计算机科学
COMPUTER SCIENCE

基于 N-Gram 静态分析技术的恶意软件分类研究

张光华，高天娇，陈振国，于乃文

引用本文

张光华，高天娇，陈振国，于乃文. 基于 N-Gram 静态分析技术的恶意软件分类研究[J]. 计算机科学, 2022, 49(8): 336-343.

ZHANG Guang-hua, GAO Tian-jiao, CHEN Zhen-guo, YU Nai-wen. Study on Malware Classification Based on N-Gram Static Analysis Technology[J]. Computer Science, 2022, 49(8): 336-343.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210900203

基于 N-Gram 静态分析技术的恶意软件分类研究

张光华[1,2]　高天娇[1]　陈振国[3]　于乃文[1]
1 河北科技大学信息科学与工程学院　石家庄 050018
2 西安电子科技大学综合业务网理论及关键技术国家重点实验室　西安 710071
3 华北科技学院河北省物联网监控工程技术研究中心　河北 廊坊 065201
(zhanggh@hebust.edu.cn)

摘　要　为了解决恶意软件分类准确率不高的问题,提出了一种基于 N-Gram 静态分析技术的恶意软件分类方法。首先,通过 N-Gram 方法在恶意软件样本中提取长度为 2 的字节序列;其次,根据提取的特征利用 KNN、逻辑回归、随机森林、XGBoost 训练基于机器学习的恶意软件分类模型;然后,使用混淆矩阵和对数损失函数对恶意软件分类模型进行评价;最后,将恶意软件分类模型在 Kaggle 恶意软件数据集中进行训练和测试。实验结果表明,XGBoost 和随机森林的恶意软件分类模型准确率分别达到了 98.43%和 97.93%,Log Loss 值分别为 0.022240 和 0.026946。与已有方法相比,通过 N-Gram 进行特征提取的方法可以更准确地对恶意软件进行分类,保护计算机系统免受恶意软件的攻击。

关键词:N-Gram;静态分析;机器学习;恶意软件
中图法分类号　TP309

Study on Malware Classification Based on N-Gram Static Analysis Technology

ZHANG Guang-hua[1,2],GAO Tian-jiao[1],CHEN Zhen-guo[3] and YU Nai-wen[1]
1 School of Information Science and Engineering,Hebei University of Science and Technology,Shijiazhuang 050018,China
2 State Key Laboratory of Integrated Services Networks,Xidian University,Xi'an 710071,China
3 Hebei IoT Monitoring Engineering Technology Research Center,North China Institute of Science and Technology,Langfang,Hebei 065201,China

Abstract　In order to solve the problem of low accuracy of malware classification,this paper proposes a research on malware classification based on N-Gram static analysis technology. Firstly,the N-Gram method is used to extract the byte sequence of length 2 from the malware samples. Secondly,according to the extracted features,KNN,logistic regression,random forest and XGBoost are used to train the malware classification model based on machine learning. Thirdly,the confusion matrix and logarithmic loss function are used to evaluate the malware classification model. Finally,the malware classification model is trained and tested in the Kaggle malware data set. Experimental results show that the accuracy rates of the malware classification models of XGBoost and random forest reach 98.43% and 97.93%,and the Log Loss values are 0.022240 and 0.026946,respectively. Compared with the existing methods,the proposed method can classify malware more accurately and protect computer system from malware attack.

Keywords　N-Gram,Static analysis,Machine learning,Malware

1　引言

随着信息技术的快速发展,恶意软件通过非法访问权限来获取私有数据,恶意软件数据分析公司 Cambridge Analytica 收集 Facebook 用户的数据并滥用 5 000 万用户的数据[1]。Facebook 信息泄露丑闻发生以后,越来越多的企业和用户认识到网络安全和隐私保护的重要性。随着恶意软件数量的不断增长,恶意软件检测率不断下降[2]。2021 年恶意软件状态报告显示,Windows 恶意软件检测率下降了 12%,威胁企业的恶意软件检测率下降了 24%,威胁用户的恶意软件检测率下降了 11%[3-4]。为了避免恶意软件窃取用户的隐私信息以及错误删除计算机系统上的正常文件,恶意软件分类需要保证较高的分类准确率。

目前机器学习在恶意软件分类领域的应用非常广泛,为开发准确率更高的分类模型提供了更多的选择和空间,但恶意软件分类领域仍然面临许多挑战。机器学习需要对大量的

到稿日期:2021-09-24　返修日期:2022-03-11
基金项目:国家重点研发计划(2018YFB0804701);国家自然科学基金(62072239);河北省科技厅科技计划(20377725D)
This work was supported by the National Key Research and Development Program of China(2018YFB0804701),National Natural Science Foundation of China(62072239) and Science and Technology Program of Hebei Science and Technology Department(20377725D).
通信作者:于乃文(yunaiwen@hebust.edu.cn)

恶意软件进行分类来提高分类模型的准确率，但是在实际环境中，由于恶意软件本身具有恶意行为，网络安全相关的法律法规政策对恶意软件的限制[5]，因此很难采集到带有标注的恶意软件样本。大多数安全研究人员都关注恶意软件分类的准确率，恶意软件分类准确率不高会导致正常文件被错误删除，并且不同的安全研究人员使用的是不同的恶意软件数据集和不同的恶意软件标注方法，因此对于不同的研究结果中的分类的准确率无法进行有效的比较。

本文的主要贡献如下：首先，提出了通过 N-Gram 方法进行特征提取，使用了当 $N=2$ 时的 Bi-Gram 方法，在恶意软件样本.byte 文件中提取长度为 2 的字节序列用于比较恶意样本之间的相似性；其次，训练了基于机器学习的恶意软件分类模型，机器学习能够有效分析恶意软件的恶意特征和攻击行为，使用了 Kaggle 恶意软件数据集中的 10 868 个恶意软件样本进行训练和测试；最后，利用混淆矩阵和对数损失函数对恶意软件分类模型进行评价，通过准确率来衡量恶意软件分类模型的成功程度，结果表明恶意软件分类模型的准确率达到 98.43%，Log Loss 值为 0.022 240，有效地分析了恶意软件的特征和行为，进一步提高了恶意软件分类的准确率。

