分类号 _____ 密级 _____

UDC[注 1] _____

南京理工大学

NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

博 士 学 位 论 文

大规模网络传播行为中的关键问题研究

倪铭

指导教师姓名 张宏 教授

学 位 类 别 工学博士

学 科 名 称 计算机应用

研 究 方 向 信息安全，数据挖掘

论文提交日期 2016.10

注 1: 注明《国际十进分类法 UDC》的类号

博 士 学 位 论 文

大规模网络传播行为中的关键问题研究

作　　者：倪铭

指导教师：张宏 教授

南京理工大学

2016 年 10 月

Ph.D. Dissertation

Research on Key Issues of Spreading Behavior in Large Scale Networks

By

Ming Ni

Supervised by Prof. ***Hong Zhang***

Nanjing University of Science & Technology

October, 2016

声　明

本学位论文是我在导师的指导下取得的研究成果, 尽我所知, 在本学位论文中, 除了加以标注和致谢的部分外, 不包含其他人已经发表或公布过的研究成果, 也不包含我为获得任何教育机构的学位或学历而使用过的材料。与我一同工作的同事对本学位论文做出的贡献均已在论文中作了明确的说明。

研究生签名：_____　　年　　月　　日

学位论文使用授权说明

南京理工大学有权保存本学位论文的电子和纸质文档, 可以借阅或上网公布本学位论文的部分或全部内容, 可以向有关部门或机构送交并授权其保存、借阅或上网公布本学位论文的部分或全部内容。对于保密论文, 按保密的有关规定和程序处理。

研究生签名：_____　　年　　月　　日

摘 要

关键词： 无线传感网，微博网络

Abstract

Keywords: Wireless Sensor Network, Microblogs

目 录

图目录

表目录

1 绪论

随着科技的进步和 Internet 的迅速普及和不断发展，互联网正密切影响着人们的工作、学习和生活，即时通信、电子邮件、电子商务、在线交易等与人们的日常生活息息相关。2017 年 1 月，中国互联网络信息中心发布了第 39 次《中国互联网络发展状况统计报告》，截至 2016 年 12 月，我国网民规模达到 7.31 亿，普及率达 53.2%，网民规模已经相当于欧洲人口总量[1]。然而，伴随着信息化快速发展的同时，网络安全形势日益严峻：个人信息可能被木马非法盗取、个人账号和密码也可能被恶意获取、电子邮件含有病毒、即时通信工具传播病毒、网上下载的资源携带病毒、网页被植入木马以及快速兴起的社交平台都成为了黑客们的目标。恶意软件（Malware）伴随着飞速发展的信息技术也在快速的发展，成为了危害网络安全甚至社会安全的一个难题。

恶意软件本质上是各种恶意、侵入性的软件或程序代码，旨在未经用户知情和授权的情况访问计算机系统以安装和执行以达到不正当目的。恶意软件的目的往往是盗取用户的各类私密账号，从而从中获取不法利益，造成了普通用户在财力和人力上的损失。据估计，我国约有高达 10% 到 20% 的电脑被感染了各种恶意软件，造成的损失达到每年数十亿之多。国家互联网应急中心（CNCERT）发布的《2015 年中国互联网网络安全报告》[2] 指出：2015 年，共有 1978 万个 IP 地址的主机被植入木马或僵尸程序；6.4 万个服务器被境外的木马控制；移动互联网恶意程序数量近 148 万个，较 2014 年增长了 55.3%；针对境内网站的仿冒页面数量达 18 万余个，较 2014 年增长了 85.7%，其中 83.2% 的仿冒网站位于境外；工业控制系统也面临着严峻的网络安全威胁。

恶意软件已经成为影响世界各个方面的全球性问题。根据微软发布的安全情报报告，大部分的网络安全威胁均来自于恶意软件。随着影子互联网经济的兴起，恶意软件已不再是单纯用来损坏、破坏或闯入计算机网络系统，但现在主要被用黑客和犯罪分子用来谋取私利。全球在线犯罪的数量已经超过了毒品交易。由于经济利益的驱使，恶意软件呈现爆炸式的增长，现有安全厂商检测和分析能力已经达到极限。海量的新型恶意软件和日益庞大的恶意软件特征库大大增加了安全专家分析和检测的难度。传统安全软件通过特征码比对进行恶意软件查杀、拦截的方式已经远远不能满足新的网络安全态势。

综上所述，有必要研究新型、高效、可靠的方法和技术，实现对恶意软件进行自动化的检测分析，从而保护网络用户的安全。

1.1 恶意软件概述

1.1.1 恶意软件定义

目前恶意软件（Malware, Malicious Software）未有一个统一明确的定义。很多专家学者尝试通过描述其本质特征来对恶意软件做出定义。1986 年，Fred Cohen 在其博士论文中第一次对计算机病毒进行了定义[3]：病毒是可以通过符号序列来描述，并且能够在合适的环境中通过其自身的拷贝或衍生修改其他的符号序列。1983 年，Cohen 在 VAX11/750 系统上展示了类似病毒的程序，该程序能够自行安装或感染其他系统对象，这是实验性计算机病毒的诞生。McGraw 和 Morrisett[4] 将恶意软件定义为 ‘‘恶意软件是指具有恶意功能的程序，如对计算机系统的安全构成威胁、破坏计算机系统或者在未经计算机用户授权许可的情况下获取计算机系统的敏感信息’’。

1.1.2 恶意软件分类

根据使用目的和传播方式的不同，恶意软件通常包括计算机病毒、蠕虫、特洛伊木马、后门、间谍软件、垃圾广告软件等多种类型。

1) **计算机病毒（Virus）**。计算机病毒是最早被提出的恶意软件。早在 1949 年，冯 · 诺依曼在论文《复杂自动机组织论》[5] 中勾勒了计算机病毒的蓝图：‘‘一部事实上足够复杂度机器能够复制自身’’，当时距离第一台商用计算机问世还有好几年。《中华人民共和国计算机信息系统安全保护条例》中指出：‘‘计算机病毒，是指编制或者在计算机程序中插入的破坏计算机功能或者毁坏数据，影响计算机使用，并能自我复制的一组计算机指令或者程序代码”[6]。从上世纪 80 年代第一代计算机病毒诞生开始，至今已经发展了数代，代码的复杂程度越来越高、传播速度越来越快、生存能力也越来越强。

2) **蠕虫（Worm）**。蠕虫是具有自我繁殖能力，可以在无需人工干预的情况下自动利用计算机网络来传播的一种恶意软件[7]。蠕虫利用多种形式的网络来进行传播，例如电子邮件、即时通信、文件共享（P2P）、局域网和广域网等。蠕虫使用各种漏洞和方式来渗透目标设备并执行代码，这些方式可能包括诱使邮件收件人打开邮件的附件、不当的网络配置、对外公开内部计算机接入权限的网络或者操作系统和应用程序中的漏洞等。最初的蠕虫通常没有破坏功能，仅仅是利用网络进行自身的传播。但是随着网络的发展，蠕虫也开始采用和病毒类似的破坏技术，具备了病毒的可自我复制和传播的特征。

3) **特洛伊木马（Trojan Horse）**。特洛伊木马（简称木马）是伪装成合法的程序并在用户不知情的情况下隐秘运行来达到非法目的的一种恶意软件，名称来源于希腊神话《特洛伊战争》[8]。木马表面上看起是正常的应用程序，实际却能危害计算机安全甚至导致严重破坏。初期的木马仅仅是用来进行非法远程控制的黑客工具，自身

并没有传播能力。然而在现在大多数的木马都加入了计算机病毒的功能，从而使得其具有较大的威胁和破坏力。

4) **后门（Backdoor）**。后门是一种存在于目标系统中，并绕过正常授权流程对目标系统进行远程控制的恶意软件[7]。当系统被攻破时，一个甚至多个后门程序将被植入到系统中，使得黑客能够轻易的控制目标系统。通常计算机生产厂商在生产计算机时会在系统中植入后门以便为客户提供便捷的技术服务，但是这些后门从未进行可靠性验证。

5) **Rootkit**。Rootkit 是一种持久并毫无察觉地驻留在目标计算机中，长久的维持对系统进行操纵、并通过隐秘渠道收集数据的程序。Rootkit 的作用是长时间的保持对目标系统的远程控制。

6) **间谍软件（Spyware）**。间谍软件是一种在计算机用户不知情的情况下，将用户的私密信息收集并发送给攻击者的程序。间谍软件通常捆绑在正常的软件中，在安装正常软件的同时也安装了间谍软件。

事实上，目前对恶意软件的分类并不十分明确，在信息技术快速发展的今天，恶意软件的形式多样，功能清大，很多恶意软件都兼有几种类别的恶意行为。随着互联网的快速发展和各类恶意软件技术的快速成熟，新一代的恶意软件已经由过去以制作者炫耀技术为主的个人作品转变为网络罪犯们牟取私利的一种工具。在巨大的利益诱惑面前，进而逐渐发展成为了一个庞大的黑色产业链。

1.1.3 恶意软件传播介质

1.1.3.1 网络服务漏洞

在服务器上运行的网络服务为网络中的客户提供共享的资源和服务。例如，DNS 服务提供域名解析服务，文件服务器在网络上提供共享存储。许多商业化的操作系统具有已经安装并运行的各种网络服务。攻击者可以利用此类服务中的漏洞对提供服务的目标机器实施攻击。这类操作系统（例如 Microsoft Windows）的大面积安装部署为攻击者利用漏洞植入恶意软件提供了机会。这个特征使网络服务漏洞成为蠕虫感染的首选方法。此外，提供对远程用户使用密码认证进行远程系统访问的服务（例如，SSH）经常暴露于字典攻击，这样的攻击不断地尝试使用存储在字典中的密码登录系统。

1.1.3.2 网络下载

网络下载方式通常以受攻击者的网页浏览器为目标，通过浏览器程序的漏洞，从网络上获取恶意代码并在目标系统上执行。这一过程通常无需与用户进行交互。与网络服务漏洞被动接受感染所不同的是，网络下载是主动的，无法通过防火墙进行防

御。目前，常用的攻击技术有 API 误用和网页浏览器漏洞利用。攻击者可以利用特定的 API 从网络上下载任一文件，并在目标系统上运行所下载文件。广泛使用的浏览器插件给了攻击者大量的简单可行的攻击方式。

1.1.3.3 移动存储

恶意软件通过自复制将自身文件复制到任何接入到已感染系统的移动存储介质中。在复制的同时，在移动存储中创建 autorun.inf 文件已保证自身能够自动运行。

1.2 恶意软件检测与分析研究进展

1.2.1 特征码匹配技术

目前，各大安全厂商的反病毒产品主要是基于特征码匹配的。特征码是可执行程序中用作标识的唯一代码片段，通常以字节序列或者指令序列的形式表示。通过反向工程工具反编译可执行文件，信息安全专家人工测试分析反编译得到的代码以选取和确定唯一的特征码。特征码存储在特征码数据库中。基于特征码匹配的恶意软件检测在扫描文件时将搜索特征码数据库进行模式匹配，查找当前文件是否在特征码数据库中有相同或者相似的特征码，若有则判定当前文件为恶意软件，反之则判定为良性。图1.1

