基于一维卷积神经网络的多类恶意软件分类集成方法

袁海根[1] 李红丽[1]
YUAN Haigen LI Hongli

摘 要 随着互联网和计算机程序在现代社会中的重要性日益凸显，恶意软件开发活动呈现出数量激增和种类多样化的趋势。相较于传统机器学习，深度学习技术因其在处理海量病毒变种行为分析方面的优势而备受研究者青睐。文章针对应用程序编程接口（API）序列中的 API 调用关系进行分类研究，采用一维卷积神经网络（1D-CNN）模型解决多类分类问题。首先，使用 Word2Vec 词嵌入方法和 Skip-gram 模型为独特 API 创建特征向量；然后，通过一对多方法训练 1D-CNN 模型进行恶意软件分类；最后，将所有模型与提出的 ModifiedSoftVoting 算法相结合以提高分类效果。在 Mal-API-2019 公开基准数据集上的实验结果表明，该集成 1D-CNN 架构在所有恶意软件类别上都取得了显著提升，分类准确率达到 90%，加权平均 F_1 分数为 0.90，AUC 分数超过 0.96。

关键词 恶意软件分类；动态分析；API 序列；1D-CNN；Skip-gram；集成学习

doi：10.3969/j.issn.1672-9528.2026.04.020

0 引言

数字化时代，互联网的普及使电子商务、电子银行和电子医疗等在线服务成为日常生活的重要组成部分，但庞大的用户群体吸引了恶意程序员通过各种手段对用户进行情感和经济剥削，其中恶意软件作为网络攻击的主要工具，其数量和种类持续激增——根据卡巴斯基 2021 年安全报告显示，仅 2020 年 11 月—2021 年 10 月，就检测到 64 559 357 种恶意程序，涵盖后门、木马等 20 个类别，而 SonicWall 的数据更表明已知恶意软件实例已达 55 亿，且年增长率为 2%，这种态势给安全防护带来了严峻挑战 [1]。

目前主流的检测方法包括基于特征的检测（通过特征库比对识别已知恶意软件，但存在漏检风险）和基于行为的检测（通过分析程序行为识别未知威胁 [2]，但存在性能开销），为此研究人员正积极探索机器学习（如随机森林、支持向量机等）和深度学习（如 CNN、RNN 等）技术来提升检测效果，其中深度学习虽在图像和自然语言处理领域表现优异，但其训练成本较高的问题仍需解决 [3]；论文研究创新性地采用 API 调用序列作为特征，构建基于 1D-CNN 的集成分类模型，通过 Word2Vec 方法生成特征向量，并结合 ModifiedSoftVoting 算法来处理数据不平衡问题，在 Mal-API-2019 数据集上实现了 90% 的准确率和 F_1 值，AUC 更达到 0.96 以上。

1. 滇西科技师范学院 云南临沧 677000

1 相关工作

在恶意软件分析领域，随着研究者们不断探索创新方法，基于深度学习的分类技术已取得显著进展。Vinod 等 [4] 最早通过对 4 种变形恶意软件家族的动态分析，利用 API 序列生成签名并应用卡方检验，实现了 75%~80% 的分类准确率；随后研究者采用多种机器学习算法进行改进，如 Gourmand 特征选择结合 WEKA 工具使 J-48 分类器准确率达到 99.1%[5]。

近年来，深度学习技术展现出明显优势：Zhang 等 [6] 通过集成学习处理不平衡数据集；Catak 等 [7] 采用 CNN-RNN 混合架构实现 85.6% 的精度；Han 等 [8] 使用融合 API 序列使 XGBoost 准确率提升至 93.33%。特别值得关注的是 Catak 团队的工作，其发布的 Mal-API-2019 数据集包含 8 类 7 107 个 API 序列，为后续研究奠定基础。基于此数据集，Li 等 [9] 采用 LSTM/GRU 模型取得 0.59 召回率，而 Demirkiran 等 [10] 提出的 RTF 模型则将 AUC 提升至 0.881 1。在可视化分析方向，Vasan 等 [11] 集成 VGG16 和 ResNet-50 模型，Tekerek 开发 CycleGAN 转换方法，Catak 团队通过图像增强使 CNN 准确率达到 96%。论文研究创新性地结合 Word-2Vec 词嵌入与 1D-CNN 集成架构，采用 ModifiedSoftVoting 算法在 Mal-API-2019 数据集上实现准确率 90% 和 AUC 值 0.96，显著优于传统 LSTM 等时序模型。由此表明，融合语义分析与深度学习的多模态方法正在成为恶意软件分类研究的新范式。