2 相关工作

在数据集方面，本文选取的是 Kaggle 微软恶意软件分类挑战赛数据集[6]，该数据集是恶意软件领域的权威数据源；在特征提取方面，本文选取 N-Gram 进行特征提取，虽然其他领域已经成功应用了 N-Gram，但是网络安全领域对其应用较少；在机器学习方面，本文利用 KNN、逻辑回归、随机森林、XGBoost 这 4 种机器学习算法训练恶意软件分类模型；在恶意软件分析技术方面，本文主要在静态分析技术下研究 PE 格式的恶意软件样本。

2.1 恶意软件数据集

文献[7]在 Kaggle 数据集和 Malimg 数据集上进行了恶意软件分类测试，在测试结果中 *F-Score* 分别为 0.982 0 和 0.966 1，虽然 RCNF(Random CapsNet Forest)在 Malimg 中的 F-Score 和准确率达到了最新技术的水准，但是可训练参数较少；Kaggle 则以更多的参数超过了 Malimg，显然 Kaggle 数据集的有效性更好。文献[8]同样在 Kaggle 和 Malimg 上进行了恶意软件分类测试，在测试结果中分别获得 1.000 和 0.997 的精度。在 Malimg 中进行 10 倍交叉验证，准确率和 *F1-Score* 分别为 0.997 2 和 0.999 1；在 Kaggle 中使用训练集训练特征提取器并通过 SVM 分类器进行 10 倍交叉验证，准确率和 *F1-Score* 都为 1，在测试集中对数损失为 0.031 42，进一步证明了 Kaggle 数据集的有效性。在 2015 年的 Kaggle 微软恶意软件分类挑战赛中，挑战赛主题为恶意软件分类并且要求使用静态分析技术，获得此次挑战赛第一名中的 3 位参赛人员的专业并不是网络安全，所使用的方法与常见的方法存在很大的差异，但是提取的 Operation Code、Headers 个数和恶意样本图像(.asm 文件像素强度)3 种特征比较有创新性，同时提取 Byte Code、N-Gram、指令频数等特征，采用随机森林并用 XGBoost 和 PyPy 加快训练速度，体现出机器学习在恶意软件分类领域中的巨大发展潜力。本文使用的是 Kaggle 微软恶意软件分类挑战赛数据集，文献[9-12]与本文使用了相同的数据集，该数据集是恶意软件领域的权威数据源，因此本文的实验结果可以令人信服。

2.2 特征提取方法

文献[13]通过 N-Gram 生成伪标签对图片进行说明。通常情况下考虑样本之间的相似性来生成伪标签，在恶意软件分类过程中，相似度越高的两个样本越有可能是同一恶意软件家族，但是两张相似的图片也可能表达了不同的寓意。与考虑样本间相似性生成伪标签的方法相比，N-Gram 可以更合理、准确地描述两张相似的图片，使描述文字更接近人类的自然语言。文献[14]对 N-Gram 用于恶意软件分类进行了研究，将文件看作字节序列，将 n 个连续字节的组合视为单个特征进行提取，当 $N=4$ 和 6 时，交叉验证分数分别为 97.4%和 95.6%。实验结果表明，当 N 大于 2 时，会导致恶意软件分类器过度拟合，并且特征在测试数据中的分布也不均匀。文献[15]为了确定 SARS-CoV-2 病毒的来源，需要将 SARS-CoV-2 的基因组序列与冠状病毒进行对比，N-Gram 可以从给定的基因组序列中提取相关信息，给出了当 $N=2,3,4$ 时 N-Gram 在 SARS-CoV-2 上的实验结果。结果表明，当 $N=2$ 时，给定的 2-g 出现的频率非常高，在给定的基因组中可能找到一个仅由两个特定核苷酸组成的序列。N-Gram 在解释 DNA 序列的隐藏信息等医学领域方面具有广泛的应用。自然语言处理和医学等领域已经成功应用了 N-Gram，但其在网络安全领域的应用较少，因此本文在恶意样本中选取 N-Gram 进行特征提取，使用了当 $N=2$ 时的 Bi-Gram 的方法。

2.3 选取分类算法

文献[16]将恶意软件生成 256×16 像素的灰度图像并提取了 N-Gram 来描述不同的恶意软件家族，文中研究了 8 种机器学习分类算法并测得 KNN 具有最佳的分类性能。文献[17]提出了基于随机森林的混合模型，该模型使用了 12 个隐藏层，主要判断给定的样本是恶意样本还是正常样本，模型的分类准确率为 85.1%。文献[9]使用的分类算法中，K 最近邻的分类效果最好，K 最近邻通过计算 K 个最近训练的恶意软件家族对输入的恶意样本进行分类，同一恶意软件家族的灰度图像相似度较高，距离很小；不同恶意软件家族的灰度图像相似度较低，距离很大。文献[9]重点介绍了 Kaggle 数据集训练数据中的.byte 文件，将恶意样本转换为 256×16 的灰度图像，在 K 最近邻中使用了 PCA 特征，准确率达到 96.6%。文献[18]使用了来自 CA Technologies VET Zoo 的数据集，收集了 967 个恶意软件样本，介绍了 4 种不同的恶意软件家族，在静态分析技术中，随机森林对标记数据的最高准确率为 93.557%，随机森林不受数据集变化的影响，因此能得到较好的分类结果。在学术界，机器学习在恶意软件分类领域具有广泛的应用，许多安全研究人员利用机器学习算法训练恶意软件分类模型[16-21]，对恶意软件进行家族分类能够有效分析恶意软件的恶意特征和攻击行为。在工业界，大多数安全厂商在对恶意软件进行命名时会参考计算机反病毒研究组织恶意软件命名格式(Computer Antivirus Research Organization，CARO)，一般格式为：Malware_Type：Platform/Family_Name.Group_Name.Major_Variant.Minor_Variant[[:Modifier]，同时工业界会根据目前网络安全现状和防恶意软件引擎的需求场景，对恶意软件的分类机制进行优化调整。