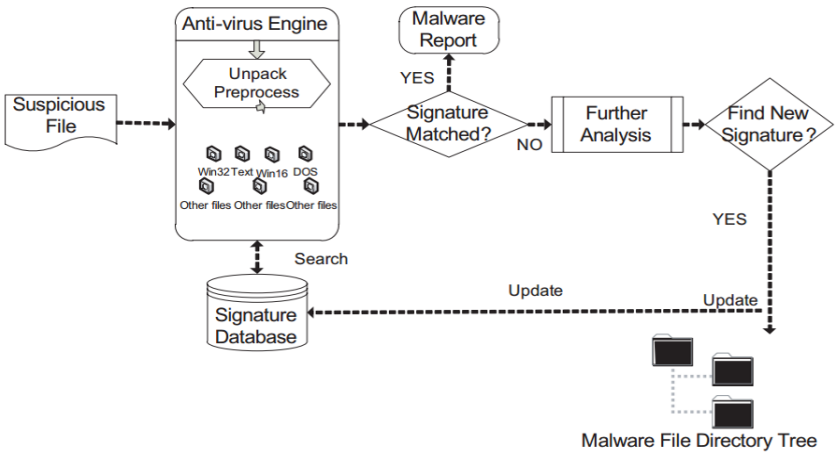


图 1.1: 特征码匹配流程

1.2.2 静态分析与检测技术

1.2.3 动态分析与检测技术

2 基于图的恶意软件静态分析方法研究

2.1 研究背景

2.2 基于图的恶意软件静态分析

本方法的体系框架如图2.1所示，由几个阶段构成。首先，反编译原始 PE 文件得到标准的汇编代码，经过预处理后抽象出控制流图（Control flow graph, CFG）。控制流图是用在编译器中的一个抽象数据结构，它是一个有向图，是过程或程序的抽象表现。在图中的每个节点代表一个基本块；每条有向边被用于代表在控制流中的跳跃，跳跃目标以一个块开始，以另一个块结束。其次，用调用的 API 或函数标记图中的每一条边，得到 FA-CFG 图；下一步，根据 FA-CFG 图建立特征向量，运用数据挖掘分类算法做出决策，判断该 PE 文件是否为恶意软件。

如图2.1所示，该方法由以下四个主要部分构成：PE 文件反编译，FA-CFG 图生成，特征提取、分类学习算法。第一步，通过反编译，得到输入 PE 文件的汇编代码；第二步，对得到的汇编代码进行预处理，移除一些不需要的指令代码，根据剩下的指令代码生成 CFG 图。根据 API 类库以及统计得到的调用函数的信息生成 FA-CFG 图。图中的一些边被对应的 API 或者调用函数的信息所标记。第三步，根据 FA-CFG 图提取特征，并采取主成分分析法（PCA）进行降维从而得到合理的特征向量。第四步，运用数据挖掘算法进行训练，形成学习模型。决策模块根据此模型判别一个输入 PE 文件是否为恶意。

2.2.1 PE 文件反编译及构建 FA-CFG 图

静态分析方法中最主要的步骤是逆向工程。在静态分析中，通过检测代码来抓取恶意软件的行为。在不执行可疑软件的基础上使用逆向工程软件或反编译器来进行分析。逆向工程是编译的反过程，是从程序中以源代码的形式获得高层次的工程性描述或者说明的过程。它可以被用来对程序的执行计划、程序结构、算法进行仔细分析。在逆向工程的过程中，底层机器代码被反编译为汇编代码，进而汇编代码被反汇编成高级语言。通过逆向工程的分析可以得到程序的函数长度、可打印字符串、基本块、指令集、控制流图、调用关系图等信息。在控制流图（CFG）中，每个节点代表一个基本块；每条有向边被用于代表在控制流中的跳跃，跳跃目标以一个块开始，以另一个块结束。在传统的控制流图中，每条边仅标识每次跳跃的开始与结束，没有反应跳跃的详细信息，包括调用的函数或 API 的信息等，因此，我们将函数调用和 API 调用产生的跳跃边用调用的详细进行标记，即可得到 FA-CFG 图。其中，函数调用所

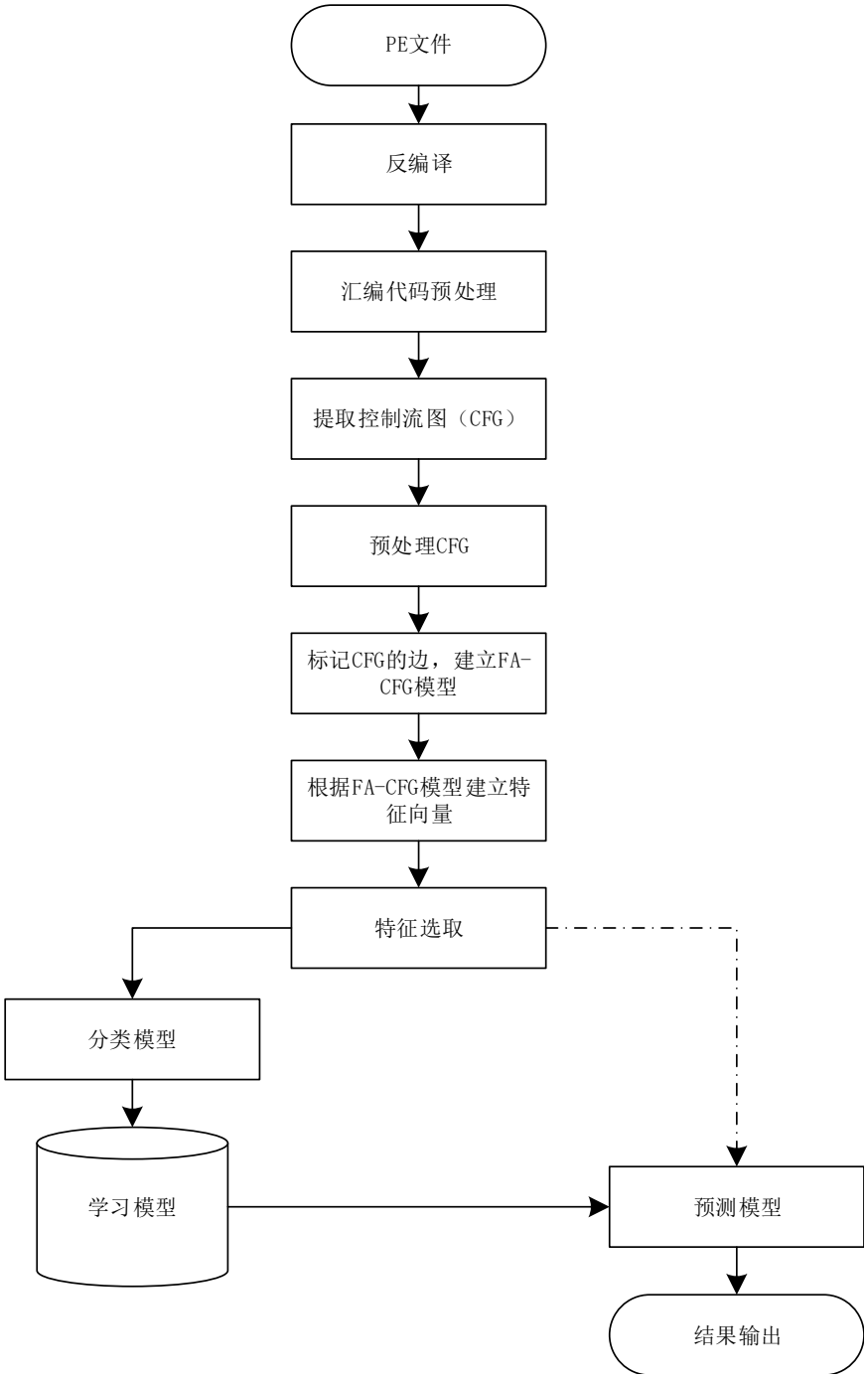


图 2.1: 体系框架图

产生的边用所对应的函数的长度频率值标记，API 调用所产生的边用对应 API 的 ID 标记。

2.2.2 函数长度频率

函数长度频率指函数代码长度在不同长度区间内的数量。将函数长度规模划分为几个不同的间隔，每个间隔称为一个容器。为每一个例子计算所有函数的长度在不同容器中的数量。由于函数长度的数量级差异，我们以指数增长的方式增加容器涵盖的范围。统计函数长度在 1 到 e 字节的数量，e 字节到 e^2 字节的数量，以此类推。

3 基于标签传播的恶意软件检测算法研究

恶意软件编程技术的快速发展已经给计算机和网络安全提出了巨大的挑战。因此，反病毒企业与市场急需发展新的有效的方法和框架来保护用户以应对更新的病毒威胁。现有的病毒软件检测技术主要将文件样本作为单一的个体，分析提取文件样本的特征，例如 API 调用序列、指令序列和二进制字符串等，再运用数据挖掘算法，如朴素贝叶斯、支持向量机等进行病毒检测。文件样本之间的关联关系隐含了大量有价值的信息，忽略他们的关联关系使得分析和检测恶意软件具有一定的局限性。本章从文件样本间关系的角度出发，研究文件样本之间的关联关系和关系类型，提出了基于标签传播的恶意软件检测模型，通过实验验证了标签传播算法在文件关系图上对检测恶意软件的准确性和有效性。

3.1 相关研究

3.2 文件关联图的构建

3.2.1 文件关联图类型

文件关联图根据关联类型以及参与者的不同而不同，

1) 文件—文件 (File-File)。在文件—文件关联图中，文件与文件之间直接建立关联，若两个文件有共存关系或者有一定的相似度，则就在图中用一条边连接这两个文件对应的节点。

2) 文件—设备 (File-Machine)。在文件—设备关联图中，与文件直接相连的是文件所存在的设备（个人电脑、计算机、手机、智能终端等）。通过设备作为过渡点，文件与文件之间建立间接的联系。

3) 文件—聚类 (File-Cluster)。在文件—聚类关联图中，通过某种相似度评价算法（例如局部敏感哈希，LSH）将文件在不同维度上进行聚类操作，因此可能会导致同一个文件在不同的维度上会被分到多个聚类中。根据文件与聚类之间的关系，建立文件—聚类关联图。通过聚类作为过渡，文件与文件之间建立间接的联系。

4) 文件—特征 (File-Feature)。在文件—特征关联图中，提取文件的特征属性并根据文件属性的紧密程度建立文件与特征间的关系。例如，对于每个 Android 程序，在 AndroidManifest.xml 需要声明改程序所需的系统操作权限，将每一种权限作为一个特征即可建立文件与特征之间的关系，即某一权限在程序中被声明则就建立程序与这个特征的联系。

后三种图均可为二部图，即图中的节点可以分为两部分，一部分为文件，另一部

分为另一种实体（即设备、聚类或特征），分属两个部分内的节点互相连接，同一部分内部的节点没有直接连接。本章采用文件—文件关系类型构建文件关联图。

3.2.2 图的构建策略

在基于图的半监督学习中，一个良好的图构建方法至关重要。在长期的理论和实践中发现，图的构建方法对最终算法的性能有着重要的影响，甚至在某些情况下构建图的方法比算法更加重要。Maier[9] 等人研究发现，对于相同的数据以及相同的聚类算法，不同的图构建方法可能会得到不同的结果。Zhu[10] 认为能够体现半监督学习假设（半监督平滑性假设、聚类假设和流行假设等）的图才是一个好图。下文列举几种图的构造方法：

1) 全连图 (Fully connected graphs)。将任意两个节点 v_i 和 v_j 用一条边连接起来所形成的图称为全连图。相似度较高的两个节点之间的边有较高的权值。全连图的优点是构造方法简单，每条边都被赋予一个权重值（一个可导的权重函数），通过求导的方法即可快速的求解。但是全连图也有明显的缺点，由于任意两个点之间都有边相连，导致了求解过程中庞大的运算量。

