DOI:10.19651/j.cnki.emt.2105981

结合应用接口可达性特征的 Android 恶意软件检测

金泽宇[1] 朱正伟[2]
(1.常州大学 计算机与人工智能学院 常州 213000;2.常州大学 微电子与控制工程学院 常州 213000)

摘 要:针对 Android 恶意软件检测,现有的研究多数提出了多种类特征结合机器学习来提高恶意程序检测中检测率,却少有考虑程序内应用接口调用之间的关联和程序调用图中的边信息。设计了基于应用接口可达性特征的 Android 恶意软件检测的方法,该方法基于恶意行为提取了应用接口的可达性特征,有效地使特征集包含边信息。在 VirusShare 所收集的 2018 年的恶意程序集中的 1 151 个恶意程序与来自 Google Player 的 1 021 个良性程序上进行了实验,结果为采用随机森林方法得到的模型相比其他 4 种方法更有效实现恶意程序的检测,并且模型整体的准确率达到了 98.90%。结果表明应用接口可达性特征使得模型的召回率和精度均有所提高,并且相比实验中的其他特征更为重要。

关键词:恶意程序检测;机器学习;边信息;恶意行为;可达性特征
中图分类号:TP311.5 **文献标识码**:A **国家标准学科分类代码**:520.4070

Android malware detection based on accessibility features of application interface

Jin Zeyu[1] Zhu Zhengwei[2]
(1. School of Computer Science and Artificial Intelligence, Changzhou University, Changzhou 213000, China;
2. School of Microelectronics and Control Engineering, Changzhou University, Changzhou 213000, China)

Abstract: For Android malware detection, most of research proposed multi-type features combined with machine learning to improve the detection rate of malware detection, but rarely considered association between application interface and edge information in call graph. Designs a method of Android malware detection based on accessibility feature of application interface. This method extracts accessibility features of application interface based on malicious behaviors, effectively makes feature set contain edge information. Experiments were conducted on 1 151 malware collected by VirusShare in 2018 and 1 021 benign programs from Google Player. Experiments show that random forest is more effective than other four methods in malware detection, and accuracy of model reaches 98.90%. Results show that accessibility features improved recall rate and precision of the model, and is more important than other features in the experiment.

Keywords: malware detection;machine learning;edge information;malicious behavior;accessibility feature

0 引 言

随着近几年智能手机的普及,手机移动应用得到了迅速发展,其多样化的功能涉及了生活的方方面面。然而 Android 系统以其本身的开放性作为最大的应用开发平台,一些不法分子在应用程序(application,APP)加入恶意代码,从而涌现大量恶意软件。恶意软件不及时处理会引发严重的信息安全问题,例如用户的隐私泄露和经济的损失。因此,恶意程序检测的研究具有重要的意义。

恶意程序检测的特征主要分为静态与动态两种检测方法。静态检测方法通过反编译,反汇编等技术从程序文件中提取语义特征或从 Manifest 文件中提取意图和权限特征[1-2]等。动态检测方法是在程序运行过程中监控手机中的资源消耗[3-4]和系统调用[5-6]这类动态行为特征。动态检测方法在程序执行过程中提取的信息较为可靠,却受到代码覆盖限制的阻碍,只有程序运行过的代码能得到保证。本文主要提出了一种应用接口之间的可达性特征,更好地表现出恶意良性程序的区别。

现有基于静态检测技术提取关于恶意程序检测特征的研究,多数是通过提取多种类型的特征来提高检测率,却少

收稿日期:2021-03-15

有考虑程序调用图中的边信息。应用程序中的功能多数需要多个应用接口来协同完成,因此,本文基于边信息提取出可达性特征,更好地区分良性与恶意程序。然后使用 5 种常见的机器学习方法进行消融实验,确认可达性特征的作用。实验结果表明,加入可达性特征后的特征集合,恶意程序的召回率、精度和模型整体的准确率均有所提高,并且使用随机森林算法得到的模型取得最高的准确率,达到了 98.90%。

1 相关研究

近几年,不少研究通过机器学习的方法来解决恶意程序检测的问题,主要集中在两个大方向。一方面是提出新的算法模型来提高分类精度,另一方面是通过提取多种特征来提高区分度,从而提升模型的准确率。

一方面研究者使用多样化的方法来训练出好的检测模型。2011 年,张鹏涛等[7]提出了引入惩罚因子摆脱恶意程序的变化带来的影响,有效地调节了特征的表征性,加强模型泛化能力。2017 年 Yerima 等[8]提出一种新的分类器融合方法 DroidFusion,采用多层结构和多种排序方法来衡量模型的有效性,从而融合分类器来提高分类精度。2017 年,苏志达等[9]基于深度置信网络设计了 DeepDroid 算法,相比朴素贝叶斯和 K 最近邻算法更能有效检测出恶意程序。2017 年,Hou 等[10]提出了一种结构化的异构信息网络表示方法来描述应用程序和 API,有效地衡量应用程序之间的相似性,并基于相似度提出了一种多核学习算法来对应用程序进行分类。同年,Zhang 等[11]提出了利用程序调用图的子图,进行同构匹配来抵抗恶意程序的变化,提高检测精度。

