为经典的分类函数，借助Tensorflow框架的支持，可以快速实现。CNN卷积神经网络模型是深度学习中常用的模型，借鉴图像处理的思路，将词袋模型表示的软件行为进一步转换为行为矩阵，利用CNN卷积神经网络局部感知的能力，抽取各个API之间关系特征。结果表明，CNN卷积神经网络效果胜过Softmax，有明显的效果提升。

⑷ 在最后的实验阶段，本文针对聚类分析结果进一步熟悉Windows平台下应用软件的API行为特点。结果表明，不同类别的应用软件在行为表现上有比较明显的差异。另外在软件行为预测实验上面，Softmax分类模型和CNN卷积神经网络模型的都有一定结果，CNN卷积神经网络得益于局部感知的能力有着更优异的效果。

**关键词**：恶意软件；软件行为；机器学习；CNN卷积神经网络；

**Abstract**

With the rapid development of Internet, communication is getting more and more convenient. However, it brings more and more security problems in the meantime. For example, on May 12th this year, the malware which is named “wannacry” contributes to millions of computers getting infected. Due to the exploits leaked from the NSA, many hackers make use of them and do evil. Computers with no relevant patches are more likely to get infected.

Although malware detection techniques are improving, while the priest climbs a foot, the devil climbs ten. Traditional ways of static code recognizing are efficient in most situations, but for variation virus, it is not that efficient as expected.

Thanks to the great development of big data techniques and data sciences, applying Machine Learning techniques to computer security is becoming more and more popular. Different from traditional ways of static code analyzing, applying ML is much more flexible, especially on analyzing variation virus. By the run-time monitoring of program execution behavior, get the pattern of virus behavior, and then use the pattern found to recognize other virus and normal software to raise the final alarm.

On the research of malware behavior, much more researchers choose the way of nature language processing, and these two problem do have much similarities, thus making it efficient when applying NLP technique on malware behavior recognition. In this paper, a different way is chosen. Transferring this problem into image recognition shows that this way worth trying. By mapping the calls of system apis into image, behaviors can be transferred into image and CNN is very suitable for this problem, thus making it efficient in recognizing malware behavior.

**Key Words:** Machine Learning; Malware; CNN; Program Execution Behavior;

**目 录**

[图清单 7](#_Toc483852382)

[表清单 8](#_Toc483852383)

[第一章 绪论 1](#_Toc483852384)

[1.1 研究的背景和意义 1](#_Toc483852385)

[1.2 国内外研究现状 2](#_Toc483852386)

[1.3.1 国外研究现状 2](#_Toc483852387)

[1.3.1 国内研究现状 3](#_Toc483852388)

[1.3 本文的目标和实现 3](#_Toc483852389)

[1.3.1 技术路线图 5](#_Toc483852390)

[1.4 论文的结构安排 6](#_Toc483852391)

[第二章 数据技术准备 7](#_Toc483852392)

[2.1 数据准备 7](#_Toc483852393)

[2.1.1 正常软件应用行为样本的获取 7](#_Toc483852394)

[2.1.2 异常软件应用行为样本的获取 7](#_Toc483852395)

[2.2 数据清洗 8](#_Toc483852396)

[2.2.1 根据用户进程切分数据 8](#_Toc483852397)

[2.2.2 行为特征表示-BOW模型 8](#_Toc483852398)

[2.2.3 提取API词袋模型 9](#_Toc483852399)

[2.3 技术框架准备 10](#_Toc483852400)

[2.3.1 Tensorflow机器学习框架 10](#_Toc483852401)

[2.4 本章小结 12](#_Toc483852402)

[第三章 聚类分析威胁行为特征 14](#_Toc483852403)

[3.1 威胁程序行为特点 14](#_Toc483852404)

[3.1.1 Windows系统API 14](#_Toc483852405)

[3.1.2 威胁程序行为特点分析 15](#_Toc483852406)

[3.2聚类分析算法 15](#_Toc483852407)

[3.2.1 主流聚类算法介绍 15](#_Toc483852408)

[3.2.2 K-means聚类算法 16](#_Toc483852409)

[3.3 使用Tensorflow聚类分析 17](#_Toc483852410)

[3.3.1 行为相似度定义 17](#_Toc483852411)

[3.3.2 特殊API弱化处理 18](#_Toc483852412)

[3.4 本章小结 19](#_Toc483852413)

[第四章 基于API调用特征威胁行为分类预测 20](#_Toc483852414)

[4.1 分类类别确定 20](#_Toc483852415)

[4.2 Softmax分类 20](#_Toc483852416)

[4.2.1 Softmax模型 21](#_Toc483852417)

[4.2.2 Softmax实现威胁行为分类 22](#_Toc483852418)

[4.3 CNN卷积神经网络实现分类 25](#_Toc483852419)

[4.3.1 CNN卷积神经网络 25](#_Toc483852420)

[4.3.2 CNN神经网络设计 27](#_Toc483852421)

[4.3.3 权值初始化 29](#_Toc483852422)

[4.3.4 ReLU激活函数 29](#_Toc483852423)

[4.3.5 卷积和池化 30](#_Toc483852424)

[4.3.6 Dropout降低过拟合程度 30](#_Toc483852425)

[4.4 本章小结 31](#_Toc483852426)

[第五章 实验结果与分析 32](#_Toc483852427)

[5.1 实验环境介绍 32](#_Toc483852428)

[5.2 行为聚类分析和分类预测测评 32](#_Toc483852429)

[5.2.1 聚类分析测评 32](#_Toc483852430)

[5.2.2 Softmax分类测评 32](#_Toc483852431)

[5.2.2 CNN卷积神经网络分类测评 33](#_Toc483852432)

[5.3 实验结果分析 33](#_Toc483852433)

[5.3.1 聚类结果分析 33](#_Toc483852434)

[5.3.2 分类结果分析 36](#_Toc483852435)

[5.4 本章小结 37](#_Toc483852436)

[第六章 总结与展望 39](#_Toc483852437)

[6.1 总结 39](#_Toc483852438)

[6.2 展望 40](#_Toc483852439)

[致谢 41](#_Toc483852440)

[参考文献 42](#_Toc483852441)

# 图清单

[图 1-1 安全事件趋势图 1](#_Toc483839823)

[图 1-2 安全感知与攻击取证 2](#_Toc483839824)

[图 1-3 技术路线图 5](#_Toc483839825)

[图 2-1 应用程序API行为图 10](#_Toc483839826)

[图 3-1 Tensorflow 数据流图示例 11](#_Toc483839827)

[图 3-2 神经网络数据流图示例 12](#_Toc483839828)

[图 3-3 聚类 16](#_Toc483839829)

[图 3-4 对数曲线变化图 18](#_Toc483839830)

[图 4-1 sigmoid函数 21](#_Toc483839831)

[图 4-2 Softmax原理图 22](#_Toc483839832)

[图 4-3 样本输入 23](#_Toc483839833)

[图 4-4 样本标签矩阵 23](#_Toc483839834)

[图 4-5 神经元示例 25](#_Toc483839835)

[图 4-6 神经网络示例 26](#_Toc483839836)

[图 4-7 CNN图像识别示例 27](#_Toc483839837)

[图 4-8 行为向量转换示意图 28](#_Toc483839838)

[图 4-9 CNN卷积神经网络设计 28](#_Toc483839839)

[图 4-10 ReLU激活函数效果图 29](#_Toc483839840)

[图 4-11 卷积核大小和准确率关系图 30](#_Toc483839841)

[图 5-1 IM类应用API图谱 33](#_Toc483839842)

[图 5-2 单机工具类应用API图谱 34](#_Toc483839843)

[图 5-3 网络类应用API图谱 34](#_Toc483839844)

[图 5-4 间谍软件API图谱 35](#_Toc483839845)

[图 5-5 木马API图谱 35](#_Toc483839846)

[图 5-6 Softmax和CNN准确率对比图 36](#_Toc483839847)

[图 5-7 识别易错软件分布图 37](#_Toc483839848)

# 表清单

[表 2-1 恶意软件样本 7](#_Toc483839849)

[表 2-2 API行为记录格式 8](#_Toc483839850)

[表 2-3 SOW模型和BOW模型对比 9](#_Toc483839851)

[表 3-1 Windows系统API分类表 14](#_Toc483839852)