2) k 近邻图 (k-nearest-neighborhood graphs)。若节点 v_j 是节点 v_i 最近的 k 个近邻之一，则用一条边将这两个节点相连。k 是控制整个图密度的超参数。k 近邻图具有良好的自适应性，当样本空间的密度较大时，k 近邻图的半径就比较小；反之，当密度较小时，图的半径就比较大。在近邻图中，节点 v_i 是节点 v_j 的 k 近邻并不代表节点 v_j 也是节点 v_i 的 k 近邻。在构建 k 近邻图时，只要其中一个节点是另一个节点的 k 近邻，则就用一条边连接两个节点，因此在 k 近邻图中每个节点的度 (即关联的边数) 可能大于 k。

3) ϵ 近邻图 (ϵ-neighbors graphs)。若节点 v_i 和 v_j 的距离 $d(v_i, v_j) \leqslant \epsilon$，则用一条边来连接两个节点。超参数 ϵ 控制了近邻的半径大小，影响了构造图的连通性。

4) exp 权重图 (exp-weighted graphs)。上述三种图构造方法主要集中于图的结构构建，没有详细规范图中边权重的计算方法。而 exp 权重图提出了不同的权重计算方法。根据公式 $w_{ij} = exp(-d(v_i, v_j)^2/\sigma^2)$ 计算连接两个节点的边的权重。超参数 σ 影响了权值的衰变。拓展开来，可以为特征的每一维都指定一个超参数 σ。exp 权重图可以和上述三种构造方法配合使用完成图的构造过程。

3.2.3 构建文件关联图

在本章中，我们通过文件样本之间的共存度作为衡量文件样本相似度的标准，以此构建文件样本关联图。定义文件关联图为 $G = (V, E, W)$，其中 V 为代表文件样本的节点的集合，E 为文件样本节点间关系的集合，$e(v_i, vj) \in E$，$v_i, v_j \in V$，表示存在一条边连接节点 v_i 和 v_j，W 为每一条边的权重，即边连接的两个节点的相似度。

设 C_i 为节点 v_i 对应的文件 f_i 存在的终端集合。这里我们运用 Jaccard 相似度来衡量两个文件的共存度，公式如下：

$$sim(f_i, f_j) = \frac{|C_i \cap C_j|}{|C_i \cup C_j|}, \tag{3.1}$$

其中，$|C|$ 是集合 C 的大小，即文件存在的终端总数。可以看出，共存度的取值是一个介于 0 和 1 之间的数。“0’’ 表示两个文件之间没有共存关系，即从未同时存储在同一台终端上；反之，“1’’ 则表示两个文件之间存在完全的共存关系，可能两个文件存在某种依赖关系。

在实际应用中，文件样本及其存储信息是从实际的用户客户端中采集而来，采集所有的文件样本以及所有的共存信息是不现实、不可行的，因此我们只采集可疑的文件样本以及与已标记文件的共存关系，这就导致了未标记文件样本之间共存关系的丢失。我们使用未标记文件与已标记文件的共存关系来估算未标记文件间的相似度。令 M_i 为未标记文件 f_i 共存的已标记样本的集合，则未标记文件 f_i 和 f_j 的相似度为：

$$sim(f_i, f_j) = \sum_{m \in M_i \cap M_j} sim(m, i) * sim(m, j). \tag{3.2}$$

在3.2.2小节，我们讨论了图的构建策略和常用的几种方法。在本章中，我们使用 k 近邻图的构建策略来构建文件关联关系图，若文件样本 f_i 是文件 f_j 的 k 近邻，则用一条边连接它们，且边的权重为两者的相似度。

为清晰的说明文件关联图构建过程，这里我们选取了一个文件关联的数据集作为例子，如表3.1所示。第一列为文件的 ID；第二列为文件的标记，“0’’ 代表当前文件为未标记样本，“1’’ 代表当前文件为良性文件，“-1’’ 代表当前文件为恶意软件；第三列为当前文件样本与恶意软件样本的共存记录；第四列为当前样本与良性文件样本的共存记录；第五列为文件样本存在的终端总数。举例来说，ID 为 2 的文件样本，它是一个存在于两个终端的未标记样本，与 ID 为 4 的恶意软件样本共存于一台终端中，与 ID 为 3 和 ID 为 6 的良性文件样本分别共存于两台终端中。根据所给示例数据，我们选择 $k = 3$（选择每个节点最近的 3 个近邻）构建文件关联图，如图3.1所示。它是一个无向带权图，图中节点旁标注的数值代表了节点对应的文件样本的编号，同理，每条边上的数值为边的权重，即两节点的相似度。

3.3 标签传播算法

数据挖掘算法中，标签数据对于监督学习算法是必需的。然而在很多模式分类的实际应用中，由于样本的标记工作需要昂贵的人力、财力和时间成本，获取大量的标记样本是不实际的。因此，在通常情况下我们面对的是少量的标记样本和大量的未标记样本。如何将未标记样本和标记样本结合起来共同训练模型是近几年研究的热点。

表 3.1: 文件关联数据库范例

ID	标签	与恶意软件的共存信息	与良性文件的共存信息	客户端数量
1	0	7(1),8(1),10(1)	6(1)	2
2	0	4(1)	3(2),6(2)	2
3	1	4(1),8(1)	5(2),6(3)	4
4	-1	8(1),10(1)	3(1),6(1)	2
5	1	8(1)	3(2),6(1)	2
6	1	4(1),8(1)	3(3),5(1)	4
7	-1	10(1)	——	1
8	-1	4(1),10(1)	3(1),5(1),6(1)	3
9	0	——	3(1),5(1),6(1)	1
10	-1	4(1),7(1),8(1)	——	2

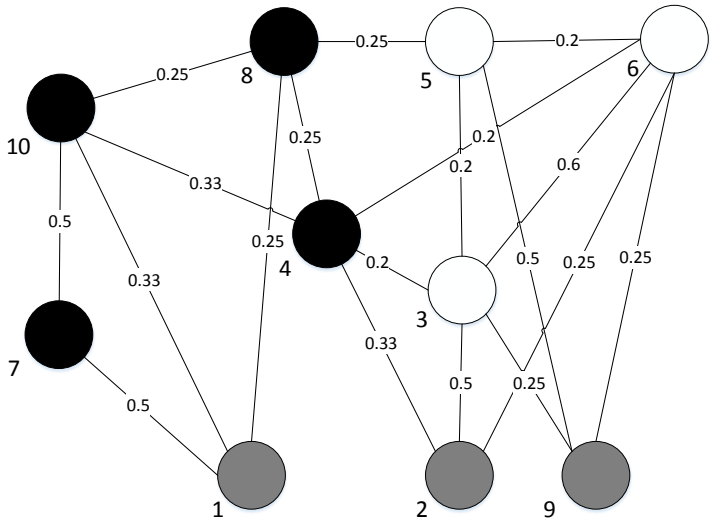


图 3.1: 基于表3.1的文件关联图示例

2002 年，Zhu 等人提出了标签传播算法（LPA，Label Propagation Algorithm）[11]，它是一种基于图的半监督学习方法，其基本思想为将已标记节点的标签信息传递给未标记节点以预测未标记节点的标签，高相似度的数据节点倾向于属于同一类别。根据样本间的关系构建关系完全图模型，其中包括已标记节点和未标记节点，节点与节点间的关系则用边来表示，节点间的相似度则表示为边的权重。在标签传递的过程中，已标记节点的标签信息根据节点间的相似度进行传递给邻接节点乃至整个图直至所有的未标记节点达到稳定的标签状态。

标签算法具体如下：设 $D_L = \{(x_1, y_1)...(x_l, y_l)\}$ 为已标记数据，其中 $\{y_1...y_l\}$ 是类别标签。假设类别总数 $|C|$ 已知，且所有的类别在已标记数据中均存在。令 $D_U =$

$\{(x_{l+1}, y_{l+1})...(x_{l+u}, y_{l+u})\}$ 为未标记数据，$\{y_{l+1}...y_{l+u}\}$ 是不可观测的，通常 $l \ll u$（即已标记数据占比很低，未标记数据的数量远大于已标记数据的数量）。该算法的最终目的即为，根据数据集 X 和已知数据标签 $\{y_1...y_l\}$ 来估计 $\{y_{l+1}...y_{l+u}\}$ 的取值。

根据数据集中的数据（包括已标记数据和未标记数据）创建一个完全连通图，其中图中每一个节点表示了数据集中的一条数据。节点 i 和 j 之间边的权重用节点间的相似度表示，如下公式：

$$w_{ij} = \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) = \exp\left(-\frac{\sum_{d=1}^{D}(x_i^d - x_j d)^2}{\sigma^2}\right), \tag{3.3}$$

其中，d_{ij} 为两点之间的距离。这里使用欧式距离计算，也可以使用其他的距离计算方法。

定义一个 $(l+U) \times C$ 标签矩阵 Y，令 Y_{ij} 为节点 x_i 被标记为类别 y_i 的概率。换言之，矩阵 Y 表示了图中每一个节点的标签概率分布。为衡量节点将标签信息传递给邻接节点的能力，定义概率传递矩阵 T，

$$T_{ij} = P(j \to i) = \frac{w_{ij}}{\sum_{k=1}^{l+u} w_{ij}}, \tag{3.4}$$

其中，T_{ij} 表示从节点 j 跳转到节点 i 的概率，即节点 j 传递标签信息给节点 i 的概率。该算法描述如下：

1) 根据公式3.3计算数据间的相似度以初始化权重矩阵 W, 并根据矩阵 W 和公式3.4计算节点 j 到 i 的传播概率，得到标签传递矩阵 T。

2) 根据已标记数据的标签信息初始化标签矩阵 Y；若节点 y_i 属于类别 C_j，则 $Y_{ij} = 1$，否则 Y_{ij} 为 0。

3) 每个节点将其标签信息传递给邻接节点，其中矩阵 $\overline{T}$ 由矩阵 T 按行归一化得到。

4) 将已标记数据的标签重置为初始值，保证原始的标签信息能够正确传播。重复上一步骤，直至收敛。

5) 根据 $y_i = \arg\max_j Y_{ij}$，为未标记节点分配标签。

算法1描述了标签传播算法 (LPA) 的具体步骤。根据已标记数据和未标记数据的排列顺序，划分行规一化矩阵 $\overline{T}$ 为如下 4 个子矩阵：

$$\overline{T} = \begin{bmatrix} \overline{T}_{ll} & \overline{T}_{lu} \\ \overline{T}_{ul} & \overline{T}_{uu} \end{bmatrix} \tag{3.5}$$

在算法迭代过程中，由于已标记数据的标签信息始终被限定为原始信息，因此标签矩阵 Y 中前 l 行即子矩阵 Y_L 保持不变，算法改变的是未标记数据的部分即子矩阵 Y_U。故算法第8行可改写为

$$Y_U \leftarrow \overline{T}_{uu} Y_U + \overline{T}_{ul} Y_L \tag{3.6}$$

算法 1 标签传播算法 (LPA)

Input: 未标记数据 $D_U = (x_{l+1}, y_{l+1})...(x_{l+u}, y_{l+u})$, 标记数据 $D_L = (x_1, y_1)...(x_l, y_l)$ 及类别集 C

Output: 未标记数据的类别

```
initial(D_U, D_L)                          ▷ 初始化相似度矩阵 W 和标签传递矩阵 T
for i = 0; i < l + u; i++ do               ▷ 初始化标签矩阵 Y
    for j = 0; j < |C|; j++ do
        Y_ij = p(y_i, C_j)
    end for
end for
repeat
    Y ← T̄Y                                 ▷ 传播标签信息
    clamp(D_L)                             ▷ 限定已标记数据
until Y converges
assignLabel(D_U)
```

