基于移动软件行为大数据挖掘的恶意软件检测技术

张 巍[1,2] 任 环[1,3] 张 凯[1,3] 李成明[1] 姜青山[1]

[1](中国科学院深圳先进技术研究院 深圳 518055)
[2](中国科学院大学深圳先进技术学院 深圳 518055)
[3](中国科学技术大学 合肥 230026)

摘 要 目前移动恶意软件数量呈爆炸式增长，变种层出不穷，日益庞大的特征库增加了安全厂商在恶意软件样本处理方面的难度，传统的检测方式已经不能及时有效地处理软件行为样本大数据。基于机器学习的移动恶意软件检测方法存在特征数量多、检测准确率低和不平衡数据的问题。针对现存的问题，文章提出了基于均值和方差的特征选择方法，以减少对分类无效的特征；实现了基于不同特征提取技术的集合分类方法，包括主成分分析、Kaehunen-Loeve 变换和独立成分分析，以提高检测的准确性。针对软件样本的不平衡数据，文章提出了基于决策树的多级分类集成模型。实验结果表明，文章提出的三种检测方法都可以有效地检测 Android 平台中的恶意软件样本，准确率分别提高了 6.41%、3.96% 和 3.36%。

关键词 Android 恶意软件检测；特征选择；特征提取；集合分类算法
中图分类号 TP 391.3 **文献标志码** A

Malware Detection Techniques by Mining Massive Behavioral Data of Mobile Apps

ZHANG Wei[1,2] REN Huan[1,3] ZHANG Kai[1,3] LI Chengming[1] JIANG Qingshan[1]

[1](*Shenzhen Institutes of Advanced Technology*, *Chinese Academy of Sciences*, *Shenzhen* 518055, *China*)
[2](*Shenzhen College of Advanced Technology*, *University of Chinese Academy of Sciences*, *Shenzhen* 518055, *China*)
[3](*University of Science and Technology of China*, *Hefei* 230026, *China*)

Abstract Currently, the number of mobile malware programs is explosively growing, and the increasingly large feature library poses challenges to security solution providers. Traditional detection methods cannot deal with the huge amount of data promptly and effectively. Mobile malware detection methods based on machine learning have problems of excessive numbers of features, low detection accuracy and unbalanced data. In this paper, a feature selection method based on the mean and variance of samples was proposed to reduce the features without affecting classification. Different feature extraction algorithms were implemented to construct an ensemble learning model for high detection accuracy, including Principal Component Analysis,

收稿日期：2015-12-11 **修回日期**：2016-01-25
基金项目：广东省部产学研结合项目(2013B091300019)
作者简介：张巍，博士研究生，研究方向为数据挖掘、钓鱼网站识别、移动恶意软件检测；任环，硕士研究生，研究方向为数据挖掘、恶意软件检测；张凯，硕士，研究方向为数据挖掘、移动恶意软件检测；李成明，博士，助理研究员，研究方向为互联网技术；姜青山(通讯作者)，研究员，博士研究生导师，研究方向为数据挖掘、网络安全，E-mail：qs.jiang@siat.ac.cn。

Kaehunen-Loeve Transformation and Independent Component Analysis. To solve the problem of unbalanced data of Android App samples, a multi-level classification model based on the decision tree was also developed. Experimental results show that the proposed methods can detect Android malware effectively, and the accuracy is increased by 6.41%, 3.96% and 3.36%, respectively.

Keywords Android malware detection; feature selection; feature extraction; ensemble classification

1 引 言

随着移动互联网的快速发展，移动终端用户群体日益庞大。《中国互联网络发展状况统计报告》[1]指出：2014 年我国网民整体规模稳步上升，手机网民规模达 5.57 亿，其中使用手机上网的人群占 85.8%，较去年提高了 4.8%。但是在经济利益的驱使下，近几年移动恶意软件的数量呈爆炸式增长，各种移动恶意软件家族更是千变万化，严重威胁着移动互联网的健康发展，给用户和智能终端带来了不可忽视的危害。同时，由于我国目前尚无针对移动应用商店安全的要求准则出台，这使得一些应用程序商店安全门槛过低，大量的恶意应用软件可以很轻易进驻应用商店并提供下载，移动互联网安全已经成为制约移动互联网发展的瓶颈。

移动恶意软件是指所有能够在智能手机或者平板计算机上执行恶意操作的应用程序，会导致系统崩溃、用户机密信息的损失或泄漏。中国反网络病毒联盟将手机恶意软件分为八个类别，包括：资费消耗、隐私窃取、恶意扣费、诱骗欺诈、流氓行为、系统破坏、远程控制以及恶意传播。奇虎 360 公司发布的《2014 年中国手机安全状况报告》[2]指出，2014 年 Android 平台的恶意软件的新增量和感染量都大幅增加。其中，新增恶意程序样本 326.0 万个，较去年增长了 3.86 倍；Android 用户感染恶意程序 3.19 亿人次，较去年增长了 2.27 倍。

移动恶意软件数据量的快速增长与变种的不断出现给恶意软件自动鉴别带来挑战，只有面向大数据的技术不断发展，才能将大数据时代带来的挑战变为机遇，更好地运用这些不断积累的样本，真正将海量数据变化为有效信息，并有效构建相适应的数据挖掘模型，才能实现对海量恶意软件样本的快速鉴别。针对 Android 恶意软件严峻的发展趋势与庞大的数据量，国内外研究学者提出了许多不同的解决方案。当前 Android 恶意软件检测技术主要分为基于特征码(signature-based)的检测技术和基于行为(behavior-based)的检测技术。基于特征码的检测技术和黑白名单技术被主要的安全软件厂商广泛使用，但该技术需要对已知的恶意软件样本进行全面的分析，无法抵御未知的或经过代码混淆、加壳技术处理过的恶意软件。基于行为的检测技术主要通过 Android 运行时的 API(Application Programming Interface，应用程序编程接口)调用、系统调用序列以及系统日志等特征，来分析软件的恶意行为。这种方法能够抵御混淆加密的恶意软件，但是系统实时性要求高。因此，目前研究集中于将模式识别技术应用到 Android 恶意软件，其主要流程包括恶意软件特征表示、特征提取，并结合相关分类算法，学习其中的鉴别规则。

