

基于静态多特征融合的恶意软件分类方法

孙博文¹, 黄炎裔¹, 温俏琨², 田斌³, 吴鹏⁴, 李祺¹

(1. 北京邮电大学网络空间安全学院天地互联与融合北京市重点实验室, 北京 100876;

2. 北京邮电大学国际学院, 北京 100876;

3. 中国信息安全测评中心, 北京 100085;

4. 四川大学计算机学院, 四川 成都 610015)

摘要: 近年来, 恶意软件呈现出爆发式增长势头, 新型恶意样本携带变异性和多态性, 通过多态、加壳、混淆等方式规避传统恶意代码检测方法。基于大规模恶意样本, 设计了一种安全、高效的恶意软件分类的方法, 通过提取可执行文件字节视图、汇编视图、PE 视图 3 个方面的静态特征, 并利用特征融合和分类器集成学习 2 种方式, 提高模型的泛化能力, 实现了特征与分类器之间的互补, 实验证明, 在样本上取得了稳定的 F1-score (93.56%)。

关键词: 恶意软件; 家族分类; 静态分析; 机器学习; 模型融合

中图分类号: TP309.5

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2017.00217

Malware classification method based on static multiple-feature fusion

SUN Bo-wen¹, HUANG Yan-yi¹, WEN Qiao-kun², TIAN Bin³, WU Peng⁴, LI Qi¹

(1. Beijing Key Laboratory of Interconnection and Integration, School of Cyberspace Security,

Beijing University of Post and Telecommunications, Beijing 100876, China;

2. International School, Beijing University of Post and Telecommunications, Beijing 100876, China;

3. China Information Technology Security Evaluation Center, Beijing 100085, China;

4. College of Computer Science Sichuan University, Chengdu 610015, China)

Abstract: In recent years, the amount of the malwares has tended to rise explosively. New malicious samples emerge as variability and polymorphism. By means of polymorphism, shelling and confusion, traditional ways of detecting can be avoided. On the basis of massive malicious samples, a safe and efficient method was designed to classify the malwares. Extracting three static features including file byte features, assembly features and PE features, as well as improving generalization of the model through feature fusion and ensemble learning, which realized the complementarity between the features and the classifier. The experiments show that the sample achieve a stable F1-score (93.56%).

Key words: malware, family classification, static analysis, machine learning, model fusion

1 引言

在互联网飞速发展的今天, 网络的安全形势

却不容乐观, 各类安全威胁飞速增长, 其中传统 PC 平台上的恶意样本传播最为频繁。卡巴斯基公司于 2016 年底发布的年度报告显示, 卡巴斯基云

收稿日期: 2017-09-20; 修回日期: 2017-10-29. 通信作者: 孙博文, 273908200@qq.com

基金项目: 国家自然科学基金资助项目 (No.U1536119, No.61401038)

Foundation Item: The National Natural Science Foundation of China (No.U1536119, No.61401038)

端恶意样本数据库总量已达 10 亿, 包括病毒、木马、蠕虫等各类恶意对象, 其增长速度也从 2012 年的 7.53% 增长到 2016 年的 40.5%, 每天发现的恶意文件数量从 2011 年的 70 000 个增长到 2016 年的 323 000 个^[1]。2016 年, 赛门铁克公司共监测到超过 35 亿个恶意软件的新型变种, 及 100 多个肆意传播的新型恶意软件家族, 数量是过去的 3 倍^[2]。在全球范围内, 恶意软件攻击事件的数量较 2016 年增长了 36%。在恶意样本呈飞速增长趋势的同时, 恶意样本所使用的技术也在迅速提升。随着复杂的加壳、混淆、反沙箱、虚拟穿透等技术的出现, 恶意样本呈现出多样化、多态化的变化趋势, 研究人员发现大量样本是已有样本的变种, 这些变异样本通过各种静态变化来规避传统的基于病毒特征码与文件散列值的查杀方式。虽然部分反病毒检测引擎采用启发式的检测方法, 使用 API 执行序列、系统全局钩子等方式监测软件行为, 但这样的方式速度慢、效率低、成本高昂, 且存在安全隐患, 不适合大规模的恶意样本检测。在这样的背景下, 研究人员提出在原有静态检测的基础上, 通过对恶意样本家族进行深入研究, 试图以恶意样本家族分类的方式确定恶意样本之间的相似度^[3-7], 进而判定恶意样本是否是已知家族变异样本, 这是一种可行的对抗恶意样本多态与多样性的研究方法。传统的恶意样本分析方法受限于计算速度与成本消耗等原因难以适应大规模的家族分析, 但随着大数据安全与机器学习技术的发展, 海量数据的融合分析将是一种最为有效的检测与防范手段, 因此如何有效、准确地进行恶意样本家族分类是一个值得研究的课题。