经过有限次迭代后，可以证明该算法收敛至唯一值 $Y_U = (I - \overline{T}_{uu})^{-1}\overline{T}_{ul}Y_L$。

标签传播算法 (LPA) 只需利用少量的标记数据作为学习导向，利用未标记数据的内部结构和分布规律，即可传播已标记数据的标签信息以及预测未标记数据的标签。该算法操作简单、运算量小，适合大规模数据信息的挖掘和处理。

3.4 算法描述

结合3.2.3节构建的文件关联图和3.3节描述的标签传播算法，我们提出基于标签传播的恶意软件检测算法。在原始的标签传播算法中，根据原始数据点构造了一个完全连通图，在此图的基础上传播节点的标签信息以完成未标记节点的标签学习。本章提出的算法是基于文件样本之间的关联信息的，因此我们单独提出了文件关联关系图的构造策略和算法，以此改进原始标签传播算法中关系图的构造方法。算法2阐述了该方法的具体步骤。

3.5 实验验证

在本章中，我们提出了一种基于标签传播的恶意软件检测方法，本节将通过两组实验分别来验证算法的可行性和有效性。在第一组实验中，将验证 k 近邻图对于算法准确率的提升并通过实验选择最佳的 k 值；第二组实验将对比本算法与其他基准算法，以验证本算法对恶意软件检测的有效性。

算法 2 基于标签传播的恶意软件检测算法

Input: 原始文件列表数据 $FileList$

Output: 文件样本的标签

1: **for** each f_i in $FileList$ **do** ▷ 计算每个文件与已知关联文件之间的相似度
2: $similaityASS(f_i)$
3: **end for**
4: **for** each f_i in $FileList$ **do** ▷ 计算未标记文件样本之间的相似度
5: **for** each f_j in $FileList$ **do**
6: **if** f_i is unlabled and f_j is unlabled **then**
7: $similaityUnLabel(f_i, f_j)$
8: **end if**
9: **end for**
10: **end for**
11: $createGraph(V, E, W, k)$ ▷ 构造 k 近邻图
12: 根据算法1所述的算法进行标签传播
13: 根据标签矩阵标记未标记文件样本的标签

本章中实验的实验环境基于一台普通 PC，设备配置为 Intel Core i7 2.7G Hz Duo CPU、8GB RAM 和 500GB 硬盘存储，搭建的实验平台为 Windows 8 操作系统以及 JAVA 1.7 编程环境。实验过程中使用的性能评价指标如下：

- **True Positive(TP)**: 被正确标记为恶意的样本数量。
- **True Negative(TN)**: 被正确标记为良性的样本数量。
- **False Positive(FP)**: 被错误标记为恶意的样本数量。
- **False Negative(FN)**: 被错误标记为良性的样本数量。
- **TP Rate(TPR)**: $\frac{TP}{TP+FN}$
- **FP Rate(FPR)**: $\frac{FP}{TN+FP}$
- **Accuracy(ACC)**: $\frac{TP+TN}{TP+TN+FP+FN}$

3.5.1 实验数据

本节介绍实验中所用数据集的相关信息。该数据集由某安全软件企业从真实客户中收集而来，包含 69,165 个文件样本（其中包括 3,095 个恶意软件、22,583 个良性文件以及 43,487 个未知文件）以及这些样本之间的关联关系[12]。图3.2为文件关联数

据库的结构，包括 8 个字段：文件 ID，文件标签 (“1” 表示当前文件为良性文件, “-1’’ 表示当前文件为恶意软件, “0’’ 代表当前文件为未知文件)，文件名，与当前文件共存的恶意软件数量，与当前文件共存的恶意软件 ID 列表，与当前文件共存的良性文件数量，与当前文件共存的良性文件 ID 列表，文件存储的客户端数量。

id	file_sort	file_md5crc	ref_black_count	ref_black_ids	ref_white_cour	ref_white_ids	ref_file_count
1	-1	5841481 7dbd783…	10	19821:1,19822:1,19837:1,…	14	138:1,140:1,141:1,14535:1,3177:1,32…	00000000002
2	-1	c3baaf5afa8cba8…	9	1:1,13980:1,18575:1,1857…	313	10198:1,10927:1,10930:1,11:1,11276…	00000000010
3	1	6b967b59d4d6a4…	441	1002:2,1003:1,10243:1,10…	6047	10:121,1000:4,1001:1,10029:3,10031…	00000000351
4	1	b786825902bd49…	78	13939:1,14811:1,16171:1,…	456	10:15,10183:1,10198:1,10268:1,1142…	00000000034
5	1	a8dc6cc4115c0d5…	47	11906:1,14340:1,15381:1,…	538	10:9,10055:1,10198:1,1028:1,10282:…	00000000027
6	1	41d5501224adae…	594	1002:1,10064:1,10189:1,1…	6420	10:159,1000:1,10022:1,10024:1,1002…	00000000394
7	1	0043cbcf4410bb3…	302	10505:2,10634:3,10635:1,…	3666	10:55,10022:1,10025:1,10056:1,1018…	00000000141
8	1	6929f9ff15a8f3b0…	1069	1002:1,10033:1,10062:1,1…	10644	10:382,1000:6,10020:7,10021:1,1002…	00000001276
9	1	90b16c00d94e7f7…	581	1002:1,10062:1,10063:1,1…	6790	10:196,1000:4,10020:2,10023:2,1002…	00000000644
10	1	7763b669cda651…	913	1002:1,1003:1,10064:1,10…	8671	1000:5,1001:1,10020:2,10022:1,1002…	00000000897
11	1	d0cb8f4df3fa8b1…	64	11029:1,12305:1,14573:1,…	1359	10:16,10023:1,10024:1,10027:1,1003…	00000000040
12	1	0020553f141db51…	285	10505:2,10634:3,10635:1,…	3428	10:59,10022:1,10056:1,10186:2,1023…	00000000134
13	1	a8c7a7c0bca0e16…	29	11029:1,12728:1,15471:1,…	695	10:24,10029:1,10183:1,10241:1,1024…	00000000030

图 3.2: 文件关联关系数据库

通过对数据集的分析发现，良性文件的数量远比恶意软件大的多，因此导致了数据分布的不均衡，当然，不仅仅是文件样本数量分布的不均衡，文件之间的关联关系也存在着不均衡分布。图3.3和图3.4分别汇总了恶意软件与恶意软件之间的共存关系以及恶意软件与良性文件之间的共存关系。

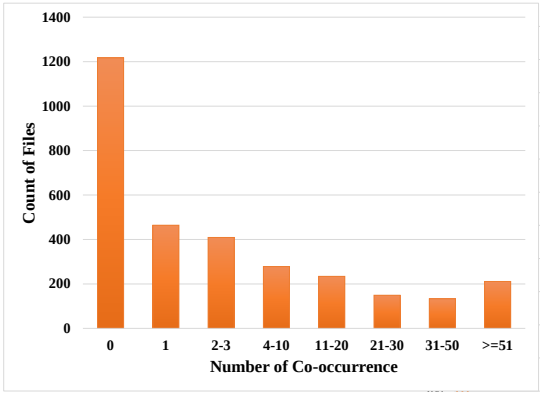


图 3.3: 恶意软件与恶意软件之间的共存关系

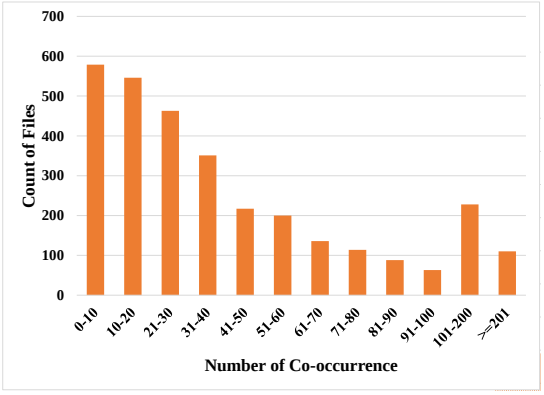


图 3.4: 恶意软件与良性文件之间的共存关系

在本章的实验中，我们从已标记文件样本中随机抽取了 3,986 个文件（其中包括 289 个恶意软件，3,697 个良性文件）作为测试集，剩余样本作为训练集。

3.5.2 kNN 图构建性能对比分析

在构建文件关联图时，我们使用了 k 近邻策略过滤数据噪声并选择每个文件样本最相似的邻居样本。本节选择与全连图、ϵ 近邻图进行实验对比，验证 k 近邻图的有效性。这里分别选择 $\epsilon = 0.01$、$\epsilon = 0.02$、$\epsilon = 0.05$、$\epsilon = 0.07$ 和 $\epsilon = 0.1$ 以及 $k = 10$、$k = 30$、$k = 50$、$k = 70$ 和 $k = 100$ 作为构图参数。通过表3.2和图3.5可以看出，运用 k 近邻策略能够明显的提升算法的有效性和准确率。当 $k = 50$ 时，对比全连图策略，k 近邻策略在 TP(True Positive) 和 TN(True Negative) 测度上分别能够获得 41% 和 6.9% 的提高，准确率则可以提升 7.96%。

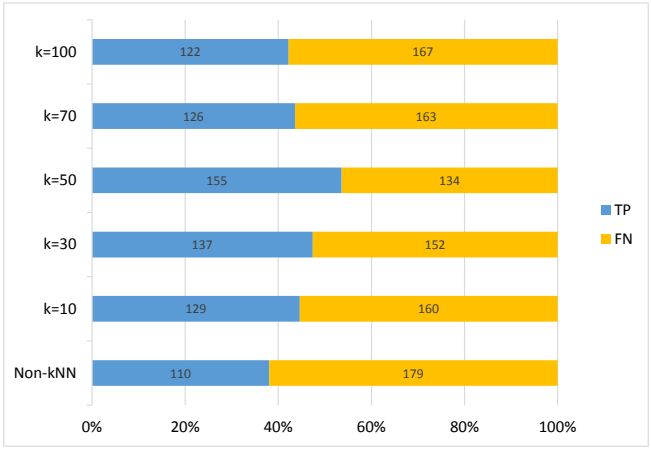
表 3.2: k 近邻图性能对比

	TP	FP	TN	FN	TPR	FPR	ACC
全连图	110	301	3,396	179	0.38	0.08	0.8796
$k = 10$	109	224	3,473	180	0.38	0.06	0.8986
$k = 30$	113	179	3,518	176	0.39	0.05	0.9109
$k = 50$	155	67	3,630	134	**0.54**	0.02	**0.9496**
$k = 70$	126	139	3,558	163	0.44	0.04	0.9242
$k = 100$	122	199	3,498	167	0.42	0.05	0.9082

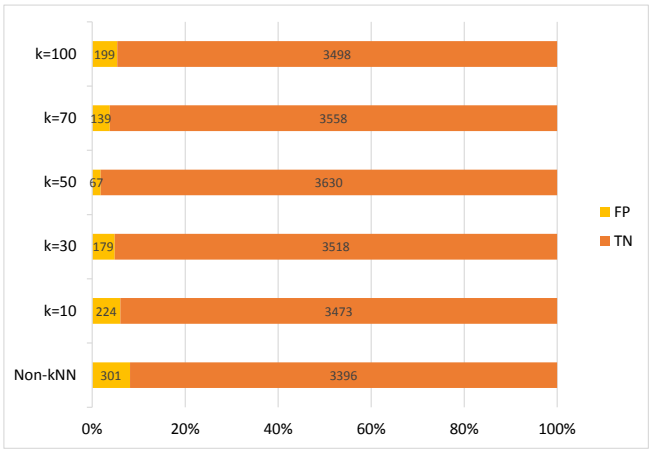
同时，对比表3.3和表3.2的结果，可以看出 k 近邻策略的在每个测度上普遍都优于 ϵ 近邻图的测度。尤其当 k 取值为 50 时，算法的准确率明显的高于 ϵ 近邻图的最优情况。以上实验也验证了 k 取值为 50 时本算法能够取得最优性能。