[表 3-2主流聚类算法对比 15](#_Toc483839853)

[表 3-3 kmeans聚类算法复杂度分析 17](#_Toc483839854)

[表 4-1 应用软件分类 20](#_Toc483852777)

[表 5-1 硬件配置平台说明 32](#_Toc483839855)

# 第一章 绪论

## 1.1 研究的背景和意义

随着互联网的急速发展，随之带来的安全隐患也逐渐明显。据xxx发布的互联网安全白皮书发布的最新计算机安全局势，安全事件的发生呈现急速增长趋势：

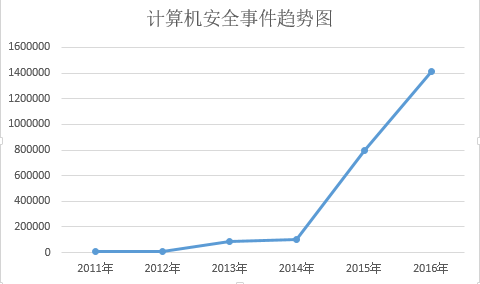


图 1-1 安全事件趋势图

病毒，木马，蠕虫，各类间谍软件，勒索软件等恶意程序已给个人以及企业造成了极大的损伤。据不完全统计，仅2016年一年，网络安全事件事件已在中国造成了915亿元的直接经济损失。以2017年5月12号爆发的“wannacry”蠕虫病毒为例，该恶意软件是一款典型的勒索软件，应用蠕虫技术广泛传播，在宿主机上寻找重要资产文件并加密，导致用户数据丢失，然后以解密数据要挟受害者支付价值300美金的比特币。截止至5月15号，已有几十万计算机中毒，包括很多政府职能部门和企业机构，影响甚是宽广。而近几年，从“熊猫烧香”到“灰鸽子”，再到近期的“wannacry”恶意软件，大规模计算机安全事件的发生越来越频繁。再加之美国国家安全局NSA此前泄露的大量黑客武器以及漏洞，其中不乏各种尚未修复的0day漏洞，军工级黑客武器的泄露直接给当前的计算机安全局势造成严重影响。因此，完善计算机的安全防御工程刻不容缓。

就目前的主流防御手段而言，大都采用的是一种偏防守偏被动的防御方式。这种被动一方面表现在很多杀毒软件采取检测恶意程序静态特征的方式判别，这种方式依赖于病毒库的丰富程度，检测效率高但同时缺陷也很明显，就是不易检测变种后的恶意程序。另一方面，被动也表现在检测手段依赖于现有计算机漏洞的发现，对于还未公开的0day漏洞而言，几乎没有抵御能力。

得益于大数据技术，数据科学的发展，基于机器学习的主动防御方式逐渐进入人们的眼球。黑客的攻击手段再多变始终都是实现自己的目的，得到自己想要的利益。基于黑客攻击时候留下的蛛丝马迹，从大量的攻击记录中分析攻击者的攻击模式，实现恶意攻击感知预警。

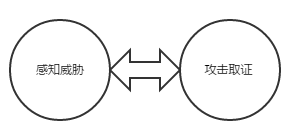


图 1-2 安全感知与攻击取证

一方面通过现有的攻击样本分析获取攻击模式，一方面通过构建的攻击模式拦截未来可能发生的潜在威胁，二者相辅相成。本文所探究的应用机器学习的方式来识别可疑进程，正式实现图中感知威胁那一环。无论恶意软件多么复杂，最终都是要在宿主机系统上运行才能达到攻击者的目的，这样也就避免不了对系统API的调用，这里系统API的调用加之利用就可以分析恶意软件的行为模式特征，从而实现运行时检测，及时发出预警。

## 1.2 国内外研究现状

### 1.3.1 国外研究现状

在国外，病毒行为的研究已经进入到了非常深入的层次。尤其在大数据时代的背景下，人工智能识别的方式得到了空前的发展。

静态检查层面，文件MD5校检和静态代码特征检测技术一直在不断的发展。病毒库日趋丰富，检测速度越来越快，各种云杀毒引擎的问世也使得静态检测越来越高效。除此之外，将机器学习应用到静态检查上的研究也有不少，其中较为出名的的检测案例是使用逆向工程技术分析汇编代码，将病毒代码映射成一张图片，在利用图像识别的方式，基于图像相似度分析的手段，通过深度学习的方式进行再识别取得了不错的成效。

动态检查层面，依托操作系统运行时API调用记录，分析操作系统API调用序列也是一个热门的方向。一方面，有人使用传统的模式识别，根据安全专家的经验设定具有威胁的行为模式，用该模式来匹配运行中的程序从而达到威胁程序检测目的。另一方面，在机器学习火热的当下，越来越多的人尝试结合机器学习的方式，首先从已获取的行为样本中找寻威胁行为模式，再用来学习来的模型做分类，区分开恶意程序和正常程序。在这一方面，Faraz Ahmed【1】取得了不错的成效。在Faraz Ahmed【1】的研究中，使用了基于词向量分析的方式，通过分析各个API在各类中出现的概率对软件应用进行分类，取得了可观的成果。

### 1.3.1 国内研究现状

国内对于恶意软件的识别也正在往人工智能代替人工的方向发展。传统的模式识别的方式，通过构建好的行为特征规则去匹配现实场景中的攻击行为。这样做准确度会更高但是会缺乏灵活性，而且很大程度上依赖专家对病毒行为的了解。目前大部分企业中使用的还是基于这种传统思路的专家系统在起着主导作用。

再谈国内在机器学习恶意软件识别领域的研究现状，和国外差不多，国内主流也是针对恶意软件的静态代码做机器学习，对反汇编得到的汇编代码进行分类。在效果上对变种恶意软件的识别同样很有效。对于恶意软件行为的研究，小金[3]的研究深度解析了当前主流的恶意软件的行为特点。奚小溪[4], 孙荣会[4]的研究中进一步归纳总结了可行的恶意软件行为检测技术。

在针对的平台上，国内对Android等移动新兴平台的恶意软件行为检测也做了深入的研究，应用当前发展火热的移动计算技术，在效果上取得了不错的成就。

## 1.3 本文的目标和实现

现实场景中的恶意软件查杀工具普遍基于原恶意软件的静态代码特征进行检测，但是在应对恶意软件变体以及新兴的恶意软件都存在不足。本文的主要目标实现基于基于机器学习的方法通过对软件程序的API调用行为识别区分恶意软件以及正常软件。本文借鉴自然语言处理的方式，依托应用软件在运行过程中的API调用记录，在TensorFlow机器学习框架的支持下，分析其运行时的行为模式特征区分开恶意软件和正常软件。具体研究目标和实现如下：

1. 词袋模型下的软件行为表示

应用软件在执行过程中，无论实现怎样的功能都避免不了对系统内核API的调用。本文在研究过程中，通过记录应用软件执行过程中调用的API并将得到的调用序列转换为标准化的词袋模型，为聚类分析和分类预测提供了标准化的数据输入。

1. 应用软件行为聚类分析

研究各个类型应用软件在行为上的差异。本文采用K-means聚类分析算法，基于词袋模型表示下的应用软件行为，将不同类型的软件应用的运行时行为做聚类分析。旨在通过这种聚类分析的方式深入挖掘各个类型软件应用的行为特征。

1. 应用软件行为分类预测

针对应用软件行为模式特征，使用机器学习中的Softmax分类模型和深度学习中的CNN卷积神经网络从大量的数据样本中学习正常软件和恶意软件行为模式的特点，实现不同类型应用软件的分类预测。

1. 实验平台构建

依托Google开源机器学习框架TensorFlow，在该平台下开发用于聚类分析应用软件行为的K-means算法和用于分类预测的Softmax模型和CNN卷积神经网络模型。利用该平台的易用性和高效性，实现本文研究目标。