一些研究集中在特征类型上进行多种尝试。2013 年,DroidAPIMiner[12]分析了权限、关键 API 以及其依赖包和参数评估了多种机器学习。2014 年,Arp 等[13]使用混合方法从 Manifest 文件中提取硬件组件、请求权限、程序组件和意图 4 部分,另一方面从反汇编的代码中提取了 API 调用和网络地址等特征,最后使用支持向量机作为分类器。2017 年,徐林溪等[14]提出混合语法和语义特征,将权限和 Intent 声明作为语法特征,将污点传播路径作为语义特征,使用 K-means 算法进行分类。2020 年,刘晓建等[15]通过考虑敏感资源是否由 UI 调用、敏感 API 与回调函数的关系、敏感权限与组件的关系来生成上下文特征,从而提高检测的准确率。

在提取多种特征的方面,部分研究者提取与数据流相关的特征来进行恶意程序的检测。

为了进一步提高精度,一些研究者逐渐在特征流分析方面进行了探讨。2014 年,Rasthofer 等[16]以隐私敏感数据流为研究点,设计了名为 SuSi 的工具,一种通过机器学习的方法为 Android 中应用接口进行 Source 和 Sink 的识别。同年,Azrt 等[17]提出 FlowDroid 以 Android 生命周期的精确模型和 SuSi 为架构,从而发现程序中的隐私敏感数据流。2015 年,Elish 等[18]提出了应用接口的触发点,发现良性程序的应用接口多以 UI 有关的接口触发,并提出了以触发点为特征区分良性和恶意程序,提高了分类精度。因此 2017 年,RepassDroid[19]结合隐私敏感数据流和触发点,同时提取了敏感应用接口及其有关权限作为特征集合,比较多种机器学习方法,通过消融实验确认了特征集合各部分的作用。2020 年,汪洁等[20]则是考虑设备中的文件,网络,数据库等实体,动态捕获程序中实体之间的数据流,提取出恶意行为的特征子图,构建子图向量之间的相似形函数来检测恶意程序。

本文提出了基于应用接口之间可达性特征的 Android 的恶意检测方法。之前关于数据流特征的研究主要考虑程序中的边信息,来提取应用接口所在路径上的特征。本文注重应用接口之间的联系,提取了程序中应用接口之间的可达性特征,最后使用随机森林对恶意程序进行分类,有效提升了模型分类的准确率。

2 特征工程

2.1 基于 Androguard3 的静态分析

对于恶意程序的静态分析,特征的提取和处理是恶意程序分类模型的重中之重,安卓安装包(Android application package, APK)的静态分析已经较为成熟。Androguard3 是基于 python 的工具,包含了对 dex 文件进行反汇编和反编译的功能,将 APK 文件中的 dex 文件、类、方法等都映射为 Python 的对象。如图 1 所示,Androguard3 被用来作为静态分析的工具,得到每个程序的 AndroidManifest.xml 文件和调用图 Callgraph,并从中提取出多种类的特征,这些特征会作为恶意行为特征和用于最终的恶意程序检测的分类。

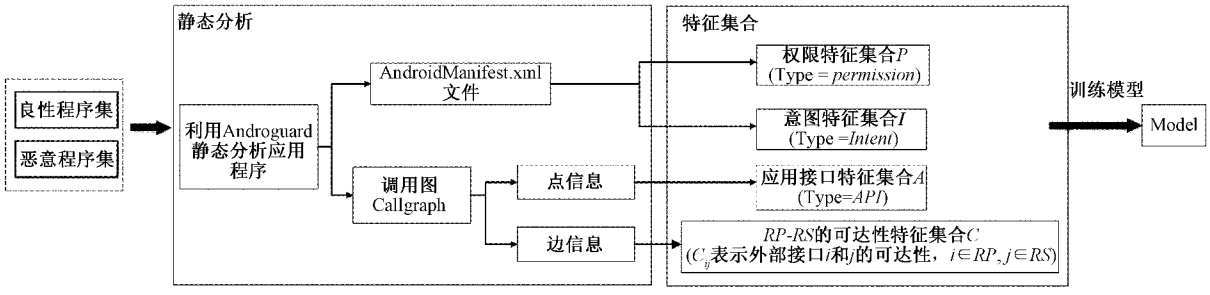


图 1 基于 Androguard3 的静态分析图

大部分研究主要都使用了 Callgraph 中的点信息来提取应用接口(application interface,API)作为特征,缺乏考虑这些包含了 API 特征之间的边信息。本文利用调用图中边信息来构建可达性特征,使得提取的特征包含更多的信息,从而提高模型的精度。