恶意软件行为是指对恶意软件在操作系统中的执行流程，及其对操作系统资源的影响进行分析，然后采用数据挖掘技术对其进行鉴别。研究移动恶意软件行为特征与检测技术，实现对移动恶意软件自动、快速准确的鉴别是安全领域一项重要的研究内容。本文主要研究基于移动

软件行为大数据的恶意软件检测技术，包括：基于 Dalvik、API 和权限的 Android 恶意软件特征表示方式、基于频率均值和方差的特征选择算法、基于主成分分析(Principal Component Analysis，PCA)、Kaehunen-Loeve 变换(KLT)和独立成分分析(Independent Component Analysis，ICA)的特征提取算法，并研究了基于决策树的多级分类集成算法，实现 Android 平台移动恶意软件的快速准确鉴别。

2　移动恶意软件检测方法研究现状

针对智能手机的安全问题，各大杀毒软件公司都积极应对，纷纷投入到手机安全杀毒软件的研究之中。国内方面，奇虎 360[3]、腾讯[4]等手机安全软件则主要采用云端和本地双引擎扫描技术，给智能终端给予安全保护。360 手机卫士采用本地和云查杀的方法[3]，本地查杀调用手机卫士内置的杀毒功能，本地扫描已安装软件的信息，根据软件的包名、UID、版本号、证书以及特征码，与病毒库进行比对，判断软件的安全性，以便完成查杀。腾讯手机管家采用多引擎查杀[4]，具备双引擎的本地查杀功能和云查杀功能。本地查杀引擎使用 QQ 手机管家查杀病毒引擎和卡巴斯基查杀病毒引擎，在无需联网的情况下，可以快速地对本地已安装软件和即将安装软件进行病毒查杀。云查杀引擎在用户允许的前提下，终端会联网将本地的软件信息及行为特征上传到云端服务器，服务器根据所上传的信息进行精准的病毒扫描，将最终精确的查杀结果返回给终端。

在国外方面，诸多安全厂商依靠其传统查杀的先进技术和经验，在本地查杀有着较为领先的优势。如著名的 Kaspersky[5]实验室推出了卡巴斯基手机安全软件，该软件结合了传统的特征码技术和主动的启发式分析技术，并提供云安全扫描的实时保护，确保智能手机不受恶意软件的侵害。McAfee[6]通过扫描文件、内存卡、应用程序、Internet 下载项、文本消息和附件并清除其中的恶意软件来保护移动设备。

针对恶意软件频繁变种、数量暴涨的现状，以提高软件行为鉴别精度、降低恶意软件的误报率、有效解决客户端与云端动态数据交互过程中的隐私保护问题为目标，我们提出了基于云计算平台的软件行为鉴别技术实用系统以及在线匿名化隐私保护系统[7]。通过在线环境隐私模型、数据分析模型及挖掘方法等技术，解决在线动态隐私保护问题以及从海量软件样本中鉴别出恶意软件，是应对当前恶意软件层出不穷、频繁变种现状的一种现实选择，也必将是业界雄心勃勃的“云安全”计划的一条重要实现途径。基于云计算平台的软件行为鉴别系统结构如图 1 所示[7]。

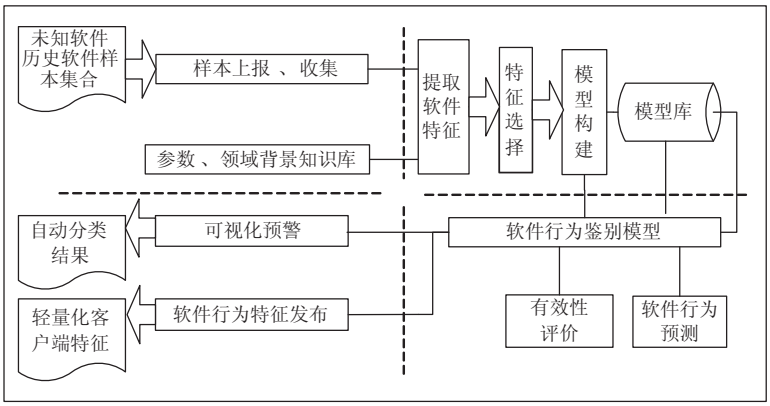

图 1　恶意软件检测系统结构图

Fig. 1　Framework of malware detection system

训练过程首先从已知的软件行为大数据样本恶意文件和正常文件中抽取相应的特征来表示样本，然后将特征数据存储到数据库中，再通过相应的学习算法构建分类模型。模型在初次生成之后，还可以通过一些专家经验对模型中不够准确的部分进行修正，以提高检测精度。而对于未知文件，同样经过特征提取选择后，结合决策引擎及生成的检测模型对样本进行预测，判断出文件类别，并给出预测结果。移动恶意软件是恶意软件的一部分，也可以用恶意软件的检测流程进行处理。目前，机器学习和数据挖掘技术被广泛应用到 Android 恶意软件检测中。基于数据挖掘的恶意软件检测方法流程如图 2 所示[8,9]。其中，检测流程包括：使用特征提取对恶意软件样本特征量化，如空间向量模型，将权限、API、URL 以及字符串等特征转换成数值特征；然后采用特征选择算法选取特征子集；最后，结合相关的分类、聚类算法和已知的样本类别标签构建最优化的检测模型，并运用该检测模型对未知软件进行预测。基于数据挖掘的恶意软件检测技术的主要工作在于特征与模型的构建，它不依赖人工制定的相关安全策略，能够有效地发现不同类别样本特征间的区别与联系，并以此作为分类的依据，具有推广性强，能够对抗混淆加密的恶意软件等优点。