通过对文献进行分析，机器学习为训练更准确的恶意软件分类模型提供了很大的帮助，因此本文利用 KNN、逻辑回归、随机森林和 XGBoost 这 4 种机器学习算法进行恶意软件分类。

2.4 静态分析技术

静态分析技术和动态分析技术是恶意软件分析的两种基本方法[22-23]。首先，静态分析技术的检测速度快，可以检测存储在检测器数据库中的已知恶意软件，未知的恶意软件需要开发新的签名。其次，动态分析技术通过判断给定软件的特征和行为是否对计算机系统具有恶意行为来检测给定软件是否为恶意软件，但是检测过程非常耗时，检测结果的假阳性率较高。静态分析技术通常是分析恶意软件的第一步，通过分析恶意软件可执行程序的函数和结构来查看其功能，关注恶意软件对计算机系统和用户做了哪些操作。文献[24]提出了一种新的基于云的半监督迁移学习模型，该模型通过静态分析技术对恶意软件进行了分类，其中.asm 分类器的准确率为 99.69%，.byte 分类器的准确率由 94.3% 提高到 96.90%。文献[25]提取了 API 调用序列和操作码序列，训练了隐马尔可夫 HMM 模型，比较了静态分析、动态分析和混合分析下模型的检测率，静态分析技术下使用 IDA Pro 从.asm 文件中提取 API 调用和操作码。静态特征有哈希值、操作码、字符串、PE 标头信息等。PE(Portable Executable)是 Windows 操作系统中的可执行文件格式，本文主要在静态分析技术下研究 PE 格式的恶意软件样本。

3 恶意软件分类模型

构建基于机器学习的恶意软件分类模型可以归结为 4 个步骤：收集样本、提取特征、训练模型和模型测试，框架图如图 1 所示。收集大量的恶意样本并从每个恶意样本中提取特征，利用这些样本训练机器学习分类模型使其能够更准确地对恶意样本进行家族分类，使用训练好的恶意软件分类模型对样本进行分类并检测其分类效果。本节将详细讨论以上 4 个步骤中的内容。

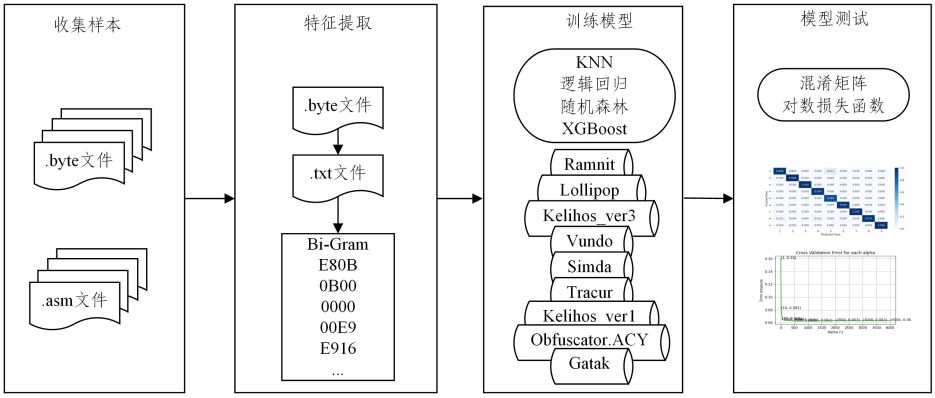


图 1 基于机器学习的恶意软件分类模型框架图

Fig. 1 Framework of malware classification model based on machine learning

3.1 收集样本

基于机器学习的恶意软件分类模型是否有效取决于数据集的数量和质量，数据集的样本越多，分类效果就越准确，同样，数据集的质量也很重要。本文使用的 Kaggle 微软恶意软件数据集将近 0.5 TB，恶意软件样本达到 21 741 个，其中包含有标记的样本 10868 个和未标记的样本 10873 个。每个样本都有类标签和 ID 值[26]，类标签指这个样本所属的恶意软件家族类别，ID 值是唯一标识这个样本的 20 个字符的哈希值，哈希值可以比作恶意代码的指纹。数据集中收集了 9 类不同家族的恶意软件：Ramnit，Lollipop，Kelihos_ver3，Vundo，Simda，Tracur，Kelihos_ver1，Obfuscator. ACY，Gatak。这 9 类不同家族的恶意软件在 Kaggle 数据集中的分布如图 2 所示。

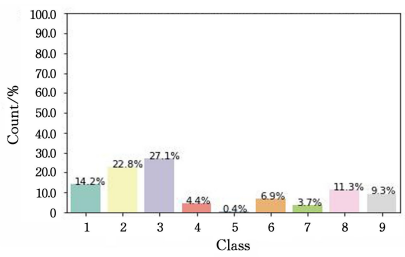


图 2 恶意软件在 Kaggle 数据集中的分布

Fig. 2 Distribution of malware in Kaggle dataset

Kaggle 数据集被安全研究人员认为是恶意软件领域的权威数据源，每个恶意软件样本去除了 PE 头，训练数据中的 21 736 个样本共 200 GB，其中.byte 文件 10 868 个，共 50 GB，.asm 文件 10 868 个，共 150 GB。.byte 文件由十六进制表示，.asm 文件由反汇编工具生成。本文使用了训练数据中的 10 868 个.byte 文件，样本数量和样本类型如表 1 所列。

表 1 Kaggle 数据集中 9 类恶意软件家族

Table 1 9 malwares in Kaggle dataset

Class	Family	Quantity	Type
1	Ramnit	1 541	Worm
2	Lollipop	2 478	Adware
3	Kelihos_ver3	2 942	Backdoor
4	Vundo	475	Trojan
5	Simda	42	Backdoor
6	Tracur	751	Trojan Downloader
7	Kelihos_ver1	398	Backdoor
8	Obfuscator. ACY	1 228	Malware
9	Gatak	1 013	Backdoor