2 相关工作

对恶意样本早期的一些工作主要关注如何鉴别正常软件与恶意软件, 2001 年, Schultz 等^[3]使用机器学习的方法如朴素贝叶斯检测恶意样本, 主要使用了字符串序列、动态链接库序列、系统调用 3 种典型特征, 在他们所使用的数据集上达到了 98% 的准确率。2010 年 Mehdi 等^[4]使用原始的 n -gram 方法得到了 87.85% 的准确率。随着恶意样本种类的增多与技术的发展, 恶意样本

分类问题成为一个棘手的问题, 更多的研究开始关注恶意样本分类问题, 根据分类所利用特征的不同主要分为静态分析和动态分析两方面的研究。在静态分析方面, 2010 年 park 等^[5]提出通过检测恶意样本行为图像中最大公共子图的方式进行分类, 并做出了探索性的研究。2013 年 Santos 等^[6]深入研究恶意代码 opcode 序列, 提取恶意样本 1-gram opcode 序列和 2-gram opcode 序列特征, 并使用支持向量机 (SVM) 的方法达到了 95.90% 的准确率。2016 年 Grini 等^[7]使用大数据统计分析的方式对 500 多个家族 100 多万多个样本进行了分类实验, 使用了多种机器学习算法最终达到 89% 的准确率。上述静态分析方法的主要优势在于, 恶意代码无需动态执行, 对分析系统不会造成破坏, 较为安全, 且其可以在执行文件前对代码流程有一定掌握, 另外此种方法不会受到具体进程执行流程的制约, 可以对代码进行详尽的细粒度分析。而其问题在于难以应对逐渐复杂的恶意样本加壳与混淆技术, 在对混淆代码的处理上无法得到预期的效果, 因而随着虚拟化技术的不断发展, 逐渐有研究者开始对恶意样本进行动态分析。2008 年 Rieck^[8]等使用基于行为分析的特征, 用支持向量机训练测试特征, 在 14 个家族中达到了 88% 的分类准确率, 但这种方式还没有让样本充分执行。2014 年 Zahra 等^[9]利用沙箱提取恶意样本 API 调用序列作为特征对恶意样本进行分类, 在小样本上取得了不错的效果。2017 年 Kolosnjaji^[10]等使用开源沙箱提取系统调用序列、使用卷积神经网络对恶意样本进行深度学习从而完成恶意样本分类, 达到了 89.4% 的准确率, 这种方式具有一定的创新性, 但在特征提取和模型训练速度上效率较低, 需要有更多进一步研究。

动态分析使用虚拟释放等机制应对软件脱壳与混淆技术, 让恶意样本充分释放, 并采用执行核心代码的方式观测其恶意行为, 这在一定程度上解决了软件加壳与混淆的问题, 但随着攻防技术的对抗, 这种方式也逐渐暴露出弊端。首先恶意软件开发对于虚拟环境、沙箱机制研究更加透彻, 他们采取多种检测与对抗技术, 使虚拟化及沙箱环境难以执行恶意样本; 其次恶意样本采用套用正常软件数字签名, 复用正常软件代码等

方式伪装宿主行为, 并使用穿透虚拟机技术攻击恶意代码分析人员; 最后动态分析方法提取特征复杂、检测时间较长、占用系统资源较多和误报率较高。对资源消耗巨大, 在应用到大规模数据时, 这种方式的弊端尤为凸显。而随着人工智能以及机器学习算法的发展, 静态分析的研究环境逐渐改善, 大量基于加壳样本分析的静态分析方法被提出, 2015年, 在微软恶意样本分类大赛中, 获奖团队提出使用文件图像化特征、深层汇编特征等静态特征对恶意样本进行分类, 并在应用中被证明有效, 韩晓光等^[11]也在恶意代码图像纹理聚类上有突破性研究, 这些研究有效地减少了之前静态分析方式难以解决样本加壳、混淆的弊端。同时随着恶意代码数量上的爆发式增长, 静态分析速度快、效率高、适合大规模实时分析的特点获得了优势。综上所述, 本文希望利用静态分析的优势, 以多源视角提取大量恶意样本静态特征, 并通过特征融合以及算法集成的方式, 提出一种应用上可行的适合大规模机器学习的恶意软件分类方法。

3 特征和模型

3.1 特征工程

为了有效进行恶意样本家族分类, 需要提取恶意样本各方面的特征, 并将其序列化表示输入分类器中。本文重点关注多维恶意样本静态特征, 主要从字节视图(即16进制字节码)、汇编视图(即asm汇编代码)、PE视图(即PE结构化信息)3个方面进行特征的提取。本文希望从这几个方面, 通过多源化的特征提取对抗恶意样本加壳、混淆等技术, 并最终进行有效的恶意软件家族分类, 下面分别介绍所选取的特征。