## 1.4 论文的结构安排

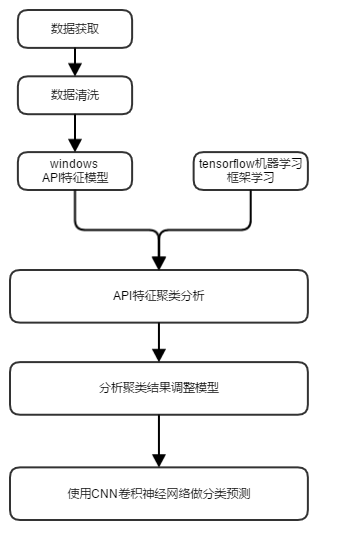


图 1-3 技术路线图

本文主要研究使用机器学习的方式通过对应用程序的行为表现，也就是API调用情况区分正常程序和异常程序，实现通过一段应用程序的运行时API调用实现行为预测。文完成的主要工作包括以下：

⑴ Windows平台软件应用内核API调用记录的获取，

⑵ 恶意软件特征了解，了解整理当前主流的安全威胁软件行为特征，如病毒，木马，间谍软件勒索软件等。

⑶ Tensorflow开源机器学习框架的学习与使用。

⑷ K-means聚类算法的学习与使用。

⑸ Softmax模型与CNN卷积神经网络模型的学习与使用。

## 1.5 论文的结构安排

本文主要的研究工作是使用机器学习的方式对病毒行为进行识别，论文共分为六章：

第一章中，首先介绍了当前互联网环境下的安全态势，并描述了国内外关于恶意程序识别上的研究状况。并在此基础上阐明了本选题的意义和最终希望达到的目标。最后介绍了本文所做的工作，所采取的技术路线图和论文的框架安排。

第二章中，介绍了开展工作前的数据预处理工作和行为的表示方法。

在第三章中，介绍了分类预测之前的聚类分析工作，以及所需要用的机器学习技术框架Tensorflow。为特征选择和聚类算法提供指导意义。

第四章，主要介绍了分类预测的实现，介绍了Softmax和CNN卷积神经网络模型，这两个模型为实验的最终解决方案。

第五章中，主要对实验的结果进行了分析，从中总结了本课题所研究的课题的实现状况以及实验中还存在的问题。

第六章，作为最后一章总结了本课题研究过程中的收获和不足，进一步展望未来还需要完成的工作。

# 第二章 数据技术准备

## 2.1 数据准备

### 2.1.1 正常软件应用行为样本的获取

本文大部分正常程序行为数据来源于腾讯公司，腾讯公司内部所有接入内网终端都会安装一个监控程序iOA，IOA在内核层面监控系统API调用并记录。

另一方面，为了增加程序行为的多样性，本文还特意使用Windows系统工具api monitor 实现程序运行期间Windows内核API调用监控，记录下API调用记录，生成API行为日志。

### 2.1.2 异常软件应用行为样本的获取

通常情况下，计算机中发现病毒的情况很少见，本文的主要目的是从正常程序中找到异常威胁程序，这里自然还需要获取一些负样本，以供学习威胁程序的特征。

在获取的方式上，一方面通过运行现有的常见病毒木马，如冰河木马，网络刺客，远控王，灰鸽子，SSS流光扫描器等等黑客工具，通过API monitor监控其API调用情况，必要时使用内存取证分析工具Volatility获取起完整的行为特征。

另一方面，为了更深入的了解恶意程序的行为特征本文尝试编写简单的恶意程序或者在恶意程序的基础上进行改写来得到而已程序，如改造Soldier【2】木马。同样利用API monitor监控程序运行，将运行时的API调用记录导出得到负样本。下表为本文用来获取恶意软件行为记录的案例程序。

表 2-1 恶意软件样本

|  |  |
| --- | --- |
| 恶意软件类别 | 测试案例 |
| 网络嗅探类 | 网络刺客，SSS扫描器 |
| 木马病毒类 | Soldier-t，冰河木马，远控王，灰鸽子 |
| 间谍软件类 | SpyTest |

表2-1中Soldier-t为开源木马Soldier的修改版，削弱了部分功能。间谍软件SpyTest为本文单独实现的简单版间谍软件。由于此类威胁程序具有破坏性，所有的操作都是在虚拟机中完成。

## 2.2 数据清洗

### 2.2.1 根据用户进程切分数据

所得到的行为日志程序有点多且庞杂，这里需要进一步将采集到的日志进行切分。为得到易于区分的行为日志，还需要将日志按照进程切分开，得到每个进程的API调用记录。

行为记录样本示例说明：

表 2-2 API行为记录格式

|  |  |
| --- | --- |
| 说明 | 键值示例 |
| 时间 | 2016-12-06 21:57:43.309 |
| 机器名 | CHRISZHAI-PC1 |
| IP | 10.123.119.77 |
| 进程id | 264172 |
| 线程id | 256668 |
| 进程名 | Explorer.exe |
| 进程文件地址 | C:\Windows\Explorer.exe |
| API | NtQueryValueKey |
| Key | \registry\machine\software\wow6432node\microsoft\ |
| Value | 360Safetray |

上表表示Windows Explorer进程的一次API调用活动，访问的API为NtQueryValueKey，即一次注册表查询。

由于最终需要区分的是恶意软件，起初得到的大量监控数据都是以电脑用户为单位，最终还需要将用户数据以进程为粒度做切分。

### 2.2.2 行为特征表示-BOW模型

使用机器学习的方式来分析处理应用软件行为很重要的一步是要解决数据的输入问题，即如何将原始数据转换为特征向量。本文在表示应用软件行为时借鉴文本处理的思路，借鉴文本建模的方法对应用软件行为进行表示。通常情况下，SOW（Set Of Word）模型和BOW（Bag Of Word）模型都是不错的选择。

表 2-3 SOW模型和BOW模型对比

|  |  |  |
| --- | --- | --- |
| 模型名称 | 表达方式 | 优缺点 |
| SOW模型 | 单词构成的集合，集合中每个元素只有一个。 | 处理数据简单快速，但是缺少表达频率信息。 |
| BOW模型 | 单词和出现次数构成的键值对。 | 表达信息更全面，但是处理稍显复杂，需要事先确定统计广度。 |

如表2-3所示，SOW虽然在处理速度有优势，但是表达的信息有缺省，相比之下，BOW模型能更多表示应用软件行为中的包含的信息，为行为分析提供更多的依据。最终本文在选择模型上选择了BOW词袋模型。

### 2.2.3 提取API词袋模型

顾名思义，词袋模型通过统计一段文本内各个API的调用次数得到一个频率字典。

例如程序A执行过程中依次调用情况如下：

API1，API1，API2，API3，API4，API4，API1，API2

得到如下的一个字典：

{

“API1”：3,

“API2”：2,

“API3”：1,

“API4”：2

}

其中，key表示所调用的API名，value为捕捉到的运行期间各个系统API调用次数，即词袋模型。词袋模型描述的是应用程序在一段时间内的API调用次数情况，普通程序在系统内核API调用情况上如下图所示：

图 2-1 应用程序API行为图

为了简化模型，API调用频率还可以进一步简化为向量作为输入,如上述的字典可以进一步转换为如下向量：

[3, 2, 1, 2]

至此，API调用记录最终转换为向量作为数据输入。

## 2.3 技术框架准备

### 2.3.1 Tensorflow机器学习框架

Tensorflow是谷歌公司在2015开源的一个机器学习基础框架，起初为谷歌大脑团队为了研究机器学习和深度神经网络而开发。目前该系统可以应用到语音识别，图像识别，自然语言处理等领域。

Tensorflow具有高灵活性，可移植等特性，能显著提高开发效率。支持主流机器学习开发语言Python和C++。

在性能上Tensorflow也有极大的优势，Tensorflow先天给予了线程，队列，异步操作等优秀的支持，是的Tensorflow能在最大程度上发挥出硬件的能力。支持GPU计算以及分布式计算，能在集群上发挥更大的潜力。同时谷歌公司也针对Tensorflow设计定制了专用的处理芯片TPU，使得Tensorflow在该平台下发挥出更佳的性能。

使用上，Tensorflow的核心是数据流图的构建，由用户完成计算流程的构建。例如要在Tensorflow实现如下函数计算逻辑:

|  |  |  |
| --- | --- | --- |
|  |  | （3.1） |

那么用户只需要构建好如下的数据流图：

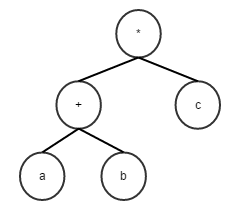


图 3-1 Tensorflow 数据流图示例

剩下的只需要往其中填充数据就可以完成计算任务。这个过程等价为机器学习中训练模型时不断的往其中输入训练所需要的数据。下图是一个典型的机器学习算法的数据流图：



图 3-2 神经网络数据流图示例

本文中关于机器学习的算法实现都是在Tensorflow框架下进行。由于Tensorflow中对于Softmax模型以及CNN卷积神经网络模型事先做了封装，在实现上非常的容易。