2.2　权限和意图特征

Android 应用程序由一个或者多个组件所组成,而每个应用程序要有 AndroidManifest. xml 文件来描述这些组件。由于安卓的沙盒机制和线程隔离机制,Android 程序中的组件必须进行一些声明,而恶意性倾向会反映在 AndroidManifest. xml 文件的权限和意图信息中。

权限是 Android 引入的最重要的安全机制之一。用户在安装时主动授予权限,并允许应用程序访问安全相关资源。恶意软件比无害的应用程序更倾向于请求某些权限。意图主要是来实现 Android 上的进程间和进程内通信,允许不同组件和应用程序之间共享有关事件的信息。恶意软件经常监听特定意图,恶意软件中涉及的意图消息的典型示例是引导完成,用于在重新启动智能手机后直接触发恶意活动。

2.3　应用接口特征及可达性特征

1)应用接口特征

应用接口特征主要来自于 Androguard3 中包含的 Android 官方资源所匹配出的程序中的函数调用。在该过程中,同时生成每个程序的调用图,有助于提取应用接口特征。由于每个程序所包含的应用接口非常多,其中不少与恶意行为并没有多大的关联。如图 2 所示,本文通过 VirusTotal 分析一部分的恶意程序,提供了 K 种恶意行为。提取每一组恶意行为的特征后,使用 SVM 得到的权重来衡量特征重要性,从而筛选特征,得到了多组筛选过后的 K 组 V_i 特征集合。

2)基于恶意行为的可达性特征

K 组特征集合没有包含可达性的特征,主要包括了权限、意图和应用接口 3 种特征,而本文所提出的可达性特征主要是基于恶意行为来提取特征后的结果。图 2 中,多组的特征集合存在交集,因此统计了各个特征在 K 组特征集合中出现的次数,记为 times。本文的可达性特征考虑的是外部应用接口之间的可达性,基于恶意行为提取到的重复的 API 与各组恶意行为的独有特征之间的可达性。

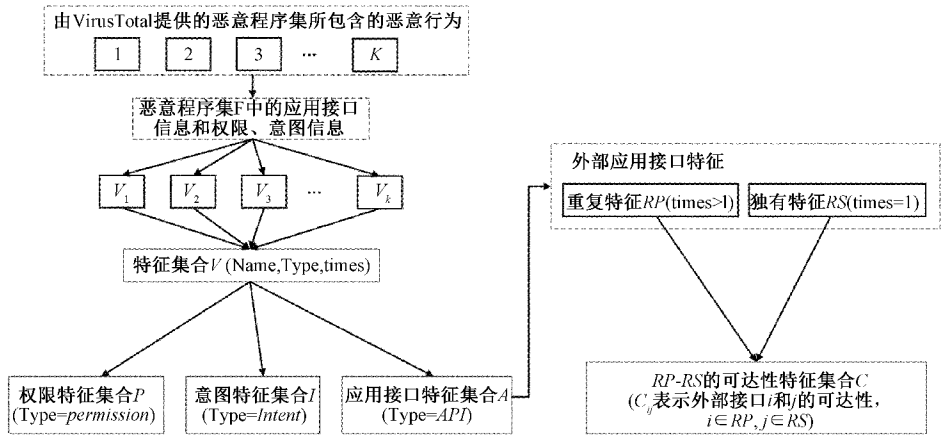


图 2　基于恶意行为的多种类特征提取

可达性特征需要提取调用图中的边信息,而应用程序调用图的节点和边数量都很多,会导致可达性特征的提取耗时多。因此,考虑所有应用接口之间的可达性特征会耗费大量时间,而很多特征对分类精度没有太多帮助,所以本文基于恶意行为来考虑重复的 API 与各组恶意行为的独有特征之间的可达性,减少可达性特征的数量。重复特征在不同恶意行为中作用会有所区别,考虑重复特征与独有特征之间的可达性,能够有效区分重复特征在不同恶意行为中的作用。

此外,本文提取的是外部应用接口之间的可达性,提取应用接口特征的过程中,使用属性 external 来表示应用接口的外部属性,当应用接口为外部接口,external 属性值即为 1,否则记为 0。

如图 3 所示,存在可达性的两个外部接口,白点为外部接口,external 的属性为 1,黑点相应的 external 属性为 0,外部应用接口大部分是获取用户信息或者手机信息的外部 API。图中虚线内部的那些非外部应用接口为 APK 中数据流的路径。外部应用接口包含了 sendTextMessage 这类在安全方面常见的重要 API,主要考虑这些 API 之间可达性是合理的。

事实上,对于任意外部接口之间没有直接的路径连通,但是外部接口之间确实有一定的联系。以图 3 中局部调用图来分析,若两个外部接口为获取手机中不同类型的信息,而右侧的外部接口得到的信息 1,会随着非外部应用接口向前传递,而左侧的外部接口中得到的信息 2 也会向前传递。信息 1 和信息 2 可能均为非过度敏感的信息,但信息组合就可能暴露用户敏感信息,后续也会有其他信息加入,来达到恶意行为。