3.1.1 字节视图特征

字节视图能观测到的是一系列的16进制字符排列, 可以直接通过16进制读取的方式进行特征采集, 本文选取的特征包括以下几点。

1) 文件大小: 包括文件行数、文件字符总数等文件基本信息。

2) 可见字符串: 统计各类可见字符串出现的长度、频率、分布等信息; 值得关注的是, 一些有意义的可见字符串值得更深入的研究^[12]。

3) 程序熵值: 程序的熵值作为一种描述文件混乱程度的指标, 可以很好地反映恶意样本是否经过加壳、混淆等变化, 因此熵值是一个有意义的特征。

3.1.2 汇编视图特征

二进制代码本质上由一系列操作码组成, 又具体分为操作符、操作数2部分, 不同操作系统对于机器指令有不同映射, 本文利用第三方工具IDA Pro^[13]以及其提供的静态批处理功能批量得到恶意样本的反汇编视图数据, 主要提取本文需要的汇编视图特征。

1) 操作码特征: 操作码是机器码的一种形象化表示, 是汇编视图中出现的最为频繁的元素, 在众多的相关研究中都证明了基于操作码的分类具有良好的效果。本文采用了opcode的1-gram、2-gram特征, 选取在本文所用样本中出现的全部汇编指令进行统计作为操作码特征。

2) 寄存器特征: 寄存器是汇编语言中重要的一部分, 不同编译环境、编译函数使用的寄存器有较大差异, 因而本文提取了各类寄存器出现的次数。

3) 函数特征: IDA具有强大的交叉编译以及系统函数识别功能, 因而可以利用IDA分析的结果进行软件自定义函数和系统函数的提取, 这对于分析恶意样本的内部调用流程及混淆等行为具有极大帮助。

4) 数据定义特征: 本文从助记符中单独提出这一特征进行阐述, 重点关注样本中出现的db、dd、dw的相关统计信息, 针对这3个与数据相关的指令进行了分类统计。在分析加壳样本过程中, 样本主要存在的就是db、dd、dw这样的数据, 通过分析这一部分的特征, 并结合样本大小、样本压缩率、样本熵值等特征, 可以发现加壳样本的相互关联, 本文希望通过对这一部分特征进行分析, 来对加壳样本有一种新的静态分析方式。

3.1.3 PE视图特征

PE文件头包含大量与样本相关的信息, 以往的研究中这一部分由于处理难度大, 常被研究者忽略。本文借助了工具PE Exeinfo^[14]和开源工具Mastiff^[15], 提取了大量PE结构化特征以及编译

环境特征作为 PE 视图的主要特征。

1) PE 结构化特征：PE 头包含大量软件相关的深层次数据，可以分析出一定的语义信息，本文在这一部分提取了包括 API 调用数量、DLL 调用数量、导入函数数量、导出函数数量、各方法起始虚拟地址、虚拟大小、语言及编码方式等重要的结构化特征。

2) 反检测引擎特征：Mastiff 检测结果包含 Yara-rules^[16]等一系列对恶意样本针对各类杀毒方式所做出的隐藏措施的检测，本文选择了恶意样本所采用的虚拟逃逸手段、反调试手段、反病毒手段、恶意片段匹配等几种典型恶意特征进行提取。

3) 编译特征：包括软件的编译时间与编译环境以及加壳信息。根据编译信息的细微差异，可以推断软件之间的演进关联。

4) 恶意 API：研究人员在分析了 50 余万恶意样本，多达 5 TB 数据后对恶意样本所使用的 API 进行了统计^[17]，本文取其中前 100 个 API 出现的次数作为特征。表 1 列出了本文使用的前 10 个恶意 API 特征名称。

表 1 恶意 API

恶意 API 名称
API_GetProcAddress
API_LoadLibraryA
API_GetModuleHandleA
API_ExitProcess
API_VirtualAlloc
API_WriteFile
API_GetModuleFileNameA
API_CloseHandle
API_RegCloseKey
API_VirtualFree

上述主要特征符号标签及含义如表 2 所示。

3.2 模型简介

模型融合是机器学习任务中最常用的方法，它通常可以在各种不同的应用场景中使结果获得提升。本文针对恶意软件提出了一种多特征融合的方法，通过对特征进行融合来构建不同基分类器、丰富特征以及算法的多样性，以提高模型的精度与稳定性，并通过交叉验证选择最优的融合方法。整体思路如图 1 所示。

表 2 部分选取特征

特征名称	特征描述
hex_filesize	样本大小
hex_filelines	样本字节视图行数
hex_str_len_*	样本字符串相应长度的次数
hex_lgram_*	样本字符出现次数
hex_entropy_*	样本熵值
asm_register_*	寄存器出现次数
asm_opcode_*	操作码出现次数
asm_dd_*	dd 出现次数
asm_dw_*	dw 出现次数
asm_symbol_*	特殊符号出现次数
pe_API	API 出现次数
pe_dll	dll 出现次数
pe_imports	引用数量
pe_exports	导出数量
pe_mastiff_*	mastiff 报告信息
pe_alertAPI	恶意 API 统计次数
pe_compile	编译环境类别