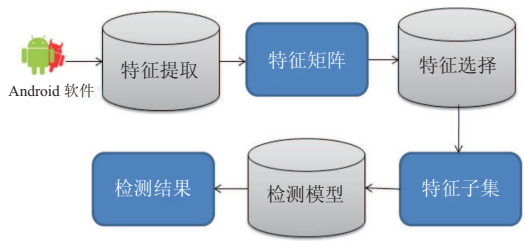

图 2 基于数据挖掘技术的 Android 恶意软件检测流程

Fig. 2 Android malware detection based on data mining

目前，Android 恶意软件检测的理论研究工作主要集中在 Android 恶意软件的特征行为分析，包括动态分析和静态分析。动态分析是指在安装程序执行的过程中，对软件行为进行分析；静态分析是指在安装程序执行之前，抽取程序的特征，如 Dalvik、API 和权限特征。在动态分析方面，Bläsing 等[10]提出了使用 Android 应用程序检测沙箱，通过系统调用重定向方法，监视系统和库函数的调用，记录其运行时的行为。他们使用 Monkey 工具，生成伪随机事件流，对应用程序进行模拟运行。沙箱在内核中，记录应用程序的行为，追踪应用程序在系统中运行时的相关系统调用。Shabtai 等[11]实现了基于行为的恶意软件检测系统，该系统可以检测手机设备的各种行为和特征，并通过机器学习算法，将这些行为和特征进行分类。然而，在程序运行时执行动态检测，对实时性要求较高，必须确保在恶意程序对系统产生损害前检测出威胁。此外，利用沙盒、虚拟机来模拟执行程序，不可避免地带来了更大的能耗。

与动态行为检测相比，静态行为检测通过逆向工程手段，无须使用沙盒、虚拟机，检测能耗更低，风险更小，对实时性要求更低。Apvrille 等[12]利用权限、API、系统命令以及 URL 等特征的权重，计算恶意软件与正常软件之间的差异。佟得天[13]对 Android 手机木马在系统日志记录中的具体表现形式进行研究，归纳出手机木马行为检测规则。路程[14]通过静态分析 Android 应用程序反编译后的 Java 源代码，以及动态监控软件行为。李涛等[15]通过拦截 Android 的广播信息和智能分析机制，统计和分析用户的行为习惯，从而拦截可疑的恶意行为。部分研究者[16-20]则将 Android 应用软件运行时的系统调用序列应用于恶意软件检测中。Xu 等[21]提出了一个新的框架 Permlyzer 来追踪 Android 权限的使用情况。Dai 等[22]使用 API 拦截技术来动态分析应用程序的行为，并匹配恶意行为特征库来检测恶意软件。

现有的研究工作主要依赖 Android 应用软件的权限、API、系统调用序列以及其他字符特

征，这些特征能够有效表示 Android 应用软件的行为，具有良好的检测效果，但是这些特征都需要经过人工的统计筛选。Dalvik 指令作为 Android 虚拟机运行时必要的指令集信息，反映了 Android 应用程序直接对寄存器的操作行为。Kang 等[23]首次利用 Dalvik 指令特征对 Android 恶意软件家族进行分类，取得了良好的分类结果。应用更多的软件行为特征及其组合能够有效地提高模型的泛化能力，为此，我们提出了基于 Dalvik 指令、API 和权限多种特征的移动恶意软件检测技术[24]。

为了实现对海量未知的 Android 应用软件准确地鉴别，软件行为特征的提取一定要有效、自动和高效。为了实现以上目标，本文研究基于均值和方差的特征选择算法，以提取利用分类的特征、降低特征维数、减少分类训练时间；提出移动软件行为样本特征提取新方法，将原始特征映射到不同特征空间，以提高移动恶意软件检测的准确性。

3　移动恶意软件检测技术与系统架构计

移动恶意软件检测系统主要由基于均值与方差的特征选择算法、基于特征提取的恶意软件检测算法和基于多级集成的恶意软件检测算法组成，我们研发的系统结构图如图 3 所示。该系统各个部分的功能如下：

(1)特征抽取：该部分从所收集的样本中抽取 Permission 特征、API 特征与 Dalvik 特征；

(2)特征提取：使用不同的特征提取方法对原始特征进行处理；

(3)集合学习：采用了多级集成方式进行分类融合。

3.1　基于均值与方差的特征选择算法

Dalvik 指令作为 Android 应用软件在 Dalvik 虚拟机运行时必要的信息，比 API 更加接近系统的底层，反映了 Android 应用软件对寄存器的操作行为，具有指令数量少、易于提取等优点，是一种有效的 Android 恶意软件鉴别特征。本章将研究 Dalvik 指令特征对恶意软件检测的有效性。Android 应用软件在编译时，Java 字节码被自动转换成 Dalvik 虚拟机(Dalvik VM)所使用的指令，并保存于 classes.dex 中。classes.dex 文件存在于每个 Android 应用软件中，是可以在 Dalvik VM 上直接运行的可执行文件。Dalivk VM 在运行时通过 Dalvik 指令来访问寄存器操作数，主要包括：空操作指令(nop)、数据操作指令(move)、返回指令(return)、数据定义指令

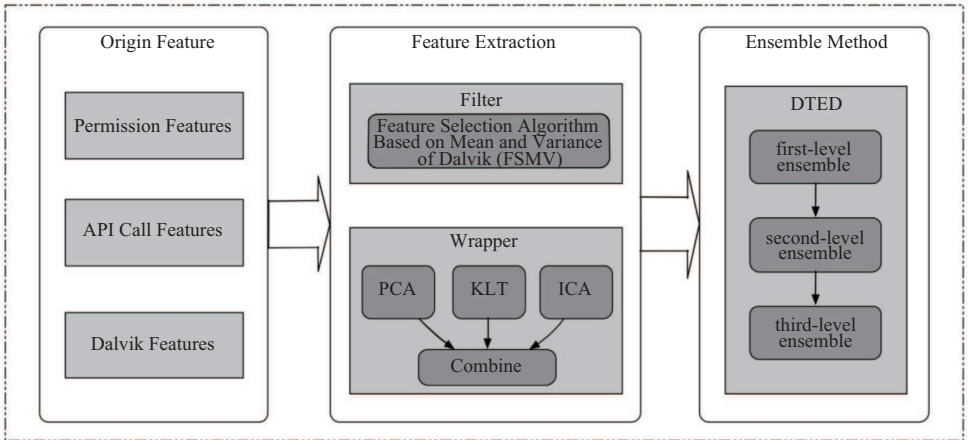

图 3　移动恶意软件检测系统框架图

Fig. 3　Mobile malware detection system

(const)、方法调用指令(invoke)、数据转换指令以及数据运算指令等。本文基于 Dalvik 指令相对频率的表示方法及特征选择算法，该部分的主要成果详细论述参见文献[24]，提出了基于频率均值和方差的特征选择算法。