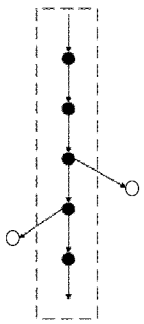


图 3 两个外部接口的可达性示意图

3)可达性特征的获取

可达性特征的获取,本文主要分为两个部分,一部分为获取外部接口的可达点集合,另一部分为比较可达点集合来确定外部接口之间的可达性,其中获取可达点集合是主要部分。

考虑到程序的设计,大部分会分为以功能为单位来单独设计,完成同一功能的多个应用接口在调用图中的会较为接近。所以在提取可达性特征时,只需要考虑外部应用接口在调用图中与其较为接近的节点。

第 1 部分获取某个应用接口的可达点的集合如算法 1 所示,算法 1 主要以递归的方法来找到搜索范围阈值 k 内的可达点集合。

算法 1 获取应用接口 a 的可达点的集合

输入:应用接口 a
调用图边信息 E
输出:应用接口 a 的可达点的集合 $arrive$
应用接口 a 的可达点的步长 $step$

```
初始化:arrive, step 为空,
        node_set=a,k=0
def helper(node_set, E, arrive, step, k):
    k=k+1
    if node_set 不为空 &k<10 do
      able_arrive←∅,arrive_step←∅
      foreach t∈node_set do
        foreach e∈E do
          if t==e.target & e.source∉arrove do
            able_arrive.append(e.spirce)
            arrive_step.append(k)
    arrive.append(able_arrive)
    step.append(arrive_step)
    node_set←able_arrive
    helper(node_set, E, arrive, step, k)
return arrive, step
```

首先初始化可达点集合 $arrive$ 和可达点的步长 $step$ 为空集,$step$ 中记录了可达点到应用接口 a 的步长,即可达点到应用接口的路径上的边数。$node\_set$ 集合为第 k 次搜索时的需要搜索的点,$able\_arrive$ 记录了每次搜索到的可达点,$arrive\_step$ 记录了每次搜索到的可达点的步长。第 k 次搜索的可达点,需要满足存在边信息的 $target$ 中和不存在 $k-1$ 次搜索后的可达点集合 $arrive$ 中,则第 $k-1$ 次搜索到的 $able\_arrive$ 为第 k 搜索的 $node\_set$ 集合。k 为当前已搜索的范围,当搜索的范围阈值过大时,搜索可达点会耗时过大,而且搜索到的可达点与应用接口 a 的联系不会特别密切。因此,本文实验的搜索范围阈值记为 10。

对于两个外部接口的可达性,当两者的 $arrive$ 集合存在交集,即视为该两个外部应用接口可达。当存在交集时,交集记为 I。

$$I = arrive_1 \cap arrive_2 \tag{1}$$

将重复特征和独有特征到交集中的非外部应用节点的步长分别记为 L_1 和 L_2。

$$L_1 = \{step_{1i} \mid i \in I\} \tag{2}$$

$$L_2 = \{step_{2i} \mid i \in I\} \tag{3}$$

则两个外部接口的可达距离记为 l。

$$L = \{step_{1i} + step_{2i} \mid i \in I\} \tag{4}$$

$$l = \min(L) \tag{5}$$

同时给定某一阈值 l_k,当两个外部应用接口的可达距离记为 l,当 $l<l_k$,则两个外部应用接口可达,记为 1,否则记为 0。

3 实验与分析

3.1 实验数据与环境说明

本文实验环境为 Windows10 64 位操作系统,内存大小为 8 GB,处理器为 Inter(R) Core(TM) i5-8500 CPU@ 2.20 GHz。使用的开发工具包括 Python、Androguard3 和 Sklearn。

在本文实验中共采集了 3 334 个样本,其中良性样本 1 021 个,来自 Google Play 应用市场近几年上传的现实安卓应用,包含了多种应用类别。恶意样本是来自于 VirusShare,由于 2019 年的样本集大小达到 820 GB 和 2020 年的恶意样本集官方暂未上传,本文选择的是 2018 年的安卓恶意程序集。该恶意样本集大小为 126.9 GB,包含了 28 000 多个恶意程序,本文随机提取了 1 162 个和 1 151 个程序分为两组恶意程序集 1,2。

恶意程序集 1 中 1 162 个恶意程序用来提供多个恶意行为的特征集合 $V_i(i=1,2,\cdots,K)$,来确定需要从恶意程序集 2 和良性样本中提取的权限特征 P,意图特征 I,应用接口特征 A 和可达性特征 C 4 组特征。表 1 所示为通过 VirusTotal 获取的恶意程序集 1 中恶意行为的信息。