3.2.1 特征融合

为保证基分类器的准确性与差异性，需要对特征进行采样与融合。本文采用对这些特征进行加权的融合方式，即给不同特征赋予不同的权值，权值大的特征表示对当前分类任务的贡献相对较大。随机森林是一种使用广泛的强大机器学习算法，能够执行回归和分类的任务。同时，它也是一种高效的数据降维手段，用于处理缺失值、异常值以及其他数据探索中的重要步骤，并取得了不错的成效。对于分类问题，通常采用基尼不纯度或信息增益作为衡量分类的效果，当训练决策树的时候，可以计算出每个特征减少了多少树的不纯度。对于一个决策树森林来说，可以算出每个特征平均减少了多少不纯度，并把它平均减少的不纯度作为特征融合的评价标准。因此可以通过随机森林算法对特征进行融合，按重要程度给特征由大到小分配权重，并舍弃负面特征。

本文 3.1 节中共选取了 3 组特征，由于单组特征无法体现特征融合效果，选取全部 3 组特征融合则会造成基分类器性能降低，因而本文

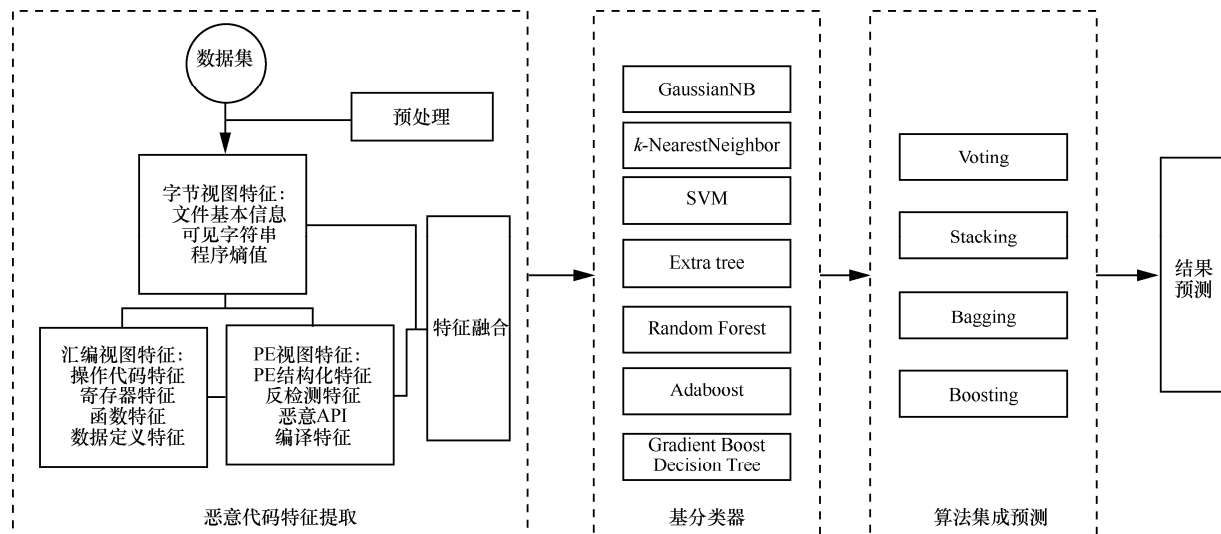


图1 模型思路

采取的抽样策略为任取 2 组特征进行融合，并在每组特征中再次随机抽取总体特征 2/3 的组内特征，这样组间特征相同的概率与相同特征百分比均大大降低。本文实验采用离线训练的方式，训练速度快，特征融合之后分类效果存在明显的提高。

3.2.2 算法集成

针对部分传统机器学习算法中会将所有特征串联而导致的过拟合以及降低泛化能力的问题，本文提出了一种应用于恶意软件的集成学习监督分类方法。该框架的思路是在进行分类的时候，把若干基本分类器进行集成，本文采用了基于 python 的 Scikit-learn 库^[18]所提供的封装分类器进行数据的分类，选择了机器学习研究领域 8 种典型算法：GNB、GBDT、RF、ET、DT、SVM、KNN 和 Adaboost。完成上文所述的特征融合后，采用上述算法训练多个基分类器，而后选择 4 种结果较优的算法进行算法集成，集成过程中，对 4 种不同的方法（Voting、Stacking、Bagging、Boosting）及每种方法中的不同策略进行测试，通过实验寻找最优的集成方案。在使用 Voting 方法时，测试 Average of probabilities 和 Majority voting 这 2 种策略；使用 Stacking 时，则测试 BayesNet、Random Forest；而在使用 Boosting 过程中，测试 Adaboost、GBDT、XGBoost 这 3 种策略。根据实验结果决定最终的分类，以取得比