2 新的集成一维卷积神经网络（1D-CNN）架构

在本文中提出的集成架构如图 1 所示，通过整合 8 个独立训练的一维卷积神经网络（1D-CNN）模型来解决恶意软件多类分类问题。该架构采用三阶段处理流程：首先基于 Mal-API-2019 数据集，运用 Word2Vec 的 Skip-gram 模型对 API 序列进行向量化处理，通过分析 API 间的语义关联为不同 API 分配权重向量；其次训练 8 个 1D-CNN 模型作为一对多分类器，分别学习特定类别与其他所有类别的区分特征；最后创新性地提出 ModifiedSoftVoting 算法，通过集成所有分类器的决策能力来有效解决多类分类问题。整个系统在保持各模型独立性的同时，通过算法层面的优化实现了分类性能的协同提升。

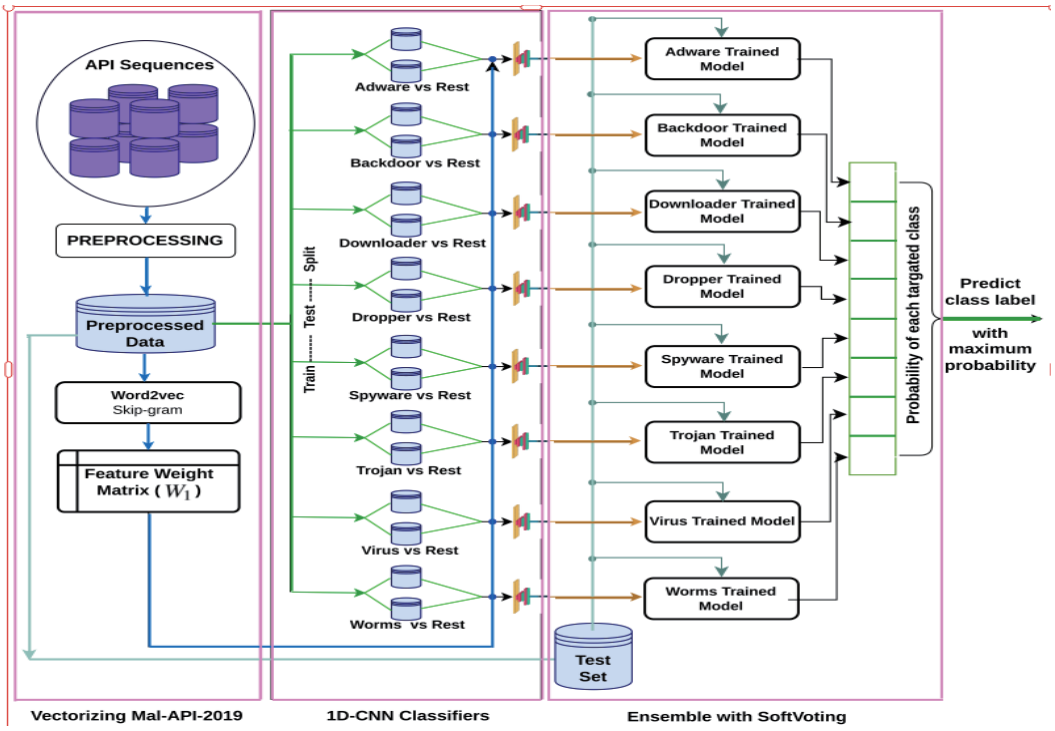


图 1 新的集成卷积神经网络架构

2.1 第一阶段：Mal-API-2019 进行向量化

Mal-API-2019 数据集作为恶意软件分类研究的重要基准，包含 8 个类别的 7 107 个恶意软件 API 调用序列。一类是数据集呈现显著的不平衡分布特征，不同类别样本量差异悬殊；另一类是原始 API 序列长度分布极不均匀，这种数据特性对分类模型性能构成挑战。数据集包含两个核心属性：API 序列（由重复 API 调用组成的类语句结构）和类别标签（0~7 的整型值对应 8 种恶意软件家族）。为构建有效的特征表示，论文采用词嵌入技术处理 API 序列：首先通过去重操作消除冗余 API 调用，随后应用 Word2Vec 的 Skip-gram 模型生成特征向量。Skip-gram 算法因其在小数据集上的优异表现和对低频词的良好表征能力，可有效捕捉 API 调用的语义特征，通过嵌入矩阵提取 API 的深层语义关联，而非直接使用权重矩阵进行预测任务。

2.2 第二阶段：训练一维卷积神经网络（1D-CNN）模型

深度学习模型凭借其卓越的性能表现正获得越来越广泛的应用，其中一维卷积神经网络（1D-CNN）作为典型架构，主要由卷积层、池化层和全连接层（Dense Layer）3 个核心组件构成：卷积层通过多种卷积核在文本序列等空间输入数据上执行卷积运算，生成具有判别性的一维特征图；池化层则对这些特征进行降维和关键信息提取；最终全连接层利用学习到的特征实现精准的二分类或多分类任务，这种端到端的特征学习机制使其在序列数据处理领域展现出独特优势。