Dalvik 指令相对频率之间的差异是鉴别恶意软件重要的评价指标，恶意软件部分 Dalvik 指令相对频率的均值要明显高于正常软件，即恶意软件比正常软件更多地调用部分 Dalvik 指令。如图 4 所示，恶意软件部分指令的方差也明显高于正常软件，这说明少数恶意软件调用 Dalvik 指令与其他恶意软件具有明显不同。文章提出基于 Dalvik 指令的均值和方差的特征选择算法(The Feature Selection Algorithm Based on the Mean and Variance of Dalvik Instructions，FSMV)。

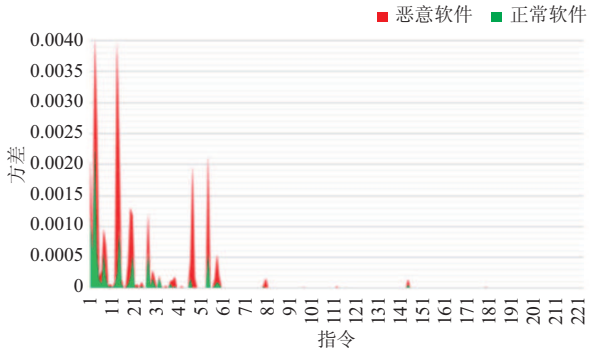

图 4 正常软件与恶意软件 Dalvik 指令相对频率的方差

Fig. 4 Comparison of Dalvik instructions variance relative frequency between normal and malicious APP

恶意软件与正常软件调用 Dalvik 指令的相对频率之间的差异越大越好，同时同类别样本的相对频率越聚集越好，即同类别样本点距离越近越好，不同类别间的样本点距离越远越好。文章引用线性判别分析(Linear Discriminant Analysis，LDA)[25]中的公式来评价不同 Dalvik 指令分类能力的强弱，如公式(1)所示 。

$$J(A)=\frac{|\mu_b-\mu_m|^2}{S_b^2+S_m^2} \tag{1}$$

其中，μ_b 和 μ_m 分别表示正常软件与恶意软件相对频率的平均值；S_b 和 S_m 分别表示正常软件与恶意软件相对频率的方差。由公式(1)可以看出：特征选择的目标是最大化 $J(A)$ 值，即最小化两类样本间的方差和及最大化两类样本间的平均值之差。我们对每个 Dalvik 指令计算 $J(A)$ 值，$J(A)$ 值越大表示特征 A 的分类能力越强，并按照 $J(A)$ 值的大小对特征进行降序排序，最终选择前 k 个 Dalvik 指令特征作为有效的特征子集。详细的算法流程如图 5 所示。

输入：样本集 S，Dalvik 指令 A；

输出：特征子集 W_k；

开始：

(1) 计算正常软件的 Dalvik 指令相对频率的平均值 μ_b 和方差 S_b；

(2) 计算恶意软件的 Dalvik 指令相对频率的平均值 μ_m 和方差 S_m；

(3) 按照公式(1)计算指令的权重；

(4) 对 Dalvik 指令进行降序排序；

(5) fork←1 to p do //p 为 Dalvik 指令的总数

a)取前 k 个指令作为 W_k；

b)不同分类器 Cls 验证 W_k 结果；

(6) 选择该分类器下最优的特征子集；

结束

图 5 基于 Dalvik 指令均值和方差的特征选择算法

Fig. 5 Feature selection algorithm based on mean and variance of Dalvik instructions

3.2 基于特征提取的恶意软件检测算法

为了提高检测精度，采用特征提取算法将特征映射到不同的特征空间，并提出了一个新的恶意软件检测框架，如图 6 所示。

首先从源码中提取 Dalvik 指令；然后使用主成分分析，Kaehunen-Loeve 变换与独立成分分析，将原始的 Dalvik 指令映射到相应的特征空间，得到 3 个新的特征；最后，使用极速学习机算法学习单层神经网络，将其作为基分类器。使用 Stacking 方法融合每一个基分类器，该部分的主要成果可参考文献[26]。

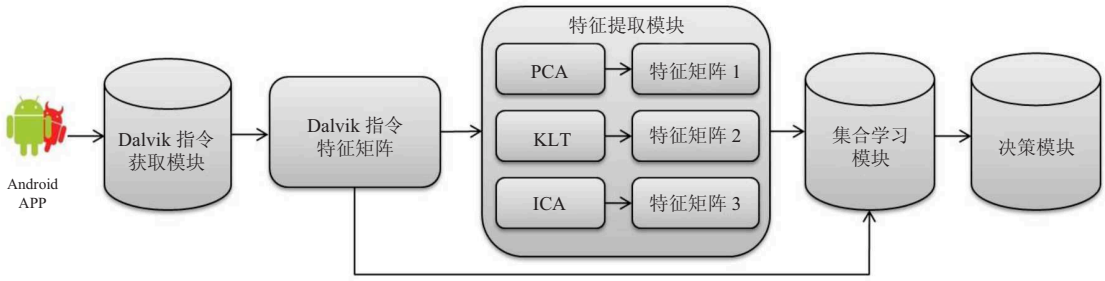

图 6　基于特征提取的恶意软件检测系统流程图

Fig. 6　Framework of feature extraction-based malware detection system

3.2.1 特征提取算法

本文使用了三种不同的特征提取算法，将特征映射到不同的特征空间。三种算法的简要介绍如下。

(1) 主成分分析：主成分分析是一种线性投影，将高维特征映射到低维的一种技术。在投影过程中，数据的方差最大。定义 N 个样本 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_N\}$，其中 $\boldsymbol{X}_i=\left[\boldsymbol{X}_{i,1},\boldsymbol{X}_{i,2},\cdots,\boldsymbol{X}_{i,d}\right]^{\mathrm{T}}\in\mathbf{R}^d$；$d$ 是特征数量。主成分分析将 $\boldsymbol{X}_i$ 线性变换为一个新的样本：

$$y_i=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{X}_i \tag{2}$$

其中，$\boldsymbol{U}$ 是正交矩阵；$\boldsymbol{U}$ 的第 i 列 u_i 是样本协方差矩阵的特征向量。主成分分析首先计算样本的协方差矩阵，然后计算协方差矩阵的特征向量，根据公式(2)将原始特征进行变换。

