基于特征生成方法的 Android 恶意软件检测方法

冯 垚,王金双,张雪涛
(陆军工程大学 指挥控制工程学院,江苏 南京 210001)

摘 要:针对传统特征工程中需要大量专家经验和人力的不足,研究了基于特征生成方法的 Android 恶意软件检测方法。基于 UC Berkeley 的 ExploreKit 自动特征生成方法,通过对原始特征计算获得大量候选特征,根据候选特征的元特征预测其性能并进行评估排序,使用贪心算法从中选出能够提升模型性能的新特征。从 APK 中提取了敏感 API、危险权限等多种特征,在根据信息增益对特征进行筛选后,输入到特征生成框架中,使用 C4.5、SVM 和随机森林等作为分类模型。实验证明,该方法使错误率平均降低了 24.6%,准确率达到了 96.5%,曲线下面积(Area Under Curve,AUC)达到了 0.99。

关键词:恶意软件检测;特征工程;特征生成;ExploreKit
中图分类号:TP393 **文献标识码**:A **DOI**:10.19358/j.issn.2096-5133.2020.11.002
引用格式:冯垚,王金双,张雪涛. 基于特征生成方法的 Android 恶意软件检测方法[J].信息技术与网络安全,2020,39(11):8-13.

Android malware detection based on feature generation method

Feng Yao, Wang Jinshuang, Zhang Xuetao
(Institute of Command Control Engineering, Army Engineering University, Nanjing 210001, China)

Abstract: Considering the expert experience and manpower required in traditional feature engineering, a detection method based on feature generation was proposed in this paper. ExploreKit was adopted as the automated feature generation method, which can improve Android malware detection model performance by generating and selecting new features. A large of candidate features are obtaioned by calculating the initial features. According to the meta-features of the candidate features, their performance is predicted and evaluated. New features that will improve the performance of the model are selected by greedy algorithms. The time spent on feature generation is reduced by filtering the initial features input to ExploreKit and limiting the number of generated features. Many features were extracted from APK, such as sensitive API and dangerous permission. By using C4.5, SVM and random forest as the classification models, our experiments show that the error rate of Android malware detection decreases 24.6% on average, the accuracy reaches 96.5%, and the AUC reaches 0.99.

Key words: malware detection; feature engineering; feature generation; ExploreKit

0 引言

机器学习在 Android 恶意软件检测中得到广泛应用,特征工程是基于机器学习的 Android 恶意软件检测的关键环节。目前使用的特征主要包括静态特征和动态特征。但特征工程的过程严重依赖于专家经验,反复试验调优才能确定候选特征集合。

针对传统特征工程需要大量专家经验和人力的不足,本文提出了基于特征生成方法的 Android 恶意软件检测方法。该方法提取了多类特征,基于 UC Berkeley 的 ExploreKit[1]方法进行自动化特征生成计算,筛选得到能够提升模型性能的新特征,得到了良好的检测性能。

1 相关工作

特征工程是使用专业知识、背景知识和技巧处理数据,使得特征在机器学习算法上发挥更好的作用的过程。特征工程主要包括特征提取、特征选择和特征构建三个方面。

1.1 特征提取

Android 恶意软件检测中的特征提取方法可以分为静态分析方法和动态分析方法。

静态分析方法是指在不执行 Android 应用的前提下获得 APK(Android 应用程序包,Android Application Package)文件的静态特征,可以实现对大规模 Android 样本的特征的快速提取。例如,可以从 APK 中的 AndroidManifest.xml 文件中提取权限、组件等特征,从 Dex 文件中提取二进制特征。常见的静态分析工具有 Androguard、Apktool。但是静态分析存在易受混淆技术影响的问题。

动态分析方法是指通过设置可控环境,安装并运行 APK,尽可能触发样本的行为,提取其执行的动作信息。文献[2]提出了 IntelliDroid,该工具能够为恶意软件动态分析工具提供输入数据,触发恶意行为。动态分析的方法需要在运行环境下监控 Android 软件的运行,并通过不断输入触发其恶意行为,甚至可能需要人工进行辅助操作,存在资源消耗大、实现困难等问题。

1.2 特征选择

特征选择是指从特征集合中选出最具统计意义的特征子集,以达到减少特征个数、提高模型精度、减小运行时间的目的。已有的恶意软件检测方法主要基于研究者经验或特征贡献度选出所用特征。特征贡献度是指某个特征在分类任务中的贡献程度,常用的评价方法有卡方检验法、信息增益法、粒子群优化算法等[3]。