3.5.3 算法性能分析

本节将本章提出的恶意软件检测算法与现有的四个恶意软件检测算法进行比较，进一步验证算法的可行性和有效性。四个基准比较算法分别为：AESOP[13]、Malware Distributor Detector(MDD)[14]、支持向量机 (Support Vector Machine, SVM) 以及随机森林 (Random Forest, RF)。其中，AESOP 算法和 Malware Distributor Detector 算法同样是基于图的算法，因此可以根据文件的关联关系构造算法所需的关联关系图；而支持向量机算法和随机森林算法是基于特征属性的训练算法，因此我们定义每个文件样



(a) TP VS FN



(b) TN VS FP

图 3.5: k 近邻图性能对比

本的关联属性向量作为训练算法的特征向量：

$$R_{v_i} = \langle v_{1i}, v_{2i}, ..., v_{ni} \rangle \tag{3.7}$$

其中，

$$v_{ji} = \begin{cases} 1 & (v_j, v_i) \in E \\ 0 & \text{其他} \end{cases} \tag{3.8}$$

根据上一组实验结果的分析，在构建文件关联关系图时，我们取 k 值为 50。

表 3.3: 三种构图策略对比

	TP	FP	TN	FN	TPR	FPR	ACC
全连图	110	301	3,396	179	0.38	0.08	0.8796
$\epsilon = 0.01$	106	246	3,451	183	0.37	0.8924	
$\epsilon = 0.02$	115	213	3,484	174	0.40	0.06	0.9029
$\epsilon = 0.05$	121	184	3,513	168	0.42	0.05	0.9117
$\epsilon = 0.07$	132	140	3,557	157	0.46	0.04	0.9255
$\epsilon = 0.1$	125	174	3,523	164	0.43	0.05	0.9152
$k = 50$	155	67	3,630	134	**0.54**	0.02	**0.9496**

3.5.3.1 预测

从表3.4和图3.6中可以看出，对比其他几种基准方法，本章提出的基于标签传播的恶意软件检测算法能够有效的发现数据集中的恶意文件样本，在大规模的现实数据中具有较好的表现。

表 3.4: 在大规模现实数据上算法的性能对比

	TP	FP	TN	FN	TPR	FPR	ACC
Label Propagation	155	67	3,630	134	0.54	0.02	**0.9496**
AESOP	198	3,385	312	91	0.69	0.92	0.1279
MDD	98	2,493	1,204	191	0.34	0.67	0.3266
SVM	81	461	3,236	208	0.28	0.12	0.8321
RF	69	398	3,299	220	0.24	0.11	0.8449

3.5.3.2 交叉验证

交叉验证 (Cross Validation) 是指在某种意义下将原始数据进行分组, 一部分做为训练集 (Train set)，另一部分做为验证集 (Test set)，首先用训练集对分类器进行训练，再利用测试集来测试训练得到的模型 (Model)，以此来做为评价分类器的性能指标。

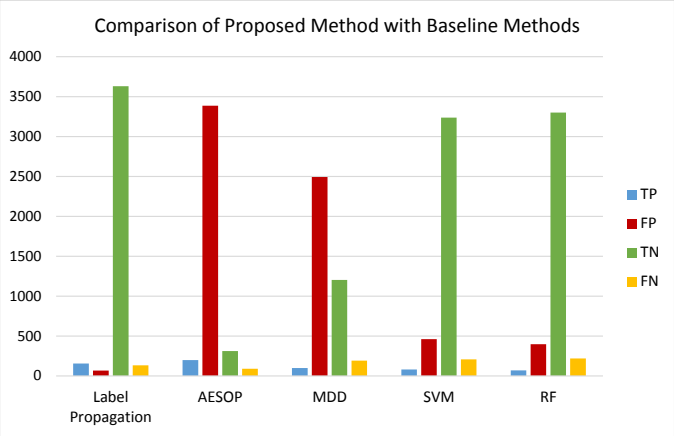


图 3.6: 在大规模现实数据上算法的性能对比

通过大量数据集以及不同学习技术进行的大量试验表明 10 折是获得相对较好误差估计的恰当选择，因此在本实验中使用 10 折交叉验证 (10-fold Cross Validation)，将数据集分成十份，轮流将其中的 9 份作为训练数据，剩余 1 份作为测试数据，进行试验。每次试验都会得出相应的正确率（或差错率）。10 次的结果的正确率（或差错率）的平均值作为对算法精度的估计。在每一次验证中，将测试集的标签设为 0(即未标记)，标签概率均为 0.5(即样本被预测为恶意软件和良性文件的概率是相等的)。基于 10 折交叉验证的数据结果对比如盒图3.7所示，图中每个数据盒子中的缺口代表了中位数的 95% 置信区间。从图中可以看出，本章提出的基于标签传播的恶意软件检测算法与其他算法相比在检测准确率上能够获得明显的改进和提升。

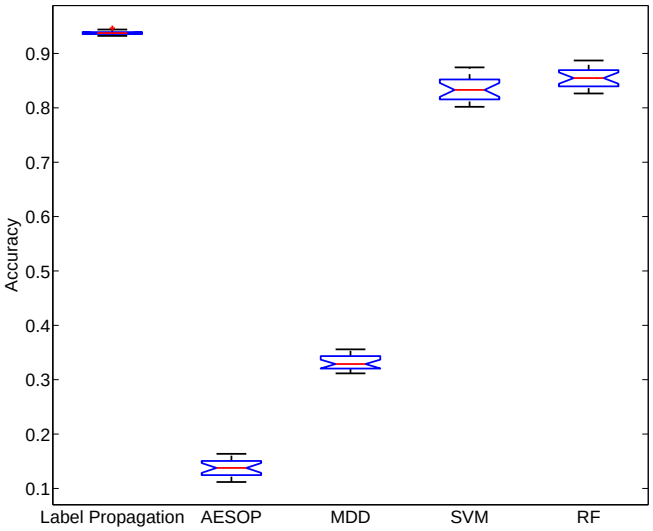


图 3.7: 10 折交叉验证准确率对比

3.6 本章小结

本章主要研究了文件样本之间关联关系在恶意软件检测中的相关应用，提出了一种基于标签传播的恶意软件检测方法。采用共存关系作为文件样本之间的关联关系，运用了 Jaccard 相似度算法来衡量文件样本之间的相似度，在此基础上通过选取每个文件样本的 k 个近邻作为邻接节点来构建文件样本的关联关系图。标签传播算法是一种将已标记节点的标签信息传递给未标记节点的基于图的半监督学习算法。在文件关联图的基础上，利用标签传播算法学习未标记文件样本的标签信息，发现恶意软件样本。将本章中的算法应用于一个从工业界采集得来的大规模真实数据进行实验，通过与真实的数据进行对比，证明提出的算法具有较高的准确性，能够精准的发现新的恶意软件样本。

在后续的研究工作中，需要进一步研究基于图的恶意软件检测算法，分析文件样本的社交关系以及改进基于图的半监督学习算法，提高恶意软件检测的准确率和可靠性。

4 基于文件社交网络的恶意软件检测方法研究

4.1 相关研究

4.2 系统架构

基于文件社交网络的恶意软件检测系统架构包含四个主要模块，如图4.1所示。文件列表采集模块 (*File List Collector*) 从用户的终端设备（包括个人电脑，手机和平板等智能设备）采集文件样本，同时提取出文件样本间的关联关系，并转存文件列表以及每个文件的关联关系至文件关系数据库 (*File Relation Database*) 中。文件关联图构造模块 (*File Relation Graph Constructor*) 根据关系数据库构造能够表示文件之间关联关系的文件关联图，在系统框架中起重要的作用。在文件关联图的基础之上，图特征提取和采样模块 (*Graph-based Feature Extractor*) 分析基于图的特征属性，在这些特征属性的基础之上选择一些具有重要的代表性的节点（对应的文件样本）进行人工标记，从图属性的层面提高分类器的准确率水平，使得模型更加健壮。作为系统结构的核心，基于主动学习 (Active Learning) 的标签传播分类器 (*Label Propagation Classifier*) 根据文件关联图学习未标记样本的标签信息（即恶意或者良性），在学习的过程中主动学习 (Active Learning) 选择重要的文件样本进行人工标记从而在算法层面上提高标签传播分类器的性能。

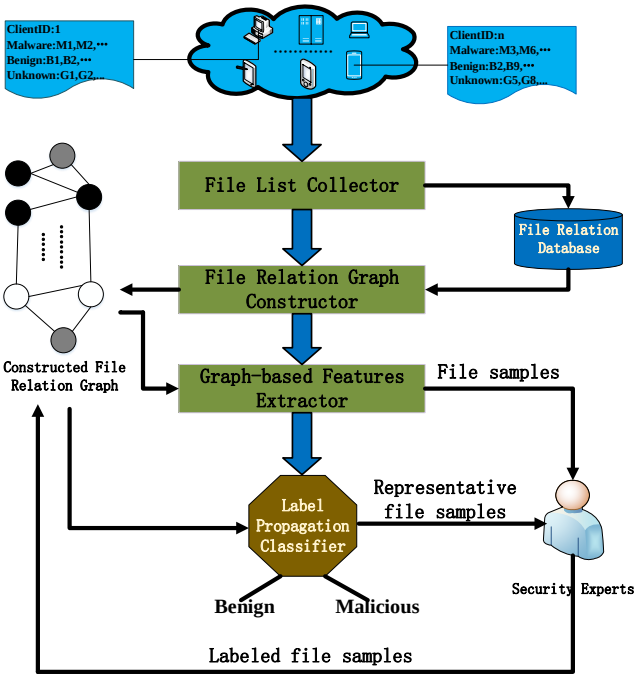
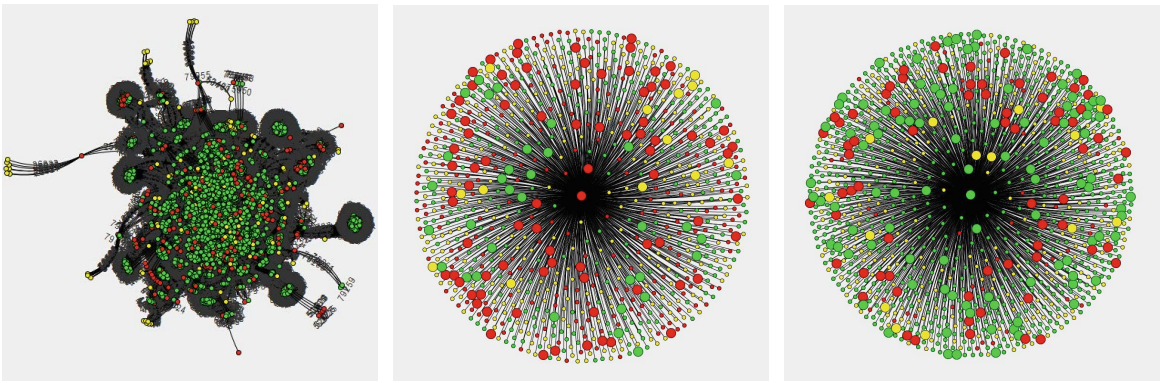


图 4.1: 总体框架

4.3 文件样本社交网络分析

4.3.1 文件关联图

本章采用上一章中文件关联图的构建算法 (k 近邻图) 构建文件关联图，图4.2(a)为构建的文件关联图，图4.2(b)和图4.2(c)分别展示了其中一个恶意文件样本和良性文件样本及其它们一跳范围内的关联信息。其中，红色点为恶意软件，绿色点为良性文件，黄色点为未标记样本。