(2) Kaehunen-Loeve 变换：是主成分分析的扩展。在主成分分析中，变换矩阵是样本的协方差矩阵。在本文的 Kaehunen-Loeve 变换中，变换矩阵是类间离散度矩阵，记为 $\boldsymbol{S}_w$。

$$\boldsymbol{S}_w=\sum_{i=1}^{L}P_iE\left[(\boldsymbol{X}-\overline{m_i})(\boldsymbol{X}-\overline{m_i})^{\mathrm{T}}\right] \tag{3}$$

其中，L 是样本类别；P_i 是第 i 个类别的概率；E 代表数学期望；$\overline{m_i}$ 是第 i 个类别的均值。计算 $\boldsymbol{S}_w$ 的特征向量，然后根据公式(2)计算得到新的特征。

(3) 独立成分分析：从原始的特征中提取独立的信息[27]。独立成分分析模型记为：

$$\boldsymbol{X}=\boldsymbol{A}s \tag{4}$$

其中，$\boldsymbol{X}$ 为原始数据；$\boldsymbol{A}$ 是满秩矩阵；s 是独立成分。独立成分分析的目的，就是从 $\boldsymbol{X}$ 中提取出独立成分 s。

$$\hat{s}=\boldsymbol{U}\boldsymbol{X} \tag{5}$$

其中，$\hat{s}$ 表示独立成分 s 的估计值；$\boldsymbol{U}$ 为变换矩阵。本文使用 FastICA 算法计算 $\boldsymbol{U}$[27]。

3.2.2 基于极速学习机的集合分类器

我们采用极速学习机(Extreme Learning Machine，ELM)算法训练单层神经网络，然后将其作为基分类器，采用 Stacking 方法组成集合分类器。极速学习机算法首先为神经网络随机分配输入权值向量与偏差，然后分析确定神经网络的输出权值，其算法流程如下。

在训练阶段，随机分配输入权值与偏差，计算隐藏层节点的输出：

$$h_{ij}=g(\boldsymbol{w}_jx_i+b_j)\quad i=1,2,\cdots,N;\ j=1,2,\cdots,k \tag{6}$$

其中，h_{ij} 是第 j 个隐藏层节点的输出；$\boldsymbol{w}_j=[w_{j1},w_{j2},\cdots,w_{jn}]^{\mathrm{T}}$是连接第 j 个隐藏节点与输入数据的权值；b_j 是第 j 个隐藏层节点偏差；N 是样本数量；k 是隐藏层节点数量；g 是激活函数。隐藏层输出矩阵记为 $\boldsymbol{H}=\{h_{ij}\}$，连接隐藏层与输出层节点的权值向量记为 $\boldsymbol{\beta}$：

$$\hat{\boldsymbol{\beta}}=\boldsymbol{H}^{\dagger}\boldsymbol{T} \tag{7}$$

其中，$\hat{\boldsymbol{\beta}}$ 是 $\boldsymbol{\beta}$ 的估计值；$\boldsymbol{H}^{\dagger}$ 是 $\boldsymbol{H}$ 的 Moore-Penrose 的广义逆矩阵；$\boldsymbol{T}$ 是分类标签。在测试阶段，对于未知样本，首先计算其隐藏层节点输

出 $\boldsymbol{H}$，然后预测其标签。

$$\boldsymbol{T}=\hat{\boldsymbol{\beta}}\boldsymbol{H} \tag{8}$$

其中，$\boldsymbol{T}$ 是未知样本的预测标签；$\boldsymbol{H}$ 是未知样本的隐藏层输出，根据公式(6)计算。本文采用 Stacking 方法建立集合分类器[28]。Stacking 基于学习的方法来融合每一个基分类器，包括两层，记为 level-0 和 level-1。在 level-0 中，使用极速学习机，建立 N 个基分类器。在 level-1 中，将 level-0 中预测的标签作为输入数据，同样采用极速学习机进行训练。

3.3 基于多级集成的移动恶意软件检测算法

基于多级集成的移动恶意软件的检测算法主要利用 Android 软件的权限和 API 特征。通过深入分析发现，权限特征和 API 特征在正常软件、恶意软件调用的频率存在一定的差异。因此，文章选取了调用频率排名前 20 的权限和 API 分别进行了比较，结果如图 7 和图 8 所示。在现实生活中，正常软件的数量远超过恶意软件的数量，这便导致数据集存在不平衡数据的问题。通常解决不平衡数据的方法主要包括少数样本的重复抽样、多数样本的欠抽样、SVM(支持向量机)、Random Forest(随机森林)等。一般情况下，针对恶意软件，通常会着重恶意软件的检测，即尽量使恶意软件误报成正常软件的比率最低。目前，常用的方法是对不同样本分配不同的训练权重、集成方法等。

从图 7 可看出，正常软件与恶意软件申请 WRITE_EXTENRNAL_STORAGE、ACCESS_NETWORK_STATE 等权限的频率很高，但两者之间的差异性较小；而恶意软件申请 READ_CONTACTS、SEND_SMS 等权限的频率要远高于正常软件申请的频率。因此，权限作为区分正常软件与恶意软件的特征是有效的。

从图 8 可看出，正常软件与恶意软件调用 android.content.Content、android.app.Activity 等 API 的频率很高，但两者之间的差异性较小。而恶意软件调用 android.graphics.Paint、android.View.KeyEvent 等 API 的频率要高于正常软件调用频率。

