基于增强灰度共生矩阵的深度恶意代码可视化分类方法

王金伟[1,2,3], 陈正嘉[1,2], 谢 雪[4,5], 罗向阳[6], 马 宾[7]

[1]南京信息工程大学数字取证教育部工程研究中心 南京 中国 210044
[2]南京信息工程大学计算机学院 南京 中国 210044
[3]数学工程与先进计算国家重点实验室 中国 450001
[4]中国科学技术大学网络空间安全学院 合肥 中国 230031
[5]中国航天系统科学与工程研究院 北京 中国 100048
[6]中国人民解放军战略支援部队信息工程大学 郑州 中国 450001
[7]齐鲁工业大学网络空间安全学院 济南 中国 250353

摘要 随着恶意代码规模和种类的增加, 传统恶意代码分析方法由于需要人工提取特征, 变得耗时且易出错。同时, 恶意代码制作者也在不断研究和使用新技术手段逃避这些传统方法, 因此传统分析方法不再适用。近年来, 恶意代码可视化方法因其能够在图像中显示恶意代码的核心特征而成为研究热点。然而, 目前恶意代码可视化方法中存在一些问题。首先, 部分算法的模型训练复杂度较高, 导致了较长的训练时间和更高的计算成本。其次, 一些算法仅关注恶意代码的二进制级别特征, 可能无法捕捉到更高层次的特征信息。另外, 现有的算法大多针对恶意代码家族分类任务设计, 而这些算法在针对恶意代码类型分类方面的适用性较低。为了解决这些问题, 本文提出了一种基于增强灰度共生矩阵的深度恶意代码可视化分类方法。该方法将常应用于机器学习的灰度共生矩阵与深度学习相结合, 避免了手动特征提取的复杂度和难度。在预处理方面, 本文首先利用 Nataraj 矢量化方法将恶意代码数据集转化为灰度图像, 随后对其提取灰度共生矩阵并转化为灰度共生矩阵灰度图, 接着采用像素值乘积以实现图像增强, 有效减少图像中黑色像素点的个数, 增加图像亮度。在模型设计方面, 本文基于残差连接和密集连接的特性, 构建了 D-ResNet18 网络模型用于灰度图分类任务, 该模型能够充分利用每个层次的特征信息, 有效提取恶意代码的核心特征。实验结果表明, 本文提出的方法取得了优越的分类效果, 具有准确率高、训练速度快等优点, 且预处理操作简单, 适用于大规模恶意代码样本的快速分类等即时性要求较高的场景。更重要的是, 该方法在恶意代码家族分类和恶意代码类型分类两个任务上均表现出优越的性能, 相较于之前的方法, 准确率分别提高了 0.22%和 4.86%, 同时训练一轮所需时间分别缩短了 52.68%和 86.11%, 具有实际应用价值。

关键词 深度学习; 数据可视化; 恶意代码检测和分类; 灰度共生矩阵
中图法分类号 TP309 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2025.03.06

A Deep Learning Visualization Classification Method for Malicious Code Based on Enhanced Gray Level Co-occurrence Matrix

WANG Jinwei[1,2,3], CHEN Zhengjia[1,2], XIE Xue[4,5], LUO Xiangyang[6], MA Bin[7]

[1]Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology, Nanjing 210044, China
[2]Department of Computer, Nanjing University of Information Science and Technology, Nanjing 210044, China
[3]State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China
[4]University of Science and Technology of China, Hefei 230031, China
[5]China Aerospace Academy of Systems Science and Engineering, Beijing 100048, China
[6]PLA Strategic Support Force Information Engineering University, Zhengzhou 450001, China
[7]School of Cyberspace Security, Qilu University of Technology, Jinan 250353, China

通讯作者: 谢雪, 博士生, Email: xuexie2008@163.com。
本课题得到国家重点研发计划(No. 2021QY0700); 国家自然科学基金(No. 62072250, No. 62172435, No. U1804263, No. U20B2065, No. 61872203, No. 71802110, No. 61802212); 中原科技创新领军人才项目(No. 214200510019); 江苏自然科学基金(No. BK20200750); 河南省网络空间态势感知重点实验室开放基金(No. HNTS2022002); 江苏省研究生研究与实践创新项目(No. KYCX200974); 广东省信息安全技术重点实验室开放项目(No. 2020B1212060078); 山东省计算机网络重点实验室开放课题基金(No. SDKLCN-2022-05)资助。
收稿日期: 2023-04-28; 修改日期: 2023-07-26; 定稿日期: 2025-01-10

Abstract With the increase in scale and variety of malicious code, traditional methods for analyzing malicious code have become time-consuming and error-prone because they require manual feature extraction. Additionally, malicious code authors are continuously researching and using new techniques to evade these traditional methods, rendering them ineffective. In recent years, visualizing malicious code has become a research hotspot because it can display the core features of malicious code in images. However, there are several issues in current malware visualization methods. Firstly, some algorithms have high complexity in model training, resulting in longer training time and higher computational costs. Secondly, some algorithms only focus on the binary-level features of malware, which may fail to capture higher-level feature information. Additionally, existing algorithms are mostly designed for malware family classification tasks, and their applicability in malware type classification is limited. To address these issues, this paper proposes a deep malicious code visualization classification method based on enhanced gray-level co-occurrence matrices. This method combines gray-level co-occurrence matrices commonly used in machine learning with deep learning, avoiding the complexity and difficulty of manual feature extraction. In terms of preprocessing, this paper first uses the Nataraj vectorization method to transform the malicious code dataset into grayscale images, then extract the gray-level co-occurrence matrices and convert them into gray-level co-occurrence matrix gray-level images. We then use pixel value multiplication to enhance the image, effectively reducing the number of black pixels in the image and increasing its brightness. In terms of model design, this paper construct a D-ResNet18 network model based on the characteristics of residual connections and dense connections for grayscale image classification tasks. This model can effectively extract the core features of malicious code by utilizing the feature information in each layer. Experimental results show that our method achieves superior classification performance, with advantages such as high accuracy and fast training speed. Moreover, the preprocessing operation is simple and suitable for fast classification of large-scale malicious code samples and other scenarios with high real-time requirements. More importantly, this method demonstrates superior performance in both malware family classification and malware type classification tasks. Compared to previous methods, it achieves an accuracy improvement of 0.22% and 4.86% in the two tasks, respectively. Furthermore, the training time per epoch is reduced by 52.68% and 86.11%, respectively. These results highlight its practical value.

Key words deep learning; data visualization; malicious code detection and classification; gray-level co-occurrence matrix

1 引言

恶意代码是一种恶意设计的计算机程序, 其目的在于未经用户授权的情况下获取计算机系统中的敏感信息、破坏系统或实施其他有害行为。随着自动化生成工具和恶意代码混淆技术的广泛使用, 大量新的恶意代码迅速生成, 它们的入侵方式以及传播方式也不断变化, 对网络环境产生了巨大的威胁, 个人和组织的信息安全和经济利益受到严重影响, 恶意代码分析师面临着巨大的挑战。根据 2020 年《国家互联网应急中心第 16 期动态周报》报道, 一周内中国境内感染网络病毒的主机数量约为 54.8 万台[1]。

为了更好地理解恶意代码的特征、行为模式和攻击方式, 研究人员通常对不同类型的恶意代码进行分类。这样的分类任务有助于有效地识别和分析新型恶意代码的类别, 进而开发出有效的防御策略和工具, 从而更好地预防和应对恶意代码的攻击。因此, 恶意代码分类任务在保障信息安全方面具有重要意义。针对不同的分类角度, 恶意代码可以按照平台、类型和家族等维度进行划分。在平台维度上, 恶意代码可以分为 Windows、Linux 或安卓系统软件等不同种类, 而本文主要关注于 Windows 平台上的恶意代码。在类型维度上, 恶意代码可以分为木马、病毒、蠕虫、间谍软件以及广告软件等多种类型。不同类型的恶意代码具有不同的攻击方式和危害程度, 因此需要根据恶意代码的类型采取相应的防范和应对措施, 以降低攻击的风险和危害。此外, 恶意代码及其变体还可以按照家族进行分类, 如 Malimg 数据集[2]中包含了 Adialer.C、Agent.FYI 等多种恶意代码家族。不同的恶意代码家族可能使用不同的感染方式, 因此通过分类恶意代码家族, 可以更好地了解感染方式, 并开发相应的防御策略, 以帮助预防感染的发生。

早些年来, 为了应对不断增长的恶意代码, 越来越多的检测和分类方法被应用。这些方法包括机器学习算法, 如随机森林、决策树、支持向量机等, 以及深度学习算法, 如卷积神经网络(Convolutional Neural Networks, CNN)、循环神经网络(Recurrent Neural Network, RNN)等。这些方法的广泛应用极大地提高了恶意代码的检测效率和准确率, 并已成为恶意代码分析工程师的重要工具。

传统非可视化恶意代码检测和分类方法通常采用静态代码分析和动态代码分析两种技术。静态分析技术是一种快速获取恶意代码语法和语义信息的方法, 无需执行实际代码。通常, 该技术使用多种静态特征进行分析和分类。例如, 文献[3-5]采用 API 调用序列, 文献[6-9]提取字节序列的 N-Gram 特征, 文献[10]利用调用函数进行分析, 文献[11-12]则使用 PE 文件头。此外, 操作码频率分布、字符串签名以及控制流图等也是常用的静态特征。这些特征能够

反映出代码的结构和行为, 从而判断其是否为恶意代码。此外, 静态检测技术还可以借助各种工具, 例如 IDA Pro 等反汇编工具可以用于逆向分析恶意可执行文件, 提供更有效的信息; 而 LordPE 内存转储工具可以在系统内存中获取受保护的代码, 对于分析有更大的帮助。相比于静态分析, 动态分析技术通过在虚拟环境中执行代码来获取恶意代码的行为报告, 包括函数调用监测、功能参数分析、信息流跟踪、指令跟踪和动态可视化分析等。这种方法需要使用自动化工具来实现, 例如文献[13]使用 Anubis, 文献[14]使用 CWSandbox。此外, TTAnalyzer、Ether 和 ThreatExpert 等也是常见的自动化工具。两种技术都有其优缺点。静态分析具有时间复杂度和空间复杂度较低、速度快、效率高的优势, 并且可以全面地对恶意代码进行分析, 捕获语法和语义信息, 但在面对混淆和加壳代码时可能会漏检。相比之下, 动态分析技术更加准确和有效, 但需要投入更多时间和空间成本。

近年来, 可视化方法作为一种新兴的恶意代码检测和分类技术备受关注。恶意代码二进制文件中包含了大量人类难以理解的二进制代码, 将它们转换成图像后, 能够可视化其结构和特征, 从而方便进行分析和研究。通过可视化方法, 我们能够发现恶意代码图像中蕴含着丰富的信息。同一类别恶意代码通常具有相似的可视化图像, 而不同类别的可视化图像则有明显的差异。此外, 将二进制文件转换成图像的方法还为使用计算机视觉和深度学习方法进行恶意代码分类和检测提供了可能。相较于传统的特征提取方法, 可视化方法减少了特征提取过程设计的复杂度, 满足大数据计算、专家系统反馈和认知复杂性等方面的需求, 从而可以更加高效地检测和分类恶意代码。

目前, 关于可视化方法的研究主要集中在机器学习和深度学习两方面。然而, 这两个方面都存在一定的问题。在机器学习方面, 手动提取和选择特征的过程较为复杂和耗时, 而且分类效果受特征质量的影响较大。例如, 使用灰度共生矩阵(Gray-Level co-Occurrence Matrix, GLCM)[15]进行恶意代码分类需要提取角二阶矩、对比度和熵等特征, 这一过程较为繁琐。而在深度学习方面, 部分分类模型存在特征提取能力偏弱的问题, 这导致可视化方法或网络模型结构相对复杂。另外, 现有的算法大多针对恶意代码家族分类任务设计, 而这些算法在针对恶意代码类型分类方面的适用性较低。

为了解决上述问题, 本文采用可视化思想, 结合二进制程序静态文件结构, 提出了一种基于增强灰度共生矩阵的深度恶意代码可视化分类方法。该方法将恶意代码利用 Nataraj 矢量化转化为灰度图像后, 提取其灰度共生矩阵, 并转化为灰度共生矩阵灰度图, 随后利用像素值乘积以增强图像分类特征, 接着使用 D-ResNet18 神经网络模型进行训练与分类。