已有研究中,文献[4]通过研究良性样本和恶意样本的区别,选择意图和权限作为特征。文献[5]根据信息增益从开发者信息、组件、权限和动态分析结果等特征中进行选择,通过非监督学习算法进一步提升性能。文献[6]将 APK 中的特征划分为基于语法和基于资源的特征,使用 Extra Tree 对特征进行排序,筛选出不受混淆技术影响的特征。文献[7]使用 opcode 序列作为特征,利用 n-gram 从简化的 opcode 序列中提取特征,根据信息增益对特征进行选择,使用 AP 聚类算法压缩样本的数量。文献[8]通过对比某个特征在良性样本和恶意样本中出现的数量,选出基于方法调用和组件间通信的动态特征。

1.3 特征生成

特征生成是指从原始数据中通过构造新的特征,挖掘原始特征中的有效信息,用于训练机器学习模型。已有的恶意软件检测方法主要是基于研究者经验设计新特征。

文献[9]提取了 545 000 个特征,根据种类将其划分为 8 个特征向量集合,统计每个特征向量集合中相关特征的数量以作为特征。文献[10]通过构建 API 调用的加权有向图作为特征,使用信息增益筛选特征。文献[11]使用由应用组件和 API 构成的行为模式图作为特征。文献[12]通过将 Dex 文件转化为 RGB 图像,并将图像作为特征。文献[13]使用抽象的 API 调用序列构建马尔科夫链行为模型作为特征。

2 自动特征生成框架 ExploreKit

特征生成方法对已有的特征进行组合、计算以进一步提升机器学习模型的分类性能。常用的生成新特征的方法有离散化、归一化、四则运算、聚合运算等。但是如何运用以上方法对特征进行处理以获得性能更好的新特征依赖于研究者的专家经验和大量的实验验证。如何快速有效地对已有特征进行处理,得到性能表现更好的新特征,是一个亟需解决的问题。

本文基于 UC Berkeley 的 ExploreKit[1]自动特征生成框架进行研究。如图 1 所示,该框架集成了离散化、归一化、四则运算等生成新特征的方法,将各类数据集输入到该框架中,通过以上方法生成大量的候选新特征。该框架使用一种基于机器学习的新特征性能评估方法以预测新特征的性能表现,从而有效筛选出性能表现更好的特征。在根据候选特征排序模型评分对候选特征进行评分后,该框架使用贪心算法按序评估候选特征在目标分类模型上的性能表现,并根据用户需要从中选择提升目标分类模型性能的新特征。ExploreKit 将本次特征生成的初始特征和选出的表现最优的新特征进行组合,作为下一次迭代的输入,从而不断挖掘有效的新特

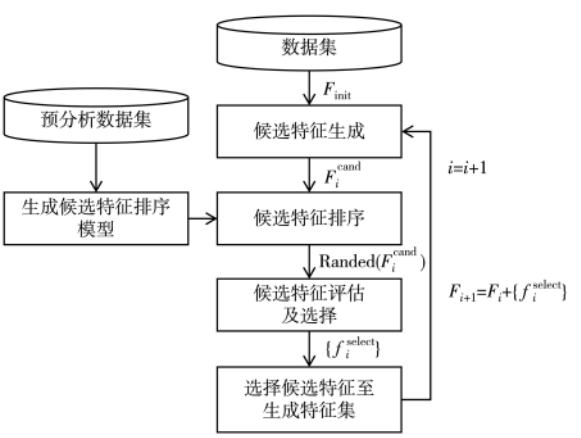

图 1 ExploreKit 特征生成框架

征。ExploreKit 在 25 组大小、分布等差异较大的数据集上，平均减少了 20%的错误率。

本文将 ExploreKit 特征生成框架应用于基于机器学习的 Android 恶意软件检测中，但是已有的特征生成方法存在生成候选特征过多的问题。假设输入框架的特征数量为 n，生成新特征的数量级为 $O(n^3)$。该框架需要对新生成的大量特征进行评估排序，使得特征生成时间过长。