针对以上两个问题以及为了提高检测准确率，我们提出了一种新的基于决策树的集成学习方法(Decision Tree Ensembles based Detection，

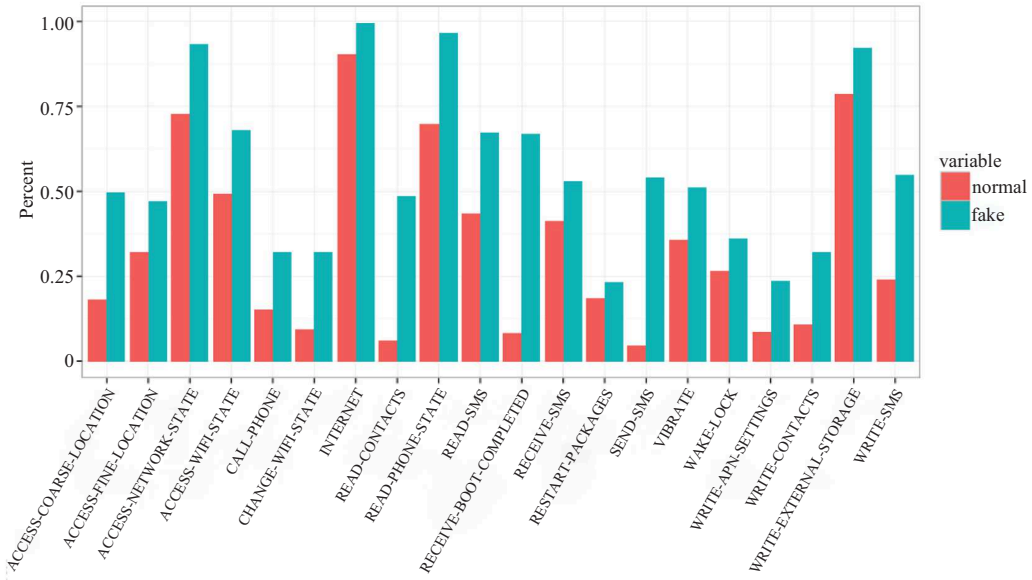

图 7 正常软件与恶意软件的权限调用频率差异比较

Fig. 7 Comparison of requested permission by normal APPs and malwares

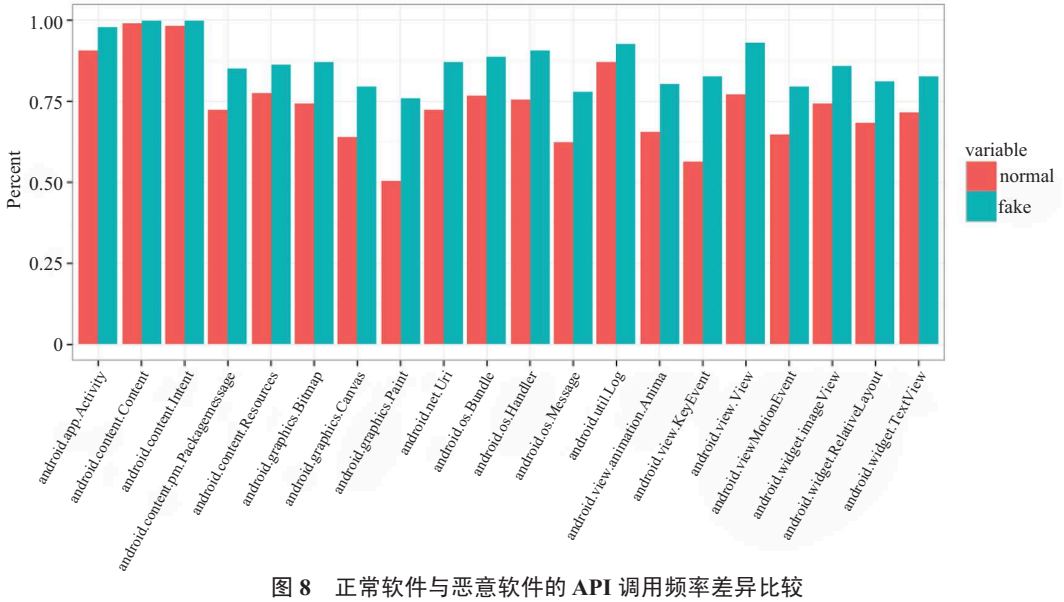

图 8　正常软件与恶意软件的 API 调用频率差异比较

Fig. 8　Comparison of requested API by normal APPs and malwares

DTED)。该集成方法是由三层决策树集成构成，其框架如图 9 所示。文章提出的移动恶意软件检测框架基于如下三部分组成：

(1) 第一层集成采用全投票的方法，即所有决策树的分类结果一样，这样基本可以使不平衡数据达到平衡，能很好地解决数据不平衡问题；

(2) 第二层集成采用多数投票方法，针对第一层未检测出的样本，结合第一层决策树的多数投票，将第二层集成结果和第一层决策树的多数投票结果不同的样本交给第三层集成方法处理，该技术很大程度上降低了恶意软件的误报率；

(3) 第三层集成采用少数投票，一般通过上述两层的检测，剩余未检测出的样本比较顽固，在缺少很好的特征刻画情况下，采用经验的少数投票，能提升一定的准确率。

该部分的主要成果已被 2015 5th International Conference on Electrical Engineering and Informatics (ELEEI 2015) 接收并发表[29]。

通过比较经典的数据挖掘算法，DTED 在不平衡数据处理、高准确率以及低漏报率方面有很好的效果。其原因在于：第一层集成方法能够使

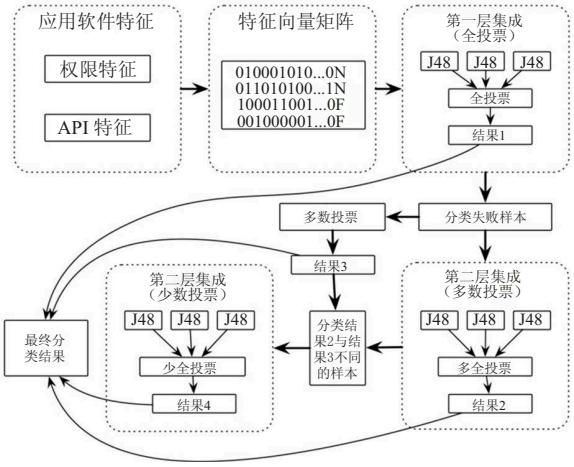

图 9　基于多级集成的恶意软件检测框架

Fig. 9　Framework of malware detection based on multi-level ensemble classification