通过与相关文献实验结果对比, 本研究提出的基于增强灰度共生矩阵的深度恶意代码可视化分类方法在恶意代码家族分类和恶意代码类型分类两个任务上均表现出优越的性能。该方法不仅具有准确率高、训练速度快的特点, 而且在处理经过加壳操作的恶意代码时也展现出较好的分类准确度。在后续的实验中, 我们选择了包含加壳操作的 PE 数据集以及不包含加壳操作的 Malimg 数据集进行实验验证。实验结果证明, 无论恶意代码是否经过加壳操作, 我们的方法都能够提供可靠的分类结果, 使其在实际应用中具有广泛的适用性和可靠性。通过在多个数据集上的实验验证, 我们进一步确认了该方法的稳定性和有效性, 为其在实际场景中的应用奠定了坚实的基础。此外, 该方法将常用于机器学习的灰度共生矩阵与深度学习相结合, 利用深度学习网络自动提取图像特征, 避免了手动提取特征的工作量和难度。

本文主要贡献如下:

本文提出了一种预处理方法, 将常用于机器学习的灰度共生矩阵作为深度学习模型的输入。对于 Nataraj 矢量化生成的灰度图像, 本文对其提取灰度共生矩阵后并转化为灰度共生矩阵灰度图, 随后利用像素值乘积以增强图像分类特征, 提升分类效果。实验结果表明, 这种预处理方法可以提高恶意代码家族分类和恶意代码类型分类的准确率和效率。

本文设计了一种基于残差连接和密集连接特性的网络模型结构 D-ResNet18, 该模型能够在每个层次充分利用网络提取的特征信息, 从而实现更加丰富和准确的特征表达。

实验结果表明, 本文设计的基于增强灰度共生矩阵的深度恶意代码可视化分类方法在恶意代码家族分类和恶意代码类型分类两个任务上均表现出最佳效果, 拥有准确率高、训练速度快等优点。特别是在恶意代码类型分类任务上, 相较于先前的方法, 准确率提高了 4.86%, 一轮训练时间缩短了 86.11%。

本文的结构分为五个部分: 第二节概述了相关研究工作; 第三节详细介绍了基于增强灰度共生矩阵的深度恶意代码可视化分类方法; 第四节介绍了

实验的详细过程, 并对实验结果进行分析; 第五节对本文的工作进行了总结和展望。

2 相关工作

近年来, 恶意代码的不断演变给恶意代码检测带来了很大挑战。相比传统的非可视化方法, 将恶意代码转化为可视化图像后, 可以获得更优越的检测和分类效果。恶意代码图像中包含了丰富的信息, 同类恶意代码的可视化图像具有相似性, 而不同类恶意代码的可视化图像则存在明显差异。针对这些特点, 研究学者开展了恶意代码可视化研究, 主要采用两种方式: 一种主流方式是从图像的纹理或结构等方面入手, 提取特征并结合机器学习进行恶意代码的检测和分类; 另一种主流方式是使用深度学习自动提取特征进行学习, 这种方式可以有效对抗混淆技术。本文将从上述两个角度对相关工作进行介绍。

2.1 基于机器学习的恶意代码检测

近年来, 恶意代码可视化结合机器学习的方法已经展开了广泛而深入的研究。该研究方向主要致力于提取出能够实现良好分类效果且不易受到干扰的恶意代码可视化特征。在特征提取的基础上, 可以利用多种分类器对图像进行分类, 从而更好地实现恶意代码的分类。Nataraj 等人[16]的研究开启了结合可视化技术的恶意软件检测和分类的新兴领域。该研究首先将恶意软件.text 区块的二进制数据通过 Nataraj 矢量化技术转化为灰度图像, 再基于 GIST 算法[17-19]对转化后的灰度图特征进行提取, 随后使用 KNN 算法对提取的特征进行分类。此外, 在文献[20]中, 研究者还表明使用图像处理的二进制纹理分析技术可以更快速地对恶意软件进行分类。然而, 由于纹理分析方法具有较大的计算开销, 因此在处理大量的恶意软件时存在问题。Naeem 等人[21]提出了一种名为 LGMP 的特征提取方法, 该方法分为三个步骤: 首先, 采用 DSIFT 方法提取局部特征并进行选择; 其次, 提取 GIST 作为全局特征; 最后, 使用高斯权重将局部和全局特征进行集成, 得到 LGMP 特征。这种方法可以提高恶意软件的检测和分类准确率, 并且在实验中取得了良好的效果。Liu 等人[22]提出了一种基于机器学习的恶意软件分析系统, 该系统由数据处理模块、决策模块和检测模块三个主要组成部分构成。数据处理模块利用操作码 N-gram 和导入函数对灰度图像进行特征提取; 决策模块负责对恶意软件进行分类和识别可疑恶意软件; 检测模块使用 SNN 聚类算法来发现新的恶意软件家族。Fu 等人[23]提出了一种新的方法通过将熵、字节值和相对大小这三个特征分别映射到 RGB 三通道, 把恶意软件可视化为 RGB 彩色图像。在特征提取方面, 该方法采用了全局特征(如灰度共生矩阵和颜色矩)和局部特征(如部分字节码序列), 并使用随机森林、K 近邻和支持向量机等机器学习方法对恶意软件进行分类。这种方法不仅提高了模型的鲁棒性, 而且还为研究者提供了一种新的可视化手段, 能够更加直观地展示恶意软件的特征。李劭杰等人[24]提出了一种恶意代码检测方法, 该方法结合了多种特征和随机森林算法。特征包括 N-Gram 算法提取的文本特征、从灰度图中提取的纹理特征 GLCM 以及灰度直方图提取的颜色特征。作者将三种特征结合起来, 并利用随机森林算法对恶意代码进行检测。

近年来, 机器学习结合可视化方法在恶意代码检测和分类领域得到了快速发展, 但仍有多个方面可以进一步完善。首先, 手动选择并提取特征是机器学习的一大瓶颈, 该过程较为耗时, 需要寻求更有效的特征提取方法。第二, 目前在提取复杂的恶意代码图像特征时, 例如 GIST、SURF、DSIFT 和 LBP 等, 需要高计算成本, 同时这些特征提取技术在处理大规模数据集时效率较低, 需要寻求更高效的方法。

2.2 基于深度学习的恶意代码检测

目前, 深度学习技术, 如卷积神经网络和循环神经网络等, 已经成为恶意代码识别领域广泛研究的热点。这些技术能够自动学习数据集中的特征, 并实现高效的分类和检测。近年来, 基于深度学习的恶意代码检测得到了较快的发展。其中, 常用的方法是将恶意代码转换为灰度图或彩色图, 并将其输入到卷积神经网络中进行学习和分类。Kalash 等人[25]利用文献[16]的方法将恶意软件的二进制文件转换为灰度图像, 并使用卷积神经网络进行分类。在测试中, 研究者在 Malimg 和 Big2015[26]两个数据集上进行了验证, 结果显示其分类准确率分别高达 98.52%和 98.99%。由此可见该方法具有较高的分类精度, 能够有效地检测恶意软件。王博等人[27]提出了一种新的恶意代码检测方法, 将每个二进制 bit 串切割成长度为 8bit 的子串, 并将每连续三个子串分别对应 RGB 通道。这种方法可以对任意长度的恶意代码进行处理, 并将其转换为图像数据。然而, 该文章所设计的卷积神经网络结构参数量较大, 且采用的数据集样本量过少, 未能完全体现所设计的卷积神经网络结构的优势。因此, 需要更多的研究来验证该方法的性能和可靠性。Vasan 等人[28]在恶意代码检测领域提出了一种新的方法, 首先使用文献[16]中的 Nataraj 矢量化方法将恶意软件转化为二维数组, 然后添加彩

色映射生成彩色图片, 并利用调整后的 VGG16 网络模型对其进行检测和分类。该方法可以高效地识别混淆的恶意软件及其变种, 具有很高的效率和实用性。蒋考林等人[29]提出了一种将恶意代码转化为彩色图像的方法, 类似于文献[27], 但不同之处在于末尾数量不足的情况下使用 0 进行填充。作者使用 Alexnet 进行训练和分类, 相比于文献[27], 分类准确率提高了 1.8%, 模型参数也得到了减少。Ren 等人[30]提出了一种基于恶意软件二进制序列的可视化分析方法, 分为归一化、映射、学习与分类三个步骤。在归一化步骤中, 使用绿色、黑色、白色、紫色标记可显示字符、字节值为 0 的字符、字节值为 255 的字符和其他字节, 以区分可显示和不可显示字符。在映射步骤中, 采用六种空间填充曲线对恶意软件进行可视化。在学习与分类步骤中, 利用 VGG19 卷积神经网络提取特征并对恶意代码进行分类。实验结果表明, 该方法能够有效地提取恶意代码的特征。王润正等人[31]在其研究中采用了反汇编工具来提取恶意代码中的不同区块数据, 并对代码段和数据段进行分离和可视化操作。由于每个区块代表的恶意代码信息不同, 这种方法可以更直观地展现不同恶意家族之间的差异性。实验结果表明, 采用这种方法可以进一步提高恶意代码分类的准确性。

深度学习结合可视化方法在恶意代码检测与分类方面展现出了高效快速的特点, 但仍存在一些问题。首先, 在一些深度学习算法中, 模型结构参数量过大, 需要进一步优化模型结构以减少参数量。其次, 部分可视化方法中, 预处理方法过于复杂或者模型训练时间过长, 从而导致效率不高。因此, 我们需要持续探索和优化深度学习与可视化方法的结合, 以实现更高效的恶意代码检测与分类。

3 基于增强灰度共生矩阵的深度恶意代码可视化分类模型

本节将详细介绍本文提出的基于增强灰度共生矩阵的深度恶意代码可视化分类方法。该方法包括三个部分: 恶意代码可视化、神经网络模型构建以及模型训练与评价, 总体流程如图 1 所示。在第 3.1 小节和第 3.2 小节中, 我们分别详细介绍了所提出的恶意代码可视化方法和所构建的 D-ResNet18 网络模型。这两个部分是该方法的核心内容, 为恶意代码分类任务的成功实现提供了重要保障。其中, 恶意代码可视化方法能够将原始的恶意代码转化为像素增强的灰度共生矩阵灰度图, 使得模型可以从中提取更为丰富、更有实用价值的特征信息。而 D-ResNet18 网络模型能够高效地提取图片特征并将其成功应用于恶意代码分类任务中。

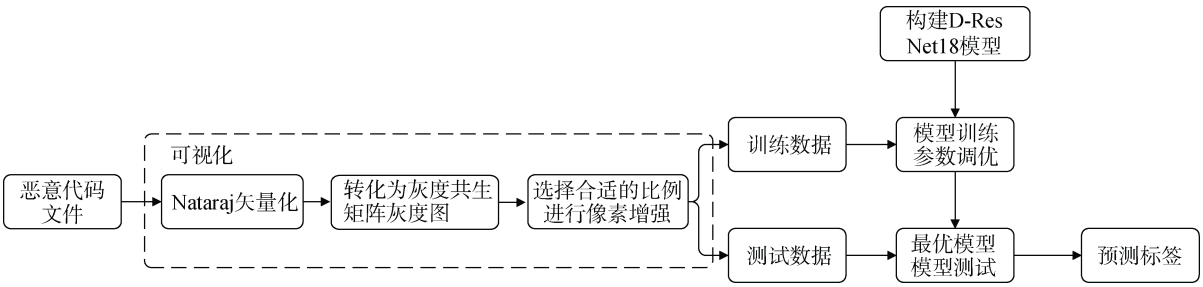


图 1 基于增强灰度共生矩阵的深度恶意代码可视化分类方法流程图

Figure 1 Flow chart of deep malicious code visualization classification method based on enhanced gray co-occurrence matrix

3.1 恶意代码可视化

本文所提出的可视化过程包含以下三个步骤: (1) 利用 Nataraj 矢量化方法将原始数据集转化为灰度图像。(2) 对灰度图像提取灰度共生矩阵, 并转化为灰度共生矩阵灰度图。(3) 采用像素值乘积以实现图像增强。