从图 2 和表 1 中可以观察到，不同恶意软件家族的样本数量大不相同，Kelihos_ver3 有 2 949 个样本，而 Simda 只有 42 个样本，Kelihos_ver3 的样本数量是 Simda 样本数量的 70 倍。文献[27]中研究了通过恶意软件变形技术增加了训练集中的恶意软件样本数量，数据扩充训练集中包含原始样本和原始样本的一个或多个通过变形技术修改后的样本，解决了恶意软件家族中数据不平衡的问题，变形技术包括 dead code

插入技术、寄存器重新分配技术、子程序重新排序技术和通过跳转技术重新排序代码。

采用单位标准化的方法对 Kaggle 数据集中. byte 文件大小的范围进行标准化处理。. byte 文件的大小以字节为单位时，数值变化的幅度很大。在恶意软件分类过程中，分类模型对数值非常敏感，分类模型会认为数值大的特征数据比其他特征数据重要，但是在特征提取的过程中，并没有强调哪种特征更重要，因此需要对数值进行标准化。通过 statinfo. stat_size 返回每个. byte 文件以字节为单位的文件大小，将其除以 1 024，转换为以 kB 为单位的文件大小，再将其除以 1 024，转换为以 MB 为单位的文件大小。. byte 文件的大小经过处理后数值在较小的范围内上下浮动，. byte 文件的文件大小如图 3 所示。

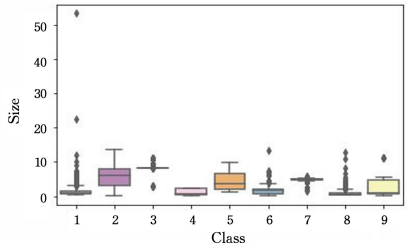


图 3　9 类恶意软件家族. byte 文件的大小范围

Fig. 3　Size range of . byte files in 9 malwares

从图 3 中可以发现第 2 类软件和第 5 类软件比较相似，但是第 2 类的样本数量比第 5 类的样本数量多，说明第 2 类广告软件比第 5 类后门发生的频率高。

3.2　提取特征

为了区分恶意软件家族或识别来自同一恶意软件家族的样本，通过 N-Gram 来处理序列数据用于比较恶意样本之间的相似性。首先，提取有助于恶意软件分类模型区分恶意软件家族的. byte 文件特征；其次，提取的特征数量不能多于收集的恶意样本数量；最后，确保提取的特征能够代表一系列恶意软件家族。

本文 Bi-Gram 的基本思想是. byte 文件转化为. txt 文件以后，将. txt 文件里面的内容生成长度为 2 的字节序列。N-Gram 假设第 m 个词出现的概率只与前$(m-1)$个词有关，整个句子出现的概率与每个词出现的概率的乘积有关。通常情况下使用单元的 Uni-Gram 和二元的 Bi-Gram 进行特征提取，本文 4.2 节分别对 $N=1$ 和 $N=2$ 进行了实验验证，实验结果表明，当 $N=2$ 时恶意软件分类模型效果更好。

假设一个句子由 m 个词组成，整个句子出现的概率为 $P(W_1,W_2,\cdots,W_m)$，根据条件概率的乘法法则，整个句子出现的概率如式(1)所示：

$$P(W_1,W_2,\cdots,W_m)=P(W_1)*P(W_2|W_1)*P(W_3|W_1,W_2)\cdots P(W_m|W_1,\cdots,W_{m-1}) \tag{1}$$

在 N-Gram 中，当 $N=2$ 时，即基于二元的 Bi-Gram 的方法，第 m 个词出现的概率只与前两个词有关，与最开始出现的词无关，二元模型的概率如式(2)所示：

$$P_2(W_1,W_2,\cdots,W_m)=P(W_1)*P(W_2|W_1)*P(W_3|W_2)\cdots P(W_m|W_{m-1}) \tag{2}$$

提取特征过程中创建了名为 bigram_keys 的字典，创建了包含 bigram 特征频数的. csv 文件，将以字典形式给出的 bigram 特征成功写入 bigrams_df，bigram_df 读取结果如表 2 所列。

表 2　bigrams_df 读取结果

Table 2　bigrams_df read results

E80B	0B00	0000	00E9	E916	1600	0090	9090	…	ID
18	1 427	273 053	922	11	1 300	992	1 614	…	01kcPWA9K2BOxQeS5Rju
9	293	19 852	34	0	144	27	4	…	04EjIdbPV5e1XroFOpiN
48	126	16 032	134	7	132	164	2	…	05EeG39MTRrI6VY21DPd
1	21	9 903	24	1	16	32	2	…	05rJTUWYAKNegBk2wE8X
0	5	15 288	7	1	4	3	0	…	0AnoOZDNbPXIr2MRBSCJ

除了特征频数和文件名称以外，. csv 文件中还增加了文件大小和家族类别两个标签，每个恶意样本的类别含有一个数字，数字对应该恶意样本所属的恶意软件家族类别。提取特征过程中计算了特征频数和文件大小两个标签的最大值和最小值，然后按照最小最大标准化对数值进行计算，最终的. csv 文件读取结果如表 3 所列。本文通过提取. byte 文件的特征来训练恶意软件分类模型，数据集中. byte 文件的特征有助于对恶意样本进行分类。

表 3　. csv 文件读取结果

Table 3　. csv file read results

	ID	0	1	2	3	4	5	…	Size	Class
0	01IsoiSMh5gxyDYTl4CB	39 755	8 337	7 249	7 186	8 663	6 844	…	6.556 152	2
1	01jsnpXSAlgw6aPeDxrU	93 506	9 542	2 568	2 438	8 925	9 330	…	4.602 051	9
2	01kcPWA9K2BOxQeS5Rju	21 091	1 213	726	817	1 257	625	…	0.679 688	1
3	01SuzwMJEIXsK7A8dQbl	19 764	710	302	433	559	410	…	0.438 965	8
4	02IOCvYEy8mjiuAQHax3	85 090	414	340	331	350	324	…	0.792 969	6