## 2.4 本章小结

本文研究的是从Windows程序的行为中识别出恶意具有威胁的程序，以从Windows平台采集而来的API调用记录为依据。本章主要介绍了处理的准备工作，包括训练样本的获取和数据切分以及表示模型的选择，为主要的为后续的数据训练准备好统一的格式化的向量数据。至此，恶意软件行为识别已经完成了数据输入的标准化表示步骤。另外，本章还介绍了Google开源的机器学习框架Tensorflow，也是本文实验中最核心的计算框架。

# 第三章 聚类分析威胁行为特征

## 3.1 威胁程序行为特点

### 3.1.1 Windows系统API

对于Windows平台而言，威胁程序在运行期间一般都会直接或者间接的调用系统API，这里针对一般涉及到安全威胁行为的61个Windows系统内核API，主要涉及到以下类别的API：

表 3-1 Windows系统API分类表

|  |  |
| --- | --- |
| API类别 | 介绍 |
| 文件操作类 | 包括对文件打开：NtOpenFile，创建：NtCreateFile，读取：NtReadFile，写入：NtWriteFile. . . |
| 注册表操作类 | 包括对注册表进行打开：RegOpenKeyEx，更新设置：RegSetValueKey，删除键值对：RegDeleValueKey，查询：RegQueryValueKey，删除键值：RegDeleteKey，枚举键值对：RegEnumKey 等API |
| 网络操作类 | 主要针对端口监听：Accept，绑定：Bind，连接操作：Connect，数据发送:Send等API |
| 浏览器操作类 | 包括对浏览器Cookies数据的获取和修改以及缓存数据和代理设置相关的API |
| 进程操作类 | 包括进程和线程的打开：NtOpenProcess，创建：NtCreateUserProcess，终止操作：TerminateProcess，进程权限令牌：NtAdjustPrivilegesToken等操作的API |
| 服务操作类 | 包括Windows服务的操作，例如定时任务：NetScheduleJobAdd等API |
| 其他 | 其他硬件资源的访问，如GetKeyboardStat等 |

其中，文件操作类API涉及到计算机上文件的访问情况，通过对文件操作API的监控，能有效的监控资产文件的访问情况是否正常。

网络操作类API是区分恶意软件的重要依据，木马类，蠕虫类恶意代码在网络操作上都具有非常鲜明的特点。

浏览器操作类是涉及到用户的隐私的API，这类API的访问直接涉及到用户隐私的安全。监控浏览器类API可以有效的监控隐私安全。

进程线程操作类是极为重要的API，涉及到内存访问控制，进程权限等高危操作。监控进程线程操作类API能有效的监控识别进程活动的安全性。

服务操作类API是参考价值极大的一类API，恶意程序中时常会伴随着这类API的调用。如存在潜伏期的恶意程序，有触发条件的恶意程序等。通过对这类API的监控能有效判断程序的行为是否安全。

其中，部分API在所有程序中都会有很频繁的调用，如注册表查询的API。另外部分API如清除系统日志API，调整进程权限，设置钩子函数以及硬件资源访问等都是可疑度非常高的行为。

### 3.1.2 威胁程序行为特点分析

木马：木马在宿主终端上运行后，其特征主要表现为会在宿主机上留下后门，也就是打开网络监听某个端口，从而实现远程黑客的联入。同时为了隐藏自己大都会将自身进程注入到其他进程中。

病毒：传统意义上的病毒主要以破坏为目的，落地到宿主机后，会各种抢占系统资源，如网络资源，磁盘资源，CPU以及内存等。

间谍/勒索软件：这类恶意程序有明显的利益目的，偷取宿主机重要数据或者加密重要数据勒索为主要目的。落地宿主机之后，一般都会搜索注册表查询有价值的信息，例如Office软件，代码编辑器以及PS等涉及到个人资产工作软件的工作目录。少部分此类会暴力以匹配后缀名的方式匹配宿主机上的文件。

蠕虫：这类恶意程序最大的特点是会在网络中扩散，显而易见，对网络的访问势必会相当的频繁。

需要注意的是，随着恶意软件的发展，各个类别的恶意软件已经不再单纯只是具有一方面的特点，而是可能同时具备蠕虫和木马的特点或者其他组合特点，功能上只会越来越复杂，行为上越来越诡异。

## 3.2聚类分析算法

### 3.2.1 主流聚类算法介绍

表 3-2主流聚类算法对比

|  |  |  |
| --- | --- | --- |
| 算法类型 | 优点 | 缺点 |
| 划分聚类 | 算法效率高，设计简单 | 需要事先给定类簇的数量，对异常点敏感。 |
| 层次聚类 | 能得到一次得到整个聚类过程，分多少个类簇数量都可以通过改变树聚类树来实现。 | 计算量大，执行效率低，需要事先规定聚类算法停止时间 |
| 密度聚类 | 对噪音不敏感，可发现任意形状的聚类。 | 计算量巨大，对输入参数敏感。 |

从表3-2中可知，常用的聚类算法有3中，分别是划分聚类法，层次聚类法以及密度聚类法。本文在选取聚类算法时，一方面，由于层次聚类法和密度聚类法的算法在实现上比较困难，不容易实现，同时算法复杂度太高，计算时间太长。另一方面，层次聚类算法的停止时机不容易定义，密度聚类法中参数不容易确定。最后选择了最高效，算法设计最简单的划分聚类法，其中典型的算法就是K-means聚类算法。

本文最终选择了计算量较小，实现上容易的K-means聚类算法。

### 3.2.2 K-means聚类算法

K-means聚类算法是一种无监督学习算法，主要用来计算数据的聚集情况。如下图所示，K-means算法根据点之间的距离，将点分为四个簇。

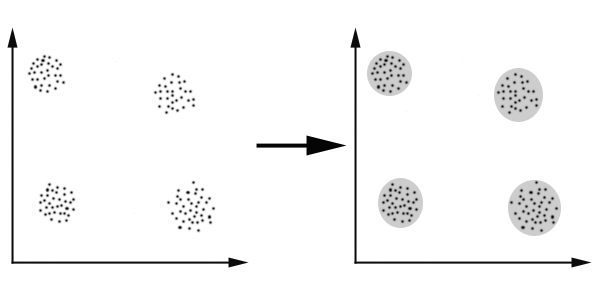


图 3-3 聚类

其算法思想如下：

1. 随机选择K个点作为类簇的中心
2. 将每个点指派到最近的中心，形成K个类簇
3. 重新计算每个类簇的中心
4. 重复执行步骤 2，3直到类簇中心不再变化

需要注意的是，K-means聚类算法需要事先直到类簇的个数k。在不确定类簇个数的情况下可以在一定范围内尝试多个k值直到得到较好的聚类效果。

时空复杂度如下表：

表 3-3 kmeans聚类算法复杂度分析

|  |  |  |
| --- | --- | --- |
| 复杂度类型 | 复杂度 | 参数说明 |
| 时间复杂度 | O(tkmn) | t为迭代次数，k为类簇的个数，m为训练集数据的条数，n为单条记录的维数。 |
| 空间复杂度 | O((m+k)n) | m为训练集数据的条数，n为单条记录的维数，k为类簇个数 |

## 3.3 使用Tensorflow聚类分析

### 3.3.1 行为相似度定义

目前主流的计算向量距离的方式有多种，对于向量[x1, x2 , … xn],有以下几种计算距离方式：

欧式距离，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （3.2） |

曼哈顿距离，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （3.3） |

向量夹角，公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （3.4） |

考虑到向量夹角接近0，但是模不相等的向量在本文场景中所代表的含义基本没差，所以本文主要主要分析了表3-1中提到的61个关键的系统API，所以这里对于每份样本，转换为一个61维的向量，以向量之间的夹角作为行为相似度的参考。

### 3.3.2 特殊API弱化处理

对于有些常用的API，如查询注册表类的API。由于出现频率太高，会直接导致本身行为很相似但是计算相似度却差别很大。

考虑到对数函数的变化特点,随着自变量的增大，函数的变化率越来越小的特点。计算公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （3.5） |