通过恶意程序集 1 的恶意行为确定提取的特征集,主要包含权限特征、意图特征和应用接口特征,特别地,恶意程序集 1 不考虑可达性特征。后续在恶意程序集 2 和良性

表 1　恶意程序集 1 中的恶意行为

反病毒引擎	恶意行为	样本数量
Qihoo-360	Trojan. Android. Gen	776
SymantecMobileInsight	Other: Android. Reputation. 1	629
Sophos	Andr/Rootnik-AI	627
Ikarus	Trojan-Dropper. AndroidOS. Shedun	560
McAfee-GW-Edition	RDN/Generic. gjk	549
AhnLab-V3	PUP/Android. Agent. 839002	535
Fortinet	Android/Shedun. AC! tr	535
NANO-Antivirus	Trojan. Android. MLW. ebzlbe	534
Cyren	AndroidOS/GhostPush. C. gen! Eldorado	532
CAT-QuickHeal	Android. Smsreg. EJ (PUP)	526

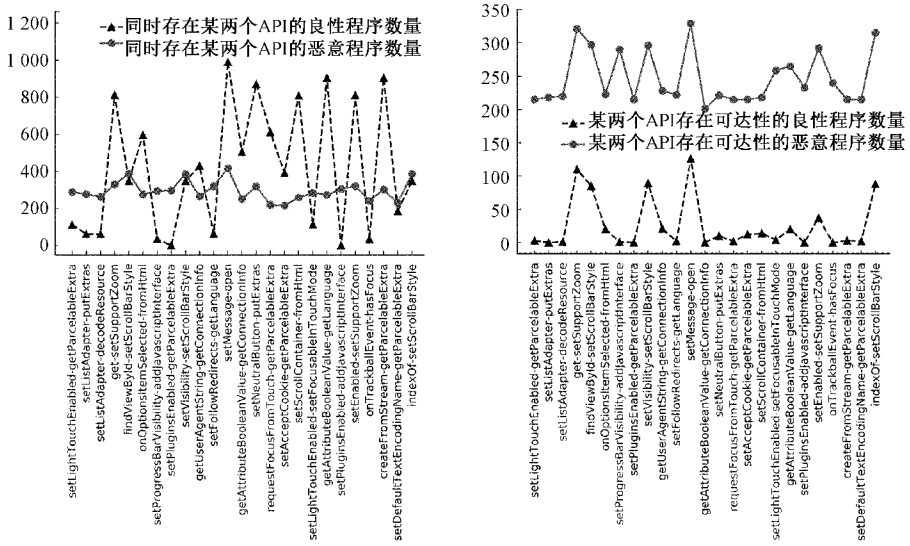
程序集中才提取了本文提及的 4 种特征。由于确认特征的恶意程序集 1 和使用特征的恶意程序集 2 样本不同，一定程度上说明基于恶意行为提取特征的有效性。

3.2　可达性特征分析

为了更清晰说明可达性特征的作用，除了 2018 年的恶意程序，另外选择了来自 VirusShare 的 2013 年恶意程序集，最后对两个年份的恶意程序进行了可达性特征的提取。

该实验的目的是验证可达性特征是否有更好的区分作用。对于可达性特征的每组外部应用接口，本文一方面不考虑边信息，分别统计恶意程序和良性程序中同时两个应用接口的程序数量，另一方面，统计了恶意程序和良性程序中每组应用接口存在可达性的良性程序数量。

实验分别从 2013 年和 2018 年的恶意程序样本挑选了 25 个可达性特征。图 4(a)、(b)分别表现了 2013 年恶意程序集得到的 25 组应用接口，从不考虑边信息和考虑边信息统计的良性和恶意程序数量。从图 4(a)可以看出如果不考虑边信息，某些组特征良性程序的数量高于恶意程序，某些却低于恶意程序，没有明显的区别，这样的特征难以说明是趋于良性还是恶意。而图 4(b)考虑边信息得到的结果表明了 25 个可达性特征均是趋于恶意的特征，这些考虑边信息的可达性特征有了明显的区别良性与恶意的作用。原本 25 组特征中不少是多数良性程序数量偏多的特征，考虑了边信息后，对于这 25 个可达性特征，良性程序的数量都小于 150。