单个分类器更好的性能。

4 实验

4.1 数据

如今，海量的恶意样本数据成为各大反病毒公司赖以生存的关键。而数据集的选择一直是恶意样本研究中最关键的一环，没有一个标准的数据集是截至目前恶意样本研究难以统一、各自为战的主要原因。为了保证实验具有一定意义上的通用性，本文使用了国外知名恶意代码数据库网站 VirusShare^[19]提供的恶意样本作为本次实验的实验数据，样本标识为 VirusShare_00271，样本总量达到了 65 536。

4.2 家族划分

在本文的研究过程中，需要对恶意样本家族进行划分，如何对恶意样本家族进行判定是恶意样本研究中另一个聚焦的问题，目前文献研究中尚未提出一个权威的解决办法。文献[7]提出使用微软所述的 CARO 命名规则^[20]，这是一种通用的病毒命名规则，然而由于反病毒公司难以共享数据，使各个杀毒引擎在对样本命名过程中均有不小的改变。表 3 为 Virustotal 平台^[21]上不同引擎对同一样本的标注结果。在阅读大量参考文献^[3~11]后，发现其中绝大部分使用了卡巴斯基扫描引擎的家族划分结果，基于成本、效率等方面考虑本文同样延续采用这种家族划分方式。

表 3 同一恶意样本扫描引擎结果

检测引擎	家族名称
Bkav	HW32.Packed.F519
McAfee	Ransomware-FUO!638C98B55FEE
Baidu	Win32.Trojan.WisdomEyes.16070401.9500.9925
Symantec	Trojan.Gen.2
Kaspersky	not-a-virus:HEUR:Downloader.Win32.LMN.gen
Microsoft	Ransom:Win32/Cerber!rfn
Tencent	Win32.Trojan.Cerber.Wnwm
Panda	Trj/GdSda.A
Qihoo-360	HEUR/QVM20.1.7BA9.Malware.Gen

4.3 实验设计

本文实验过程如图 2 所示,首先从 VirusShare 网站获取原始数据集,使用 PE Exeinfo、卡巴斯基引擎进行样本的初次筛选和恶意家族的划分。接下来,从 3 个视图提取恶意样本特征,输入分类器进行训练和测试,根据初次实验结果,分析模型与数据不足,改进模型,进行特征融合。基于融合后的特征使用 8 种常用分类器对数据再次进行训练和测试,根据结果择优进行算法集成。在结果评价方面使用 F1-score 作为评价标准,并使用 5-cross-validation 作为最终的评价标准展示不同机器学习算法的效果。

4.4 实验结果与分析

本文所选用的“VirusShare_00271”数据集总量为 65 536,由于本文研究是针对 Windows 平台下的 PE 文件,因而需要对原始数据进行筛选,本文借助 PE Exeinfo 这款软件和命令行 file 指令,筛选掉非 PE 文件后,剩余 13 909 样本。接下来通过卡巴斯基引擎的扫描,保留了卡巴斯基所能明

确检测出来的恶意家族 600 余个,样本 11 651 个。而后对家族名称进行筛选,保留了卡巴斯基能明确划分的家族名,去掉了诸如“not a virus adware”这类的不明确样本;最终得到了 182 个不同的家族种类,2 798 个样本,其中排名靠前的几个家族如表 4 所示。

表 4 家族名称及数量

卡巴斯基检测家族名称	数量
Backdoor.Win32.Hlux	27
HEUR:Trojan-Downloader.Win32.Generic	41
HEUR:Trojan.Win32.Generic	578
HEUR:Trojan.Win32.Invader	53
Packed.Win32.Tpyn	20
Trojan-Clicker.Win32.Agent	39
Trojan-Downloader.Win32.Agent	41
Trojan-Dropper.Win32.Injector	18
Trojan.Win32.Agent	196
Trojan.Win32.AntiFW	1 480
Trojan.Win32.Buzus	46
Trojan.Win32.Inject	28

对这部分样本进行数据的预处理及特征提取,并进行初步的训练和测试实验,分析初步的分类结果,发现其中大量的错误出现在与“Generic”相关的类别,错误对应关系见表 5。查阅资料后发现,卡巴斯基引擎将这个类别定义为启发式引擎命名,也就是说它并不是一个单独的类别,而是大量难以归类的样本。因而本文暂时去掉了“Trojan.Win32.Generic”“UDS: DangerousObject.Multi.Generic”这 2 种类别,并将