在 1D-CNN 架构中，卷积层的数学表达如式（1）所示：通过 k 个卷积核（j 表示核的数量）对 M 个输入通道执行卷积运算（* 为卷积算子），每个核配有对应的偏置 b，并经过激活函数 $f()$ 进行非线性变换；随后池化层采用平均池化或最大池化方法，利用预设窗口对卷积特征进行降维处理，其输出作为第 l+1 层的特征表示；最终全连接层如式（2）所示，通过权重 w 和偏置 b 的线性组合，将学习到的特征映射为分类结果，这种层级结构实现了从局部特征提取到全局决策的端到端学习。

$$x_j^l = f\left(\sum_{i=1}^{N} x_i^{l-1} * w_{ij}^l + b_j^l\right) \quad (1)$$

$$h(x) = f(w^{l+1}x^{l+1} + b^{l+1}) \quad (2)$$

本文采用 1D-CNN 模型架构，其训练流程严格遵循算法 1 的规范：首先基于 Mal_API_c（预处理后的中间数据集）对图 1 中标注为 Model 的各个 1D-CNN 分类器进行训练；通过算法 1 构建 OvR_Mal_API_Datasets 集合，该集合中的每个中间数据集均采用“一对多”（OvR）标注策略——将目标类别样本标记为 1（正例），其余类别样本标记为 0（负例）；继而通过数据集训练专属的 1D-CNN 分类器，最终生成包含所有分类器的 1D_CNN_Classifier_List；在架构的最终阶段，

通过集成学习策略有效融合这些分类器的判别能力，从而构建完整的恶意软件多分类系统。

算法 1：算法主要用于基于 API 序列的恶意软件分类。

（1）初始化：首先初始化嵌入矩阵（EmbeddingMatrix）和空的 OvR（一对多）恶意 API 数据集（OvR_MalAPI_Datasets）。

（2）数据收集：遍历每个恶意软件类别（Mclass），针对每个类别收集对应的 API 序列。如果记录中的类别与当前遍历的类别相同，则标记为 1；否则，标记为 0。

（3）数据分割与模型训练：对每个收集到的恶意 API 数据集，使用训练测试分割方法（train_test_split）将其分为训练集和测试集。然后，使用 Conv1D 模型（一维卷积模型）和嵌入矩阵进行训练，得到分类器。