函数变化曲线如下：

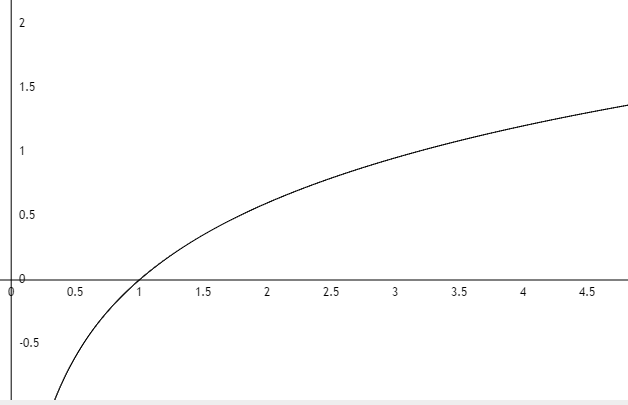


图 3-4 对数曲线变化图

如图易知，应用对数函数之后，出现词频过高的API对结果的影响会越来越小。本文对这类API在计算数量后取对数来表示词频，从而使得这类API对结果的影响越来越小。

## 3.4 本章小结

本章主要介绍了行为分类前的准备工作，首先通过聚类分析来了解数据的特点，为后续分类工作提供有力基础。其中聚类使用到的技术主要包含了google的开源机器学习系统Tensorflow以及K-means聚类算法。通过聚类分析就能发现数据的聚集情况，发现异常数据，为后续工作打下基础。

# 第四章 基于API调用特征威胁行为分类预测

## 4.1 分类类别确定

通常情况下，构建分类算法之前需要首先确定类别的数量。本文中，得益于聚类分析过程的结果，主要将应用软件行为分为以下七种：

表 4-1 应用软件分类

|  |  |
| --- | --- |
| 类别名称 | 介绍 |
| IM即时聊天应用类 | 主要为QQ，微信，阿里旺旺等软件 |
| 网络应用类 | 主要为Chorme浏览器，360浏览器，QQ浏览器，IE浏览器等网页浏览类应用 |
| 单机工具类 | 主要为Office文档编辑工具，WPS，PS等办公工具 |
| 游戏类 | 主要为LOL，Starcraft，贪吃蛇，植物大战僵尸等游戏应用 |
| 影音娱乐类 | 主要为QQ音乐播放器，网易云音乐，暴风影音，迅雷等工具 |
| 系统工具类 | 主要为360安全卫士，腾讯电脑管家，驱动精灵，鲁大师等系统辅助工具 |
| 恶意软件类 | 主要为木马，间谍软件，病毒等恶意程序 |

如表4-1所示，本文实验数据上处理了日常生活中常用的应用，通过第三章的聚类分析主要分为以上7类。

## 4.2 Softmax分类

### 4.2.1 Softmax模型

通常，实现二分类问题使用Logistic函数实现，也就是sigmoid函数，其公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.1） |

Sigmoid曲线如下：

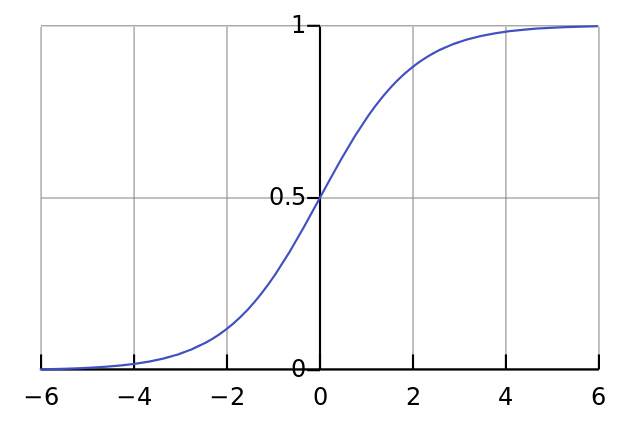


图 4-1 sigmoid函数

从曲线特性易知，该函数主要功能：对于任意实数x，sigmoid(x) (0, 1)。

本文研究的行为分类存在多种类型的程序，属于多项Logistic回归问题，使用Softmax非常合适。Softmax 定义如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.2） |