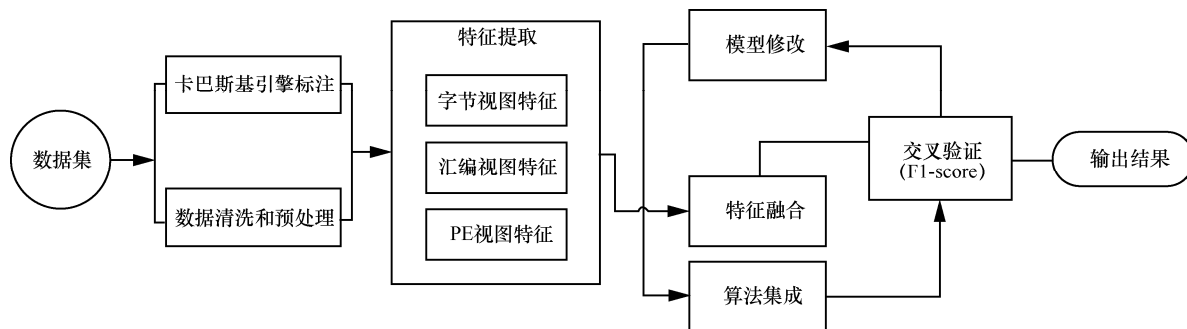


图 2 实验流程

“Trojan.Win32.Agent”“Trojan.Win32.Agentb”这类相近的家族归为一类。最终得到了 12 个类别共计 1 848 个样本, 将这部分数据提取出的 3 类特征使用多种分类器进行训练测试, 单独计算准确率。

表 5 错误预测情况

预测类别	实际类别	次数
Trojan.Win32.AntiFW.a	HEUR:Trojan.Win32.Generic	271
Trojan.Win32.AntiFW.a	Trojan.Win32.Agent	93
Trojan.Win33.AntiFW.a	HEUR:Trojan.Win32.Invader	25
Trojan.Win34.AntiFW.a	Trojan.Win32.Inject.vbet	19
Trojan.Win35.AntiFW.a	Trojan-Downloader.Win32.Agent.hfug	19
HEUR:Trojan.Win32.Generic	Trojan.Win32.AntiFW.a	8
HEUR:Trojan.Win32.Generic	Trojan.Win32.Agent	4

接下来, 采用 3.1 中所述的特征融合方法, 对 3 个视图的特征进行抽样融合, 去掉了对分类结果具有负面影响的特征, 同时得到了如表 6 所示的对于分类影响最大的几个特征, 可以看到本文重点强调的数据定义指令特征以及恶意 API、寄存器 opcode 等特征确实为重要的一批特征, 结合编译环境与查壳信息, 并考虑样本大小熵值等信息, 也可进行单独的样本类型推断。最后, 按照各特征重要程度为其分配权重, 利用融合后的特征再次进行实验。

表 6 各特征分数

特征类别	特征分数
asm_dd_para5	0.045 3
asm_opcode_wait	0.039 2
pe_alertapi_StartupInfoW	0.025 4
pe_alertapi_GetModuleHandleW	0.022 4
ColInitialize	0.010 7
asm_dd_para4	0.009 5
pe_alertapi_LoadLibraryW	0.008
asm_opcodes_fdiv	0.007 5
pe_compile	0.004 9
pe_alertapi_GetProcAddress	0.004 5
asm_opcodes_cwd	0.004 3
asm_regs_ecx	0.002 6

在建立完模型之后需要对模型的好坏进行评价, 为了实验的可靠性(排除一次测试的偶然性), 要进行多次测试验证, 也就是交叉验证。为了避免对验证数据的过拟合, 本文选择 5-fold 交叉校验的方式验证本文所提出方法的准确率和稳定性, 最终得到各分类器效果如表 7 和表 8 所示。表 7 中的数据显示特征融合前每类特征的单独分类效果, 而表 8 则是在完成特征融合后重新计算得出, 准确率相较前者有了显著的提升。其中, 数据使用 ExtraTrees 分类器得到了最高的 92.69% 的准确率。

表 7 特征融合前准确率

分类器/特征集合	字节视图特征	汇编视图特征	PE 视图特征	全部特征
GNB	61.467 2%	86.301 8%	52.026 4%	73.730 0%
RF	85.904 5%	89.977 6%	85.421 6%	87.520 0%
GBDT	85.124 5%	89.456 4%	83.751 4%	88.339 0%
ET	81.900 2%	90.152 7%	79.174 8%	85.582 7%
DT	77.506 1%	44.114 8%	57.567 0%	81.882 9%
SVM	65.399 9%	52.872 7%	60.235 7%	64.114 8%
KNN	78.561 6%	82.634 1%	78.797 2%	81.791 6%
AB	57.312 8%	64.015 7%	48.605 8%	64.913 2%