以下是对每个步骤的详细介绍。为了便于叙述, 我们将第二步和第三步合并叙述:

1) Nataraj 矢量化方法转化为灰度图

为了将二进制恶意代码文件可视化为灰度图像, 本研究采用了 Nataraj 矢量化方法。具体而言, 首先, 将二进制文件分割成长度为 n 个 8bit 的子序列(不足 8bit 的后几位被舍去), 并将每个子序列转换为[0,255]之间的数。这些数按顺序排成一行, 然后对其总长度取平方根, 取整得到整数 m 。其次, 将这些数排列成一个 $m \times m$ 的正方形数组(多余部分被舍去), 其中像素值 0 表示黑色, 像素值 255 表示白色。通过这一转换, 可以成功将二进制恶意代码文件转换为一张正方形灰度图像, 这为后续的图像处理和分类任务提供了便利。具体转换过程如图 2 所示。

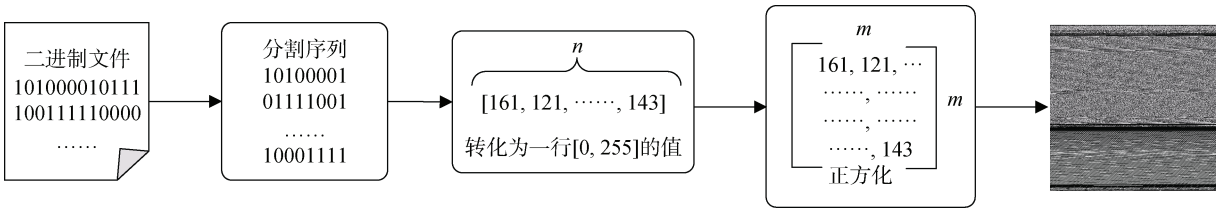


图 2 Nataraj 矢量化示意图

Figure 2 Nataraj Vectorization diagram

2) 转化为灰度共生矩阵灰度图并进行像素值乘积

灰度共生矩阵是一种用于描述图像纹理特征的统计工具。它可以衡量不同灰度级别像素之间的空间关系, 一般被定义为从灰度级为 i 的像素点离开某个固定位置关系到达灰度级为 j 的像素点的概率。如公式 1 所示, 灰度共生矩阵中每个值用 $p(i,j,d,\theta)$ 表示, 其中 i, j 分别表示像素点的灰度; d 表示两个像素点间的空间位置关系, 不同的 d 决定了两个像素点间的距离和方向; θ 表示灰度共生矩阵的生成方向, 在每一方向上都可以得到多类特征, 常取 0°, 45°, 90°和 135°这四个方向; N 表示原图像尺寸。

$$p(i,j,d,\theta)=\{(x,y),(x+dx,y+dy)\in N\times N \mid f(x,y)=i, f(x+dx,y+dy)=j\} \tag{1}$$

在常见的机器学习应用中, 通过对灰度共生矩阵进行计算, 可以获得多个统计量参数, 例如角二阶矩、对比度和熵等, 以反映样本的纹理特征。这些参数在恶意代码分类中可以体现不同恶意代码类别之间的差异性, 以提高分类和检测的准确性和鲁棒性。然而, 灰度共生矩阵结合机器学习的主要缺陷在于需要手动提取灰度共生矩阵中的统计量参数作为特征向量, 并进行分类器的选择。这种方法需要研究人员具有专业领域知识和进行大量的实验设计和调试, 增加了工作量和难度。

为了解决这些问题, 本文提出将灰度共生矩阵与深度学习相结合, 无需手动提取特征。在本文中, 我们的灰度共生矩阵是通过统计灰度级别相邻的像素对出现的次数而得到的矩阵, 其中每个元素表示了一对像素在 $\theta=0°$, $d=1$ 空间位置关系上出现的次数。接下来, 我们将灰度共生矩阵转化为灰度图进行处理, 并将图像像素值乘以经过优化的系数来增强图像特征, 以利用深度学习网络自动提取图像特征和语义信息, 提高恶意代码分类的准确性和鲁棒性。

由于在本研究提出的方法中, 灰度共生矩阵的数值代表了一对像素在 $\theta=0°$, $d=1$ 空间位置关系上出现的次数, 因此矩阵中数值差异较大。研究过程中, 我们对 Malimg 数据集与 PE 数据集中训练集的所有灰度共生矩阵最大数值与最小数值进行了统计, 在 Malimg 数据集和 PE 数据集中, 矩阵中最大数值平均值分别为 19452 和 200272, 而最小数值平均值分别为 1 和 32。若直接对其可视化, 一方面, 矩阵中的数值差异较大, 可能导致图像中的颜色变化过于剧烈, 难以有效地展示不同数值之间的差异。另一方面, 一些数据的细节特征会被掩盖, 从而导致数据的信息丢失。因此, 为了避免数值差异过大导致的图像质量问题, 更好地展示灰度共生矩阵的特征, 我们采用了归一化的方法, 对矩阵中的每个数值除以矩阵中最大数值, 从而将矩阵中的每个数值映射到[0,1]区间内。随后, 我们对归一化后的每个数值乘以 255, 将其映射到[0,255]内, 生成了转化后的灰度图, 本文称其为灰度共生矩阵灰度图。在该过程中, 每个灰度共生矩阵中的元素被有效地转换为图像上对应像素的灰度值, 为进一步分析和识别灰度共生矩阵的特征提供了便利。值得一提的是, 本文中我们选择了 256×256 的灰度共生矩阵尺寸, 即对于每个 Nataraj 灰度图, 我们都能将其转化为大小为 256×256 的灰度共生矩阵灰度图, 以进一步提升训练效果和分类准确度。

但是在转化过程中, 我们发现图像存在黑色像素点比例较高、暗淡的问题。这种情况可能会影响恶意代码分类的效果, 因为黑色像素点过多会掩盖灰度共生矩阵的细节特征。为了解决这一问题, 本文采用了像素值增强的方法, 即将灰度共生矩阵灰度图中每个像素值乘以一个增强比例系数(若像素值超过 255 则截断为 255), 该方法可以有效减少黑色像素点的比例, 增加图像亮度, 使恶意代码图像更加清晰。研究过程中, 我们对 Malimg 数据集与 PE 数据集中训练集的所有灰度共生矩阵灰度图进行了统计, 在 Malimg 数据集和 PE 数据集中, 增强前黑色像素点占比平均值高达 93.96%和 99.23%。然而, 将矩阵中每个像素值乘以 100 进行增强后, 黑色像素点的比例显著减少, 分别减少到了 46.72%和 47.57%。由

此可见, 增强图像可以大幅度减小黑色像素点所占比例, 从而更加突出灰度共生矩阵的特征, 提升分类效果, 有助于分类模型更好地学习和识别恶意代码的特征。

上述操作过程如公式 2 所示, 其中 x 表示灰度共生矩阵中的每一个数值, $\max\_x$ 表示矩阵中所有数值的最大值, a 表示增强比例系数。

$$x = x / \max\_x \times 255 \times a \tag{2}$$

此外, 采用灰度共生矩阵灰度图的处理方法还具有训练加速的优势。在直接对未处理的 Nataraj 矢量化灰度图进行训练时, 由于其较大的图片尺寸, 训练速度较慢, 使得高效的分类难以实现。而在进行灰度共生矩阵灰度图处理后, 图片尺寸得到了大幅减小, 从而提高了训练速度和分类效率, 有助于提高模型的性能和泛化能力, 同时该处理方式还可以降低存储成本。

为了进一步展示本文提出的可视化方法, 方便读者理解, 我们对其进行了图片绘制, 如图 3 所示。

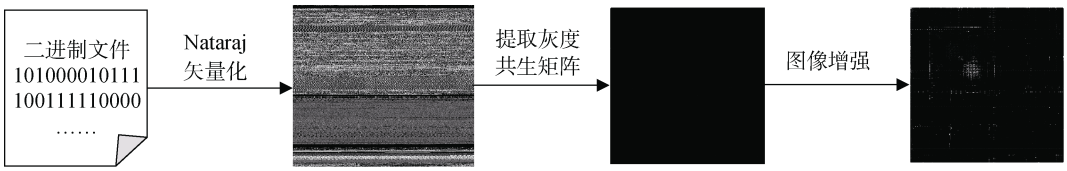


图 3 所提可视化方法示意图

Figure 3 Nataraj Vectorization diagram

3.2 神经网络模型构建

本文旨在探索一种能够在恶意代码家族分类和恶意代码类型分类两个任务上均表现出优越性能的恶意代码分类模型。其中, 在恶意代码家族分类任务中, 我们使用了公开的 Malimg 数据集, 该数据集包含 9339 张图片, 具有相对较大的数据规模。在恶意代码类型分类任务中, 我们使用了一个非公开的 PE 数据集, 该数据集包含 5094 张图片, 数据规模相对较小。因此, 本文旨在构建的模型需要在小型和大型数据集上均适用, 具有较强泛化能力。

在处理不同规模的数据集时, 我们应根据实际情况选择适合的深度学习模型, 因为不同的模型具有各自的优势。ResNet18[32]具有较少的参数, 通过残差连接使信息在网络中自由流动, 可有效缓解过拟合问题, 提高了训练效果和泛化能力。而 DenseNet121[33]采用密集连接的方法构建网络, 通过特征在 channel 上的连接来实现特征重用, 使信息更充分地传递, 并通过每个卷积层的输出直接连接到后续所有层的输入中, 增加了梯度在整个网络中的流动, 提高了训练效率和稳定性。然而, 为了在多个不同规模的数据集上都能表现出较好的性能, 我们可以考虑将残差连接和密集连接结合起来, 以充分发挥两种模型的优势。

为此, 本文提出了一种新型网络模型—D-ResNet18。该模型在 ResNet18 网络模型的基础上, 引入了 DenseNet 的密集连接优势进行改进。D-ResNet18 结合了两种模型的优点, 以实现在不同任务和数据集上拥有更好的适用性和泛化能力。通过引入 DenseNet 中的密集连接和特征重用机制, D-ResNet18 网络可以更好地利用数据的信息, 提高特征提取和分类的准确性。同时, 相较于 DenseNet, D-ResNet18 在模型结构上相对简单, 使其在小型数据集上也具有较强的适用性。

为了方便读者对 D-ResNet18 模型的结构进行理解, 本文绘制了该模型的示意图, 如图 4 所示。其中, 图 4(a)展示了 D-ResNet18 网络结构的整体示意图, 而图 4(b)、(c)、(d)和(e)则分别对应了图 4(a)中 D-BasicBlock-A、D-BasicBlock-B、D-BasicBlock-C 和 D-BasicBlock-D 四个部分。需要注意的是, 图 4(b)、(c)、(d)和(e)示意图中左侧部分代表密集连接, 右侧部分代表残差连接, 虚线则表示需要进行下采样操作以保持图像面积或通道数的一致性, 从而方便后续相加(add)和拼接(concat)操作的进行。

如图 4(a)所示, D-ResNet18 网络模型主体结构与 ResNet18 网络模型相似。在前向传播过程中, 该模型首先对输入图像进行卷积、批量归一化(Batch Normalization, BN)[34]、激活函数和池化等操作, 提取特征信息。接着, 通过四个卷积层组, 每个卷积层组包含两个 BasicBlock, 对特征进行深度处理。每个 BasicBlock 包含两个 3×3 的卷积层和一个残差连接, 在残差连接中, 第二个卷积层的输入与第一个卷积层的输出相加, 得到残差块的输出。在输入和输出维度不同的情况下, 使用一个 1×1 的卷积层与一个批量归一化层进行下采样, 以保证残差块的输入和输出维度相同。最后, 通过全局平均池化层和全连接层进行分类。这一过程可以有效地提取图像特征, 为后