3 基于特征生成的 Android 恶意软件检测方法

本文基于 ExploreKit 特征生成框架，提出了基于特征生成的 Android 恶意软件检测方法，如图 2 所示。

特征生成所用时间会随着输入特征的数量的增加而快速增加，而 ExploreKit 特征生成框架中没有对输入的特征进行筛选，即使输入的特征不具有任何信息，也会对其进行计算生成大量新特征，从而导致时间的浪费。采用信息增益对输入的特征进行筛选，过滤掉信息增益过低的特征。通过实验发现，聚合过多个特征生成的新特征几乎不会被选作新特征，通过对聚合特征的数量进行限制，从而将新生成特征的数量降低到了 $O(n^2)$。表 1 给出了本实验中用到的特征生成方式。

表 1 特征生成方式

特征生成方式	描述	特征生成数量
一阶运算	标准化、离散化、激活函数（新加入）	$O(n)$
二阶运算	四则运算	$O(n^2)$
聚合运算	聚合运算（GroupbythenOperate）	$O(n^3)$

在该框架中，选取了如下特征用于 Android 恶意软件检测：

(1)Manifest 中涉及的特征，包括声明的权限、组件信息、显式意图与意图过滤器等。Android 恶意软件制作者经常利用重打包技术，将恶意负载捆绑到市场上的正常 APK 中，因此本文选取了与重打包有关的特征，如证书与签名的时间相距较近，说明该 APK 可能为重打包 APK。

(2)Dex 中涉及的特征，包括敏感 API、使用的权限、隐式意图与广播意图过滤器等。动态代码加载、反射、加密函数等常被恶意软件用于隐藏其恶意行为，从 Dex 中提取了与之相关的特征。对 Dex 中包名进行了统计，获得其信息熵，以判断是否使用混淆技术对包名进行处理。

(3)其他文件中涉及的特征，包括 APK 中文件的数量、ELF 文件信息、文件格式与后缀名不匹配出现的次数等，并对 APK 中文件进行扫描，以发现 APK 文件中的隐匿 APK，并对其特征进行提取。

在特征提取后，根据信息增益对得到的特征进行排序和筛选，从而减小输入框架内特征数量。使用过滤法对特征进行选择，设置阈值，从排序后的特征中选取高于阈值的特征输入到特征生成框架。将得到的特征输入到 ExploreKit 特征生成框架[1]，得到能够提升目标分类模型的新特征。新特征与原始特征进行组合，用于训练目标分类器并得到恶意软件检测模型。

4 实验及结果分析

4.1 评价指标

Android 恶意软件检测中常用的检测指标有准确率（Accuracy）、精确率（Precision）、召回率（Recall）、AUC 等。定义如下：

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

$$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \tag{2}$$

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

其中，TP 为将正样本判定为正类的概率，即真阳率；FP 表示将负样本判定为正类的概率，即假阳率；TN 为将负样本判定为负类的概率，即真阴率；FN 表示

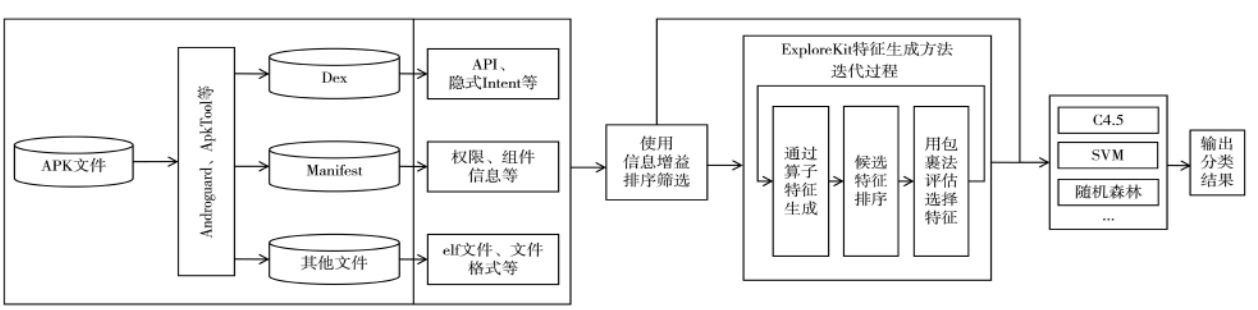

图 2 基于特征生成方法的 Android 恶意软件检测方法框架

将正样本判定为负类的概率，即假阴率。接收者运行特征(Receiver Operating Characteristic，ROC)曲线显示了给定模型的TP和FP之间的比较评定。曲线下面积(Area Under Curve，AUC)是指ROC曲线下方的面积，用于度量模型的准确率。AUC的取值范围为0.5~1，AUC越接近1，则模型的准确率越高。

