基于MP-FSCIL的恶意代码分类方法

王 坚[1],刘 强[1],王 蕾[2*]
(1. 空军工程大学防空反导学院,陕西西安 710051;2. 空军工程大学空管领航学院,陕西西安 710051)

摘 要: 恶意软件家族通过代码混淆、多态变形等技术持续变异,导致特征空间偏移与模型决策边界失效,且零日攻击的快速演化和早期小样本场景进一步加剧了传统检测模型的知识退化与适应瓶颈. 针对上述问题,本文提出一种基于多原型小样本类增量学习的恶意代码分类方法MP-FSCIL(Multi-Prototype Few-Shot Class-Incremental Learning),旨在解决动态环境下模型灾难性遗忘与过拟合问题. 在基类训练阶段,将可分离大核注意力(Large Separable Kernel Attention, LSKA)与DenseNet网络融合,设计面向恶意软件图像的专用特征提取器,通过LSKA模块的大核注意力机制捕捉恶意软件图像的全局特征,结合DenseNet的密集连接特性保留细粒度局部特征,有效解决了传统特征提取器对恶意软件图像关键特征捕捉不充分的问题,模型在Malimg数据集上实现99.36%的分类准确率,优于现有FSCIL(Few-Shot Class-Incremental Learning)方法在Malimg数据集上的特征提取效果;在新类适应阶段,构建"自适应聚类-多原型学习"协同机制:通过G-means算法基于恶意软件特征分布自动迭代确定新类最佳聚类数量,再结合多原型学习为每个新类生成多个类原型,解决了传统单原型方法对类内特征异质性较高的恶意代码家族区分能力弱的问题,该策略使模型在每个增量会话上对新类识别准确率平均提升17.23%. 在Malimg与Microsoft Big 2015数据集上的跨数据集类增量实验验证了模型在真实恶意代码演化场景中的有效性,实验结果表明,MP-FSCIL在保持旧类记忆的同时,能够较好地学习新类特征,和现有研究方法相比,模型在所有类别上分类准确率提升8.89%,在最后一个增量会话上的性能下降率降至12.21%,且模型参数量仅为16.18 M,对每个样本的推理时间仅为12.6 ms,适合在实际应用中部署,为开放动态环境下的恶意软件检测提供了鲁棒、可扩展的解决方案.

关键词: 网络安全;恶意代码;多原型;小样本类增量学习;可分离大核注意力;G-means算法
基金项目: 国家自然科学基金(No.62402521);陕西省科学基金(No.2021JM-226)
中图分类号: TP309.5 **文献标识码:** A **文章编号:** 0372-2112(2025)10-3566-13
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20250692

A Malware Classification Method Based on MP-FSCIL

WANG Jian[1], LIU Qiang[1], WANG Lei[2*]
(1. *Air Defense and Antimissile School, Air Force Engineering University, Xi'an, Shaanxi* 710051, *China*;
2. *Air Traffic Control and Navigation School, Air Force Engineering University, Xi'an, Shaanxi* 710051, *China*)

Abstract: Malware families continue to mutate through techniques such as code obfuscation and polymorphic deformation, leading to the shift of feature space and the failure of model decision boundaries. In addition, the rapid evolution of zero-day attacks and small sample scenarios in early stage further exacerbate degradation of knowledge and adaptation bottleneck of traditional detection models. In response to the above issues, this article proposes a malicious code classification method based on multi-prototype few-shot class-incremental learning, namely MP-FSCIL (Multi-Prototype Few-Shot Class-Incremental Learning), which aims to resolve the problems of catastrophic forgetting and overfitting in dynamic environments. In the base class training stage, the large separable kernel attention (LSKA) is fused with the DenseNet network to design a dedicated feature extractor for malware images. The large kernel attention mechanism of the LSKA module is capable of capturing the global features of malware images, while the dense connection feature of the DenseNet structure is competent to preserve fine-grained local features, effectively solving the problem of insufficient capture of key features in malware images by traditional feature extractors. The proposed model achieves a classification accuracy of 99.36% on the Malimg dataset, which is better than the feature extraction effect of existing FSCIL (Few-Shot Class-Incremental Learning) methods on the Malimg dataset; In the new class adaptation stage, a collaborative mechanism of "adaptive clustering and