（4）返回结果：将所有训练好的分类器整合到一个列表中，并返回该列表。

2.3 第三阶段：集成分类器的决策

算法 2：ModifiedSoftVoting 算法

（1）初始化混淆矩阵：首先，根据类别数量 n，初始化一个 $n\times n$ 的混淆矩阵 $\boldsymbol{C}_m$，所有元素设为 0。

（2）数据集划分：将处理后的数据集 ProcessedDataset 划分为测试集 $\boldsymbol{X}_{\text{test}}$ 和标签集 $\boldsymbol{Y}_{\text{test}}$。

（3）分类器预测：对于分类器列表 C_{list} 中的每个分类器 c：

① 获取其在列表中的索引 C_{index}。

② 使用分类器 c 对测试集 $\boldsymbol{X}_{\text{test}}$ 进行预测，得到预测得分 $\boldsymbol{C}_{\text{score}}$。

③ 将 $\boldsymbol{C}_{\text{score}}$ 的第一列赋值给概率矩阵 $\boldsymbol{C}_{\text{proba}}$ 的第 C_{index} 列。

（4）更新混淆矩阵：对于概率矩阵 $\boldsymbol{C}_{\text{proba}}$ 的每一行：

① 获取行索引 T_{index}。

② 找到该行最大值所在的列索引 C_{plabel}，作为预测标签。

③ 从 $\boldsymbol{Y}_{\text{test}}$ 中获取对应行索引的实际标签 C_{tlabel}。

④ 在混淆矩阵 $\boldsymbol{C}_m$ 的 $[C_{\text{tlabel}}, C_{\text{plabel}}]$ 位置上加 1。

（5）返回结果：最终返回更新后的混淆矩阵。

本文采用算法 2 提出的 ModifiedSoftVoting 集成策略，通过以下流程实现多分类器协同决策：首先基于分层抽样从预处理数据集中构建测试集 $\boldsymbol{X}_{\text{test}}$（含 m 条记录）；随后 1D_CNN_Classifier_List 中的 8 个独立分类器（model）对每条测试记录生成二元概率预测，形成（$m\times2$）维的 Classification-Score 矩阵 $\boldsymbol{C}_{\text{score}}$（其中 $C_{\text{score}}[i][0]$ 和 $C_{\text{score}}[i][1]$ 分别表示第 i 条记录的负例 / 正例概率）；同时构建（$m\times8$）维的 ClassProb-ability 矩阵 $\boldsymbol{C}_{\text{proba}}$，其列向量存储单个分类器对所有记录的正例预测（如 $\boldsymbol{C}_{\text{proba}}[:][j]$ 对应第 j 个分类器的输出），而行向量 $\boldsymbol{C}_{\text{proba}}[i][:]$ 则聚合了所有分类器对第 i 条记录的正例概率；最终算法综合这些概率矩阵，通过改进的软投票机制确定每条测试记录的最优类别标签。

在算法评估阶段，算法 2 构建（$n\times n$）维的混淆矩阵实现多分类性能量化。该矩阵为式（3）~（6）中的性能指标计算提供基础数据，具体通过四类基本参数实现：真正例（TP）指目标类别被正确识别的样本；假阴性（FN）指目标类别被误判为其他类别的样本；假阳性（FP）指其他类别被误判为目标类别的样本；真反例（TN）指其他类别被正确区分的样本，虽然模型准确率可通过式（3）计算的正确预测占比获得，但在多分类场景中该指标可能存在偏差，因此需要补充计算各类别的平均精确度、召回率和 F_1 分数以全面评估性能。

$$\text{Accuracy}=\frac{\sum_{i=1}^{n}\boldsymbol{C}[i,i]}{\sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{C}_m[i,j]} \tag{3}$$

$$\text{Recall}=\frac{\boldsymbol{C}_m[c,c]}{\sum_{i=1}^{n}\boldsymbol{c}_m[ic,i]} \tag{4}$$