本文使用AUC和错误减少率(Error Rate Reduction，ERR)作为模型性能提升的评价指标。使用Accuracy、Precision、Recall、F值和AUC作为模型性能的评价指标。错误减少率定义如下：

$$ERR=\frac{(1-AUC_i)-(1-AUC_f)}{(1-AUC_i)} \quad (4)$$

其中，AUC_i和AUC_f分别代表经过ExploreKit特征生成方法处理前后分类模型对应的AUC。

4.2 实验设置

实验采取10-fold交叉验证法。C4.5、SVM、随机森林分类模型使用Weka 3.7中的默认配置。特征生成过程迭代搜索。实验环境如表2所示。

表2 实验环境

名称	配置	名称	配置
CPU	i5-8400 2.80 GHz	Python	Python 3.6
RAM	16 GB	Weka	Weka 3.7
OS	Windows 10	Java	Java 1.8

4.3 实验所用数据

实验所用数据中，良性样本由CICAAGM[14]数据集和从安智市场获得的APK组成，共计10 000个；恶意样本由Drebin[9]数据集、CICAAGM数据集和从VirusShare获得的恶意样本组成，共计10 000个，如表3所示。

表3 实验所用数据集

数据集来源	良性样本数量	恶意样本数量
CICAAGM	1 147	306
Drebin	–	5 477
VirusShare	–	4 217
安智市场	8 853	–

4.4 实验结果及分析

4.4.1 特征选择

对实验所用数据集进行特征提取，提取了35 138组特征，大多数特征的信息增益都很小，图3给出了所有信息增益大于0.001的分布，在所提取的特征中，仅有199组特征的信息增益大于0.001，有54组特征的信息增益大于0.005。从曲线上发现，信息增益小于0.005的曲线趋于平缓，故设置阈值为0.005。采用过滤法，从中提取出54组特征，得到的部分特征及其信息增益如表4所示。

4.4.2 特征生成

图4给出了由特征生成方法得到的新特征示例。在该示例中，使用了StandardScoreUnaryOperator()和Add()两类算子。将初始特征A_1：File与A_2：URLConection进行标准化后得到B_1和B_2，B_1和B_2相加得到特征C_1，再对C_1进行标准化得到输出结果Output。File指与文件操作有关的API的数量，

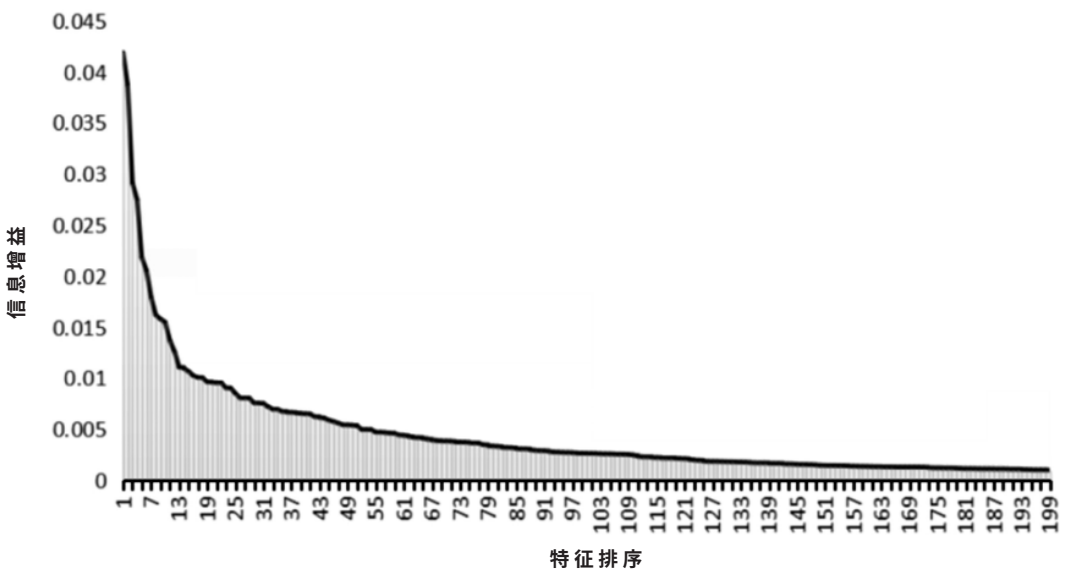

图3 信息增益分布

表 4 部分特征及其信息增益

特征名称	信息增益
file.Android	0.536
num_files	0.528
file.PNG	0.508
package.package_entropy	0.354
sensitive_API.File:exists	0.335
native_code	0.333
num_intent_consts	0.331
num_intent_const_other	0.329
PackageMissmatchIntentConsts	0.328
file.Targa	0.305
e_sh_flags.WA	0.293