大部分正样本检测出来，这就使得正样本和负样本数量达到平衡状态，即处理了不平衡问题；第二层集成是结合第一层集成，加重负样本的权重，使得负样本的检出率增高，这就能够降低负样本的漏报率；第三层集成是加强第二层集成结果，使整个检测准确率提高。总之，DTED 方法能够在不平衡问题、低漏报率和高准确率方面有着很好的表现。

4 结果分析与评估

4.1 数据集与评价指标

数据集 D1 样本来自 Google Play、豌豆荚以及安卓在线[30]，共 29 110 个正常样本；恶意软件样本来自 Android Malware Genome Project[31]和 Andro MalShare[32]，共 29 202 个样本。数据集 D2 中的恶意样本来自 Android Malware Genome Project，正常软件全部来自于 Google Play。其中，数据集 D1 用于测试特征选择算法，数据集 D2 用于评价特征提取算法，具体如表 1 所示。

表 1 数据集

Tabel 1 Experimental data sets

数据集	恶意软件	正常软件	总数
D1	29 202	29 110	58 312
D2	1 260	1 260	2 520

分类算法有很多，不同分类算法又有很多不同的变种。不同的分类算法有不同的特定，在不同的数据集上的表现效果也不同。我们需要根据特定的任务进行算法的选择，如何选择分类，如何评价一个分类算法的好坏。一般情况下，选用准确率(Accuracy)和 F-measure 作为评价指标。

首先介绍几个常见的模型评价术语，现在假设我们的分类目标只有两类，计为正例(Positive)和负例(Negative)分别是：

(1) True positives(TP)：被正确地划分为正例的个数，即实际为正例且被分类器划分为正例的实例数(样本数)；

(2) False positives(FP)：被错误地划分为正例的个数，即实际为负例但被分类器划分为正例的实例数；

(3) False negatives(FN)：被错误地划分为负例的个数，即实际为正例但被分类器划分为负例的实例数；

(4) True negatives(TN)；被正确地划分为负例的个数，即实际为负例且被分类器划分为负例的实例数。

分类评价指标如下：

(1) 正确率(Accuracy)：正确率是最常见的评价指标，Accuracy=(TP+TN)/(P+N)，表示被分为对的样本数除以所有的样本数；通常来说，正确率越高，分类器越好；

(2) 精度(Precision)：精度是精确性的度量，表示被分为正例的示例中实际为正例的比例，Precision=TP /(TP+FP)；

(3) 召回率(Recall)：召回率是覆盖面的度，Recall=TP /(TP+FN)；

(4) F-measure：F-measure 是召回率和精准率平均调和数：

$$F\text{-}measure=\frac{2\times Recall\times Precision}{Recall+Precision}$$

4.2 实验结果

为了测试提出的 FSMV 分类算法，本文使用逻辑回归、C4.5 决策树和随机森林作为分类器，并与序列前向选择算法(SFS)、爬山算法、Relief 算法进行对比，使用数据集 D1 的分类结果如表 2 所示。从表 2 可以看出：基于 Dalvik 指令均值和方差的特征选择算法(FSMV)能够有效提高多种分类算法的检测。对于 Dalvik 指令特征的检测结果，其中 FSMV 结合随机森林的算法比单独使用随机森林的算法提高 6.41%，低于 Relief 算法，而结合逻辑回归和 C4.5 决策树的检测准确率则高于 Relief。

为了测试基于特征提取的恶意软件系统，本文采用数据集 D2，随机选取 80% 的数据作为训练数据，20% 作为测试数据，结果如表 3 所示。由表可以看到，ELM+PCA 的 F-measure 值比仅使用 ELM 算法高，ELM+KLT 与 ELM+ICA 的 F-measure 比单独使用 ELM 低；仅使用 Stacking 的精度与 F-measure 均高于单独使用 ELM 的方法，使用基于特征提取的 Stacking 方法的精度和 F-measure 均高于其他组合，表明其有效性。

表 2　不同特征选择算法的分类准确率

Table 2　Experimental results of different feature delection methods

分类器	准确率（%）				
	无特征选择	SFS	爬山算法	Relief 算法	FSMV
逻辑回归	82.56	63.56	78.80	82.42	**82.69**
C4.5 决策树	87.03	77.74	86.94	87.52	**87.60**
随机森林	83.97	79.80	82.25	**90.91**	90.38

表 3　特征提取分类结果

Table 3　Experimental results of feature extraction

分类器	准确率（%）	F-measure（%）
ELM	93.56	93.43
ELM＋PCA	92.10	93.81
ELM＋KLT	93.79	90.73
ELM＋ICA	93.55	92.94
Stacking	95.11	93.58
FES	**97.52**	**97.52**

为了测试基于多级集成的恶意软件方法，本文采用数据集 D1，随机选取 70% 的数据作为训练数据，30% 作为测试数据。通常情况下，在不平衡数据、低漏报率和准确率的要求下，一般采用 F-measure 值作为评价标准，结果如表 4 所示。由表可以看到，对比常用的解决不平衡数据的方法，如 SVM、随机森林等，DTED 方法不仅在准确率上要高于 SVM 和随机森林，而且 F-measure 同样高于其他方法，这就表明 DTED 方法不仅在不平数据处理上有很好的建树，而且在低漏报率和高准确率的要求下也能很好的满足要求。因此，DTED 方法具有较好的检测效果。

表 4　不同算法的分类准确率及 F-measure 值

Table 4　Detection accuracy and F-measure of different classifiers

算法	准确率（%）	F-measure（%）
J48 决策树	88.24	87.42
SVM	88.12	87.20
随机森林	89.87	89.14
DTED	**91.60**	**91.13**

5　结　论

移动互联网的快速发展促进了移动恶意软件的增长，Android 平台因其开放性，成为恶意软件的主要攻击目标。因此，如何对发布到互联网、应用商店中的应用程序的安全性进行评价，识别出恶意软件是目前研究的一个热点。文章分析了移动终端恶意软件检测技术现状及其存在问题，提出了从源代码的角度分析软件的特征，研究了基于 Dalvik 指令、Permission 和 API 的 Android 恶意软件检测关键技术，包括特征选择算法、特征提取算法与多级集成分类算法。实验结果表明，本文提出的基于源代码的恶意软件检测技术可以有效地检测出 Android 恶意软件。

参 考 文 献

[1]　三川. CNNIC 发布第 35 次《中国互联网络发展状况统计报告》[J]. 中国远程教育, 2015, 1(2): 31.