(a) 同时存在某两个API的程序数量　　(b) 两个API存在可达性的程序数量

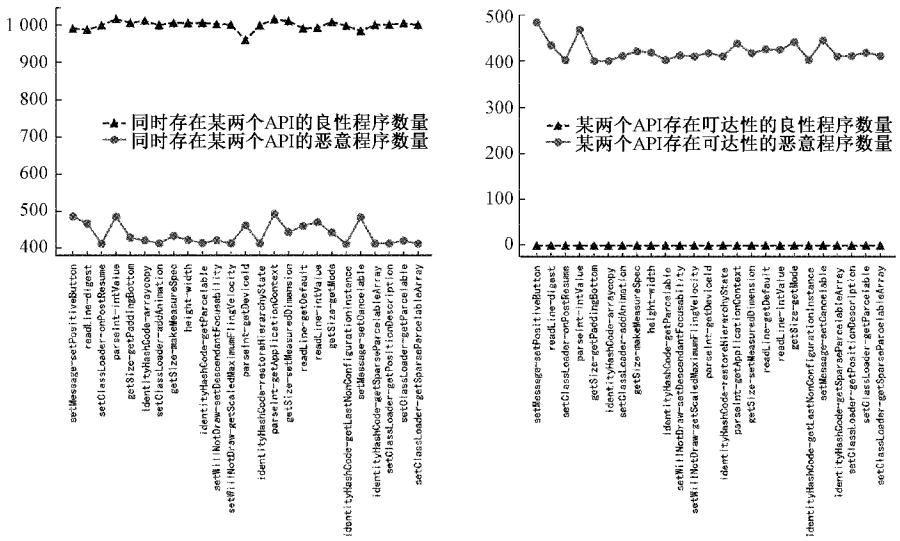
图 4　2013 恶意程序集的可达性特征分析

图 5(a)、(b)则是对 2018 年的程序集得到的 25 组应用接口，相比 2013 年的 25 组，有了更明显的效果。图 5(a)中原本 25 组应用接口，良性程序基本都含有这 25 组的特征，统计的良性程序数量都在 1 000 左右，而恶意程序对这 25 组应用接口特征程序大都只维持在 400～500 之间，与良性程序数量差别很大。但是考虑边信息之后，图 5(b)中的表现与图 4(b)的效果类似，却更为明显。考虑这 25 组特征之间的可达性后，收集的良性程序中的这 25 组应用接口均不存在可达性，统计得到数量基本都为 0。而恶意程序中，对于这 25 个可达性特征，数量仍然均维持在 400 以上。

因此，本文认为相比不考虑边信息的特征组合，可达性特征具有更强的区分性。

3.3　实验结果

本实验将恶意程序集 2 中 1 151 个恶意程序与 Google Player 的 1 021 个良性程序按标签比例随机分为训练集和



(a) 同时存在某两个API的程序数量 (b) 两个API存在可达性的程序数量

图 5 2018 恶意程序集的可达性特征分析

测试集,并重复了 5 次实验,最后统计 5 次的分类结果进行模型的评估。

在实验中,本文采用的评估标准通过机器学习常见的混淆矩阵,使用矩阵数据测量算法性能,在实验中,在恶意特征使用者中将有恶意程序标记为 1,良性程序标记为 0,在恶意特征提供者中将有相应恶意行为的程序标记为 1,反之标记为 0,表 2 所示为混淆矩阵。

表 2 恶意程序分类混淆矩阵

样本数量		实际	
		恶意程序	良性程序
预测	恶意程序	TP	FP
	良性程序	FN	TN

accuracy 是分类器的整体分类准确率。

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

precision 代表着恶意程序分类精度。

$$precision = \frac{TP}{TP + FP}$$

与精度类似,*recall* 代表着恶意程序分类召回率。

$$recall = \frac{TP}{TP + FN}$$

虚警率(false positive rate,FPR)代表着良性程序中误分类为恶意的比例。

$$FPR = \frac{FP}{TN + FP}$$

漏警率(false negative rate,FNR)代表着恶意程序中误分类为良性的比例。

$$FNR = \frac{FN}{TP + FN}$$

为了进一步验证可达性对于模型分类的效果,进行了消融实验,比较了不包含可达性特征的 3 种特征和包含了可达性特征的 4 种特征在 5 种不同机器学习下的分类效果,5 种机器学习方法为决策树(decision tree),梯度提升树(gradient boosting decision tree, GBDT),逻辑回归(logistic regression, LR),随机森林(random forest,RF),支持向量机(support vector machines, SVM)。

实验结果如表 3 所示,两组特征在 5 种机器学习的方法下均为随机森林方法达到了最高的准确率,分别为 98.41%和 98.90%。4 种特征得到的模型,在准确率、精度和召回率方面均高于 3 种特征。两个特征集相比,5 种方法的准确率都有了明显地提升。

表 3 不同特征集下实验结果数据 %

特征集	采用方法	*accuracy*	*precision*	*recall*
3 种特征(权限 P、意图 I、应用接口 A)	decision tree	96.97	98.07	96.21
	GBDT	98.35	98.62	98.28
	LR	97.98	98.95	97.24
	RF	98.44	98.96	98.10
	SVM	98.07	98.61	97.76
4 种特征(权限 P,意图 I,应用接口 A,可达性特征 C)	decision tree	97.61	98.60	96.90
	GBDT	98.44	98.79	98.28
	LR	98.62	99.13	98.28
	RF	98.90	99.13	98.79
	SVM	98.72	98.96	98.62