3.3　训练模型

从. byte 文件中提取特征之后就需要训练基于机器学习的恶意软件分类模型，分类模型是将样本分成多个类别，即将 Kaggle 数据集中的样本分成 9 类恶意软件家族。本节将详细讨论 KNN、逻辑回归、随机森林和 XGBoost 这 4 种机器学习分类算法。

KNN 的分类思想是，如果一个未知. byte 文件的 K 个最接近的文件大多属于 Ramnit 恶意软件家族，则这个. byte

文件就被判断为是 Ramnit 家族。K 指在训练模型中设置的最近邻居的数量，主要取决于确定一个.byte 文件所属恶意软件家族类别所需要的邻居数量。提取恶意样本的特征并在特征空间中找到与其最接近的 K 个样本，最常见的是使用欧几里得距离函数来计算特征空间中样本之间最短路径的长度，本文使用 KNN 的原因在于它清楚地解释了其分类思想，比较了样本之间的相似性。

逻辑回归实际上是个二分类问题，随机抽取一个样本并判断该样本是良性样本或恶意样本，或者随机抽取一个恶意样本判断该恶意样本所属恶意软件家族类别。通过查看逻辑回归模型的特征权重对样本进行检测，当权重值高时被认为是恶意软件，当权重值低时被认为是良性软件。与其他机器学习算法相比，逻辑回归是一种简单实用的算法，该算法不仅能得到较好的实验结果，还能大大降低其过拟合的风险。

随机森林由数百个或者数千个决策树组成，决策树从某个初始问题开始，以不同的问题来训练每个决策树，从而方便以不同的角度处理特征空间中的每个样本。在决策树创建的过程中限制决策树提出问题的数量可以降低决策树过拟合训练数据的风险。从训练集中随机抽取样本并通过这些样本构建决策树，运行决策树对每个恶意样本进行分类，直到完全确定训练集中每个恶意样本所属恶意软件家族类别。

XGBoost 的基本思想与梯度提升决策树(Gradient Boosting Decision Tree，GBDT)相同。在算法优化上，XGBoost 的损失函数对误差部分做了二阶泰勒展开；在运行效率上，对所有特征的值进行排序分组以便对决策树建立的过程进行并行选择；在算法鲁棒性上，对特征的缺失值进行处理以防止过拟合的现象出现。

3.4 模型测试

基于机器学习的恶意软件分类模型训练完成后需要检查模型的准确率。准确率是实验结果中真正的数量占恶意样本总数量的百分比，因此准确率可以用来衡量分类模型的效果。在分类模型评价标准中，本文使用的评价标准有混淆矩阵(Confusion Matrix)和对数损失函数(Logarithmic Loss Function)。

在恶意软件分类问题中，通过混淆矩阵对分类模型进行评价，混淆矩阵如表 4 所列，样本按照真实类和预测类可以得到 4 个一级指标。

表 4 混淆矩阵一级指标

Table 4 Primary index of confusion matrix

Primary index	Real class Positive	Real class Negative
Prediction class Positive	TP	FP
Prediction class Negative	FN	TN

真阳性(True Positive，TP)：样本真实类是第 n 类家族，预测类是第 n 类家族。

假阳性(False Positive，FP)：样本真实类不是第 n 类家族，预测类是第 n 类家族，FP 是统计学上的第一类错误，把负类判断为正类。

假阴性(False Negative，FN)：样本真实类是第 n 类家族，预测类不是第 n 类家族，FN 是统计学上的第二类错误，把正类判断为负类。

真阴性(True Negative，TN)：样本真实类不是第 n 类家族，预测类不是第 n 类家族。

通过以上 4 个一级指标可以延伸到 4 个二级指标，如表 5 所列。

表 5 混淆矩阵二级指标

Table 5 Secondary index of confusion matrix

Secondary index	Formula	Description
Accuracy	Formula(3)	判断正确的占总体的比重
Precision	Formula(4)	预测类中实际为真的比重
Recall	Formula(5)	真实类中实际为真的比重
Specificity	Formula(6)	真实类中实际为假的比重

$$Accuracy=\frac{TP+TN}{TP+FP+FN+TN} \tag{3}$$

$$Precision=\frac{TP}{TP+FP} \tag{4}$$

$$Recall=\frac{TP}{TP+FN} \tag{5}$$

$$Specificity=\frac{TN}{TN+FP} \tag{6}$$

损失函数指分类模型的预测类与真实类的不一致程度，损失函数越小，分类模型的分类效果越好。Logarithmic Loss Function 的计算式如(7)所示：

$$L(Y,P(Y|X))=-\log P(Y|X)=-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}y_{ij}\log(p_{ij}) \tag{7}$$

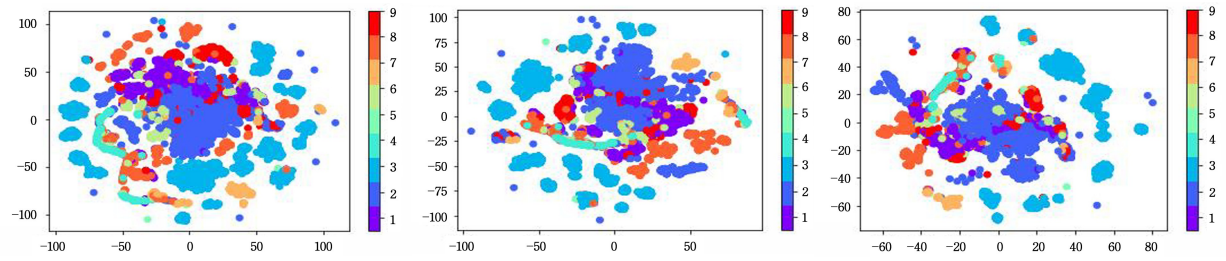
其中，L 为损失函数，Y 为输出变量，X 为输入变量，N 为输入恶意样本数量，M 为恶意样本家族的类别，y_{ij} 指类别 j 是否是输入的恶意样本 x_i 的真实类别，p_{ij} 指分类模型预测输入的恶意样本 x_i 属于类别 j 的概率。