续的分类任务提供有力的支持。

如图 4(b)、(c)、(d)和(e)所示, 与 ResNet18 网络模型不同的是, D-ResNet18 网络模型在每一个卷积组中除了 BasicBlock 的残差连接操作外, 还添加了一个密集连接操作, 即将该卷积组的输入与输出在 channel 上进行拼接, 用于下一卷积组的输入。但由于卷积组的输入与输出图片尺寸不一致, 因此需要使用一个 1×1 的卷积层进行下采样, 以保证密集连接的输入和输出图片尺寸相同。在卷积层后, 还添加了一个批量归一化层, 可以加速深度神经网络的训练速度并提高模型的泛化性能。从图 4 中可以看出, A 部分不需要下采样操作, 因为经过 A 部分后通道数和图片尺寸均不会改变; B, C 部分的残差连接和密集连接都需要进行下采样; 而 D 部分不需要密集连接, 因为它已经是最后一层卷积组, 无需将输入输出拼接用于下一卷积组的输入。

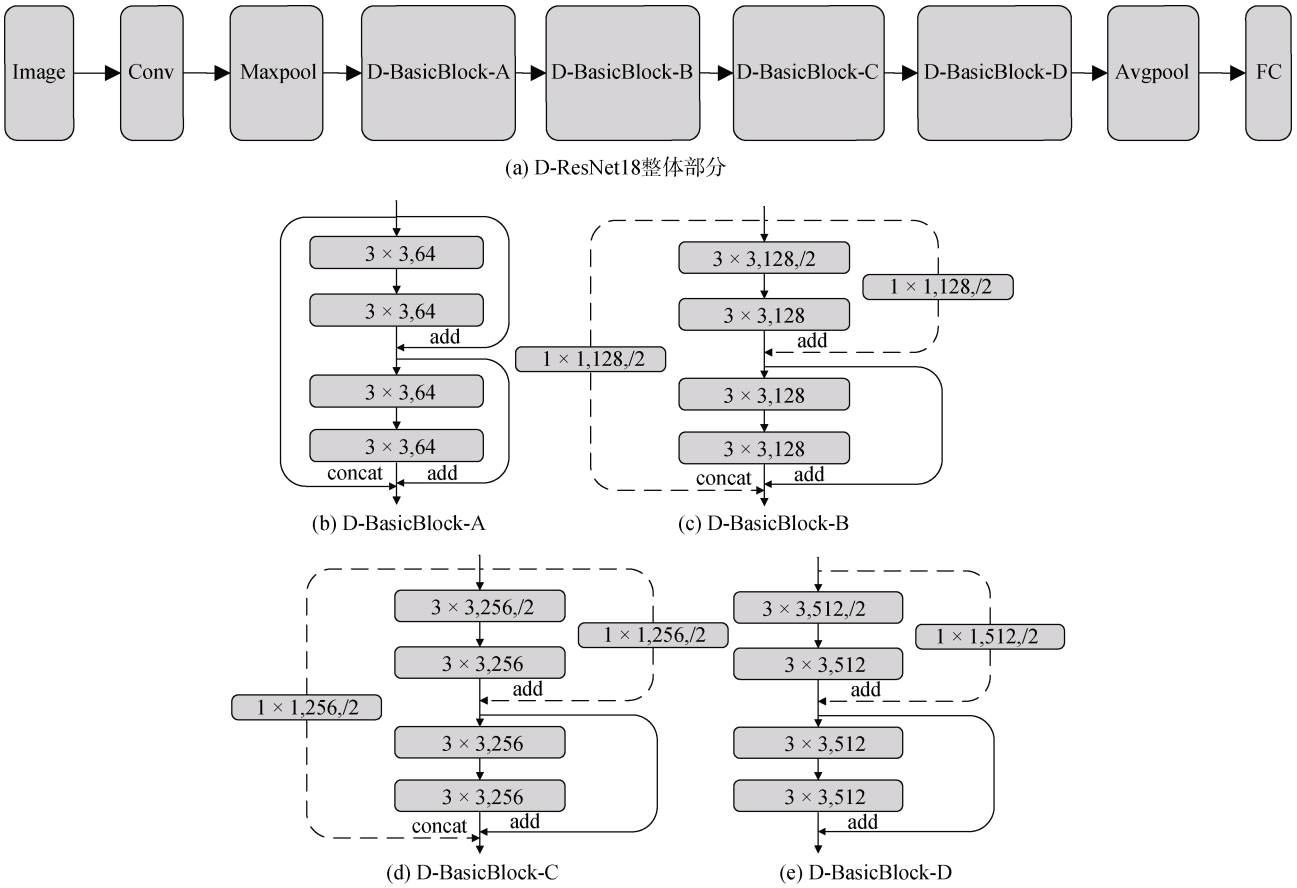


图 4 D-ResNet18 网络结构

Figure 4 D-ResNet18 network structure

为了更加详细地描述图 4(a)中 D-ResNet18 整体部分的网络结构参数, 我们绘制了表格, 如表 1 所示。需要注意的是, D-BasicBlock-A、D-BasicBlock-B、D-BasicBlock-C、D-BasicBlock-D 部分的详细参数以及下采样细节已经在图 4 中得到了展示, 在该表中我们将不再重复呈现。

本文统一将输入图像的尺寸调整为 224×224。这样的处理方式具有以下优点: 一方面, 可以使输入图像的尺寸保持一致, 简化了模型的训练过程。另一方面, 由于一些深度学习模型对输入图像的尺寸有着严格的要求, 使用统一的缩放策略可以确保模型能够在不同的图像尺寸下表现出较好的性能。

表 1 D-ResNet18 网络结构参数

Table 1 D-ResNet18 Network structure parameter

层名	输出大小	输出通道数	具体结构
Conv	112×112	64	7×7,/2
Maxpool	56×56	64	3×3,/2
D-BasicBlock-A	56×56	128	[3×3,64]×4
D-BasicBlock-B	28×28	256	[3×3,128]×4
D-BasicBlock-C	14×14	512	[3×3,256]×4
D-BasicBlock-D	7×7	512	[3×3,512]×4
Avgpool、FC	1×1	1	-

D-ResNet18 模型的优点是多方面的, 该模型采用了密集连接和残差连接相结合的方式, 充分利用

每个层次的特征信息, 提高特征提取能力, 有助于提高模型的准确性和泛化能力。同时该模型还拥有更低的计算复杂度, 相比于 DenseNet 减少了计算量 FLOPs, 具有更高的计算效率, 同时仍保持着较好的性能表现。

综上所述, D-ResNet18 模型结合了 ResNet 和 DenseNet 的优点, 不仅具备更优秀的特征提取能力, 而且具有更好的泛化能力和更低的计算复杂度。这些优点使得 D-ResNet18 模型在各种应用中表现出色。

4 实验与结果分析

4.1 数据集

本文研究的主要目标是针对恶意代码家族分类和恶意代码类型分类两个任务进行分类方法设计与优化。为了实现这一目标, 我们在实验中采用了两个不同的数据集。

4.1.1 恶意代码家族分类数据集

在恶意代码家族分类任务中, 本研究使用了公开的恶意代码家族数据集 Malimg 作为实验数据集。该数据集已经将恶意代码二进制文件转化为了 256×256 像素的灰度图像, 该图像的像素值通过将二进制数据转换为十六进制, 并将其映射到 0 到 255 的灰度值范围内得到。这种转换方式的目的是为了方便利用计算机视觉和图像处理技术对恶意代码文件进行分析和分类。换句话说, 该数据集已经实现了恶意代码的可视化操作, 通过将恶意代码二进制文件转换成灰度图片的方式构建而成。

具体来说, Malimg 数据集包含了 9339 张图片, 覆盖了 25 个不同的恶意代码家族, 是一个相对较大的数据集。每个图像都对应着一个恶意代码文件, 可以看做是该文件的可视化呈现。

表 2 中列出了 Malimg 数据集中各类别的名称以及相应的样本数量。

4.1.2 恶意代码类型分类数据集

在恶意代码类型分类任务中, 由于当前公开的恶意代码类型数据集较为有限, 因此本研究构建了一个非公开的数据集, 包含了 5094 个恶意代码 PE 文件, 这些文件均来自企业内部。该数据集扩展了现有公开数据集的规模和多样性, 为恶意代码类型分类任务提供了更加完整的数据支持。PE 文件是 Windows 操作系统上的常见可执行文件格式, 也称为“可移植执行”(Portable executable)文件, 其中包含了程序代码、数据、资源和元数据等信息。恶意代码 PE 文件指的是被恶意程序感染的 PE 文件, 即包含了恶意代码的可执行文件。这些文件可能会执行各种恶意行为, 例如窃取用户信息、加密或删除文件、篡改系统配置等。由于 PE 文件是 Windows 操作系统上的核心文件类型之一, 因此恶意程序通常会以 PE 文件的形式传播和执行, 而针对恶意代码 PE 文件的分析和检测也是恶意代码研究中的重要方向之一。

表 2 Malimg 数据集 25 个恶意代码家族及其数量
Table 2 Malimg dataset 25 malicious code families and their number

恶意代码家族	数量	恶意代码家族	数量	恶意代码家族	数量
Adialer.C	122	Fakerean	381	Swizzor.gen!E	128
Agent.FYI	116	Instantaccess	431	Swizzor.gen!I	132
Allaple.A	2949	Lolyda.AA1	213	VB.AT	408
Allaple.L	1591	Lolyda.AA2	184	Wintrim.BX	97
Alueron.gen!J	198	Lolyda.AA3	123	Yuner.A	800
Autorun.K	106	Lolyda.AT	159	总计	9339
C2LOP.gen!g	200	Malex.gen!J	136		
C2LOP.P	146	Obfus-cator.AD	142		
Dialplatform.B	177	Rbotigen	158		
Dontovo.A	162	Skintrim.N	80		

我们的数据集覆盖了 6 个恶意代码类型, 分别为后门(Backdoor)、通用(Generic)、木马(Trojan)、变形(Variant)、病毒(Virus)和蠕虫(Worm)。需要注意的是, Backdoor 和 Trojan 这两个恶意代码类型均属于卡巴斯基这一木马大类, 因此它们的样本具有较高的相似性, 难以进行准确分类, 从而导致分类准确率相对较低。

由于该数据集中的所有文件类型均为 PE 文件, 为了方便描述, 本文中我们将该数据集称为 PE 数据集。表 3 列出了 PE 数据集中各类别的名称及相应的样本数量。

表 3 PE 数据集 6 个恶意代码类型及其数量
Table 3 PE data set 6 malicious code types and their number

恶意代码类型	数量
Backdoor	898
Generic	898
Trojan	808
Variant	898
Virus	709
Worm	883
总计	5094

4.2 实验结果与分析

本研究的实验分为三部分: 第一部分旨在比较多个恶意代码家族以及恶意代码类型的可视化效果; 第二部分为消融实验, 其中包括预处理结构的消融实验、 图像增强比例的消融实验、网络结构的消融实验和 D-ResNet 模型细节的消融实验, 以验证本文实验预处理步骤三个阶段的有效性以及模型 D-ResNet18 的性能提升; 第三部分为对比实验, 旨在将本文提出的方法与其他效果较好的文献方法进行比较分析。

在实验二和实验三中, 即消融实验和对比实验中, 我们将批处理样本数设置为为 128, 初始权重设置为随机, 优化器设置为 SGD 优化器, 损失函数设置为交叉熵损失函数。我们采用经验学习率值 0.01, 并使用余弦退火调度器来降低 optimizer 的学习率从初始值到最小值, 再将其逐渐恢复到初始值, 这一过程每 200 个 epoch 执行一次。该调度器能够提高模型的训练效果, 使其更快地收敛到最优解。

我们对 Malimg 数据集和 PE 数据集进行了随机抽样, 并将其中 80%的样本作为训练集, 另外 10%的样本用于测试集, 剩余 10%的样本用于验证集。每个模型每次训练采用 200 个 epochs, 并记录每次训练在验证集上的最高分类准确率作为本次训练的分类准确率。由于实验的随机性, 我们对每个模型进行了 10 次独立的训练, 并将这 10 次实验结果的分类准确率取平均值作为该模型的表现评估结果, 从而降低了实验误差的影响。