通过实验可以发现,加入了可达性集合 C 之后,训练得到的模型的效果有了整体性的提高。在特征集合添加了可达性特征后,包含了边信息,提高了恶意程序的精度

和召回率，能够更好地区分良性与恶意程序。

图 6 所示为加入可达性特征之后 5 种机器学习准确率、虚警率和漏警率变化。其中，可达性特征对恶意程序检测中漏警率下降的效果比较明显，对虚警率方面的提升比较弱，由于可达性特征对 GBDT 在漏警率方面没有多少提升，导致最后的精度提升也不大。

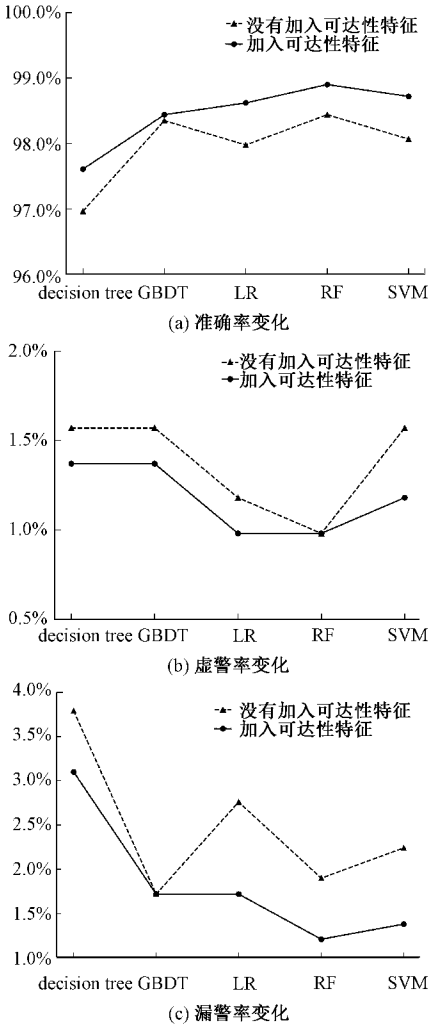


图 6　加入可达性特征后准确率、虚警率和漏警率的变化

在加入可达性特征之前，GBDT 的准确率高于 LR 的准确率。由于可达性特征的加入，明显降低了 LR 的漏警率，使得 LR 的准确率达到了 98.62%。

总体上，从 3 幅折线图中可以看出，加入了可达性特征之后，5 种机器学习的虚警率和漏警率均有所下降。

最后，本文基于随机森林训练得到了一个模型，用于衡量可达性特征的效果，主要两个方面进行衡量。一方面，计算可达性特征的特征重要性，另一方面，计算了特征在样本上的覆盖率。

图 7 所示为随机森林模型得到特征重要性排名中的前 20 个特征，其中前 18 个特征均为本文提取的可达性特征。可达性特征 valueOf-readline 获取到最高的特征重要性，到达了 0.031，而最后是 hasOnClickListeners 和 getMediaUri 两个 API 特征。

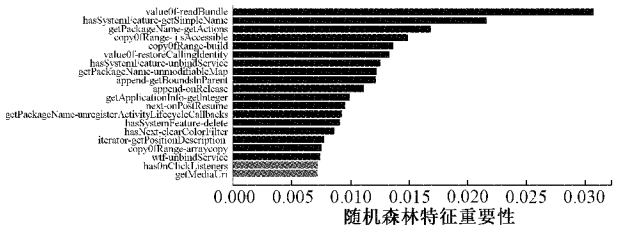


图 7　随机森林特征重要性高的前 20 个特征

由于应用程序结构复杂，并非简单的某个调用就能检测出来是否为恶意，所以特征重要性整体不高。但是通过实验可以发现，加入的可达性特征相比其余特征，在该模型中具有较高的重要性。

本实验还针对样本通过随机森林模型进行决策这个过程对可达性特征进行了说明，如表 4 所示，对具有较高特征重要性的特征进行了覆盖率的分析。

表 4　不同类型特征覆盖率

特征	覆盖率/%	特征类型	覆盖率排名
getPackageName-unmodifiableMap	90.37	可达性特征	4
valueOf-restoreCallingIdentity	88.44	可达性特征	5
valueOf-readBundle	87.71	可达性特征	6
startIntentSenderForResult	76.61	API 特征	10
copyOfRange-isAccessible	74.50	可达性特征	11
copyOfRange-build	72.20	可达性特征	13
hasSystemFeature-unbindService	71.93	可达性特征	15
getMediaUri	70.09	API 特征	16
hasOnClickListeners	61.38	API 特征	19
android.permission.CHANGE_WIFI_STATE	54.40	权限特征	29

针对样本使用树进行决策，会产生相应的路径，当路径中出现某个特征，则认为该特征覆盖到该样本，即对该样本的决策有帮助。对重要性较高的特征，其覆盖率也不低。分别对 3 种不同类型的特征进行了一些展示，覆盖率高的特征类型多数为可达性特征。而 API 特征覆盖率最高的为 76.61%，权限特征覆盖率最高的为 54.40%。