4 实验结果

本文实验环境如下：操作系统为 Windows 10 64 位，处理器为英特尔 Core i7-9750H @ 2.60GHz 六核，显卡为 Nvidia GeForce RTX 2060。软件有 Anaconda3，Python3.8.5，Ubuntu20.04.1 等。为了验证分类模型的有效性，首先，使用不同数量和不同比例的恶意样本在分类模型中进行测试；其次，根据测试结果选取分类准确率最高的一组样本进行恶意样本分类；最后，对实验结果进行分析并从数据集、样本特征、分类算法和模型准确率 4 个方面与文献中的其他实验结果进行比较。

4.1 t-SEN 结果

t-SEN(t-Distributed Stochastic Neighbor Embedding)是一种降维技术，也是一种可视化技术。在本文中，t-SEN 生成了一个缩小的特征空间，家族相似的恶意样本由附近的点进行建模，家族不相似的恶意样本由远处的点进行建模。不同的 *perplexity* 值会产生不同的 t-SEN 图，当 $perplexity=30$ 时，t-SEN 如图 4(a)所示；当 $perplexity=50$ 时，t-SEN 如图4(b)所示；当 $perplexity=100$ 时，t-SEN 如图 4(c)所示。随着 *perplexity* 值的增加，图的形状越来越清楚，这表示恶意软件家族中不同的家族之间可以进行更好的分离。



(a)当 *perplexity*=30 时　(b)当 *perplexity*=50 时　(c)当 *perplexity*=100 时

图 4　t-SEN 结果

Fig. 4　t-SEN results

4.2　恶意样本实验结果

在 Kaggle 数据集每类恶意软件家族中随机选取 10 个恶意样本，共 90 个恶意样本，分别对 $N=1$ 和 $N=2$ 进行了实验验证，最终选择 $N=2$，实验结果包括训练集、交叉验证集、测试集的 Log Loss 值、错误分类点值和准确率，实验结果如表 6 所列。可以看出，Log Loss 值和错误分类点值越小，准确率越高，恶意软件分类模型效果越好，实验结果表明当 $N=2$ 时，恶意软件分类模型效果更好。

表 6　当 $N=1$ 和 $N=2$ 时的实验结果比较

Table 6　Comparison of experimental results when $N=1$ and $N=2$

ML	$N=1$					$N=2$				
	Train	CV	Test	Misclassified	Accuracy	Train	CV	Test	Misclassified	Accuracy
KNN	1.13	1.31	1.49	27.78	0.72	1.11	1.28	1.35	16.66	0.83
Logistic Regression	1.69	1.69	1.80	38.89	0.61	1.33	1.73	1.46	11.11	0.89
Random Forest	0.78	1.00	1.12	22.22	0.78	0.59	1.13	0.98	11.11	0.89
XGBoost	0.89	1.14	1.29	22.22	0.78	0.78	1.23	1.13	16.66	0.83

为了检测分类效果，将 Kaggle 数据集训练数据中的 .byte 文件按不同比例随机分成测试集和训练集，恶意样本的提取数量以及分配比例如下，实验结果准确率如表 7 所列。

表 7　实验结果准确率

Table 7　Accuracy of experimental results

(单位：%)

ML	A	B	C	D
KNN	55.56	48.14	90.40	91.90
Logistic Regression	27.78	33.33	58.97	61.82
RandomForest	72.22	77.78	97.55	97.93
XGBoost	66.67	74.07	97.73	98.43

A 组样本：在每类恶意软件家族中手动提取 10 个恶意样本，共 90 个恶意样本，按 8∶2 的比例随机分成训练集和测试集。

B 组样本：在每类恶意软件家族中自动提取 10 个恶意样本，共 89 个恶意样本（自动提取过程中含有重复的恶意样本，重复的恶意样本会自动删除，经过 20 次自动提取，89 个恶意样本与 A 组手动提取的 90 个恶意样本数量最接近），按 8∶2 的比例随机分成训练集和测试集。

C 组样本：Kaggle 数据集训练数据中的 .byte 文件，共 10868 个恶意样本，按 7∶3 的比例随机分成训练集和测试集。

D 组样本：Kaggle 数据集训练数据中的 .byte 文件，共 10868 个恶意样本，按 8∶2 的比例随机分成训练集和测试集。

表 7 中的测试结果表明：D 组分类准确率比其他组高；自动提取恶意软件样本比手动提取准确率高；机器学习对大量的恶意样本进行分类时分类模型的准确率会提高；与按 7∶3 的比例相比，按 8∶2 的比例进行实验时随机森林的准确率由 97.55% 提高到了 97.93%，XGBoost 的准确率由 97.73% 提高到了 98.43%。

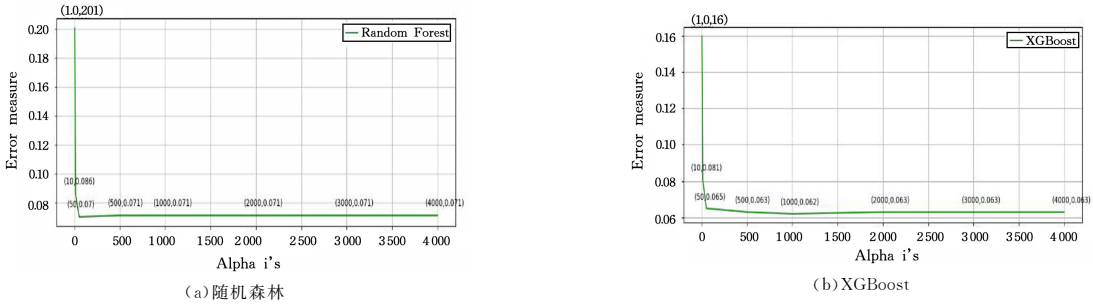
4.3　D 组样本的实验结果

在 D 组样本中，首先将 .byte 文件按 8∶2 的比例随机分成训练集和测试集，然后将训练集按 8∶2 的比例随机分成训练集和交叉验证集。D 组样本的 Log Loss 结果如表 8 所列，随机森林和 XGBoost 的超参数调整如图 5 所示。