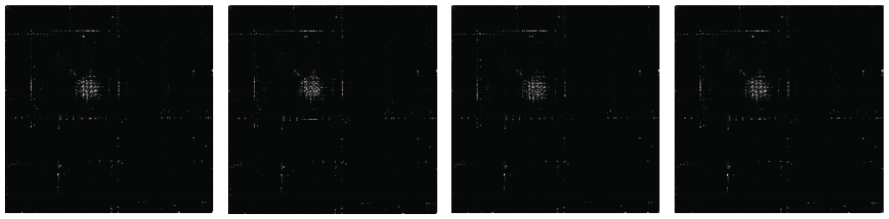
4.2.1 可视化实验

为了验证恶意代码家族分类和恶意代码类型分类的可视化效果, 我们从 Malimg 数据集中选取 Adialer.C 类和 Agent.FYI 类, 并从 PE 数据集中选取 Variant 类和 Worm 类, 每个类均选取 4 个样本进行可视化展示。将这些样本转换为增强后的灰度共生矩阵灰度图, 观察各恶意代码生成图像的可视化效果与纹理特征, 其中 Malimg 数据集如图 5 所示, PE 数据集如图 6 所示。

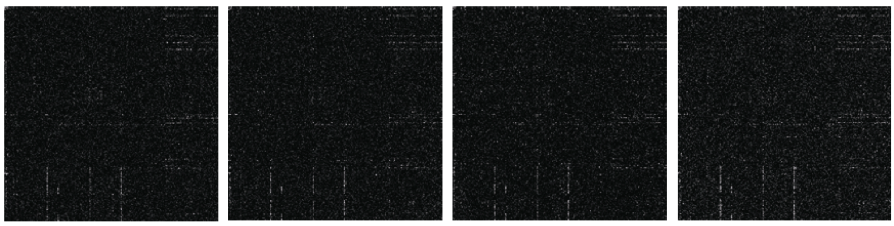
在恶意代码家族分类中, 图 5(a)展示了 Adialer.C 恶意代码类内的 4 个不同样本实例的可视化生成图, 可以看出它们之间存在高度的相似性; 图 5(b)展示了 Allaple.L 恶意代码类内的 4 个不同样本实例的可视化生成图, 也可以看出它们之间也存在高度的相似性。然而, 图 5(a)与图 5(b)之间存在较大的图片差异性。由此我们可以发现, 在恶意代码家族分类中, 不同恶意代码家族之间差别较大, 而相同家族之间往往具有高度的相似性。

在恶意代码类型分类中, 图 6(a)展示了 Variant 恶意代码类内的 4 个不同样本实例的可视化生成图, 可以看出它们之间存在高度的相似性; 图 6(b)展示了 Worm 恶意代码类内的 4 个不同样本实例的可视化生成图, 也可以看出它们之间也存在高度的相似性。同样地, 图 6(a)与图 6(b)之间也存在较大的图片差异性。由此我们也可以发现, 在恶意代码类型分类中, 不同恶意代码类型之间差别较大, 而相同类型之间往往具有高度的相似性。

因此, 我们可以得出结论, 无论是在恶意代码家族分类还是恶意代码类型分类任务中, 本文提出的基于增强灰度共生矩阵的深度恶意代码可视化分类方法均表现出较好的可视化效果。通过可视化后, 相同类恶意代码样本的图像纹理特征表现出相似性,



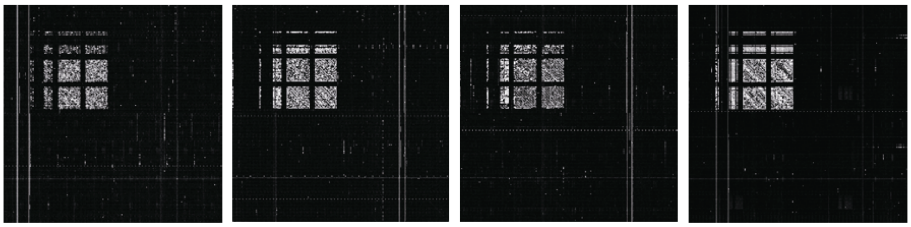
(a) Adialer.C 类可视化样本实例



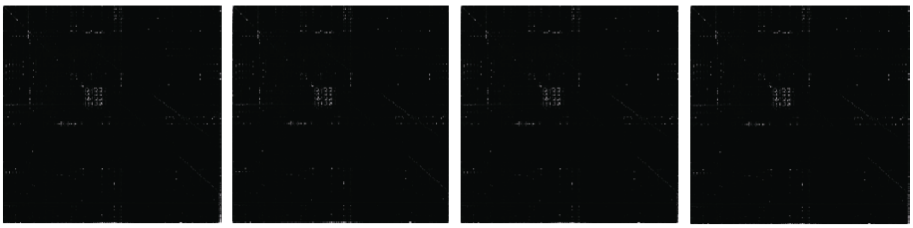
(b) Allaple.L 类可视化样本实例

图 5 Malimg 数据集可视化示意图

Figure 5 Malimg data set visualization diagram



(a) Variant 类可视化样本实例



(b) Worm 类可视化样本实例

图 6　PE 数据集可视化示意图

Figure 6　PE data set visualization diagram

而不同类恶意代码样本的纹理特征则呈现出相异性。因此, 我们认为这种可视化方法具有可行性, 并能够有效提取恶意代码样本的纹理特征。

4.2.2　消融实验

本小节展示了 4 个消融实验: 预处理结构的消融实验、图像增强比例的消融实验、网络结构的消融实验和 D-ResNet 模型细节的消融实验。

(1) 预处理结构消融实验

在本文的第三节中, 我们提出了一种预处理方法, 其中包括 3 个阶段: Nataraj 矢量化、灰度共生矩阵灰度图和图像增强。为了进一步探究本文设计的预处理结构对分类准确率的影响, 我们采用了 D-ResNet18 作为网络模型, 并将预处理的 3 个阶段作为网络模型的输入进行实验。

具体而言, 首先, 我们使用 Nataraj 矢量化将原数据集转化为灰度图像, 得到对照组 A, 在本次实验中, 对于 Malimg 数据集而言, 由于其本身为灰度图像数据集, 因此无需进行 Nataraj 矢量化, 直接将其作为对照组 A。其次, 我们从对照组 A 中提取灰度共生矩阵, 并转化为灰度共生矩阵灰度图, 生成对照组 B。第三, 我们对灰度共生矩阵灰度图进行像素值增强, 得到实验组 C。为评估预处理方法中这 3 个阶段对图像分类任务的影响, 我们使用 D-ResNet18 作为网络模型, 在对照组 A、对照组 B 和实验组 C 三个数据集上进行了 10 次训练, 每次训练进行 200 轮, 并将 10 次训练在验证集上的分类准确率的平均值作为实验结果。

表 4 显示了对照组 A、对照组 B、实验组 C 对分类准确率以及分类效率的影响。

表 4　预处理结构对分类准确率的影响

Table 4　The influence of preprocessing structure on classification accuracy

数据集	衡量标准	对照组 A	对照组 B	实验组 C
Malimg 数据集	分类准确率/(%)	99.39	94.77	99.68
	训练一轮所需时间/second	77	47	53
PE 数据集	分类准确率/(%)	72.29	70.72	79.03
	训练一轮所需时间/second	153	28	32

通过观察对照组 A 与实验组 C 的实验结果我们可以看出, 针对 Malimg 数据集, 直接对 Nataraj 矢量化得到的灰度图(对照组 A)进行训练, 分类准确率为 99.39%, 一轮训练时长为 77s。然而, 对灰度共生矩阵灰度图每个像素乘以 100 进行图像增强的数据集(实验组 C)进行训练, 其分类准确率可达到 99.68%, 相较于对照组 A 提高了 0.29%; 训练时间缩短为 53s, 相较于对照组 A 减少了 24s。而对于 PE 数据集, 未经任何处理的灰度图数据集(对照组 A)的分类准确率为 72.29%, 一轮训练时长为 153s, 而对灰度共生矩阵灰度图每个像素乘以 100 进行图像增强的数据集(实验组 C)的分类准确率可达到 79.03%, 相较于对照组 A 提高了 6.74%; 训练时间缩短为 32s, 相较于对照组 A 减少了 121s。因此, 无论是恶意代码家族分类任务还是恶意代码类型分类任务, 实验组 C 相较于对照组 A 都能够显著提高分类准确率, 并能够大大减少训练时长, 提升训练速度。因此, 本文提出的将 Nataraj 矢量化灰度图转化为灰度共生矩阵灰度图, 并进行合理像素值增强的方法可以有效地提升恶意代码分类准确率和效率。

通过观察对照组B与实验组C的实验结果我们可以看出, 针对Malimg数据集, 如果直接转化为灰度共生矩阵灰度图后不作处理(对照组 B)进行训练, 分类准确率为 94.77%, 相比对照组 A 反而下降了 4.62%; 而如果进行像素值图像增强后(实验组 C), 分类准确率可以达到99.68%, 相比对照组 B 提升了 4.91%。对于 PE 数据集, 转化为灰度共生矩阵灰度图后不作处理(对照组 B)的分类准确率为 70.72%, 相比对照组 A 下降了 1.57%, 而进行像素值图像增强后(实验组 C)的分类准确率可达到 79.03%, 相比对照组 B 提高了 8.31%。由此可见, 不论是恶意代码家族分类任务还是恶意代码类型分类任务, 使用灰度共生矩阵灰度图和像素值增强相结合的方法可以显著提高分类准确率, 而如果只是转化为灰度共生矩阵灰度图后不作处理并不能提供足够的有用信息以支持分类器的准确分类。这是因为直接转化的灰度共生矩阵灰度图过于黑暗, 不易区分。在应用像素值图像增强技术时, 将图像的明度值调整到合适的范围可以使得各个代码类别之间的特点更加明显, 从而能够提供更多有用的特征以支持分类器进行准确分类。

(2) 图像增强比例消融实验

由于灰度共生矩阵灰度图本身呈现较暗的特点, 需要进行图像增强, 以凸显不同恶意代码家族与恶意代码类型的特征。通过增强图像的明暗交错特征, 可以为分类模型提供更具辨识度的特征。

本文采用遍历的方式进行图像增强比例的选取。具体地, 在数据集转化为灰度共生矩阵灰度图后, 对每个像素乘以增强比例 a 进行图像增强。对于Malimg数据集和PE数据集, a 均选取 10^n, 其中 n 选取[0,8]之间的整数。随后对每一个 a 值所对应的增强灰度图像, 使用 D-ResNet18 进行训练, 记录对应分类准确率, 实验结果如表 5 所示。

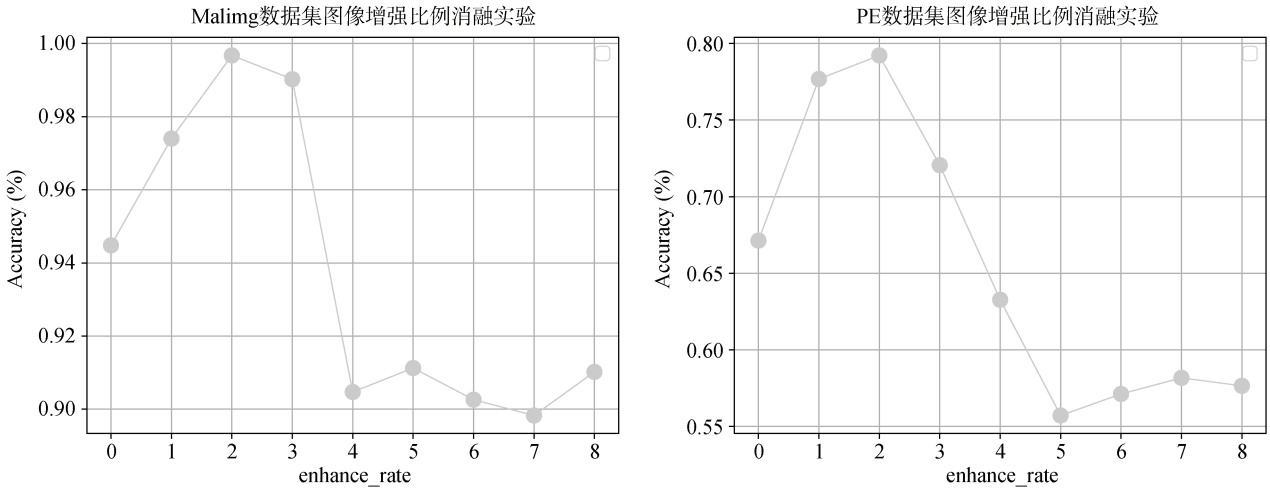
表 5 图像增强比例对分类准确率的影响

Table 5 Effect of image enhancement ratio on classification accuracy

增强比例	1	10	100	1000	10000	100000	1000000	10000000	100000000
Malimg 数据集分类准确率/(%)	94.49	97.40	99.68	99.03	90.47	91.13	90.26	89.83	91.02
PE 数据集分类准确率/(%)	67.14	77.68	79.22	72.06	63.27	55.71	57.12	58.17	57.65