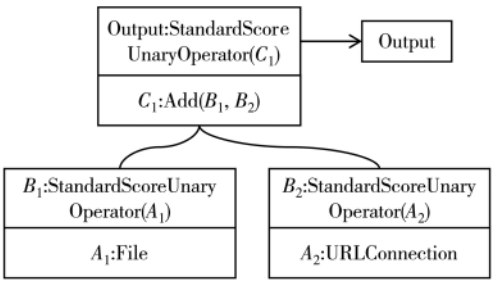

图 4 新生成特征示例

URLConection 指与网络连接有关的 API 的数量，二者特征组合得到新特征，该特征可能描述了通过接收外部信息对本地文件进行操作的恶意行为。

本文选取 C4.5、SVM 和随机森林作为目标分类模型。将提取的特征输入到改进后 ExploreKit 特征生成框架中得到实验提升结果，如表 5 所示。

根据 ExploreKit 特征生成结果，获得了新生成的特征，图 5 给出了目标分类模型为随机森林的新特征生成过程。涉及的符号如表 6 所示。

该特征生成过程在三次迭代后结束，将生成的特征与原始特征组合，对目标分类模型进行训练得到的检测模型性能如表 7 所示。

表 5 实验结果

分类模型	AUC_i	AUC_f	ERR/%
C4.5	0.932	0.950	26.47
SVM	0.948	0.967	32.69
随机森林	0.980	0.991	55.00

表 6 符号及其含义

符号	描述
A_1	num_intent_objects
A_2	string_emulator
A_3	permission.ACCESS
A_4	permission.READ_LOGS
StandardScoreUnaryOperator()	标准化
EqualRangeDiscretizer()	等频离散化
Add()	相加
GroupByThenAvg(source(), target())	根据源列对目标列聚合后求平均值

表 7 检测器性能

分类模型	Accuracy/%	Precision/%	Recall/%	AUC
C4.5	94.8	94.5	95.2	0.932
SVM	93.2	94.6	94.9	0.966
随机森林	96.5	97.8	95.1	0.992

针对 C4.5、SVM 和随机森林三种分类模型，在实验所用数据集上，特征生成所用时间分别为 6 599 s、9 126 s 和 21 116 s。图 6 给出了针对不同规模的 APK 提取特征所用时间。