$$\text{Precision}=\frac{\boldsymbol{C}_m[c,c]}{\sum_{i=1}^{n}\boldsymbol{C}_m[i,c]} \tag{5}$$

$$F_1=\frac{2}{\frac{1}{\text{Recall}}+\frac{1}{\text{Precision}}} \tag{6}$$

在机器学习模型评估中，针对特定类别 c 的性能分析需要综合考察精确率（Precision）、召回率（Recall）和 F_1 分数三项核心指标。精确率（真阳性数 / 预测阳性数）重点反映假阳性（FP）的影响，适用于 FP 比 FN 更关键的应用场景；召回率（真阳性数 / 实际阳性总数）则侧重衡量假阴性（FN）的代价，适用于漏检后果严重的场景。二者通过调和平均得到的 F_1 分数，能有效平衡精确率与召回率的矛盾需求。对于多分类问题，论文采用宏观平均（未加权平均）和加权平均（以类别样本量为权重）两种聚合方式：宏观平均平等对待所有类别，而加权平均则能反映数据分布不平衡时的真实性能表现。这种分层评估体系既保障了对关键错误类型的针对性监控，又确保了整体性能评估的全面性。

3 实验与结果

实验是在配备 32 GB 内存的 Intel Xeon E5-2620 v4 处理器、NVIDIA GTX 1080 Ti 显卡、Ubuntu 18.04 系统和 Python 3.8 环境下，构建了基于词嵌入的集成 1D-CNN 恶意软件检测架构。第一阶段采用 Skip-gram 模型生成 API 词嵌入矩阵 $\boldsymbol{W}_1[500\times281]$，通过参数调优确定最优窗口大小和向量维度为 500。针对八类恶意软件（广告软件 / 后门程序 / 下载器 / 投递器 / 间谍软件 / 木马 / 病毒 / 蠕虫）的分类任务，第二阶段采用一对多策略训练多个 1D-CNN 模型，经实验验证最优配

置包含：Adam 优化器、特定参数的卷积 / 池化 / 全连接层，并使用 OvR_MalAPI_Datasets 数据集按 80:20 比例进行分层训练验证。如表 1 所示，该架构对各类恶意软件均展现出良好性能指标。

表 1 新架构在恶意软件性能指标

数据集	模型类别	1D CNN 分类器列表	精确率	召回率	F_1 分数	准确率
MAL_API_0	model_0	其余（0）	0.99	0.99	0.99	0.98
		广告软件（1）	0.88	0.83	0.85	
MAL_API_1	model_1	其余（0）	0.91	0.95	0.93	0.88
		后门程序（1）	0.61	0.46	0.52	
MAL_API_2	model_2	其余（0）	0.95	0.97	0.96	0.93
		下载器（1）	0.78	0.66	0.72	
MAL_API_3	model_3	其余（0）	0.95	0.95	0.95	0.91
		投放器（1）	0.65	0.63	0.64	
MAL_API_4	model_4	其余（0）	0.93	0.95	0.94	0.89
		间谍软件（1）	0.52	0.43	0.47	
MAL_API_5	model_5	其余（0）	0.90	0.93	0.91	0.85
		木马（1）	0.46	0.36	0.40	
MAL_API_6	model_6	其余（0）	0.95	0.96	0.95	0.92
		病毒（1）	0.73	0.70	0.72	
MAL_API_7	model_7	其余（0）	0.93	0.94	0.93	0.89
		蠕虫（1）	0.60	0.56	0.58	

在架构的第三阶段，基于第一阶段构建的 ProcessedDataset 数据集，采用 20% 分层抽样生成多类测试集，并运用 ModifiedSoftVoting 算法整合 8 个训练完成的 1D-CNN 分类器预测结果。通过对比实验验证集成模型的优越性，集成模型在分类报告中各项指标显著提升（如表 2 所示）。