其中，公式中的分母起到了正则的作用，使得概率之和为1：

|  |  |  |
| --- | --- | --- |
|  |  | （4.3） |

对于数组V，Vi表示V中的第i项，对应的Si的值为：

|  |  |  |
| --- | --- | --- |
|  |  | （4.4） |

从功能角度来讲，使用Softmax函数可以将一个k维的向量[x1, x2, …xn]映射成[y1, y2, …yn]，其中yn (0, 1)，如下图所示：

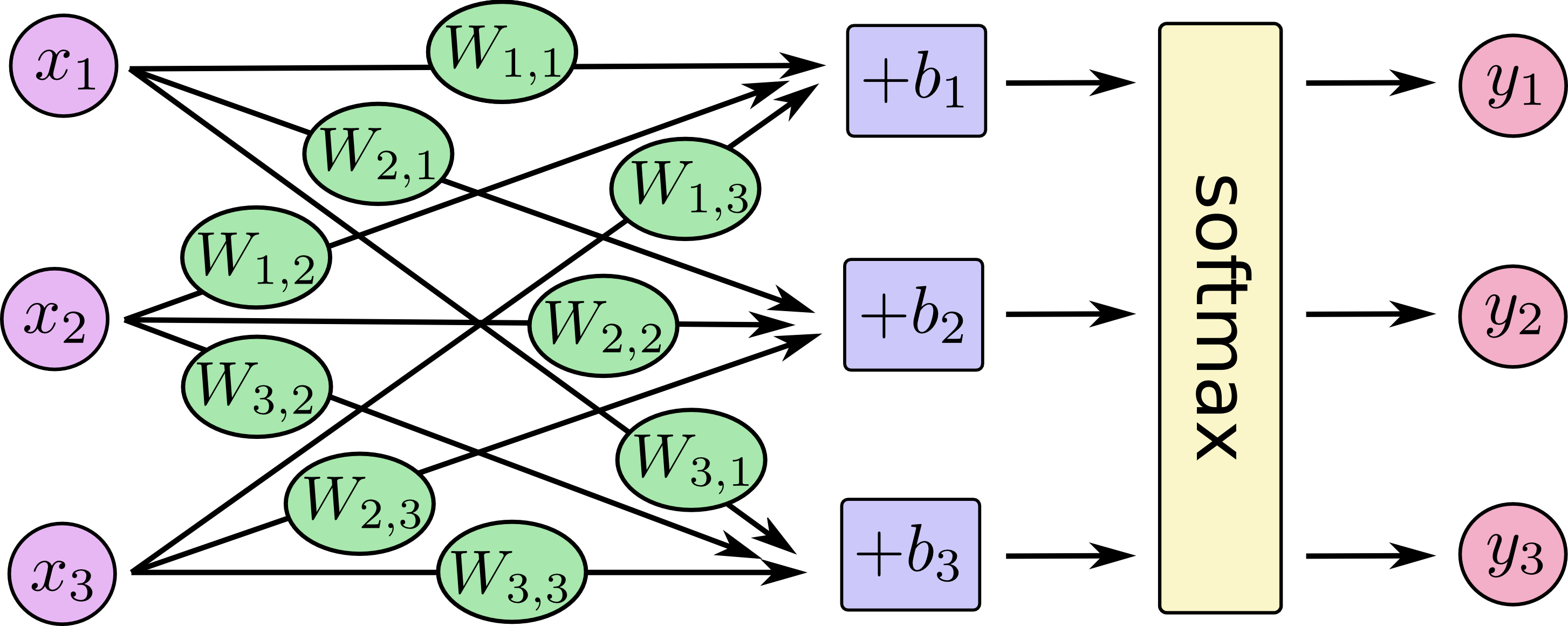


图 4-2 Softmax原理图

最终根据yi的大小进行分类。对于本文而言，其中yi表示当前输入为类簇i的可能性大小。

### 4.2.2 Softmax实现威胁行为分类

本文借助Tensorflow框架实现Softmax分类。数据集的输入如下：

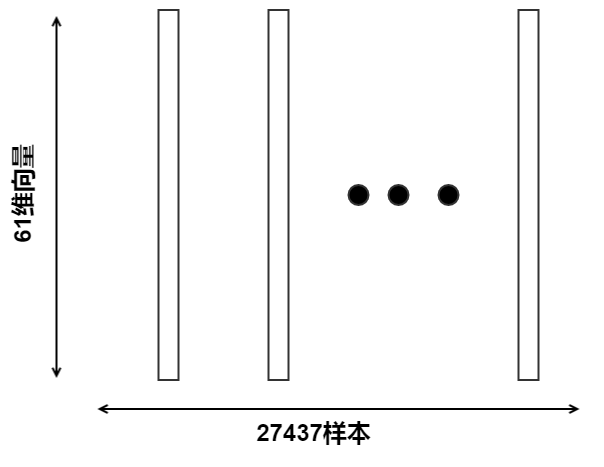


图 4-3 样本输入

由于本文中数据的表示方式是将一段行为文本转换为一个61维的向量，在图中，每一个矩形代表一个61维的向量。共输入27437个样本。

在聚类过程中，本文为了简化问题，只考虑了常用的软件，聚类时分为7类软件时聚类效果最好。所以分类的结果标签设置为如下的一个矩阵：

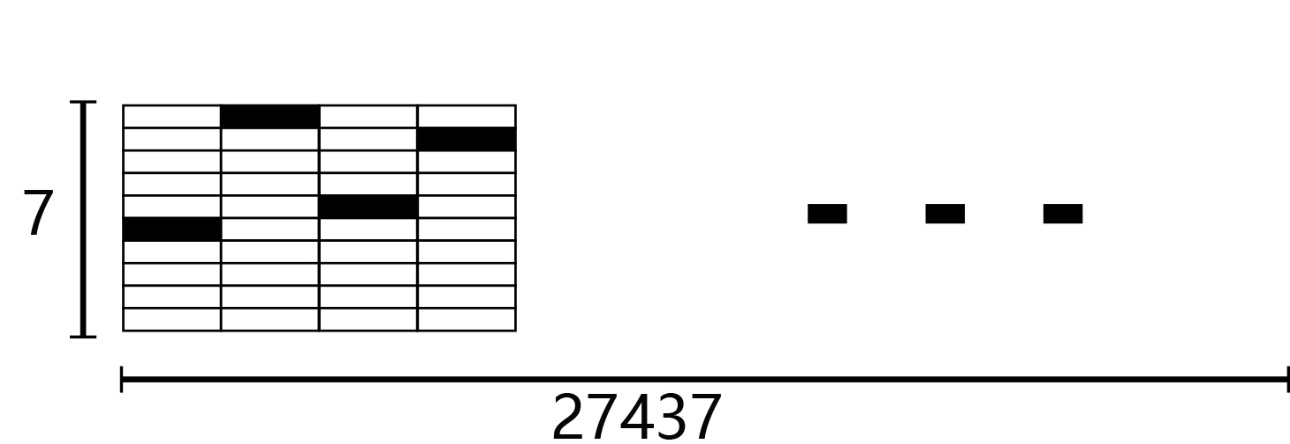


图 4-4 样本标签矩阵

回到Tensorflow实现Softmax分类，首先得构建Tensorflow数据流图，创建过程如下：



需要明确的是这里的x是一个占位符，不是一个特定的值，到需要的时候才会指定。通俗的来理解，这个x就是自变量，就会是后续需要输入的训练数据矩阵。创建占位符之后，需要确定需要训练得到的参数，即权值和偏置：



这里的W和b都会初始化为0值矩阵。W的维数是61x7，是因为需要将61维的矩阵经过训练之后映射到7个类别上的evidence 值。b时7个类别上再加上偏置。接下来实现Softmax模型，在Tensorflow中仅需一行代码：



接下来需要确定一个损失函数，用来衡量模型训练好坏的参考，本文使用交叉熵函数来描述训练结果值与真实结果值的符合程度。其函数原型如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.5） |

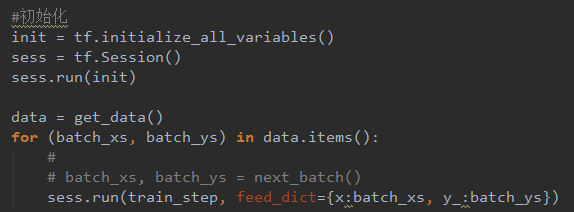
具体在Tensorflow中实现如下：



接下来以得到最小交叉熵为目标训练数据：



至此，Tensorflow数据流图构建完成，接下来只需要向其中填充数据即可完成数据的训练。在Tensorflow中首先初始化所有变量然后创建session，将操作放到session中完成训练过程。



## 4.3 CNN卷积神经网络实现分类

### 4.3.1 CNN卷积神经网络

神经网络是由多个神经元构成的层级网络，神经网络中每个单元示意图如下：



图 4-5 神经元示例

其对应的公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.6） |

其中，x1，x2表示的是输入，hw,b(x)为神经元的激活函数。多个神经元构成的具有分层结构特点的网络就是神经网络模型，如下图所示：



图 4-6 神经网络示例

其对应的公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.7） |
|  |  | （4.8） |
|  |  | （4.9） |
|  |  | （4.10） |

其中，每个神经元获得输入后，输入层将输入值乘以权值加上偏置即为第二层的神经元的输出，最后第三层将第二层的输出作为输出并作用激活函数得到第三层的输出。

以一个图像识别为例，如下图：

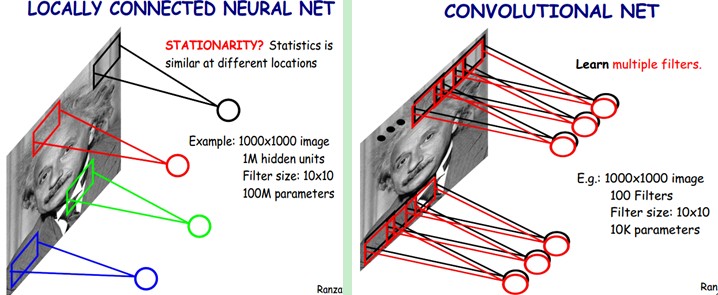


图 4-7 CNN图像识别示例

从图中可以看到多个卷积核在图上滑动，以此获得多种特征。每次局部卷积都能获取到图片某个部位的特征。在图像识别中，不同的卷积核可以代表的是不同的特征类型，如颜色通道，各种曲线轮廓特征等。

本文中所采用的CNN卷积神经网络是一种基于人工神经网络的模型，受生物自然视觉认知机制启发而来。在计算机视觉领域应用最为广泛。其核心思想是使用卷积核做特征抽取器。起初，CNN卷积神经网络是为识别二维形状而特殊设计的多层感知器，这种网络结构在应对平移变换，拉伸变换时具有高度不变的特性。本文在实践中虽然针对的问题不是图像识别，采用CNN卷积神经网络是考虑到威胁程序的行为可以转换为图像，API调用之间的联系可以与图像中的轮廓有高相似性，故采用CNN卷积神经网络。

### 4.3.2 CNN神经网络设计

本文中为了参考图像识别中的思路，本文首先将表3-1中提到的61维向量通过补位的方式将转换为64维向量，最终转换为8x8的矩阵。过程如下图所示：

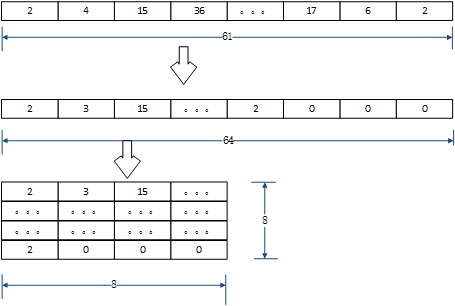


图 4-8 行为向量转换示意图

其中缺失的位置使用0填充。至此，问题转换为图像识别的思路，接下来构建神经网络，本文采用了一种简单的设计方式，仅含有两层隐藏层的CNN卷积神经网络：

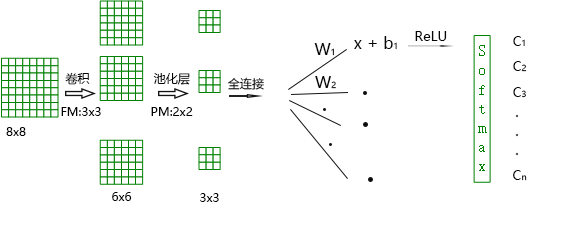
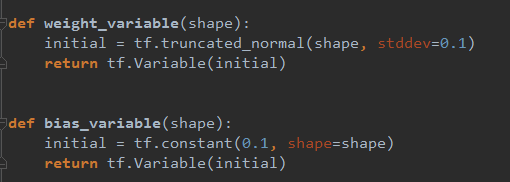