5 结论

本文研究了基于特征生成的 Android 恶意软件检测方法，该方法使用 ExploreKit 生成并筛选出能够提高分类模型性能的新特征，缓解了传统特征工

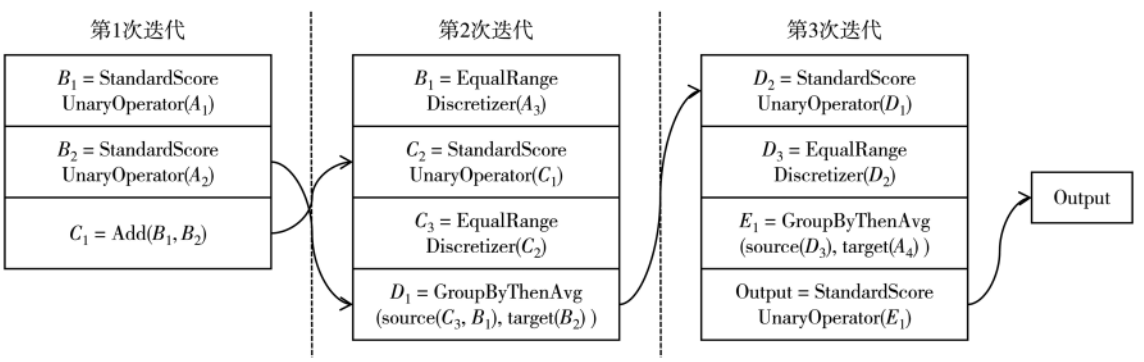

图 5 新特征生成过程

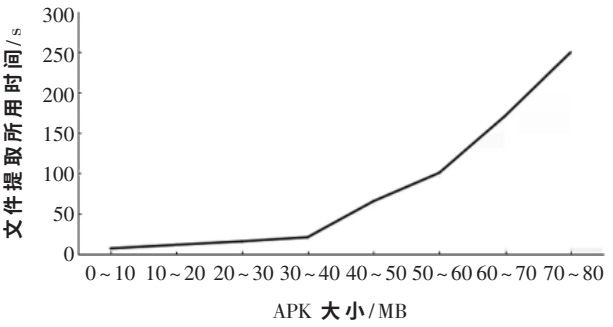

图 6 不同规模 APK 特征提取时间

程需要专家经验和大量人工筛选的困难。

ExploreKit 在应对大规模的 Android 数据集时的时间耗费较大，因此，需要继续改进 ExploreKit 以降低其时间耗费。在未来工作中，考虑到 Android 样本特征类型的特点，将 Android 特征按照特定顺序组合来引入新的算子，并根据 Android 样本选出更有效的算子组合方案，来提高候选特征的平均质量，进一步降低特征评估所用时间。

参考文献

[1] KATZ G, SHIN E C R, SONG D, et al. ExploreKit: automatic feature generation and selection[C]. IEEE 16th International Conference on Data Mining, 2016: 979-984.

[2] WONG M Y, LIE D. IntelliDroid: a targeted input generator for the dynamic analysis of Android malware[C]. Proceedings of the Annual Symposium on Network and Distributed System Security Symposium, 2016.

[3] 刘启川，覃仁超，刘玲，等. 基于特征贡献度的安卓恶意应用检测[J]. 计算机工程与设计，2020，41(4)：928-932.

[4] FEIZOLLAH A, ANUAR N B, SALLEH R, et al. AndroDialysis: analysis of Android intent effectiveness in malware detection[J]. Computers & Security, 2017, 65: 121-134.

[5] CHAKRABORTY T, PIERAZZI F, SUBRAHMANIAN V S, et al. EC2: ensemble clustering and classification for predicting Android malware families[J]. IEEE Transactions on Dependable and Secure Computing, 2020, 17(2): 1-14.

[6] SUAREZTANGIL G, DASH S K, AHMADI M, et al. DroidSieve: fast and accurate classification of obfuscated Android malware[C]. Proceedings of the 7th ACM Conference on Data and Application Security and Privacy, 2017: 309-320.

[7] CHEN T M, MAO Q Y, YANG Y M, et al. TinyDroid: a lightweight and efficient model for Android malware detection and classification[J]. Mobile Information Systems, 2018(2018): 1-9.

[8] CAI H P, MENG N, RYDER B, et al. DroidCat: effective Android malware detection and categorization via app-level profiling[J]. IEEE Transactions on Information Forensics and Security, 2019, 14(6): 1455-1470.

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

[10] IKRAM M, BEAUME P, KAAFAR M A, et al. DaDiDroid: an obfuscation resilient tool for detecting Android malware via weighted directed call graph modelling[J]. arXiv: Cryptography and Security, 2019.

[11] YANG C, XU Z Y, GU G F, et al. DroidMiner: automated mining and characterization of fine-grained malicious behaviors in Android applications[C]. European Symposium on Research in Computer Security, 2014: 163-182.

[12] HUANG T H, KAO H. R2-D2: ColoR-inspired convolutional neural network(CNN)-based Android malware detections[J]. arXiv: Cryptography and Security, 2017.

[13] MARICONTI E, ONWUZURIKE L, ANDRIOTIS P, et al. MaMaDroid: detecting Android malware by building markov chains of behavioral models[C]. Network and Distributed System Security Symposium, 2017.

[14] LASHKARI A H, KADIR A F, GONZALEZ H, et al. Towards a network-based framework for Android malware detection and characterization[C]. Conference on Privacy Security and Trust, 2017: 233-234.

（收稿日期：2020-08-19）

作者简介：

冯垚(1995-)，男，硕士研究生，主要研究方向：网络安全与恶意软件检测。

王金双(1978-)，男，博士，副教授，主要研究方向：系统安全、机器定理证明。

张雪涛(1995-)，男，硕士研究生，主要研究方向：网络安全与恶意软件检测。