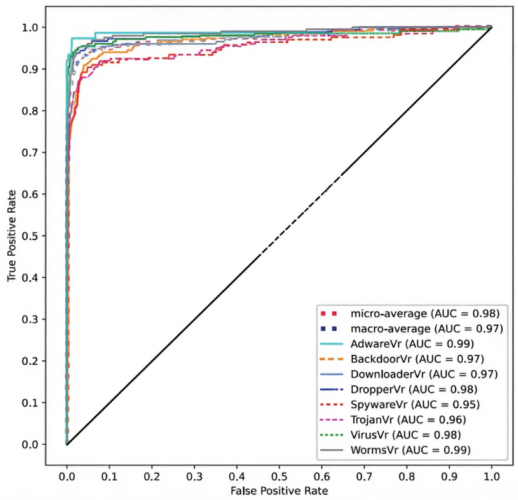
表 2 分类报告的比较

新的集成模型架构	精确率	召回率	F_1 分数
Adware（广告软件）	0.95	0.95	0.95
Backdoor（后门程序）	0.87	0.86	0.87
Downloader（下载器）	0.99	0.92	0.95
Dropper（投递器）	0.97	0.89	0.93
Spyware（间谍软件）	0.77	0.87	0.82
Trojan（木马）	0.85	0.89	0.87
Virus（病毒）	0.89	0.95	0.92
Worms（蠕虫）	0.97	0.90	0.93
Accuracy	—	—	0.90
Macro avg	0.91	0.90	0.90
Weighted avg	0.90	0.90	0.90

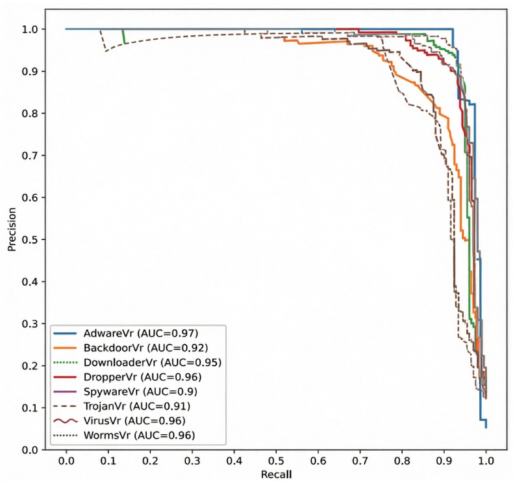
基础模型	精确率	召回率	F_1 分数
Adware（广告软件）	0.82	0.80	0.81
Backdoor（后门程序）	0.56	0.62	0.59
Downloader（下载器）	0.74	0.66	0.69
Dropper（投递器）	0.54	0.61	0.57
Spyware（间谍软件）	0.44	0.40	0.42
Trojan（木马）	0.44	0.41	0.42
Virus（病毒）	0.75	0.74	0.74
Worms（蠕虫）	0.56	0.58	0.57
Accuracy	—	—	0.59
Macro avg	0.61	0.60	0.60
Weighted avg	0.59	0.59	

为确保结论可靠性，使用 MAL-API-2019 数据集 20% 分层样本进行统计验证，结合 McNemar 检验（α=0.05），集成模型与基础模型的错误比例存在统计学显著差异，最终确立改进后集成方案的统计优势。

论文通过集成 ModifiedSoftVoting 模型在多类恶意软件检测中取得了显著性能提升。图 2（a）的 ROC 曲线（含 AUC 分数）和图 2（b）的 *P-R* 曲线进一步验证了模型对八类恶意软件（广告软件 / 后门程序等）的区分能力。如表 3 所示，横向对比显示，本模型以平均分数 F_1=0.90 显著优于现有方法提出的 RTF 模型（随机 Transformer 森林，F_1=0.61），且本方案无需依赖装袋技术（bootstrapping），仅通过优化集成策略即实现了当前最优性能。这一结果证实了 ModifiedSoftVoting 方法在恶意软件分类任务中的有效性和先进性。