为了更加直观地展示图像增强比例对分类准确率的影响, 我们针对 Malimg 数据集和 PE 数据集绘制了基于图像增强比例与分类准确率的示意图, 如图 7(a)和图 7(b)所示。这两张示意图能够清晰地显示出不同图像增强比例下, 分类准确率的变化情况。值得注意的是, 示意图中横坐标 enhance_rate 代表 10^n 中的 n 值。

如图 7(a)(b)所示, 无论是对于 Malimg 数据集还是 PE 数据集, 当 n 等于 2, 即 a 等于 100 时, 分类准确率均达到最高点, 这一结论具有一定的稳定性和普适性。因此, 我们建议在对 Malimg 数据集以及 PE 数据集进行图像增强时, 选择 100 作为增强比例, 以达到最佳的分类效果。



(a) Malimg数据集图像增强比例消融实验 (b) PE数据集图像增强比例消融实验

图 7 图像增强比例消融实验

Figure 7 Image enhancement proportional ablation experiment

本小节消融实验结合恶意代码家族分类任务和恶意代码类型分类任务, 探究了图像增强的比例对恶意代码分类准确率的影响。实验结果表明图像增强的比例对分类准确率的影响存在差异, 选择合适的比例可以大大增强恶意代码图像的分类准确率, 而选择不合适的比例则可能导致分类准确率下降。因此, 对灰度图像进行适当的像素值增强处理, 可以显著提升恶意代码家族分类和恶意代码类型分类的准确度。这一实验结果为相关领域的研究提供了一定的参考价值, 有助于优化恶意代码分类的实践应用。

(3) 网络结构消融实验

本小节的实验展示了不同网络结构对恶意代码家族分类任务和恶意代码类型分类任务的影响。经过实验分析发现, 本文所提出的 D-ResNet18 模型相比于其他模型在恶意代码分类准确率和计算效率上有一定的提升。

实验采用先前建立的实验组 C 进行训练和分类实验验证, 探讨不同网络模型对于恶意代码家族分类和恶意代码类型分类任务的影响。本研究旨在设计一个高精度且模型复杂度较低的网络模型, 以提高恶意代码分类任务的效率和准确性。为此, 我们对不同网络结构的性能进行了深入的比较和分析, 以提高分类准确性和泛化能力。

首先, 本文探究了多种经典的卷积神经网络结构在恶意代码分类任务中的性能表现, 包括 VGG 系列[35]、GoogleNet 系列[36]、AlexNet 系列[37]、ResNet 系列、DenseNet 系列、Res2Net 系列[38]、RegNet 系列[39]以及 EfficientNetV2 系列[40]等。这些测试旨在全面了解各种网络结构的性能和适用范围, 以便为恶意代码家族分类任务与恶意代码类型分类任务选择最佳的网络结构。

由于网络结构数量众多, 我们对以上网络结构进行了三次训练, 每次训练 200 轮, 并取三次训练在验证集上的分类准确率平均值作为初步筛选的实验结果。在初步筛选中, 我们发现 ResNet18、ResNet34、DenseNet121、VGG16 以及 Res2Net50 等网络结构的分类效果较好。因此, 我们选取这五个网络结构, 同时引入我们提出的 D-ResNet18 进行对比。除此之外, 我们还引入了一个简单的 CNN 网络作为对比, 该网络只包含两层卷积层和两层池化层, 我们将其命名为 Simple_CNN。在进一步实验中, 我们采用这七个网络结构进行了十次训练, 每次训练 200 轮, 并取十次训练在验证集上的分类准确率平均值作为最终实验结果, 以比较这些网络结构的性能。这样的实验设计和执行可以确保实验结果具有一定的可靠性和科学性。

实验过程中, 本文引入每秒浮点运算次数(Floating Point Operations Per Second, FLOPs)作为衡量网络结构复杂性的指标, 同时考虑参数量作为衡量网络结构大小的指标。FLOPs 指的是网络结构在进行前向传播时需要执行的浮点数运算的数量, 可以用来衡量网络结构的计算量大小。参数量指的是网络结构中需要学习的参数的数量, 包括权重和偏置项等。在深度神经网络中, 每个神经元都包括权重和偏置项, 这些参数会在训练过程中进行更新。因此, 网络结构的参数量通常与其计算量成正比。综上所述, FLOPs 和参数量是衡量深度神经网络计算量的重要指标, 它们之间存在一定的关系, 但具体的关系需要根据具体的网络结构和任务进行分析。

接下来, 我们将综合考虑分类准确率、训练时间、网络结构复杂度以及网络结构大小等多个因素, 对各个网络结构的性能进行全面评估, 实验结果如表 6 所示。通过对比分析各网络结构的表现, 我们可以更好地了解每个网络结构的优缺点, 并为选择最合适的网络结构提供有价值的指导。

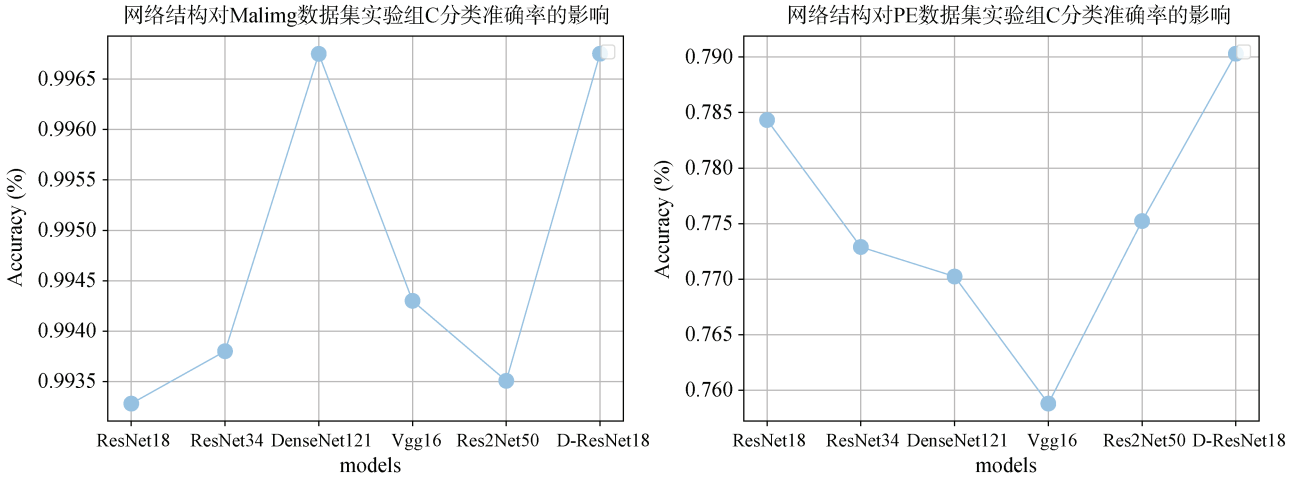
表 6 网络结构对分类准确率的影响

Table 6 The influence of network structure on classification accuracy

数据集	衡量标准	ResNet18	ResNet34	DenseNet121	Vgg16	Res2Net50	D-ResNet18	Simple_CNN
Malimg 数据集实验组 C	分类准确率/(%)	99.33	99.38	99.68	99.43	99.35	99.68	97.75
	训练一轮所需时间/second	52	65	90	104	106	53	42
	FLOPs/G	1.8186	3.6708	2.8647	15.5069	4.2035	2.0243	0.0816
	参数量大小/M	11.1893	21.2975	6.9795	134.3714	23.0620	12.9514	0.9083
PE 数据集实验组 C	分类准确率/(%)	78.43	77.29	77.02	75.88	77.52	79.03	75.70
	训练一轮所需时间/second	31	40	51	60	57	32	27
	FLOPs/G	1.8186	3.6708	2.8647	15.5068	4.2035	2.0243	0.0813
	参数量大小/M	11.1796	21.2878	6.9600	134.2936	23.0231	12.9416	0.6072

分析表 6, 我们可以发现 Simple_CNN 相对于其他深度学习模型在分类准确度上存在一定的差距。因此, 选择合适的深度学习模型对于优化分类准确度是非常必要的。通过比较不同模型的性能指标, 我们可以找到最适合特定任务的模型, 从而取得更好的分类结果。为了更加直观地展示除 Simple_CNN 外其余六种网络结构对分类准确率的影响, 我们针对 Malimg 数据集和 PE 数据集绘制了基于网络结构以及实验组 C 的分类准确率示意图, 如图 8(a)和图 8(b)所示。这两张示意图能够清晰地显示出在不同网络结构下, 分类准确率的变化情况。

分析表 6, 图 8(a)和图 8(b)可知, 对于恶意代码家族分类任务而言, 除 Simple_CNN 和我们提出的 D-ResNet18 外, 其余网络结构中最简单的 ResNet18 模型在训练时间最短的同时, 其分类准确率最低, 仅为 99.33%。相比之下, DenseNet121 模型在其余五种网络结构中表现最好, 其分类准确度可以达到 99.68%。然而, 该模型的网络模型复杂度较高, 且训练时间较长, 训练一轮需要 90s。此外, DenseNet121 模型在恶意代码类型分类任务中表现一般, 其分类准确度仅为 77.02%。而 Res2Net50 和 Vgg16 模型的训练时间更长, 其网络模型复杂度也更高, 但是分类准确率低于 DenseNet121; ResNet34 模型的训练时间相对较短, 但其分类准确率并不高。



(a) 网络结构对Malimg数据集实验组C分类准确率的影响　(b) 网络结构对PE数据集实验组C分类准确率的影响

图 8　网络结构对实验组 C 分类准确率的影响

Figure 8　The effect of network structure on classification accuracy of experimental Group C

就恶意代码类型分类任务而言, 除 Simple_CNN 和本文提出的 D-ResNet18 模型外, ResNet18 模型在其余五种网络结构中表现最好, 其分类准确率可达到最高的水平, 且模型复杂度和训练时间相对较低, 但其在恶意代码家族分类任务上分类准确度最低。

可见, 在除了本文所提出的 D-ResNet18 之外的其余六种网络结构中, 并没有一个既可以在恶意代码家族分类任务中又可以在恶意代码类型分类任务中均有较高准确率、训练时间较短且模型复杂度低的高效网络结构。

为了解决上述问题, 本文提出了一种名为 D-ResNet18 的模型, 其在恶意代码家族分类任务上的分类准确率与 DenseNet121 几乎一致, 可达到 99.68%; 同时该模型一轮训练所需时间仅为 53s, 比 DenseNet121 快了 37s, 计算量 FLOPs 也仅为 2.0243G, 比 DenseNet 少了 0.8404G, 极大地提高了训练效率。更为重要的是, D-ResNet18 在恶意代码类型分类任务中具有最高的分类准确率, 达到了 79.03%, 相较于目前最优秀的 ResNet, 提升了 0.60%。分析表 6 结果发现, 在不考虑 Simple_CNN 的情况下, 该模型在恶意代码家族分类和恶意代码类型分类任务中均拥有第二短的训练时间和第二小的模型复杂度, 仅次于最小的 ResNet18。这是因为 D-ResNet18 基于 ResNet18 结构进行改进, 添加了 DenseNet 的特性, 所以保留了 ResNet18 训练时长快、模型复杂度低等优点, 同时具备 DenseNet 的优势。

综上所述, 在不考虑 Simple_CNN 的情况下, D-ResNet18 模型在恶意代码家族分类和恶意代码类型分类两个任务中均表现出最高的准确率、第二短的训练时长以及第二小的模型复杂度, 具有精度高、训练简单、模型复杂度低等优点, 在恶意代码分类领域具有广泛的实际应用前景。