(a) 文件关联图 (b) 恶意软件关联 (c) 良性文件关联

图 4.2: 文件关联图可视化

4.3.2 文件关联网络的特征属性

4.3.2.1 节点度

在图论中，节点度是指与节点相关联的边的数量，又称为关联度。节点度能够明确的表示节点与其邻居的关联性[15]。在文件关联图中，提出节点的恶意软件度 (DoM) 和节点的良性文件度 (DoB) 来分别衡量文件与不同类型文件的关联关系，根据公式分别计算：

$$DoM(v) = |\delta_m^v|, DoB(v) = |\delta_b^v| \tag{4.1}$$

其中，$|\delta_m^v|$ 是节点 v 恶意邻居节点的数量，$|\delta_b^v|$ 是节点 v 良性邻居节点的数量。古人云：近朱者赤，近墨者黑。同样的道理，在计算机网络中恶意软件的 DoM 将比 DoB 大，反之亦然。

4.3.2.2 局部聚类系数

聚类系数表示一个图形中节点聚集程度的系数，用于描述网络中节点和其邻居节点之间互相连接的紧密程度，即网络的集团化程度。聚类系数主要分为全局聚类系数、局部聚类系数和平均聚类系数。局部聚类系数表示一个节点的相邻节点形成一个团（完全图）的紧密程度，故本章采用局部聚类系数来计算文件关联图中节点的聚集

能力，计算方法为[16]：

$$LCC(v) = \frac{2\,|e^v|}{d_v(d_v-1)}, \tag{4.2}$$

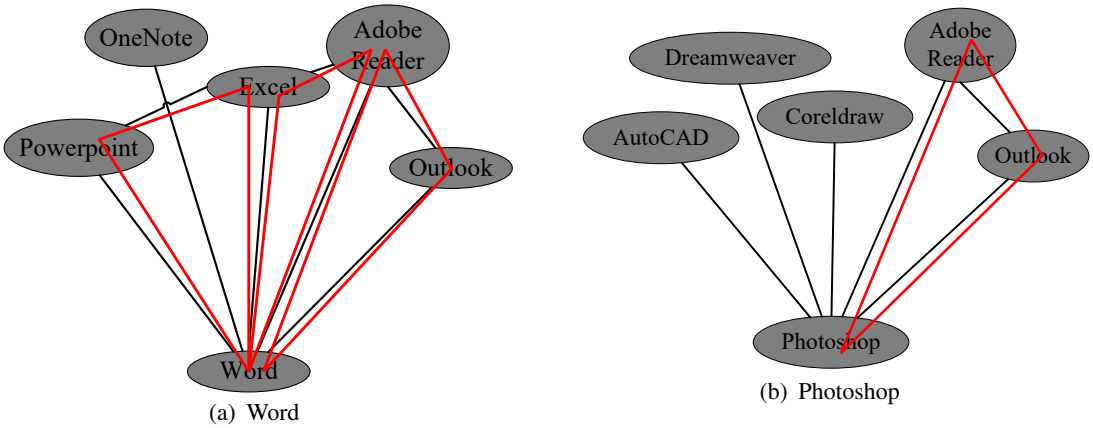
其中 $|e^v|$ 为节点 v 的所有邻居节点之间的边的总数，d_v 为节点 v 的度。这里的局部聚类系数 LCC 在计算时考虑了节点度以及邻居节点之间的边，能够较好的表达无权图中节点的局部紧密程度。而在带权图中，每条边上的权重衡量了每对节点之间的相似度，而在公式4.2的计算中每个邻居节点都被平等对待，没有考虑每对节点之间的相似程度，即两点之间边的权重，因此在带权图中，计算节点的局部聚类系数时需要根据连接至节点的边的权重来计算。Jukka-Pekka Onnela 等人[17] 提出了一种基于子图强度的带权图局部聚类系数计算方法，定义为子图中边权重的几何均值：

$$LCC(v) = \frac{1}{d_v(d_v-1)} \sum_{i,j\in N(v)} (\hat{w}_{vi}\hat{w}_{vj}\hat{w}_{ij})^{1/3}, \tag{4.3}$$

其中，$N(v)$ 是节点 v 的邻居节点集合，$\hat{w}_{vi}$ 是经过公式4.4归一化的边权重。

$$\hat{w}_{vi} = \frac{w_{vi}}{max(w)} \tag{4.4}$$

每个人对于电脑、手机等智能设备都有不同的用途和使用习惯，因此也会安装或者拷贝不同类型的程序和文件。举例来说，办公室文员经常需要使用 Word、Excel 等办公事务处理类的应用软件，而这一类的软件彼此之间也都会有较高的关联性和相似度，这就使得这些软件可以被划分为具有高关联性和相似度的群体，因此办公软件 “’Word’’ 会拥有较大的局部聚类系数 LCC。类似的，在恶意软件中，病毒 ‘‘下载者”(Trojan-Downloader) 通过用户的计算机从其作者指定的 URL 地址下载一个或多个的病毒文件并在本地运行，这些下载得到的病毒文件以及对应的 ‘‘下载者’’ 病毒有较高的共存关系和相似性，所以 ‘‘下载者’’ 病毒的局部聚类系数会较大。图4.3对比了应用程序 “Word’’ 和 “Photoshop’’ 在局部聚类系数属性上的不同。



(a) Word (b) Photoshop

图 4.3: “Word’’ 和 “Photoshop’’ 局部聚类系数对比

4.3.2.3 度中心性

度中心性 (Degree Centrality) 是在网络分析中刻画节点中心性（Centrality）的度量指标。一个节点与越多的节点发生直接联系，节点的节点度就越大，就意味着节点的度中心性越高，那么这个节点就处于中心地位，在网络中就越重要[18]。

$$DC(v) = \frac{d_v}{n-1} \tag{4.5}$$

其中，$n-1$ 为节点 v 可能连接的最大节点数，n 为图中节点的数量。

4.3.2.4 接近中心性

接近中心性反应了节点在网络中与其他节点之间的接近程度，体现了节点对图的全局认识程度。具有较高接近中心性的节点能够比其他节点更快速的触及整个图。

$$CC(v) = \frac{n-1}{\sum_{u \neq v}^{n} g(u,v)} \tag{4.6}$$

其中，$g(u,v)$ 是节点 u 和 v 之间的最短路径，n 为图中节点的数量。

4.3.2.5 中间中心性

中间中心性体现了一个节点在图中的地位。中间中心性的概念由 Linton 于 1977 年提出，用来衡量一个人在社交网络中控制他人信息交流的能力[19]。如果一个节点处在许多节点互联的路径上，可以认为此节点处于重要地位，因为该节点具有控制其他节点交互的能力，其他节点之间的交互需要通过该节点才可以进行。出现在很多其他节点之间最短路径上的节点将比其他的节点拥有更大的中间中心性。节点 v 的中间中心性可以根据公式4.7计算，δ_{st} 是从节点 s 到节点 t 的最短路径总数，$\delta_{st}(v)$ 为从节点 s 到节点 t 的最短路径中经过节点 v 的路径个数，n 为图中节点的总数。

$$BC(v) = \frac{1}{(n-1)(n-2)} \sum_{s \neq v \neq t \in V} \frac{\delta_{st}(v)}{\delta_{st}} \tag{4.7}$$

4.3.3 基于图属性的文件采样

根据文件社交网络的关系以及每个文件所处位置可以看出，每个文件的重要性是不同的，重要性较高文件的邻居通过这些文件产生关联关系。因此，根据图的特征属性从大量的未标记文件集中选取具有代表性的重要的文件进行标记对于提高恶意软件检测的准确率和性能是十分重要的。本章选择度中心性 (DC)、局部聚集系数 (LCC) 和接近中心性 (CC) 作为衡量文件重要性的因子，定义文件的重要性为：

$$importance(v_i) = \frac{DC(v_i) * LCC(v_i)}{CC(v_i)} \tag{4.8}$$

该值越大，文件的重要性越大。根据重要性对未标记文件进行降序排序，选取前 k 个文件作为采样样本交由安全专家进行标记并加入已标记样本中来训练分类器。

4.4 主动学习

在传统的数据挖掘算法中，学习算法以给定的已标记样本数据集作为训练集进行训练学习。然而在很多现实的应用中，往往面临的情况是大量的未标记样本和极少量的已标记样本，而且未标记样本比已标记样本更加容易获取。对样本进行标记工作的成本是高昂的和困难的，需要花费大量的人力、物力和财力。Zhu 等人研究表明[10]，对于训练样本的准确标记不仅需要数据的领域知识和领域专家，而且样本标记工作所需的时间是获取数据所花费时间的 10 倍以上。根据 PAC 学习理论，已标记样本的数量越多，算法的泛化误差越小，因此稀少的已标记样本使得现有的监督学习算法的应用能力大大降低。主动学习 (Active Learning) 算法被提出以有效地解决这类问题。主动学习算法模拟了人类的学习过程，通过选择重要的未标记样本进行标记并加入训练集，迭代学习来提高分类器的学习能力。

4.4.1 相关概念

主动学习是最早由 Angluin 等人在 1988 年提出[20]，其主要的思想根据数据分布情况，主动选择对分类器模型有重要意义的样本，交由领域专家进行人工标注并把这些样本加入到已标注的训练集中，通过反复的迭代学习训练，使得分类器拥有更好的泛化能力和分类准确率。

主动学习算法一般包括以下两个部分：

1) 学习引擎（Learning Engine）。学习引擎在已标记样本集上训练一个分类器，当分类器的准确率或某一度量标准满足条件时结束训练并输出结果。

2) 采样引擎（Sampling Engine）。采样引擎根据学习引擎分类器的分类结果和一种样本选取策略在未标记样本集中选取样本，将样本交由领域专家进行人工标记，并在标记完成后将样本加入到已标记样本集中。

学习引擎和选择引擎进行循环交替，经过多次循环之后学习引擎中分类器的性能逐渐提高，并在分类器达到一定的性能条件后终止。算法3以伪代码的形式描述了主动学习算法的流程。

采样引擎是整个主动学习过程的核心，根据采样引擎的不同，主动学习算法可分为三类：

1) **成员查询综合 (Membership Query Synthesis)**。成员查询综合是主动学习最早被提出的采样方案[20]。算法通过学习器来生成一个样本，并交由专家进行标注。该方法的缺点是忽略了样本集的实际分布情况，有可能生成了人类专家无法标注的样本。

算法 3 主动学习算法伪代码描述

Input: 已标记样本集 L，未标记样本集 U，学习引擎 LE，采样引擎 SE

Output: 学习引擎 LE

1: **repeat**
2: $LE.train(L)$ ▷ 在已标记样本集上训练分类模型
3: $T = LE.predict(U)$ ▷
4: $Q = SE.query(U, T)$
5: $Label(Q)$
6: $L = L + Q$
7: $U = U - Q$
8: **until** 性能条件 $Condition$ 满足