在随机森林模型训练过程，相比 API 特征，更偏向于使用可达性特征作为决策的分支。

4　结　　论

本文主要针对安卓恶意程序检测的准确率进行研究，通过提取应用接口之间的可达性特征来给特征集合中添

加边信息,从而提高精度。然而,可达性特征的提取较为耗时,主要来自于两个方面。一方面是程序中的应用接口较多,程序调用图的点数量和边数量一旦过多,可达性特征提取就会相当耗时。另一方面,应用接口的种类较多,分析每一对之间的可达性会耗时巨大,本文已通过特征选择减少特征数量和基于恶意行为来减少分析的可达性特征数量。即使减少数量后,分析可达性特征的效果会发现并非分析每对应用接口之间的可达性都能够有更好的区分性。下一步工作主要研究如何更好地选择可达性特征,避免区分能力较小的可达性特征的提取消耗时间,同时研究可达性特征与恶意行为之间的联系。

参考文献

[1] FENG Y, ANAND S, DILLIG I, et al. Apposcopy: Semantics-based detection of Android malware through static analysis[C]. ACM Sigsoft International Symposium on Foundations of Software Engineering, ACM, 2014:576-587.

[2] LI J, SUN L, YAN Q, et al. Significant permission identification for machine-learning-based Android malware detection [J]. IEEE Transactions on Industrial Informatics, 2018, 14(7): 3216-3225.

[3] MALIK J, KAUSHAL R. CREDROID: Android malware detection by network traffic analysis [C]. ACM Workshop on Privacy-Aware Mobile Computing, Paderborn: ACM, 2016: 28-36.

[4] ZULKIFLI A, HAMID I R A, SHAH W M, et al. Android malware detection based on network traffic using decision tree algorithm [C]. International Conference on Soft Computing and Data Mining, Springer, 2018: 485-494.

[5] HUANG J, ZHANG X, TAN L, et al. AsDroid: Detecting stealthy behaviors in Android applications by user interface and program behavior contradiction[C]. International Conference on Software Engineering, ACM, 2014: 1036-1046.

[6] DAMOPOULOS D, KAMBOURAKIS G, PORTOKALIDIS G. The best of both worlds: A framework for the synergistic operation of host and cloud anomaly-based IDS for smartphones [C]. European Workshop on System Security, ACM, 2014: 1-6.

[7] 张鹏涛,王维,谭营. 基于带有惩罚因子的阴性选择算法的恶意程序检测模型[J]. 中国科学:信息科学, 2011,41(7):798-812.

[8] YERIMA S Y, SEZER S. DroidFusion: A novel multilevel classifier fusion approach for Android malware detection [J]. IEEE Transactions on Cybernetics, 2019, 49(2): 453-466.

[9] 苏志达,祝跃飞,刘龙. 基于深度学习的安卓恶意应用检测[J]. 计算机应用,2017,37(6):1650-1656.

[10] HOU S, YE Y, SONG Y, et al. HinDroid: An intelligent Android malware detection system based on structured heterogeneous information network [C]. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017: 1507-1515.

[11] ZHANG W, WANG H, HE H, et al. DAMBA: Detecting Android malware by ORGB analysis [J]. IEEE Transactions on Reliability, 2020, 69(1): 55-69.

[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, 2013: 86-103.

[13] ARP D, SPREITZENBARTH M, HUBNER M, et al. DREBIN: Effective and explainable detection of Android malware in your pocket [C]. Network & Distributed System Security Symposium, 2014.

[14] 徐林溪,郭帆. 基于混合特征的恶意安卓程序检测方法[J]. 计算机工程与科学,2017,39(10):1837-1846.

[15] 刘晓建,雷倩,杜茜,等. 多上下文特征的 Android 恶意程序静态检测方法[J]. 华中科技大学学报(自然科学版),2020,48(2):85-90.

[16] RASTHOFER S, ARZT S, BODDEN E. A machine-learning approach for classifying and categorizing Android sources and sinks [C]. Network & Distributed System Security Symposium, 2014.

[17] ARZT S, RASTHORFER S, FRITZ C, et al. FlowDroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps [J]. ACM Sigplan Notices, 2014, 49(6): 259-269.

[18] ELISH K O, SHU X, YAO D D, et al. Profiling user-trigger dependence for Android malware detection [J]. Computers & Security, 2015, 49(3):255-273.

[19] XIE N, ZENG F, QIN X, et al. RepassDroid: Automatic detection of Android malware based on essential permissions and semantic features of sensitive APIs [C]. International Symposium on Theoretical Aspects of Software Engineering, 2018:52-59.

[20] 汪洁,王长青. 子图相似性的恶意程序检测方法[J]. 软件学报,2020,31(11):3436-3447.

作者简介

金泽宇(通信作者),硕士研究生,主要研究方向为恶意程序检测、机器学习。

E-mail: 2376882965@qq.com

朱正伟,教授,主要研究方向为智能检测技术及应用、嵌入式系统及应用。

E-mail: zhuzw@cczu.edu.cn