值得一提的是, 在恶意代码分类领域, 分类效果会因不同的分类任务而异。对于恶意代码家族分

类, 模型分类效果普遍较高, 最高可达 99.68%。这是由于不同恶意代码家族之间存在明显的差异, 深度学习模型能够有效地学习并利用这些差异来进行分类。然而, 对于恶意代码类型分类, 模型分类效果相对较一般, 最高可达 79.03%。这是因为 PE 数据集中虽然大多数恶意代码类型的特征已被模型很好地捕捉, 但仍有一些恶意代码类型由于样本高度相似而难以分类。

在恶意代码类型分类任务中, 为了更全面地展示各个类型的分类准确率, 本文绘制了基于网络结构与各个恶意代码类型的准确率示意图, 如图 9 所示。

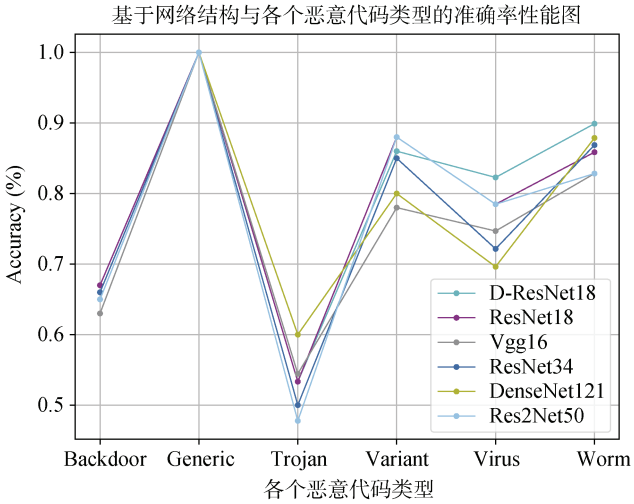


图 9 基于网络结构与各个恶意代码类型的准确率性能图

Figure 9 Accuracy performance graph based on network structure and each malicious code type

分析图 9 实验结果可以发现在恶意代码类型的分类任务中, 不同网络结构表现出了各自的优势和劣势。例如, 在 Virus 和 Worm 类型中, 本文的模型 D-ResNet18 表现最佳, 在 Backdoor 和 Variant 类型中, 模型 ResNet18 表现更为出色, 因此需要根据具体任务的需求来选择最为合适的模型来进行恶意代码分类。此外, 还需注意的是, 在 Generic 类型中, 所有模型准确率均达到了 100%。而在 Backdoor 和 Trojan 这两个类型中所有模型分类准确率均较低。这是因为这两个类型同属于木马这一大类的不同子类, 具有较高的同源性, 使得样本难以区分。因此, 在恶意代码类型分类任务中, 为了取得更好的分类效果, 还需要探索更多的研究领域, 如特征选择、模型优化等。

(4) D-ResNet 模型细节消融实验

为了更加深入地研究 D-ResNet 模型的细节, 我们设计了多组实验对网络细节进行设计和对比, 实验选取包含了残差连接与密集连接(下采样包含批量归一化层)的 D-ResNet18 作为实验组。首先, 为了探究模型深度对性能的影响, 本文选取了网络层数 34 层的 D-ResNet34 作为对照组 1。其次, 为了探究密集连接(下采样包含批量归一化层)对 ResNet 模型的影响, 本文选取了未添加密集连接的 ResNet18 作为对照组 2。第三, 为了探究残差连接在 D-ResNet 中的作用, 本文设计了未添加残差连接的 D-ResNet18 版本作为对照组 3。最后, 为了探究在密集连接下采样过程中添加批量归一化层的作用, 本文设计了未添加批量归一化层的 D-ResNet18 版本作为对照组 4。

实验过程中, 我们使用先前建立的实验组 C 对上述设计的模型进行训练和验证, 并将实验结果汇总在表 7 中。其中, D-ResNet-no-Residual 表示未添加残差连接的 D-ResNet18 版本, D-ResNet-no-BN 表示在密集连接下采样过程中未添加批量归一化层的 D-ResNet18 版本。

表 7 D-ResNet 模型细节对分类准确率的影响

Table 7 The effect of D-ResNet model details on classification accuracy

数据集	衡量标准	D-ResNet18	D-ResNet34	ResNet18	D-ResNet18 -no-Residual	D-ResNet18 -no-BN
Malimg 数据集	分类准确率/(%)	99.68	99.68	99.33	99.51	99.57
	训练一轮所需时间/second	53	68	52	53	53
PE 数据集	分类准确率/(%)	79.03	78.17	78.43	78.70	78.52
	训练一轮所需时间/second	32	40	31	32	32

经过对表 7 结果的分析, 可以发现在探究模型深度相关实验中, D-ResNet34 和 D-ResNet18 在恶意代码家族分类任务上均取得了 99.68%的高准确率。然而, 在恶意代码类型分类任务上, D-ResNet34 相比 D-ResNet18 的准确率低了 0.86%。此外, D-ResNet34 的训练时间相对于 D-ResNet18 更长, 在恶意代码家族和类型分类任务上, 一轮训练时间分别增加了 15 秒和 8 秒。因此, 本文认为网络层数较浅的 D-ResNet18 更适合用于恶意代码分类任务, 因为其具有更高的分类准确率和更快的训练速度, 尤其是在恶

意代码类型分类方面, 表现出更加优异的分类效果。

此外, 对于恶意代码家族分类和类型分类两个任务, 分析发现相较于包含残差连接和密集连接(下采样包含批量归一化层)的 D-ResNet18 模型, 未添加密集连接的 ResNet18 模型准确率分别下降了 0.35%和 0.60%; 未添加残差连接的 D-ResNet18 模型准确率分别下降了 0.17%和 0.33%; 下采样过程中未添加批量归一化层的 D-ResNet18 模型准确率分别下降了 0.11%和 0.51%。这些实验结果表明了残差连接和密集连接结合的重要性。残差连接通过引入跨层连接缓解了深度神经网络中的梯度消失问题, 提高了网络的训练效率和性能。而密集连接则通过将当前层的输出和前面所有层的输出连接在一起, 提高了特征的重用和信息流动的效率, 增强了网络的表达能力和性能。将残差连接和密集连接结合起来, 可以发挥它们各自的优势, 并弥补它们的局限性。另外, 在密集连接下采样过程中添加批量归一化层可以获得更优越的效果, 因为它可以使得网络的输出分布更加稳定, 避免了梯度消失和梯度爆炸等问题, 提高了模型的泛化性能。

综合上述实验结果, 可以得出结论: 本研究所设计的 D-ResNet18 模型在恶意代码分类任务中表现最佳。该模型不仅拥有较浅的网络层数, 同时采用了残差连接和密集连接的结合策略, 并在密集连接下采样过程中添加了批量归一化层。实验结果表明, 这些优化策略对于深度学习网络的性能具有显著的提升作用, 为深度学习网络的设计与优化提供了重要的参考和指导。

4.2.3 对比实验

为了验证本文所提出的恶意代码分类方法在效果提升方面的有效性, 我们将其与基线方法进行比较。

(1) 基线方法

首先, Kalash 等人[25]利用 Nataraj 矢量化方法将恶意代码的二进制文件转换为灰度图像, 并使用基于 VGG16 改进的卷积神经网络模型进行分类。其次, 王博等人[27]提出了一种创新的方法, 该方法将每个二进制 bit 串切割成长度为 8bit 的子串, 并将每连续三个子串分别对应 RGB 通道。通过基于 VGG16 改进的卷积神经网络模型来提取特征, 实现对恶意代码的分类。针对文献[27]中存在的模型数量过多的问题, 蒋考林等人[29]提出了一种与其类似的恶意代码可视化为彩色图像的方法。然而, 其独特之处在于在末尾不足的情况下使用 0 进行填充, 并使用 Alexnet 进行训练和分类。此外, 为了比较深度学习与机器学习的性能, 我们提取灰度共生矩阵, 并将其角二阶矩、对比度、熵以及反差分矩阵作为机器学习特征, 随后应用 KNN 分类器进行分类。

上述四种方法都被视为本文的基线方法。

(2) 实验结果与分析

通过对比实验, 对本文所提出的恶意代码分类方法的有效性进行了验证, 实验结果如表 8 所示, 其中'-'表示没有此项内容。

表 8 相关研究工作比较

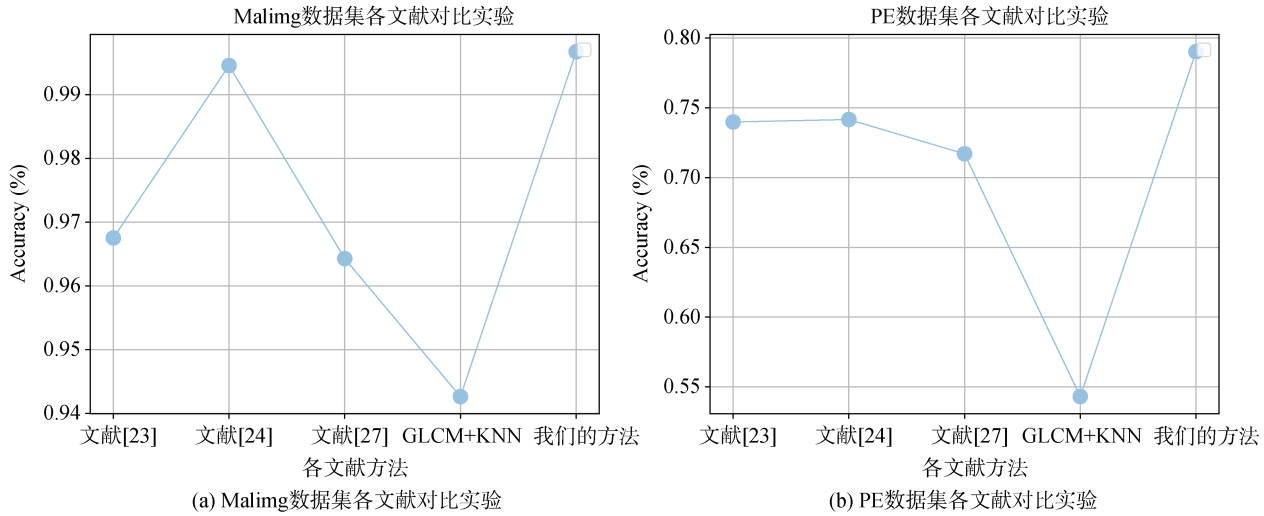
Table 8 Comparison of relevant research work

数据集	衡量标准	文献[25]	文献[27]	文献[29]	GLCM+KNN	我们的方法
Malimg 数据集	分类准确率/(%)	96.75	99.46	96.43	94.26	99.68
	训练一轮所需时间/second	129	112	68	-	53
	FLOPs/G	15.5069	15.5069	4.7902	-	2.0243
	模型参数量大小/M	134.3714	134.3714	169.0355	-	12.951385
PE 数据集	分类准确率/(%)	73.99	74.17	71.70	54.31	79.03
	训练一轮所需时间/second	171	125	108	-	32
	FLOPs/G	15.5068	15.5068	4.7902	-	2.0243
	模型参数量大小/M	134.2936	134.2936	168.9966	-	12.9416

为了更加直观地展示对比实验的结果, 本文针对 Malimg 数据集和 PE 数据集绘制了基于不同文献方法与分类准确率的示意图, 如图 10(a)和图 10(b)所示。这两张示意图能够清晰地显示出各文献方法对应分类准确率的变化情况。

对表 8 以及图 10(a)和图 10(b)的实验结果进行分析, 可以发现本文所提出的方法在恶意代码家族分类任务和恶意代码类型分类任务上的表现均优于其他对比方法。

与文献[25], 文献[27], 以及文献[29]三种深度学习方法相比, 本文方法在恶意代码分类方面表现优异, 不仅准确率最高, 而且训练模型所需时间最



(a) Malimg数据集各文献对比实验　(b) PE数据集各文献对比实验

图 10　对比实验

Figure 10　Contrast experiment

短, 具有最高的分类效率, 这使得其具有广泛的应用价值和推广意义。相较于对比方法中使用的 VGGNet 和 AlexNet 网络, 本文所采用的 D-ResNet18 神经网络模型结构简单且高效, 有效缩短了检测时间, 提高了检测效率。此外, 本文设计的预处理方法能够大幅减小图片尺寸, 便于分类模型学习特征并进行分类, 显著提升了模型训练速度, 进一步增强了恶意代码分类的效率。