图 4-9 CNN卷积神经网络设计

如图，第一层使用3x3的卷积核对原图进行卷积，第二层为池化层，使用最大池化的方式，最后为一个全连接层。

### 4.3.3 权值初始化

本文在建立模型之前，类似Softmax模型，首先需要创建权值和偏置参数。为了破坏参数整体的对称性以及避免梯度下降法训练时梯度为0的情况出现，在权值进行初始化的时候加入了部分噪声。另一方面，本文所采用的激活函数为ReLU函数，通常情况下权值和偏置参数的初始值都会很小，本文使用下列方式完成权值参数的初始化：



### 4.3.4 ReLU激活函数

常用的激活函数有很多，如前文提到的sigmoid函数，还有tanh等函数，本文中选择ReLU函数作为激活函数。其函数公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | （4.11） |

对应的函数效果如下：

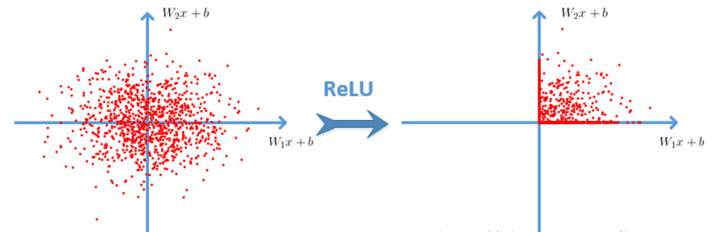


图 4-10 ReLU激活函数效果图

从图中易知，ReLU激活函数将所有非正值都映射到第一象限，对于小于0的值都映射成0，对于大于0的值都直接作为函数输出。

### 4.3.5 卷积和池化

卷积层用来产生多个局部特征，池化采样层用来减少过多的训练参数。严格上来说，这里需要训练的参数还不算太多，但是考虑到后面将会扩充的API集，还是加入了池化采样层。一方面，与传统的图像识别相比，API行为所产生的矩阵大小非常的小，后续工作中会陆续将Windows所涉及到的API都考虑在内。另一方面，考虑到API行为间联系的紧密不像图像种轮廓那样跨度大，经过几次测试之后最终使用3x3的卷积核提取特征。当选择不同大小卷积核时准确率变化如下：

图 4-11 卷积核大小和准确率关系图

如图所示，选取了三种大小的卷积核，分别是3x3，4x4以及5x5。就表现结果来看，当选取的卷积核为3x3时候准确度最高。

### 4.3.6 Dropout降低过拟合程度

Dropout即丢弃某些神经元的输出结果。本文中为了降低过拟合程度，在输出层之前使用Dropout技术。Dropout在Tensorflow中使用非常简单，得益于Tensorflow的有力支持，本文在实现时需要创建占位符来表示一个神经元的输出在Dropout操作中不被丢弃的概率。



## 4.4 本章小结

本章主要介绍了使用Softmax以及CNN卷积神经网络的方式实现威胁程序的分类，模型实现都是在Tensorflow框架下进行。其中Softmax算法是机器学习中常用来解决多项回归问题的函数，相比较其他的模型而言，Softmax有着简单高效的特点。而CNN卷积神经网络稍显复杂，这里由于时间原因，尚未构建复杂的CNN神经网络，当前实现的是一个包含卷积层，池化层和全连接层的简单卷积神经网络。通过构建CNN卷积神经网络，更深入的挖掘了API调用行为中的局部关联关系，在很大程度上增强了模型的表达能力和精确度。

在功能上二者都能实现分类，CNN卷积神经网络最后的全连接层也依靠Softmax完成。这里得益于Tensorflow对深度学习的有力支持，实现上都不是很困难。

# 第五章 实验结果与分析

## 5.1 实验环境介绍

在本文中，所采用的的计算平台配置如下：

表 5-1 硬件配置平台说明

|  |  |
| --- | --- |
| 硬件名称 | 说明 |
| CPU | Intel Xeon E5506 2.4GHz 四核 |
| 内存 | 16Gb |
| 操作系统 | Ubuntu 16.04LTS x64 |

其中，本文所使用的机器学习框架Tensorflow只能在64位系统下运行。实验期间算法对内存和CPU的占用会比较高。

## 5.2 行为聚类分析和分类预测测评

### 5.2.1 聚类分析测评

由于本文中分析的是表3-1中提到的Windows内核中关键的61个API，转换之后为一个61维的向量，在聚类分析之后得到的是一个61维的中心点，无法以图像的形式展现。测评方式是通过比对中心点所表示的API行为与实际该类簇中程序是否满足其特点。

例如，QQ类即时聊天应用，获取键盘，鼠标响应事件，网络IO类请求比较大等行为会比较频繁，同时由于聊天记录的保存，文件传送功能，本地文件IO次数也会多。对比类簇中心与单独的QQ进程活动记录相似度判断聚类效果。

### 5.2.2 Softmax分类测评

在章节4.1.2中通过Softmax模型训练后，需要对模型训练结果进行测评。由于Softmax训练后得到的是一组权值和偏置，验证时，只需要将数据与这组权值和偏置进行计算，得到该数据在各个类别上的分部概率。在Tensorflow中，本文使用argmax函数获取到概率最大的那组数据，将其与正确结果对比验证是否预测正确。

### 5.2.2 CNN卷积神经网络分类测评

和Softmax模型评测类似，通过比对预测结果与真实分类进行对比来计算分类正确率。实现上，由于CNN卷积神经网络的最后一步使用Softmax实现全连接层，在操作上也和Softmax模型类似。

## 5.3 实验结果分析

### 5.3.1 聚类结果分析

聚类分析基于尝试的思路，常见软件应用在分为表4-1中的7类时候符合程度最佳。下面介绍聚类主要产生了如下几种中心点：

图 5-1 IM类应用API图谱

这类应用主要表现在网络IO频繁，键盘事件获取很多，其他的API调用不是很多，会有一些文件读取类API调用。查看对应的日志记录，以QQ微信等聊天类应用居多。

图 5-2 单机工具类应用API图谱

从图表中可看出，创建线程很多，本地文件IO频繁，键盘事件获取较其他软件较多，查看对应的日志记录，其中以Office，代码编辑器等工作软件居多。

图 5-3 网络类应用API图谱

从上图易知，文件IO，网络IO，以及本地Cookies数据相关的API调用频繁，与浏览器类的应用行为十分相似。查看对应的日志记录，其中以浏览器，迅雷QQ下载类下载软件为主。

下面分析聚类时产生的几个异常点，包括间谍软件和木马的API行为特征：

图 5-4 间谍软件API图谱

从图中易知，这类应用特点非常明显，其中以注册表访问和本地的文件IO为主，伴随一些网络IO，行为十分诡异，查看对应的日志记录，这个类簇属于少量的几个异常类簇，对应测试使用的几个间谍软件。

图 5-5 木马API图谱

图标中是聚类时发现的几个异常点，是少有的几个脱离大多数类簇少量存在的点，其中网络IO请求频繁，搜索注册表频繁，经常访问涉及到硬件资源的API，包括键盘事件，摄像头以及录音设备访问接口，查看对应的日志，能很明显的发现是之前准备的负样本。

总的来说，聚类分析得到的结果还是令人满意，有效的区分开了不同类型程序在API行为表现上的差异，对后续分类工作提供了有力依据。同时聚类异常点的发现也标志着可疑进程的出现。

### 5.3.2 分类结果分析

无论是使用Softmax还是CNN，对于威胁行为的识别都有一定的效果，下图为两个模型在预测威胁行为上的准确率对比：

图 5-6 Softmax和CNN准确率对比图

对比Softmax 和CNN卷积神经网络，可以发现在准确率上CNN卷积神经网络会好很多。就结果分析易知，CNN卷积神经网络在很大程度上因为有局部感知的能力。Softmax不能像CNN一样挖掘出更多的相邻API的频率大小关系，直接导致了精度上赶不上CNN卷积神经网络模型。