收稿日期:2025-08-11;录用日期:2025-10-09;责任编辑:覃怀银
*通讯作者:王蕾

multi prototype learning" is constructed: the G-means algorithm is used to automatically iterate based on the distribution of malicious software features to determine the optimal number of clusters for the new class, and then combined with multi prototype learning to generate multiple class prototypes for each class. This strategy addresses the weakness of traditional single-prototype methods in distinguishing malware families with high intra-class feature heterogeneity, and increases the model's average accuracy in identifying new classes by 17.23% in each incremental session. The cross-dataset class increment experiment on the Malimg and Microsoft Big 2015 datasets validated the effectiveness of the model in real scenarios of malicious code evolution. The experimental results show that MP-FSCIL can learn new class features well while maintaining the memory of old classes. Compared with existing research methods, the model improves classification accuracy by 8.89% on all classes, and the performance degradation rate on the last incremental session drops to 12.21%. Besides, The parameter size of the model is only 16.18 M, and the inference time for each sample is only 12.6 ms. It is suitable for deployment in practical applications and provides a robust and scalable solution for malware detection in open dynamic environments.

Key words: network security; malware; multi-prototype; few-shot class-incremental learning; large separable kernel attention; G-means algorithm

Foundation Item(s): National Natural Science Foundation of China (No.62402521); Science Foundation of Shaanxi (No.2021JM-226)

1　引言

随着信息技术的快速发展,网络安全面临的挑战日益严峻,其中,恶意代码问题尤为突出,已成为全球范围内广泛关注的焦点. 卡巴斯基安全网发布的《2024年移动端恶意软件演变》[1]报告显示,2024年,网络犯罪分子平均每月发起约280万次针对移动设备的恶意软件、广告软件和风险软件的攻击,在监测过程中,系统共检测到约110万个存在安全隐患的安装包,其中6.9万个与移动银行木马有关. 此外报告中披露了多个新的恶意软件家族及其技术演进,如Mamont银行木马通过社会工程手段进行传播,能够窃取用户的银行凭证和资金;CMoon USB蠕虫具备数据盗窃和远程控制功能,通过USB设备进行传播,对用户的设备造成危害;SparkCat恶意SDK植入程序能够绕过应用商店的安全机制,窃取屏幕截图和设备图库中的图像;还有植入Triada木马的WhatsApp修改版,可以下载并运行额外的恶意软件或广告软件模块. 以上数据和案例表明,在当前网络环境中,恶意代码大量存在且不断发展变异,对用户的日常生活产生了严重影响,甚至危及国家的网络安全,因此,研究和开发先进的恶意软件检测技术具有重要现实意义和紧迫性.

近年来,深度学习技术凭借其强大的特征学习能力,在恶意代码检测领域展现出显著优势,为该领域的问题提供了一系列解决方案[2]. 现有研究表明,通过构建深度神经网络对恶意软件数据集中的图像特征和序列特征进行多层次表征学习,能够有效捕获恶意样本的深层特征,从而实现较高的检测准确率和出色的泛化能力. 然而,网络攻防对抗的持续升级对现有检测体系构成严峻挑战. 首先,恶意软件家族通过代码混淆、多态变形等技术持续变异,这种动态演化过程不断改变恶意软件的统计特征分布,从而导致检测模型的性能随时间的推移而逐步退化[3]. 其次,零日攻击(Zero-day Attack)和新型恶意代码变种的快速演化,暴露出现有检测模型在开放动态环境中的适应瓶颈[4]. 以WannaCry勒索病毒为例,其早期传播阶段可供分析的样本数量通常很少,这种小样本场景不仅导致传统监督学习难以构建有效的特征表征,更使得模型难以及时迭代更新以响应新型攻击模式[5]. 因此,探索小样本类增量场景下的恶意代码分类方法具有重要意义.