表 8 特征融合后准确率

分类器/特征集合	字节视图特征	汇编视图特征	PE 视图特征	全部特征
GNB	62.677 0%	83.354 4%	48.371 8%	75.485 4%
RF	85.879 2%	89.184 0%	83.674 4%	92.694 2%
GBDT	85.534 7%	91.668 6%	83.897 3%	90.374 6%
ET	81.739 9%	91.234 9%	76.766 5%	86.940 2%
DT	79.534 0%	49.371 6%	57.130 9%	83.379 5%
SVM	69.568 6%	56.180 3%	66.807 0%	71.711 8%
KNN	74.747 9%	80.255 2%	74.232 4%	84.218 6%
AB	64.125 9%	68.835 9%	49.153 2%	71.271 1%

接下来, 选择最优的 4 种分类方法(RF、ET、DT、KNN)并使用 3.2.2 节中提到的融合方法进行算法融合, 各方法及策略的分类效果如表 9 所示, 从中可见, Voting 方法过于简单, 存在一定欠拟合的情况, 因而效果欠佳; Bagging 和 Boosting 均采用抽样的集成方式, 前者采用均匀取样, 而

表 9		算法集成结果			
融合方法	集成策略	字节视图特征融合后	汇编视图特征融合后	PE 视图特征融合后	全部特征融合后
Voting	Average of Probabilities	78.4720%	82.4035%	86.1256%	87.7300%
	Majority Voting	63.5502%	71.7669%	69.7124%	67.1198%
Stacking	J48	80.2417%	87.1749%	81.6689%	92.0740%
	BayesNet	87.6839%	88.3260%	81.3020%	93.5660%
Bagging	Random Forest	85.1330%	87.3974%	78.1261%	86.4870%
Boosting	Adaboost	84.6372%	79.1950%	86.4609%	90.7526%
	GBDT	76.4684%	89.8672%	79.2185%	86.3200%
	XGBoost	84.1474%	90.8446%	76.2524%	90.6736%

后者根据错误率取样，因此 Boosting 的分类精度要优于 Bagging。Stacking 的工作分为 2 个阶段：首先使用多个基础分类器预测分类，然后用一个新的学习模块与它们的预测结果结合起来，以降低泛化误差，从表 9 中可以看出，使用 Stacking 方法的集成效果明显优于另外 3 种，其中集成策略 BayesNet 得到了最高的 93.56% 的准确率。

5 设备与时间

为了进行效率对比分析以及更大规模数据的实验，本文对实验过程中的各项时间进行了整理。本文全部实验均使用一台个人计算机进行，设备型号为 macbookpro2013，处理器为 Intel Core I5-4287U 2.60 GHz，内存为 8 GB。操作系统为 Windows7 旗舰版，软件为 Python2.7-64 位。整个实验主要分为引擎扫描、汇编视图转换、字节特征提取、汇编视图特征提取、PE 视图特征提取、特征融合、算法集成、训练与分类几个步骤，表 10 展示了整个实验的消耗时间，并单独展示特征提取 3 个部分所消耗的时间。

分析时间消耗可知，整体实验时间消耗在卡巴斯基引擎扫描和样本预处理上的时间较多，当采用多台机器分布式处理时这一部分有较大的优化空间。在特征提取部分存在大量反复文件读写，后续拟采用数据库存储的方式加快特征组合。在算法运行时间上集成学习算法与传统线性方法存在一定差距，但对整体效率上的影响微乎其微。

表 10		实验消耗时间	
类别	数量/个	时间/s	
卡巴斯基引擎检测时间	65 536	7 023	
样本预处理时间	2 798	6 013	
PE Exeinfo 处理时间	11 651	1 141	
字节视图特征提取时间	2 798	2 822	
汇编视图特征提取时间	2 798	4 020	
PE 视图特征提取时间	2 798	2 035	
特征融合时间	2 798	447	
算法运行时间	2 798	<5	
算法融合时间	2 798	669	

6 结束语

本文提出了一种基于多特征融合的恶意样本家族分类方法，利用恶意样本网站 Virusshare 提供的最新恶意样本以及卡巴斯基引擎扫描得到的分类结果，从字节视图、汇编视图、PE 视图 3 个方面进行特征提取，设计了特征融合方法和算法集成模型，对包括 SVM、决策树、随机森林等 8 种典型算法进行测试与对比，经过多次优化与调整最终达到了 93.56% 的准确率。本文的主要贡献在于提出了完整的思路与实验设计，实现了大规模恶意样本的特征提取与分类检测的模型，并提出使用 3 种视图进行多维度特征提取与特征融合的思想以提高准确率。本文选取的特征包括在相关文献中被证实有效的特征，也包括针对加壳文件进行区分的新式特征，最终取得了很好的效果