再来单独看Softmax的实验结果，由于算法模式比较固定，在实现上几乎有没多少的优化空间，从53.7%的预测精确率来看，有一定效果，准确率上不理想，分析主要会预测错误的进程，识别错误的几类程序统计结果如下：

图 5-7 识别易错软件分布图

其中以安全杀毒类软件识别错误率最高，其次Windows Explorer进程。

对比分析杀毒类软件的API行为，由于其存在进程注入行为和设置钩子函数的操作，这类API在恶意程序中出现概率最高，因此在权值计算的时候权值非常大，直接导致这类软件通常被分类到恶意威胁程序中。

再查看分析Explorer进程，这个进程是Windows自身一个非常核心的进程，又叫Windows资源管理器，在系统中有举足轻重的地位。查看API行为日志，其中注册表查询类API以及文件IO相关API调用远超其他进程，这样以来，API调用行为上着实与间谍类软件类似。

最后来看QQ，按照之前的设想，这个分类基本不应该出错，经分析预测分类错误的情况普遍出现在QQ使用过程中使用QQ远程链接功能，语音通话或者视频通话功能的日志中，这类获取硬件信息的API在训练时权值太大，容易发生样本倾斜。

在看CNN卷积神经网络的预测正确率，虽然比Softmax略高一点，但是也不算是理想的结果。对比出错结果与Softmax类似，在数量上会少，猜想是CNN卷积神经网络局部感知能力的结果。

## 5.4 本章小结

本文旨在使用机器学习的方式识别恶意威胁软件行为，在本章中，主要介绍了实验情况，一方面，分析了正常程序和安全威胁程序聚类分析后的结果。另一方面，分析了Softmax算法和CNN卷积神经网络分类的结果。聚类分析结果印证了之前关于恶意软件行为的猜想，不同类型的软件在API行为上的表现区分度还是比较大。分类预测结果上，随不至于达到预期的结果，经过分析也能找到一部分原因，模型参数还需要进一步调优，负样本也需要进一步丰富。

# 第六章 总结与展望

## 6.1 总结

本文主要研究了基于机器学习的恶意软件行为检测的方法。本文以应用软件运行时的Windows系统内核API的调用记录为研究对象，以区分开恶意软件行为与正常软件行为为目的。对其中应用软件行为的表示方法，恶意软件行为特点和机器学习建模预测分析软件行为展开分析和研究。最后基于Tensorflow开源机器学习框架实现了软件行为聚类分析和软件行为预测分析。以下为本文的主要研究内容以及成果：

⑴ 使用腾讯公司内部iOA终端监控管理软件以及API monitor软件提取Windows平台应用软件的内核API调用记录。将得到的API调用记录应用BOW词向量模型进行建模。

⑵ 使用K-means算法对软件应用行为进行聚类分析，将BOW模型下应用软件行为表示转换为特征向量作为输入，最终得到了常见软件的聚类分析结果以及行为特征，同时也发现了异常数据的特征。

⑶ 研究了基于softmax模型和CNN卷积神经网络模型的分类方式在预测分析应用软件行为时的效果。Softmax模型在实现上简单很多，具有训练参数少，训练时间快的特点，但是在效果上差强人意。CNN卷积神经网络在实现上复杂一点，具有局部感知的能力，准确率上高过Softmax模型。

## 6.2 展望

本文虽然在实验中印证了机器学习在恶意行为识别上的有效性，但是还有很多有待完善的地方。通过对实验结果的分析，还有待完善的几点如下：

1. 使用的词袋模型，在一定程度上损失了API调用上下文中包含的信息，直接导致预测的精度有所降低。可以借鉴国外对API行为序列建模的方式，从而保留住调用的先后关系，增加模型的准确率。另一方面，词向量模型的表示方式也不失为一种有效的手段，将单个进程的API调用序列，以词向量的模式构建，然后应用情感分析中的思路，获取每个API表现在恶意软件行为中的概率，从而得到整体为恶意软件的概率。
2. 负样本太少，样本倾斜太严重，一方面，在后面的工作中需要获取更多的恶意行为记录，丰富负样本。另一方面，还需要关注更多类型的软件应用。进一步降低误判的可能性。
3. CNN卷积神经网络关于卷积核的选择还需要更多的实验来确定。神经网络的层数还需要更多的实验来确定。
4. 增加更多的Windows API监控。一方面，由于很多API是正常程序会调用而恶意程序不会调用，例如需要很多人机交互操作的环节，这样就降低了正常软件识别成恶意软件的概率。另一方面，能更全面的识别软件，从而对变种恶意软件以及新兴的0day漏洞有一定的防御能力。
5. 行为日志记录中尚且只是利用了其中的API调用记录，后续还需要关注进程，进程地址以及注册表等等信息。深入挖掘这部分信息对模型的准确率势必会有很大的积极作用。

# 致谢

时间如白驹过隙，一眨眼大学四年就匆匆结束。回顾自己的大学四年学习生涯，最想感谢的还是我的导师张剑波副教授。 张老师是我的启蒙老师，整个大学期间给予了我很多帮助和指导。大二时学习数据结构课有幸认识了张老师，后来有幸跟着张老师参与了地球化学一体化处理系统的项目中。承蒙老师厚爱，利用自己的暑假时间在该项目中完成了两个模块的开发，很大程度上熟悉了编程开发工作，期间张老师还给我很多的指导让我收益匪浅。后来大三的产学研以及大四的毕业设计，得益于张老师孜孜不倦的帮助，为我解惑给我指引方向才充实的完成了大学期间的学业。

在这里我还需要感谢信息工程学院的老师们，感谢你们大学四年的培养，教会了我很多计算机的知识，为以后的工作打下了基础。同时，我还要感谢我的同学们，给我学习上的帮助以及给我生活上带来这么多的快乐，丰富充实了整个大学时光。我还要特别感谢我的父母，亲人和朋友，没有他们在背后默默支持我，我肯定不能好好完成我的学业。由衷的感谢你们，谢谢。

# 参考文献

1. Faraz Ahmed, Haider Hameed, M. Zubair Shafiq,et al. Acm Workshop on Security & Artificial [Intelligence, 2009:55-62](http://xueshu.baidu.com/usercenter/data/journal?cmd=jump&wd=confuri%3A%2879a96aa3145672fa%29%20Acm%20Workshop%20on%20Security%20%26%20Artificial%20Intelligence&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dpublish&sort=sc_cited)
2. https://github.com/hackedteam/Soldier-win [OL].2017
3. http://www.jeyzhang.com/Tensorflow-learning-notes-2.html [OL].2017
4. 小金. 程序行为记录与跟踪 解读PC深处的黑匣子[J]. 新电脑,2007,(09):92-95.
5. 奚小溪, 孙荣会. 恶意软件的行为与检测技术分析[J]. 安徽建筑大学学报, 2012, 20(3):52-55.
6. http://www.36dsj.com/archives/24006 [OL].2017
7. https://msdn.microsoft.com/library/ [OL].2017
8. Michael A.Davis，Sean M.Bodmer，Aaron Le Master.黑客大曝光：恶意软件和Rootkit安全. 第一版[M]. 机械工业出版社.2011
9. https://github.com/jivoi/awesome-ml-for-cybersecurity [OL].2017
10. Wes Mckinney.利用Python进行数据分析.[M] 机械工业出版社.2013
11. 韩兰胜, 高昆仑, 赵保华, 等. 基于 API 函数及其参数相结合的恶意软件行为检测[J]. 计算机应用研究, 2013, 30(11): 3407-3410.
12. Aafer Y, Du W, Yin H. Droidapiminer: Mining api-level features for robust malware detection in android[C]//International Conference on Security and Privacy in Communication Systems. Springer International Publishing, 2013: 86-103.
13. Lou V. Application behavior based malware detection: U.S. Patent 7,779,472[P]. 2010-8-17.
14. 赵星, 李秦. 基于进程行为分析恶意软件的识别技术[J]. 电脑知识与技术:学术交流, 2010, 06(21):6089-6090.
15. 黄文坚，唐源.Tensorflow实战.[M].电子工业出版社.2017