自Tao等人[6]首次提出小样本类增量学习(Few-Shot Class-Incremental Learning,FSCIL)这一研究方向以来,该领域已取得了一系列研究成果. Cui等人[7]将半监督学习引入到FSCIL中,在训练过程中,未标记数据与标记数据相结合,以提高FSCIL的性能. Yang等人[8]提出动态支持网络(Dynamic Support Network,DSN),在每次训练过程中,DSN有选择地扩展网络节点,增强了增量类的特征表示能力. Song等人[9]提出了幻想空间的概念来增强语义知识,通过学习识别和对比虚拟类培育的幻想空间,促进了基类分离和新类的泛化. Liu等人[10]从神经正切核(Neural Tangent Kernel,NTK)理论框架出发,通过构建元学习机制优化全局NTK收敛,显著增强了FSCIL模型的抗遗忘性和跨任务泛化能力.

尽管当前FSCIL框架在通用图像识别领域已取得重大进展,但在恶意软件检测领域,其应用尚处于探索阶段. 这主要源于恶意软件检测对模型精度要求较高,而现有FSCIL方法存在两大关键瓶颈:一是模型在学习新增类别时,对旧类别的记忆快速消退,引发灾难性遗忘;二是新类别可用的样本数量匮乏,致使模型极易过拟合. 这两个问题直接制约着模型精度的提升,进而限制了FSCIL技术在恶意软件检测领域的实际应用.

针对上述问题,本文提出了一种基于多原型小样本类增量学习的恶意代码分类方法MP-FSCIL(Multi-

Prototype Few-Shot Class-Incremental Learning)，主要创新点如下：

(1)结合可分离大核注意力(Large Separable Kernel Attention，LSKA)，设计了一个改进的DenseNet网络作为特征提取器，能够有效提取恶意软件图像特征，在Malimg数据集上实现了99.36%的分类准确率.

(2)引入G-means算法自动确定每个新类的最佳聚类数量，聚类中心作为该类的原型，随后采用多原型学习的策略，有效提升了模型在新类上的准确率.

(3)根据数据集发布的时间顺序，跨数据集设置小样本类增量学习实验，以模拟更加真实的恶意代码类增量场景，验证了MP-FSCIL的有效性.

2 相关工作

恶意代码检测技术主要分为静态分析与动态分析两大类. 静态分析通过直接解析可疑文件结构，在不执行代码的前提下识别其特征与潜在风险，具有检测效率高的优势，但面对采用加密算法或代码混淆技术的复杂恶意软件时，这一方法具有局限性. 动态分析则依托沙箱环境执行样本，通过实时追踪文件运行过程中的所有操作行为生成分析报告，可有效规避加密混淆技术的干扰. 然而该技术对计算资源与存储空间的需求较高，导致实施成本与分析复杂度显著增加[11]. 值得关注的是，近年来深度学习技术的发展为恶意代码检测领域提供了新的技术路径，其在静态特征提取与动态行为建模中的应用，均实现了检测效能的显著提升.

2.1 静态分析技术

在恶意代码静态分析中，通常采用的特征包括操作码特征、图像特征和图结构特征等. 操作码特征方面，任卓君等人[12]提出了一种基于操作码频率的恶意软件可视化分类方法，利用设计的色谱标记出现数量最多的前15种操作码，按照空间填充曲线遍历RGB颜色空间的顺序进行填充，使得相同颜色的操作码聚集在一起；Lee等人[13]使用操作码类别序列和熵值来创建特征，在多种机器学习模型上进行效果评估，恶意软件检测和分类的准确率超过98.0%. 图像特征方面，王硕等人[14]提出一种结合多尺度特征融合与通道注意力机制的轻量化模型，有效提升恶意代码检测的准确率和时效性，同时解决了数据类别不平衡问题；李思聪等人[15]通过三通道映射技术将恶意代码的多维信息转换为图像通道，增强了特征区分性，并通过重参数化技术提升运行效率. 图结构特征方面，Ling等人[16]提取可执行文件的函数调用图和相应的控制流图，使用基于图神经网络的端到端学习框架进行恶意软件检测，模型的准确率和鲁棒性都有所提高. Do xuan等人[17]提出基于行为画像与图神经网络的APT(Advanced Persistent Threat)检测框架，通过提取系统内核事件构建进程行为图谱，结合图同构网络实现恶意行为分类.