相较于 GLCM 结合 KNN 的机器学习方法, 本文的方法在恶意代码家族和类型分类任务中均实现了显著的分类准确率提升, 在恶意代码家族分类方面提升达到了 5.42%, 而在恶意代码类型分类方面提升甚至高达 24.72%。尽管机器学习方法具有简单易懂、容易实现的优点, 但因为需要人工选择特征和分类器, 可能无法发现数据的潜在特征, 而且会消耗大量的时间。而深度学习方法无需手动进行特征提取和选择, 而是通过模型自动提取特征, 更加便捷, 从而可以更好地适应不同的数据分布和任务。因此, 本文提出的方法能够更加准确地学习到恶意代码图像中的特征, 具有更好的泛化能力和分类准确率。

综上所述, 在恶意代码家族分类和恶意代码类型分类两个任务的研究上, 本文所提出的方法在分类准确率和训练速度方面均显著优于机器学习方法以及对比文献中的深度学习方法, 具有较高的实用价值。

5　总结与展望

本文提出了一种基于增强灰度共生矩阵的深度恶意代码可视化分类方法。该方法通过 Nataraj 矢量化方法将恶意代码转化为灰度图像, 接着转化为灰度共生矩阵灰度图, 并通过像素值乘积进行图像增强。最后, 利用 D-ResNet18 模型进行训练, 实现恶意代码分类。

实验结果表明, 本文工作成功将恶意代码的分类问题转化为图片的分类问题, 具有较好的鲁棒性, 对于大规模恶意代码的分类任务具有实际意义, 在恶意代码家族分类任务和恶意代码类型分类任务上均表现优越, 具有较高分类准确率、较短训练时间、较低模型复杂度, 优于对比文献方法。此外, 该方法将常用于机器学习的灰度共生矩阵与深度学习相结合, 避免了手动特征提取的工作量和难度。

本文工作仍存在一些不足之处, 同时也提出了改进的方向, 具体如下:

第一, 本文研究了将恶意代码可视化为灰度图像的方法, 未来的研究可以考虑将其可视化为彩色图像。彩色图像具有三个通道, 可以反映恶意代码更丰富的特性信息, 有助于提高检测和分类效果。例如, 可以将 RGB 三通道分别代表三种不同的信息, 比如 R 通道代表灰度图像、G 通道代表字符信息, B 通道代表文件头信息。

第二, 在恶意代码类型分类任务中, Backdoor 和 Trojan 两类恶意代码属于同一木马大类, 难以进行准确分类。因此, 需要进一步研究如何提高恶意代码类型分类的准确性。

第三, 目前将可视化方法与深度学习结合用于恶意代码检测主要以卷积神经网络和循环神经网络为主, 而使用对抗生成网络(Generative Adversarial Networks, GAN)和图卷积神经网络(Graph Convolution Network, GCN)进行恶意代码检测的方法较少。因此, 未来可以考虑将可视化方法、GCN 和 GAN 相结合, 用于恶意代码分类和检测。

参考文献

[1] National Internet Emergency Center. Network Security Information and Dynamic Weekly Report [EB/OL]. https://www.cert.org.cn/publish/main/44/2020/20200423151618969661418/20200423151618969661418_.html.2020.

[2] Bhodia N, Prajapati P, Di Troia F, et al. Transfer Learning for Image-Based Malware Classification[C]. *The 5th International Conference on Information Systems Security and Privacy*, 2019: 719-726.

[3] Iwamoto K, Wasaki K. Malware Classification Based on Extracted API Sequences Using Static Analysis[C]. *The Asian Internet Engineeering Conference on - AINTEC '12*, 2012: 31-38.

[4] Imran M, Afzal M T, Qadir M A. Similarity-Based Malware Classification Using Hidden Markov Model[C]. *2015 Fourth International Conference on Cyber Security*, *Cyber Warfare*, *and Digital Forensic*, 2015: 129-134.

[5] Hardy W, Chen L, Hou S, et al. DL4MD: A Deep Learning Framework for Intelligent Malware Detection[C]. *The International Conference on Data Science, The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing*, 2016: 61.

[6] Schultz M G, Eskin E, Zadok F, et al. Data Mining Methods for Detection of New Malicious Executables[C]. *Proceedings 2001 IEEE Symposium on Security and Privacy*. *S&P 2001*, 2001: 38-49.

[7] Kolter J Z, Maloof M A. Learning to Detect Malicious Executables in the Wild[C]. *The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004: 470-478.

[8] Kolter J Z, Maloof M A. Learning to Detect and Classify Malicious Executables in the Wild[J]. *Journal of Machine Learning Research*, 2006, 6: 2721-2744.

[9] Kang B, Yerima S Y, McLaughlin K, et al. N-Opcode Analysis for Android Malware Classification and Categorization[C]. *2016 International Conference on Cyber Security and Protection of Digital Services*, 2016: 1-7.

[10] Kong D G, Yan G H, Kong D G, et al. Discriminant Malware Distance Learning on Structural Information for Automated Malware Classification[C]. *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013: 1357-1365.

[11] Li B, Roundy K, Gates C, et al. Large-Scale Identification of Malicious Singleton Files[C]. *The Seventh ACM on Conference on Data and Application Security and Privacy*, 2017: 227-238.

[12] Kumar A, Kuppusamy K S, Aghila G. A Learning Model to Detect Maliciousness of Portable Executable Using Integrated Feature Set[J]. *Journal of King Saud University - Computer and Information Sciences*, 2019, 31(2): 252-265.

[13] Firdausi I, Lim C, Erwin A, et al. Analysis of Machine Learning Techniques Used in Behavior-Based Malware Detection[C]. *2010 Second International Conference on Advances in Computing*, *Control*, *and Telecommunication Technologies*, 2010: 201-203.

[14] Zolkipli M F, Jantan A. An Approach for Malware Behavior Identification and Classification[C]. *2011 3rd International Conference on Computer Research and Development*, 2011: 191-194.

[15] Haralick R M, Shanmugam K, Dinstein I. Textural Features for Image Classification[J]. *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, 1973, SMC-3(6): 610-621.

[16] Nataraj L, Karthikeyan S, Jacob G, et al. Malware Images: Visualization and Automatic Classification[C]. *The 8th International Symposium on Visualization for Cyber Security*, 2011: 1-7.

[17] Torralba, Murphy, Freeman, et al. Context-Based Vision System for Place and Object Recognition[C]. *Ninth IEEE International Conference on Computer Vision*, 2003: 273-280.

[18] Oliva A, Torralba A. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope[J]. *International Journal of Computer Vision*, 2001, 42(3): 145-175.

[19] Siagian C, Itti L. Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007, 29(2): 300-312.

[20] Nataraj L, Yegneswaran V, Porras P, et al. A Comparative Assessment of Malware Classification Using Binary Texture Analysis and Dynamic Analysis[C]. *The 4th ACM Workshop on Security and Artificial Intelligence*, 2011: 21-30.

[21] Naeem H, Guo B, Naeem M R, et al. Identification of Malicious Code Variants Based on Image Visualization[J]. *Computers & Electrical Engineering*, 2019, 76: 225-237.

[22] Liu L, Wang B S, Yu B, et al. Automatic Malware Classification and New Malware Detection Using Machine Learning[J]. *Frontiers of Information Technology & Electronic Engineering*, 2017, 18(9): 1336-1347.

[23] Fu J W, Xue J F, Wang Y, et al. Malware Visualization for Fine-Grained Classification[J]. *IEEE Access*, 2018, 6: 14510-14523.

[24] Li S J, Wang C, Shi Y. Malicious Code Detection Based on multi-Feature Random Forest[J]. *Computer Applications and Software*, 2020, 37(10): 328-333.
(李劭杰, 王晨, 史峄. 基于多特征随机森林的恶意代码检测[J]. *计算机应用与软件*, 2020, 37(10): 328-333.)

[25] Kalash M, Rochan M, Mohammed N, et al. Malware Classification with Deep Convolutional Neural Networks[C]. *2018 9th IFIP International Conference on New Technologies*, *Mobility and Security*, 2018: 1-5.

[26] Ronen R, Radu M, Feuerstein C, et al. Microsoft Malware Classification Challenge[EB/OL]. 2018: 1802.10135. https://arxiv.org/abs/1802.10135v1.

[27] Wang B, Cai H H, Su Y. Classification of Malicious Code Variants Based on VGGNet[J]. *Journal of Computer Applications*, 2020, 40(1): 162-167.
(王博, 蔡弘昊, 苏旸. 基于 VGGNet 的恶意代码变种分类[J]. *计算机应用*, 2020, 40(1): 162-167.)

[28] Vasan D, Alazab M, Wassan S, et al. IMCFN: Image-Based Malware Classification Using Fine-Tuned Convolutional Neural Network Architecture[J]. *Computer Networks*, 2020, 171: 107138.

[29] Jiang K L, Bai W, Zhang L, et al. Malicious Code Detection Based on Multi-Channel Image Deep Learning[J]. *Journal of Computer Applications*, 2021, 41(4): 1142-1147.

(蒋考林, 白玮, 张磊, 等. 基于多通道图像深度学习的恶意代码检测[J]. *计算机应用*, 2021, 41(4): 1142-1147.)

[30] Ren Z J, Bai T. Malware Visualization Based on Deep Learning[C]. *2021 14th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics*, 2021: 1-5.

[31] Wang R Z, Gao J, Tong X, et al. Research on Malicious Code Family Classification Combining Attention Mechanism[J]. *Journal of Frontiers of Computer Science and Technology*, 2021, 15(5): 881-892.
(王润正, 高见, 仝鑫, 等. 融合注意力机制的恶意代码家族分类研究[J]. *计算机科学与探索*, 2021, 15(5): 881-892.)

[32] He K M, Zhang X Y, Ren S Q, et al. Deep Residual Learning for Image Recognition[C]. *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2016: 770-778.

[33] Huang G, Liu Z, Van Der Maaten L, et al. Densely Connected Convolutional Networks[C]. *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017: 2261-2269.

[34] Ioffe S, Szegedy C, Ioffe S, et al. Batch Normalization[C]. *The 32nd International Conference on International Conference on Machine Learning - Volume 37*, 2015: 448-456.

[35] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[EB/OL]. 2014: 1409.1556. https://arxiv.org/abs/1409.1556v6.

[36] Szegedy C, Liu W, Jia Y Q, et al. Going Deeper with Convolutions[C]. *2015 IEEE Conference on Computer Vision and Pattern Recognition*, 2015: 1-9.

[37] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks[J]. *Communications of the ACM*, 2017, 60(6): 84-90.

[38] Gao S H, Cheng M M, Zhao K, et al. Res2Net: A New Multi-Scale Backbone Architecture[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021, 43(2): 652-662.

[39] Radosavovic I, Kosaraju R P, Girshick R, et al. Designing Network Design Spaces[C]. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020: 10428-10436.

[40] Tan M, Le Q. Efficientnetv2: Smaller models and faster training[C]. *International conference on machine learning. PMLR*, 2021: 10096-10106.

王金伟 于 2007 年在南京理工大学自动化学院获得博士学位。现任南京信息工程大学教授, 博士生导师。研究领域为多媒体版权保护、多媒体取证、多媒体加密和数据认证。E-mail: wjwei_2004@163.com

陈正嘉 于 2022 年在南京信息工程大学软件工程专业获得学士学位。现在南京信息工程大学电子信息专业攻读硕士学位。研究领域为网络安全、信息安全、恶意代码检测等。E-mail: 973522457@qq.com

谢雪 于2016中国科学院大学获得硕士学位, 现在中国科学技术大学网络空间安全专业攻读博士学位, 研究领域为网络空间安全、数字媒体取证。E-mail: xuexie2008@163.com

罗向阳 中国人民解放军战略支援部队信息工程大学教授、博士生导师。研究领域为图像隐写和隐写分析技术。E-mail: luoxy_ieu@sina.com

马宾 齐鲁工业大学网络空间安全学院教授、博士生导师。研究方向为可逆信息隐藏、多媒体取证、隐写与隐写分析。E-mail: sddxmb@126.com