[2]　Lijun. 360 发布手机安全报告 恶意程序去年增长 4 倍 [J]. 计算机与网络, 2015, 41(Z1): 89.

[3]　维普. 360 发布用户隐私保护白皮书接受用户监督[EB/OL]. [2013-08-13]. http://www.360.cn/privacy/v2/360shoujiweishi.html.

[4]　腾讯移动安全实验室. 基于 MTAA 的安全解决方案白皮书 [EB/OL]. [2011-06-17]. http://www.cnbeta.com/articles/146321.htm.

[5]　卡巴斯基实验室. KSC 技术白皮书 [EB/OL]. [2015-11-01]. http://slytt.com/144574605/.

[6]　迈克菲. McAfee VirusScan Mobile Security [EB/OL]. [2015-11-01]. http://161.69.29.243/cn/

resources/data-sheets/ds-virusscan-mobile.pdf.
[7] 庄蔚蔚, 姜青山. 恶意软件鉴别技术及其应用[J]. 集成技术, 2012, 1(1): 55-64.
[8] Aafer Y, Du WL, Yin H. DroidAPIMiner: mining API-level features for robust malware detection in Android [M] // Security and Privacy in Communication Networks. Springer Internation Publishing, 2013, 127: 86-103.
[9] Cen L, Gates CS, Si L, et al. A probabilistic discriminative model for android malware detection with decompiled source code [J]. IEEE Transactions on Dependable and Secure Computing, 2014, 12(4): 400-412.
[10] Bläsing T, Batyuk L, Schmidt AD, et al. An Android Application Sandbox system for suspicious software detection [C] // The 5th IEEE International Conference on Malicious and Unwanted Software (MALWARE), 2010: 55-62.
[11] Shabtai A. Malware detection on mobile devices [C] // Eleventh International Conference on Mobile Data Management, 2010: 289-290.
[12] Apvrille A, Strazzere T. Reducing the window of opportunity for Android malware Gotta catch'em all [J]. Journal in Computer Virology, 2012, 8(1-2): 61-71.
[13] 佟得天. 基于行为分析的 Android 手机木马检测技术研究 [D]. 广州: 中山大学, 2012.
[14] 路程. Android 平台恶意软件检测系统的设计与实现 [D]. 北京: 北京邮电大学, 2012.
[15] 李涛, 秦海超. 基于 Android 手机安全防护系统的研究 [J]. 现代计算机, 2011(24): 52-55.
[16] Burguera I, Zurutuza U, Nadjm-Tehrani S. Crowdroid: behavior-based malware detection system for Android [C] // Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, 2011: 15-26.
[17] Abela KJ, Angeles DK, Alas JRD, et al. An automated malware detection system for android using behavior-based analysis AMDA [J]. International Journal of Cyber-Security and Digital Forensics (IJCSDF), 2013, 2(2): 1-11.
[18] 冯博, 戴航, 慕德俊. Android 恶意软件检测方法研究 [J]. 计算机技术与发展, 2014, 24(2): 149-152.
[19] Tchakounté F, Dayang P. System Calls Analysis of Malwares on Android [J]. International Journal of Science and Technology, 2013, 2(9): 669-674.
[20] Isohara T, Takemori K, Kubota A. Kernel-based behavior analysis for android malware detection [C] // 2011 Seventh International Conference on Computational Intelligence and Security (CIS), 2011: 1011-1015.
[21] Xu W, Zhang F, Zhu S. Permlyzer: analyzing permission usage in android applications [C] // 2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE), 2013: 400-410.
[22] Dai S, Liu Y, Wang T, et al. Behavior-based malware detection on mobile phone [C] // 2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM), 2010: 1-4.
[23] Kang B, Kang BJ, Kim J, et al. Android malware classification method: dalvik bytecode frequency analysis [C] // Proceedings of the 2013 Research in Adaptive and Convergent Systems, 2013: 349-350.
[24] Zhang K, Jiang QS, Zhang W, et al. An android malware detection method using dalvik instructions [C] // Proceedings of International Conference on Informatics, Networking and Intelligent Computing, 2014.
[25] Scholkopf B, Mullert KR. Fisher discriminant analysis with kernels [J]. Neural Network for Signal Processing IX, 19991(1): 1.
[26] Zhang W, Ren H, Jiang QS, et al. Exploring feature extraction and ELM in malware detection for Android devices [M] // Advance in Neural Network. Springer International Publishing, 2015, 9377: 489-498.
[27] Hyvarinen A. Fast and robust fixed-point algorithms for independent component analysis [J]. IEEE Transactions on Neural Networks, 1999, 10(3): 626-634.
[28] Zhou ZH. Ensemble Methods: Foundations and Algorithms [M]. CRC Press, 2012.
[29] Ren H, Zhang W, Jiang QS. A new decision tree ensembles method for Fake Apps detection in Android [C] // 2015 4th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering, 2015.
[30] 安卓在线 [EB/OL]. [2014-09-16]. www.androidonline.net.
[31] Zhou Y, Jiang X. Android and Malware Genome Project [DB/OL]. [2015-01-12]. http://www.malgenomeproject.org/.
[32] Botnet Research Team. Android Malware Share [DB/OL]. [2015-01-12]. http://202.117.54.231:8080/#home.