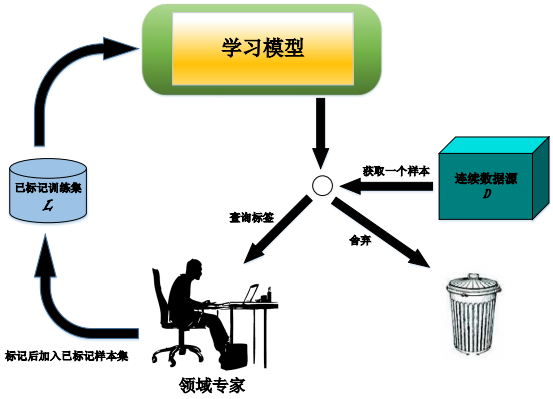
2) **基于流的主动学习 (Stream-based)**。如图4.4(a)所示，基于流的主动学习从真实的样本空间中抽取样本，由选择算法决定是否需要将其交给专家进行标注，若不需要标注则舍弃。这种采样策略通常需要设定一个评价标准来对样本进行评估（例如一个固定的阈值），缺乏对不同应用场景的普适性。同时，算法过程中每次仅选取一个样本与阀值进行比较，忽略了与其他未标记样本的对比。

3) **基于池的主动学习 (Pool-based)**。基于池的主动学习算法将未标记样本集看作一个 ‘‘池’’，每次从池中抽取最有价值的样本进行人工标注。基于池的主动学习解决了上述两种方法的缺点和不足，因此是当前研究最多、应用最广泛的样本抽取方法。图4.4(b)为基于池的主动学习算法过程。

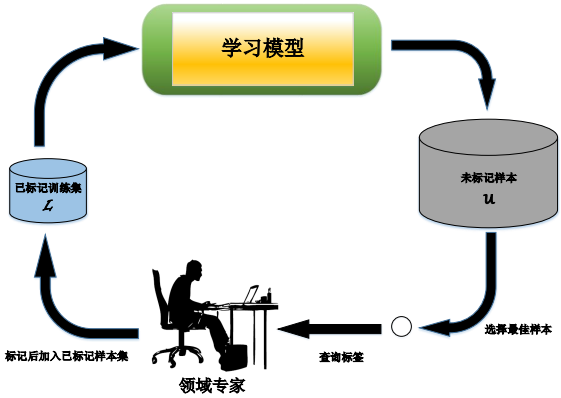
4.4.2 主动学习算法采样策略

近年来，众多的学者和工作聚焦于主动学习算法的研究，尤其是对样本选择算法的研究，希望能发掘出最有意义最具代表性的样本加入训练样本集。在上一节中，我们介绍了主动学习算法中采样方法的三种类型，其中基于池的主动学习算法是目前研究和应用最广泛的样本选择方法。因此，本章主要按照基于池的主动学习算法展开，根据样本选择标准的不同，主要介绍基于不确定性缩减、基于版本空间缩减和基于期望误差缩减三种样本选择策略。

1) **基于不确定性缩减**。基于不确定性缩减的样本选择是目前研究较多的一种基于池的样本选择方法。这种选择方法选择当前分类器最无法确定标签的样本并交由专家进行标记。这种方法对于概率学习模型来说最简单适用。例如，对于二分类问题的概率模型来说，基于不确定性缩减的采样策略将选取那些后验概率接近 0.5 的样本[21]。对于分类器来说，最无法确定标签的样本也是分类器最有可能产生分类错误的样本，选择这类样本进行人工标记，不仅能够有效减少人类专家的标注工作，也能较好地提高分类器的准确度和泛化能力。根据分类器模型的不同，衡量样本不确定性



(a) 基于流的主动学习



(b) 基于池的主动学习

图 4.4: 主动学习示意图

的方法和角度也有不同，因此采样策略也会不同，目前常用的方法包括最小边界、最低确定度、最小边缘、最大信息熵等。信息熵 (Entropy)[22] 是编码分布所需信息量的信息论度量，因此通常被认为是机器学习中不确定性或不纯度的度量：

$$x_H^* = \arg\max_x -\sum_i P_\theta(y_i|x) \log P_\theta(y_i|x) \tag{4.9}$$

其中，y 为所有的标签集合。基于信息熵的采样策略受到了广泛的研究和应用[23,24]，适用于各种复杂模型和多类别问题，在自然语言处理、文本分类等领域均有较好的性能。基于不确定缩减的采样策略可以应用于多种分类器模型，如逻辑回归 (Logistic Regression)、隐马尔科夫模型 (Hidden Markov Model) 以及支持向量机 (Support Vector Machine) 等，在多数问题上都能获得较好的性能。

2) **基于版本空间缩减**。版本空间是概念学习中与已知训练集一致的所有假设 (Hypothesis) 的子集集合，因此学习器所需要训练的目标假设肯定也包含其中。基于版本空间缩减的主动学习样本选择方法的目的就是选择那些能够最大程度地缩减版本空间的样本进行标记。委员会投票选择算法 (Query By Committee, QBC) 是一种典型的版本空间缩减的样本选择方法。首先，从当前的版本空间中随机地选择 C 个不

同的假设构成一个委员会 $Committee = \{\theta^{(1)}, ..., \theta^{(C)}\}$，委员会中的每个假设对所有未标记样本进行分类投票，选择各个假设分类结果差异最大的样本交由专家进行标记。换句话说，在当前的训练集上构建多个分类器模型，运用这些模型来预测未标记样本，并选择分类结果不一致性的样本进行标记。委员会投票选择算法的核心是如何衡量成员对样本的的不一致性程度。作为最早提出的 QBC 算法[25]，Seung 等人采用投票熵作为评价委员会成员分类差异度的标准：

$$x^*_{VE} = \arg\max_x - \sum_i \frac{V(y_i)}{C} \log \frac{V(y_i)}{C} \tag{4.10}$$

其中，y_i 为所有可能的类别，$V(y_i)$ 是投票为 y_i 的委员会分类器个数，C 是委员会的分类器个数。McCallum 等人将样本分布密度与 QBC 相结合的 EM-QBC 算法，引入 KL 分歧度 (Kullback-Leibler Divergence) 作为衡量两个概率分布差异程度的标准，计算方法如下：

$$KL(p_1(x), p_2(x)) = -\sum_{i \in L} p_1(x_i) \log \left(\frac{p_1(x_i)}{p_2(x_i)} \right) \tag{4.11}$$

该算法的主要思想是委员会成员中不一致性程度最大的样本分布区间包含拥有最大信息量的样本。算法将这一区间内的样本交由专家进行标记。另外，委员会的建立策略也是研究的一个重点，Abe 采用集成学习中 Boosting 和 Bagging 方法提出了 Boosting-QBC 和 Bagging-QBC 的委员会建立策略[26]。

3) **基于期望误差缩减**。主动学习的目的是通过选择最优样本进行标记并加入训练来降低分类器的误差和提高分类器的分类准确率。基于期望误差缩减的样本选择方法就是通过减少分类器的误差来直接提高算法的泛化能力。该方法选择使得分类器未来泛化误差最大程度缩减的样本进行标记。具体步骤为：采样算法将每一个未标记的样本都作为候选样本，将其标记后加入已标记样本训练集并训练分类器；对比分类器训练前后的误差变化，选择能够最大程度缩减分类器泛化误差的样本进行标记并加入已标记样本集。Cohn 等人[27] 提出了基于统计学的主动学习算法，采用了模型方差最小化的缩减策略，并在人工神经网络、高斯混合模型和回归模型中加以应用。基于期望误差缩减的方法，主要搜索使期望误差缩减最大化的未标注样例，来减少标注样例的次数并且有效提高分类器的性能，但是这种方法的计算量较大，尤其是数据集合较大的情况下。

大多数的主动学习算法致力于每次选择一个最优样本进行标记，在标记后重新训练模型并循环这一过程（采样 $\rightarrow$ 标记 $\rightarrow$ 训练）直至达到迭代次数或满足性能优化要求。然而，很多模型的训练优化是复杂而且耗时的，从而反复训练模型的代价是高昂的，而且如此反复的训练模型在实际的运用中也是不现实的，因此提出了批量模式的主动学习算法。批量模式的主动学习不仅能够减少采样冗余，而且还可以增加样本的多样性。Hoi 等人通过最大化查询批量样本的费舍尔信息来解释冗余问题[24,28]。

主动学习作为解决数据稀缺问题的有效范例，通过领域专家反馈信息优化了分类器的学习效果，降低了监督学习获取数据标签信息的成本，近年来已经受到广泛关注和深入研究[29,30]。特别是随着在各种应用领域内图和网络数据的丰富，基于图的主动学习已经受到很多研究的关注[31–33]。

4.4.3 最大批量网络增益的采样算法

最大批量网络增益采样是批量模式主动学习的优化采样策略，将代表性和多样性与传统的不确定性标准相结合。代表性和多样性考虑了训练集中的实例之间的相互依存关系，是不确定性标准的两个补充标准。在选择待标记样本时，样本需要具有代表性以能够提高分类器模型的性能，同时也可以具有更大的多样性以降低标记操作的冗余。最大批量网络增益采样结合代表性和多样性两个特点，并通过最大化网络增益来选择批量样本集。

对于未标记样本集 U 中的每个样本 x_i，查询样本的标签信息的个体信息增益通过信息熵计算：

$$H(x_i) = -\sum_{y} P(y|x_i) \log P(y|x_i) \tag{4.12}$$

其中，$P(y|x_i)$ 是标签传播算法计算得到的样本 x_i 标签分布概率。

根据标签传播算法，查询批次的网络增益通过传播查询本批次中每个样本的个体信息增益至图中其他的未标记样本来计算，因此定义批量网络增益 $NG(B)$ 为：

$$NG(B) = \sum_{j \in U-B, i \in B} w_{ji} H(i) - \mu \sum_{i,k \in B, i \neq k} w_{ki} H(i) \tag{4.13}$$

其中，第一部分代表了从剩余未标记样本中查询批量样本的信息增益，而第二部分表示批次内样本之间的冗余，w_{ji} 是节点 j 和 i 之间边的权重，μ 为惩罚因子。批量网络增益结合了不确定性 (节点信息熵 $H(\cdot)$)、代表性 (公式4.13中第一部分) 和多样性 (公式4.13中第二部分)。

在主动学习迭代过程的每一步，选择具有最大网络增益 $NG(B)$ 的样本批次来交由领域专家进行标记。定义 $W = [w_{ij}]$ 为图的邻接矩阵，可以将公式4.13中第一部分的最大化问题转化为在给定 B 大小的情况下最大化分割图中 B 和 $U - B$ 的问题。通过贪心算法可以求解该问题，伪代码描述如算法4所述。

4.5 基于主动学习的标签传播算法

上一节我们提出了一个基于批量模式的最大网络增益主动学习采样算法，结合标签传播算法的特点选取能够最大化提升标签传播算法性能的节点进行标记。整体算法描述如算法5所示。

算法 4 最大批量网络增益采样

Input: 未标记样本集 U, 批量大小 N

$B = \emptyset$ ▷ 初始化 B 为空集

repeat

$k = \arg\max_{i \in U} NG(B \cup i)$

$B \leftarrow B \cup k, U \leftarrow U - k$

until $|B| = N$

4.6 基于文件社交网络的恶意软件检测方法

在分析了文件社交网络特点和特征属性的基础上，运用上一节提出的基于主动学习的标签传播算法可以分析文件之间标签信息传播的过程并进行恶意软件的检测。算法6描述了整个算法的流程。

4.7 实验验证

在本章中，我们提出了一种基于文件社交网络的恶意软件检测方法，由于部分方法和技术在上一章中已经进行了实验验证，故本章不再赘述。本节重点通过两组实验分别来验证基于图属性的文件采样方法和主动学习算法的可行性和有效性。

本章实验所用平台与上一章相同，在此不再赘述。在上一章分析实验结果时我们可以发现，由于数据的不均衡性，良性文件的数量远远大于恶意软件的数量，因此在准确率 ACC 的计算上良性文件的数量占据了主要成分，考虑以下两种极端情况：第一种情况，所有的文件（包括恶意和良性）都被判为良性文件，在这种情况下，算法的准确率 ACC 就可高达 92.75%，而 TPR 则为 0；然而第二种情况，所有的文件都被判为恶意软件，此情况下算法的准确率 ACC 则为 7.25%，而 TPR 则为 100%。在本章中，综合考量各类别样本的分布情况，选择以下性能评价指标以体现算法的性能：

- **Recall**: $\frac{TP}{TP+FN}$: 正确标记为恶意的样本占恶意软件总量的比率。
- **Precision**: $\frac{TP}{TP+FP}$: 标记为恶意软件的样本中确为恶意样本的比率。
- **Accuracy(ACC)**: $\frac{TP+TN}{TP+TN+FP+FN}$
- **F1-Score**: $\frac{2 \cdot precision \cdot recall}{precision+recall} = \frac{2TP}{2TP+FP+FN}$.