表 8　D 组数据的 Log Loss 结果

Table 8　Log Loss results of group D data

ML	Log Loss			Misclassified Points
	Train	Cross Validation	Test	
KNN	0.173389	0.294453	0.324623	8.095676
Logistic Regression	1.101221	1.121656	1.118156	38.178472
Random Forest	0.026946	0.070169	0.091229	2.069917
XGBoost	0.022240	0.062490	0.078072	1.563937



(a)随机森林　(b)XGBoost

图 5　随机森林和 XGBoost 的超参数调整

Fig. 5　Hyper parametric adjustment of random forest and XGBoost

在 KNN 中，将 alpha 设置成[1,3,5,7,9,11,13,15,17,19]，当 $k=3$ 时，$Log\ Loss_{min}=0.294453907714634\ 1$。

在逻辑回归中，将 alpha 设置成[10^{-6},10^{-5},10^{-4},10^{-3},10^{-2},10^{-1},1,10,10^{2},10^{3},10^{4}]，当 $c=10$ 时，$Log\ Loss_{min}=1.121656318270487\ 3$。

在随机森林中，将 alpha 设置成[1,10,50,500,1000,2000,3 000,4 000]，当 $c=50$ 时，$Log\ Loss_{min}=0.070169602119032\ 9$。

在 XGBoost 中，将 alpha 设置成[1,10,50,500,1000,2000,3 000,4 000]，当 $c=1\ 000$ 时，$Log\ Loss_{min}=0.062490104337951\ 13$。

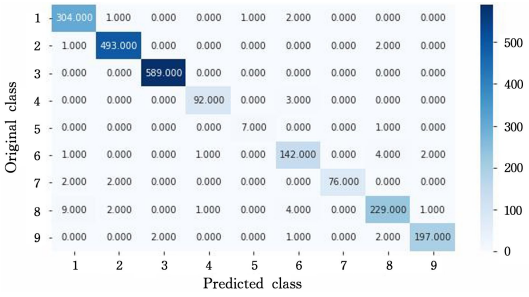
D 组样本的混淆矩阵 Accuracy，Precision 和 Recall 结果如表 9 所列。XGBoost 的分类效果最好，准确率高达 98.43%，其中第 3 类的预测值和真实值的情况完全相同。

表 9 D 组样本的混淆矩阵结果

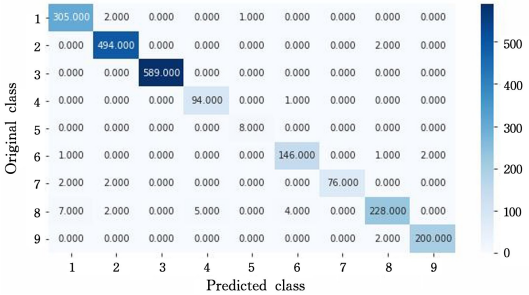
Table 9 Confusion matrix results of group D samples

ML	*Accuracy*/%		1	2	3	4	5	6	7	8	9
KNN	91.90	*Precision*	0.798	0.981	0.995	0.909	0.333	0.831	0.911	0.895	0.890
		Recall	0.925	0.913	0.997	0.947	0.250	0.887	0.900	0.902	0.762
Logistic Regression	61.82	*Precision*	0.259	0.702	0.993	0.000	0.000	0.000	0.941	0.000	0.804
		Recall	0.711	0.873	0.997	0.000	0.000	0.000	0.800	0.000	0.203
Random Forest	97.93	*Precision*	0.959	0.990	0.997	0.979	0.875	0.934	1.000	0.962	0.985
		Recall	0.987	0.994	1.000	0.968	0.875	0.947	0.950	0.931	0.975
XGBoost	98.43	*Precision*	0.968	0.988	1.000	0.949	0.889	0.967	1.000	0.979	0.990
		Recall	0.990	0.996	1.000	0.989	1.000	0.973	0.950	0.927	0.990

在混淆矩阵中，列对应预测类标签，行对应真实类标签。在恶意软件分类过程中，由于 Kaggle 数据集中收集了 9 类不同家族的恶意软件，所以混淆矩阵的大小为 9×9。主对角线指预测值与真实值相同的情况，其他部分指预测值与真实值不相同的部分。随机森林和 XGBoost 的混淆矩阵如图 6 所示。混淆矩阵显示了每种恶意软件家族分类模型预测准确的数量，此混淆矩阵还表明 XGBoost 能够正确预测第 3 类恶意软件家族。



(a)随机森林



(b)XGBoost

图 6 随机森林和 XGBoost 的混淆矩阵

Fig. 6 Confusion matrix of random forest and XGBoost

4.4 结果比较

从数据集、样本特征、分类算法和模型准确率这 4 个方面与文献中的其他实验结果进行比较，通过准确率来衡量恶意软件分类模型的成功程度，结果如表 10 所列。从表 10 中可以看出，在数据集不变的情况下，本文提取 N-Gram 特征比文献[9]、文献[11]、文献[12]、文献[24]提取的其他特征准确率高；在数据集不变且分类算法都采用 XGBoost 的情况下，本文提取 N-Gram 特征比文献[12]提取 Operation code 特征的准确率提高了 2.69%；在分类算法都采用随机森林但数据集不同的情况下，本文使用的 Kaggle 数据集比文献[18]使用的 VETZoo 数据集准确率提高了 4.373%。