（a）ROC-AUC



（b）Precision-Recall

图 2 新的模型架构性能图

表 3 各模型算法的比较

模	准确率	宏平均精确率	宏平均召回率	宏平均 F_1 分数
单层 LSTM	—	0.50	0.47	0.47
双层 LSTM	—	0.40	0.41	0.39
LSTM（Case2 变体）	0.55	0.56	0.58	0.57
GRU（Case2 变体）	0.55	0.56	0.59	0.57
RTF 模型	0.60	—	—	0.61
提出的集成模型	0.90	0.91	0.90	0.90

4 结论与未来方向

论文针对 API 调用序列在恶意软件分类中的关键作用，提出了一种基于集成 1D-CNN 的创新架构，成功解决了八类恶意软件（广告软件 / 后门程序 / 下载器 / 投递器 / 间谍软件 / 木马 / 病毒 / 蠕虫）的极端不平衡分类问题。实验采用 Mal-API-2019 基准数据集，结果显示：模型整体准确率达 90%，宏观平均指标（精度 91%/ 召回率 90%/F_1 分数 90%）显著优于现有研究（历史最佳 F_1=61%）。具体而言，图 2（a）的 ROC 曲线显示各类别 AUC 分数均超 0.95（广告软件 / 蠕虫达 0.99），图 2（b）的 *P-R* 曲线 AUC 值也稳定在 0.90~0.97 区间，验证了模型对各恶意软件亚型的强区分能力。值得一提的是，本文未采用数据增强或统计特征工程（如 PCA/ 子序列优化）等常见处理手段，其性能突破主要源于集成架构设计，为后续研究提供了新的技术路径。

参考文献：

[1] 轩勃娜，李进，宋亚飞，等．基于改进 MobileNetV2 的恶意代码分类方法 [J]. 计算机应用 ,2023,43(7):2217-2225.

[2] 王金伟，陈正嘉，谢雪，等．恶意软件检测和分类可视化技术综述 [J]. 网络与信息安全学报 ,2023,9(5):1-20.

[3] 赵海燕，马权益，曹健，等．面向任务扩展的增量学习动态神经网络：研究进展与展望 [J]. 电子学报，2023, 51(6): 1710-1724.

[4]VINOD P, JAIN H, GOLECHA Y K,et al. MEDUSA: MEtamorphic malware dynamic analysis using signature from API[C]//Proceedings of the 3rd international conference on security of information and networks.NewYork: ACM, 2010: 263-269.

[5]ALOM M Z, TAHA T M, YAKOPCIC C, et al. A state-of-the-art survey on deep learning theory and architectures[J]. Electronics, 2019, 8(3): 292.

[6]ZHANG Y N, HUANG Q J, MA X J, et al. Using multi-features and ensemble learning method for imbalanced malware classification[C]//2016 IEEE Trustcom/BigDataSE/ISPA. Piscataway: IEEE, 2016: 965-973.

[7]CATAK F O, AHMED J, SAHINBAS K,et al. Data augmentation based malware detection using convolutional neural networks[EB/OL].(2020-10-05)[2026-01-22].https://doi.org/10.48550/arXiv.2010.01862.

[8]HAN W J, XUE J F, WANG Y,et al. MalDAE: detecting and explaining malware based on correlation and fusion of static and dynamic characteristics[J].Computers and security, 2019, 83: 208-233 .

[9]LI C, ZHENG J J. API call-based malware classification using recurrent neural networks[J]. Journal of cyber security and mobility, 2021,10(3):617-640 .

[10]DEMIRKIRAN F, CAYIR A, UNAL U,et al. An ensemble of pre-trained transformer models for imbalanced multiclass malware classification[J]. Computers and security, 2022, 121: 102846.

[11]VASAN D, ALAZAB M, WASSAN S, et al. Image-based malware classification using ensemble of CNN architectures (IMCEC)[J]. Computers and security ,2020,92:101748.

【作者简介】

袁海根（1979—），男，江西宜春人，硕士研究生，副教授，研究方向：大数据、人工智能。

（收稿日期：2025-07-30 修回日期：2026-03-24）