4.7.1 基于图属性的采样策略分析

本章提出了三个基于图的特征属性，根据这些属性衡量文件的重要性，并选择具有高代表性的文件交由安全专家进行标记。本节将通过实验来验证这三个属性以及采样算法对算法提升的作用。实验中，分别比较基于局部聚集系数 (Local Clustering

算法 5 Label Propagation With Active Learning

Input: 未标记数据 $D_U = (x_{l+1}, y_{l+1})...(x_{l+u}, y_{l+u})$, 标记数据 $D_L = (x_1, y_1)...(x_l, y_l)$ 及类别集 C

Output: 未标记数据的类别

```
repeat
    initial(D_U, D_L)                          ▷ 初始化相似度矩阵 W 和标签传递矩阵 T
    for i = 0; i < l + u; i++ do               ▷ 初始化标签矩阵 Y
        for j = 0; j < |C|; j++ do
            Y_ij = p(y_i, C_j)
        end for
    end for
    repeat
        Y ← T̄Y                                 ▷ 传播标签信息
        clamp(D_L)                             ▷ 限定已标记数据
    until Y converges
    B = ∅                                      ▷ 初始化 B 为空集
    repeat
        k = arg max_{i∈U} NG(B ∪ i)
        B ← B ∪ k, U ← U − k
    until |B| = N
until the stopping criterion S is satisfied    ▷ 直至满足迭代停止条件
assignLabel(D_U)
```

Coefficient)、度中心性 (Degree Centrality)、接近中心性 (Closeness Centrality) 以及综合三个属性的 $important$ 值采样和随机抽取方法对算法的性能改善能力。从表4.1可以看出，综合三个图属性的 $important$ 对分类器性能的提升有最好的作用。为了中和方差，随机抽取方法运行十次，取平均值作为最终结果。

4.7.2 主动学习对预测的影响

在本实验中，将系统性地分析和评估主动学习算法对分类器的影响。为了验证最大批量网络增益采样策略的有效性，选取随机抽样策略作为对比策略。最大批量网络增益算法的批量大小设置为 20，即每次从未标记数据集中选择 20 个最代表性的样本进行标记。同理在随机抽样策略中，每次也从未标记数据集中抽取 20 个样本进行标记。

从图4.5中可以看出，结合了最大批量网络增益采样策略主动学习算法的标签传

算法 6 基于文件社交网络的恶意软件检测方法

Input: 原始文件列表数据 $FileList$

Output: 文件样本的标签

for each f_i in $FileList$ **do** ▷ 计算每个文件与已知关联文件之间的相似度
  $similaityASS(f_i)$
end for
for each f_i in $FileList$ **do** ▷ 计算未标记文件样本之间的相似度
  for each f_j in $FileList$ **do**
    if f_i is unlabled and f_j is unlabled **then**
      $similaityUnLabel(f_i, f_j)$
    end if
  end for
end for
$createGraph(V, E, W, k)$ ▷ 构造 k 近邻图
$samplingByGraphFeature(G)$ ▷ 根据图属性选择重要文件样本进行标记
运用基于主动学习的标签传播算法训练
根据标签矩阵标记未标记文件样本的标签

表 4.1: 基于图属性的采样策略分析

Feature	Recall(%)	Precision(%)	ACC(%)	**F1-Score(%)**
Random	50.3	67.0	94.2	57.4
DC	50.1	67.5	94.1	57.5
CC	49.8	68.4	94.4	57.7
LCC	50.8	66.3	94.2	57.5
Combination	50.7	68.0	94.2	**58.1**

播算法能够比随机采样有更好的表现。从第三次迭代开始，最大批量网络增益的主动学习算法与随机采样策略之间的差距逐渐变大，这样的趋势也表明本章提出的恶意软件检测系统的性能比其他的方法要好。

4.8 本章小结

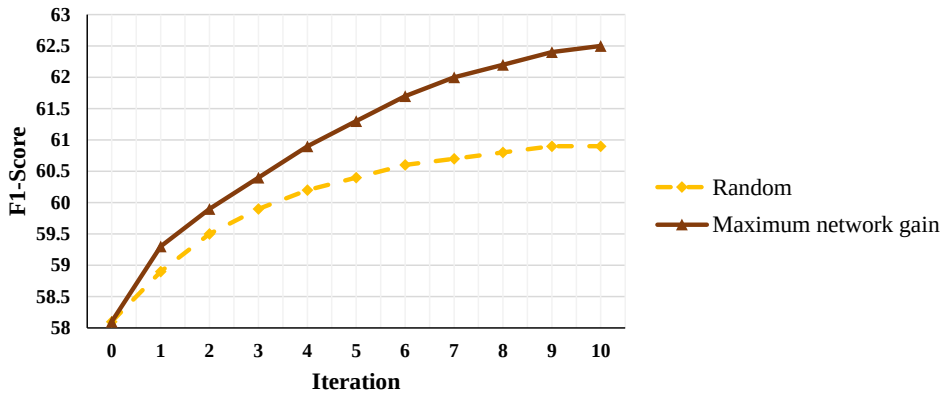


图 4.5: 主动学习算法对预测的影响

5 总结与展望

致　谢

参考文献

[1] 中国互联网络信息中心. 中国互联网络发展状况统计报告 [R]. Tech. rep., 2017. http://www.cac.gov.cn/2017-01/22/c_1120352022.htm.

[2] 国家互联网应急中心. 2015 年中国互联网网络安全报告 [R]. Tech. rep., 2016. http://www.cert.org.cn/publish/main/46/2016/20160602141337392864292/20160602141337392864292_.html.

[3] F. Cohen. Computer Viruses[D]University of Southern California, 1985.

[4] G. McGraw, G. Morrisett. Attacking Malicious Code: A Report to the Infosec Research Council[J]. IEEE software, 2000, 17(5):33–41.

[5] B. A. W. Neumann, John Von. Theory of Self-reproducing Automata[M]. University of Illinois Press, 1966:745.

[6] 中华人民共和国国务院. 中华人民共和国计算机信息系统安全保护条例, 1994.

[7] E. Skoudis, L. Zeltser. Malware: Fighting Malicious Code[M]. Prentice Hall Professional, 2004.

[8] Wikipedia. Trojan Horse. https://en.wikipedia.org/wiki/Trojan_horse_(computing).

[9] M. Maier, U. Von Luxburg, M. Hein. Influence of Graph Construction on Graph-based Clustering Measures.[C]//NIPS. 2008:1025–1032.

[10] X. Zhu, J. Lafferty, R. Rosenfeld. Semi-supervised Learning with Graphs[M]. Carnegie Mellon University, language technologies institute, school of computer science, 2005.

[11] X. Zhu, Z. Ghahramani. Learning from Labeled and Unlabeled Data with Label Propagation[J]. 2002.

[12] Y. Ye, T. Li, S. Zhu, et al. Combining File Content and File Relations for Cloud Based Malware Detection[C]//Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. 2011:222–230.

[13] A. Tamersoy, K. Roundy, D. H. Chau. Guilt by Association: Large Scale Malware Detection by Mining File-relation Graphs[C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014:1524–1533.

[14] A. Venzhega, P. Zhinalieva, N. Suboch. Graph-based Malware Distributors Detection[C]//Proceedings of the 22nd International Conference on World Wide Web. 2013:1141–1144.

[15] R. Diestel. Graph Theory[J]. volume 173 of Graduate texts in mathematics, 4th Edition, 2010.

[16] D. J. Watts, S. H. Strogatz. Collective Dynamics of ‘small-world’ networks[J]. nature, 1998, 393(6684):440–442.

[17] J.-P. Onnela, J. Saramäki, J. Kertész, et al. Intensity and Coherence of Motifs in Weighted Complex Networks[J]. Physical Review E, 2005, 71(6):065103.

[18] J. Scott. Social Network Analysis[M]. Sage Publications Ltd, 2012.

[19] L. C. Freeman. A Set of Measures of Centrality Based on Betweenness[J]. Sociometry, 1977:35–41.

[20] D. Angluin. Queries and Concept Learning[J]. Machine learning, 1988, 2(4):319–342.

[21] D. D. Lewis, J. Catlett. Heterogeneous Uncertainty Sampling for Supervised Learning[C]//Proceedings of the eleventh international conference on machine learning. 1994:148–156.

[22] C. E. Shannon. A Mathematical Theory of Communication[J]. The Bell System Technical Journal, 1948, 27(4):623–656.

[23] A. Culotta, A. McCallum. Reducing Labeling Effort for Structured Prediction Tasks[C]//AAAI. 2005, 5:746–51.

[24] S. C. Hoi, R. Jin, M. R. Lyu. Large-scale Text Categorization by Batch Mode Active Learning[C]//Proceedings of the 15th international conference on World Wide Web. 2006:633–642.

[25] H. S. Seung, M. Opper, H. Sompolinsky. Query by Committee[C]//Proceedings of the fifth annual workshop on Computational learning theory. 1992:287–294.

[26] N. Abe, H. Mamitsuka. Query Learning Strategies Using Boosting and Bagging[C]//J. W. Shavlik. Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998. Morgan Kaufmann, 1998:1–9.

[27] D. A. Cohn, Z. Ghahramani, M. I. Jordan. Active Learning with Statistical Models[J]. Journal of artificial intelligence research, 1996, 4(1):129–145.

[28] S. C. Hoi, R. Jin, J. Zhu, et al. Batch Mode Active Learning and its Application to Medical Image Classification[C]//Proceedings of the 23rd international conference on Machine learning. 2006:417–424.

[29] H. T. Nguyen, A. Smeulders. Active Learning Using Pre-clustering[C]//Proceedings of the twenty-first international conference on Machine learning. 2004:79.

[30] I. Muslea, S. Minton, C. A. Knoblock. Active Learning with Multiple Views[J]. Journal of Artificial Intelligence Research, 2006, 27:203–233.

[31] N. Cesa-Bianchi, C. Gentile, F. Vitale, et al. Active Learning on Graphs via Spanning

Trees[C]//NIPS Workshop on Networks Across Disciplines. 2010:1–27.

[32] N. Cesa-Bianchi, C. Gentile, F. Vitale, et al. Active Learning on Trees and Graphs[C]//In COLT. 2010:320–332.

[33] W. Zhao, J. Long, E. Zhu, et al. A Scalable Algorithm for Graph-based Active Learning[C]//International Workshop on Frontiers in Algorithmics. 2008:311–322.

附 录

攻读博士学位期间发表的论文和出版著作情况：

1. Kechen Zhuang, Haibo Shen, Hong Zhang: User Spread Influence Measurement in Microblog. Multimedia Tools and Applications, 2016. （SCI 在线发表, EI：20163202701002）

攻读博士学位期间参加的科学研究情况：

1. Urban Population Activity Patterns Prediction based on Mobile Phone Data，华盛顿大学，南京理工大学出国访学资助，起止时间：2013 年 1 月至 2013 年 10 月；