(feature score)。本文重点介绍了特征提取与模型设计部分,对于机器学习算法和参数的设定上没有进行详细的介绍,今后将在这一方面继续深入。后续的工作需要提高有效样本的数量,由于本文实验环境所限,以及探索阶段中出现的各种探索性实验,使大量时间被用在了特征设计、特征提取、特征选择,以及出错后的修正等阶段,接下来需要将本文的样本容量继续扩大一个数量级,争取达到1 000万的初始数据量。同时解决训练样本分布不够均匀的问题,对家族数量进行扩充,增进分类训练的准确度与稳定性。

参考文献:

- [1] 卡巴斯基年报[EB/OL]. <https://securelist.com/kaspersky-security-bulletin-2016-story-of-the-year/76757/>. Kabasiji annual report[EB/OL]. <https://securelist.com/kaspersky-security-bulletin-2016-story-of-the-year/76757/>.
- [2] 赛门铁克年报[EB/OL]. <https://www.symantec.com/products/atp-content-malware-analysis>. Symantec annual report[EB/OL]. <https://www.symantec.com/products/atp-content-malware-analysis>.
- [3] SCHULTZ M G, ESKIN E, ZADOK E, et al. Data mining methods for detection of new malicious executables[C]//2001 IEEE Symposium on Security and Privacy(S&P 2001). 2001:38-49.
- [4] MEHDI B, AHMED F, KHAYYAM S A, et al. Towards a Theory of Generalizing System Call Representation for In-Execution Malware Detection[C]//IEEE International Conference on Communications. 2010:1-5.
- [5] PARK Y, REEVES D, MULUKUTLA V, et al. Fast malware classification by automated behavioral graph matching[C]//AMIA Annu Symp Proc. 2010:1-4.
- [6] SANTOS I, BREZO F, UGARTE-PEDRERO X, et al. Opcode sequences as representation of executables for data-mining-based unknown malware detection[J]. Information Sciences, 2013, 231(9): 64-82.
- [7] GRINI L S, SHALAGINOV A, FRANKE K. Study of soft computing methods for large-scale multinomial malware types and families de-tecton[C]//The World Conference on Soft Computing. 2016.
- [8] RIECK K, HOLZ T, WILLEMS C, et al. Learning and classification of malware behavior[C]//The International Conference on Detection of Intrusions & Malware. 2008:108-125.
- [9] SALEHI Z, SAMI A, GHIASI M. Using feature generation from API calls for malware detection[J]. Computer Fraud & Security, 2014, 2014(9):9-18.
- [10] KOLOSNAJBI B, ZARRAS A, WEBSTER G, et al. Deep learning for classification of malware system call sequences[M]//AI 2016: Advances in Artificial Intelligence. Berlin: Springer, 2016.
- [11] 韩晓光, 曲武, 姚宣霞,等. 基于纹理指纹的恶意代码变种检测方法研究[J]. 通信学报, 2014, 35(8):125-136.
- HAN X G, QU W, YAO X X, et al. Research on malicious code variants detection based on texture fingerprint[J]. Journal on Communications, 2014, 35(8):125-136.
- [12] Digital bread crumbs: seven clues to identifying who's behind advanced cyber attacks[EB/OL]. <https://www.fire-eye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-digital-bread-crumbs.pdf>.
- [13] IDA pro website [EB/OL].<https://www.hex-rays.com>, accessed: 27.06.2017.
- [14] PE Exeinfo website [EB/OL]. <http://exeinfo.pe.hu/>, accessed: 12.06.2017.
- [15] Mastiff website[EB/OL]. <https://github.com/KoreLogicSecurity/mastiff>, accessed: 27.06.2017.
- [16] Yara-rules website [EB/OL].<https://github.com/Yara-Rules/rules>.
- [17] Top maliciously used APIs [EB/OL]. https://www.bnxnet.com/wp-content/uploads/2015/01/malware_APIs.pdf.
- [18] Scikit-learn website [EB/OL].<https://github.com/scikit-learn/scikit-learn>.
- [19] VirusShare website [EB/OL]. <https://virusshare.com/>.
- [20] Naming scheme-caro-computer antivirus research organization [EB/OL]. www.caro.org/naming/scheme.html.
- [21] Free online virus, malware and url scanner[EB/OL]. <https://www.virustotal.com/>.

作者简介:



孙博文(1994-),男,辽宁沈阳人,北京邮电大学硕士生,主要研究方向为网络安全、恶意软件检测。



黄炎裔(1994-),女,江西黎川人,北京邮电大学硕士生,主要研究方向为数据安全、分布式调度。

温俏琨(1996-),女,辽宁沈阳人,北京邮电大学本科生,主要研究方向为网络安全、机器学习。

田斌(1983-),男,北京人,博士,中国信息安全测评中心副研究员,主要研究方向为威胁情报、攻击溯源。

吴鹏(1982-),男,四川广元人,四川大学博士生,主要研究方向为恶意软件检测、软件抄袭、软件安全加固。

李祺(1981-),女,北京人,北京邮电大学副教授,主要研究方向为网络安全、机器学习、物联网安全。