表 10 与其他研究结果的对比

Table 10 Comparison with other research results

Reference	Dataset	Characteristics	Algorithm	Accuracy/%
文献[9]	Kaggle	PCA	KNN	96.600
文献[11]	Kaggle	Auto-encoders	DCGAN	95.700
文献[12]	Kaggle	Operation code	XGBoost	95.740
文献[18]	VETZoo	API	Random Forest	93.557
文献[24]	Kaggle	Gray image	RNN	96.900
This paper	Kaggle	N-Gram	Random Forest	97.930
This paper	Kaggle	N-Gram	XGBoost	98.430

结束语 本文提出了基于 N-Gram 静态分析技术的恶意软件分类研究，通过 N-Gram 方法在恶意软件样本中提取特征，利用 KNN、逻辑回归、随机森林、XGBoost 这 4 种机器学习算法对恶意软件样本进行分类，提高了恶意软件分类的准确率。在 Kaggle 恶意软件数据集中的测试结果表明，与已有的方法相比，借助 *N*-Gram 进行特征提取的方法可以更准确地对恶意软件进行分类。未来的研究工作将继续优化特征提取，进一步提高恶意软件分类的准确率。

参 考 文 献

[1] GUAN C, SUN K, LEI L, et al. Danger Neighbor attack: Information leakage via post Message mechanism in HTML5 [J]. Computers & Security, 2019(80): 291-305.

[2] YE Y, LI T, ADJEROH D, et al. A survey on malware detection using data mining techniques [J]. ACM Computing Surveys (CSUR), 2017, 50(3): 1-40.

[3] Malware bytes Labs. 2020 State of Malware Report [R]. Ireland: Malwarebytes, 2020.

[4] Malware bytes Labs. 2021 State of Malware Report [R]. Ireland: Malwarebytes, 2020.

[5] SINGH J, SINGH J. A survey on machine learning-based mal-

ware detection in executable files [J]. Journal of Systems Architecture,2021(112):101861.
[6] Microsoft. Microsoft Malware Classification Challenge(BIG2015)[EB/OL]. https://www.kaggle.com/c/malware-classification/overview.
[7] CAYIR A,UNAL U,DAG H. Random CapsNet forest model for imbalanced malware type classification task [J]. Computers & Security,2021(102):102133.
[8] XIAO G,LI J,CHEN Y,et al. MalFCS:An effective malware classification framework with automated feature extraction based on deep convolutional neural networks [J]. Journal of Parallel and Distributed Computing,2020(141):49-58.
[9] NARAYANAN B N,DJANEYE-BOUNDJOU O,KEBEDE T. Performance analysis of machine learning and pattern recognition algorithms for Malware classification [C]// 2016 IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit(OIS). 2016:338-342.
[10] ZHANG W,MEMBER S,WANG H,et al. DAMBA:Detecting android malware by ORGB analysis [J]. IEEE Transacyions Reliability,2020,69(1):55-69.
[11] YOUSEFI-AZAR M,VARADHARAJAN V,HAMEY L. Autoencoder-based feature learning for cyber security applications [C]// 2017 International Joint Conference on Neural Networks (IJCNN). 2017:3854-3861.
[12] KIM J,BU S,CHO S. Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders[J]. Information Sciences,2018(460/461):83-102.
[13] CHENG C,LI C,HAN Y,et al. A semi-supervised deep learning image caption model based on Pseudo Label and N-gram [J]. International Journal of Approximate Reasoning,2021(131):93-107.
[14] RAFF E,ZAK R,COX R,et al. An investigation of byte n-gram features for malware classification [J]. Journal Computer Virology and Hacking Techniques,2018(14):1-20.
[15] EI BOUJNOUNI H,RAHOUTI M,EI BOUJNOUNI M. Identification of SARS-CoV-2 origin:using Ngrams,principal component analysis and random Forest algorithm [J]. Informatics in Medicine Unlocked,2021(24):100577.
[16] HAN X,JIN F,WANG R,et al. Classification of malware for self-driving systems [J]. Neurocomputing,2021(428):352-360.
[17] YOO S,KIM S,KIM S,et al. AI-HydRa:Advanced hybrid approach using random forest and deep learning for malware classification [J]. Information Sciences,2021(546):420-435.
[18] HUDA S,MIAH S,HASSAN M,et al. Defending unknown attacks on cyber-physical systems by semi-supervised approach and available unlabeled data [J]. Information Sciences,2017(379):211-228.
[19] CUI Z,DU L,WANG P,et al. Malicious code detection based on CNNs and multi-objective algorithm [J]. Journal of Parallel and Distributed Computing,2019(129):50-58.
[20] Lracker. Worm. WhBoy. cw-Killer[EB/OL]. (2020-01-09)[2021-09-28]. https://github.com/lracker/Worm.WhBoy.cw-Killer.
[21] SURENDRAN R,THOMAS T,EMMANUEL S. A TAN based hybrid model for android malware detection [J]. Journal of Information Security and Applications,2020(54):102483.
[22] Baidubaike. WannaCry[EB/OL]. [2021-09-28]. https://baike.baidu.com/item/WannaCry/20797421.
[23] SIKORSKI M,HONIG A. Practical malware analysis:The hands-on guide to dissecting malicious software [M]. San Francisco:No Starch Press,2014.
[24] GAO X,HU C,SHAN C,et al. Malware classification for the cloud via semi-supervised transfer learning [J]. Journal of Information Security and Applications,2020(55):102661.
[25] DAMODARAN A,TROIA F,DI TROIA F,et al. A comparison of static,dynamic,and hybrid analysis for malware detection [J]. Journal of Computer Virology and Hacking Techniques,2017,13(1):1-12.
[26] RONEN R,RADU M,FEUERSTEIN C,et al. Microsoft malware classification challenge[J]. arXiv:1802.10135,2018.
[27] GIBERT D,MATEU C,PLANES J,et al. Auditing static machine learning anti-Malware tools against metamorphic attacks [J]. Computers & Security,2021(102):102159.

ZHANG Guang-hua, born in 1979, Ph.D, professor, master supervisor, is a senior member of China Computer Fede-ration. His main research interests include network and information security.

YU Nai-wen, born in 1983, master, assistant researcher. Her main research interests include computer network management and so on.

(责任编辑:何杨)