2.2 动态分析技术

在动态分析方面，系统API(Application Programming Interface)调用轨迹因其能够完整记录目标样本的网络通信、内存访问和文件操作等关键行为，已成为恶意软件动态检测的核心特征依据. Zhang等人[18]提出的Mal-ASSF模型通过整合API调用的语义信息与序列特征，在关键行为序列的语义解析与识别层面展现出显著优势. Trizna等人[19]构建的Nebula架构采用基于Transformer的自注意力机制，从动态日志中捕获API调用、内存访问轨迹及文件操作等多维度特征，在检测精度与分类效能上超越传统CNN(Convolutional Neural Networks)与LSTM(Long Short-Term Memory)模型. Li等人[20]提出的DMalNet框架则采用分阶段处理策略，初始阶段从API名称及参数中提取语义特征，随后基于调用序列构建结构化图模型，将API间的关联关系转化为图结构信息，最终通过专门设计的图神经网络实现恶意软件的精准检测与分类.

2.3 动静结合分析技术

为了充分利用恶意代码多维度的特征，提高检测性能，也有许多研究将动态分析与静态分析相结合. Han等人[21]通过语义映射机制揭示了恶意软件动态API序列与静态API序列的内在关联性，构建了动态-静态API混合特征向量空间，实现两类特征的有机融合. Alhashmi等人[22]提出基于相似度度量的混合检测模型，创新性地将静态特征提取与动态行为分析相结合，通过挖掘API调用与函数关联模式的跨域相关性，显著增强了混合模型的特征表征能力. Huang等人[23]开发了恶意软件混合可视化框架，利用Cuckoo沙盒执行动态分析，将生成的动态行为日志转化为结构化图像，随后使用包含静态和动态特征的混合可视化图像训练神经网络. Jeon等人[24]提取操作码序列的静态语法特征，并捕获程序运行时的API调用时序特征，最终通过双向LSTM与空间金字塔池化网络的协同训练，实现了静态语义特征与动态行为特征的深度耦合.

以上方法主要集中在从已知的恶意软件类别中训练模型，取得了较高的检测精度，然而在现实网络环境中，未知恶意软件和变种不断出现，使得这些方法的应用受到限制. 为提高模型对未知恶意代码的检测能力，Qiang等人[25]首次提出基于FSCIL的增量恶意软件分类方法IMC，通过少量样本将预训练模型有效地动态扩展到未知类别，而不会失去识别已知类别的能力，然而这种方法的局限性在于仅考虑单个增量会话，而未考虑多个增量会话的情况. Chai等人[5]提出恶意软件检测框架MalFSCIL，采用解耦的训练策略，结合变分自动编码器(Variational AutoEncoders，VAE)来抑制灾难性遗忘，并

设计了一种基于类原型的动态边界刻画方法来实现增量决策边界的精确刻画. 虽然FSCIL已被应用于恶意软件检测领域,但相关研究仍然较少,且效果不够理想,随着新类别的持续增加,模型分类准确率快速下降,难以满足实际应用中对检测精度的需求. 为此,本文提出了一种基于MP-FSCIL的恶意代码分类方法,在现有研究工作基础上,进一步提升了模型精度.

3　MP-FSCIL框架概述

MP-FSCIL模型整体框架如图1所示,由基类训练阶段、伪增量学习阶段和新类适应阶段三个阶段组成,这种三阶段的设计延续了文献[26]提出的经典框架,并在其基础上进行创新性拓展. MP-FSCIL的具体训练和测试流程如下:在基类训练阶段,将恶意代码原始二进制文件转化而来的灰度图输入到Densenet-LSKA模型进行训练,训练完毕后模型解耦为特征提取器和分类器两部分,在后续阶段,特征提取器的参数被冻结,而分类器随着新类别的到来不断进行参数扩展,以适应分类的需求;在伪增量学习阶段,采用元学习的方法从基类训练阶段的数据集中采样多个*N*-way *K*-shot任务作为特征提取器的输入,随后训练一个图注意力网络(Graph Attention Network,GAT),利用训练好的GAT更新基类训练阶段得到的分类器,增强分类器的泛化性能;在新类适应阶段,每一类的训练集样本经过特征提取器后,进入G-means模块,得到该类的多个原型表示,随后输入到GAT网络,得到更新后的分类器,在当前所有已知类别上进行测试,输出预测类别.

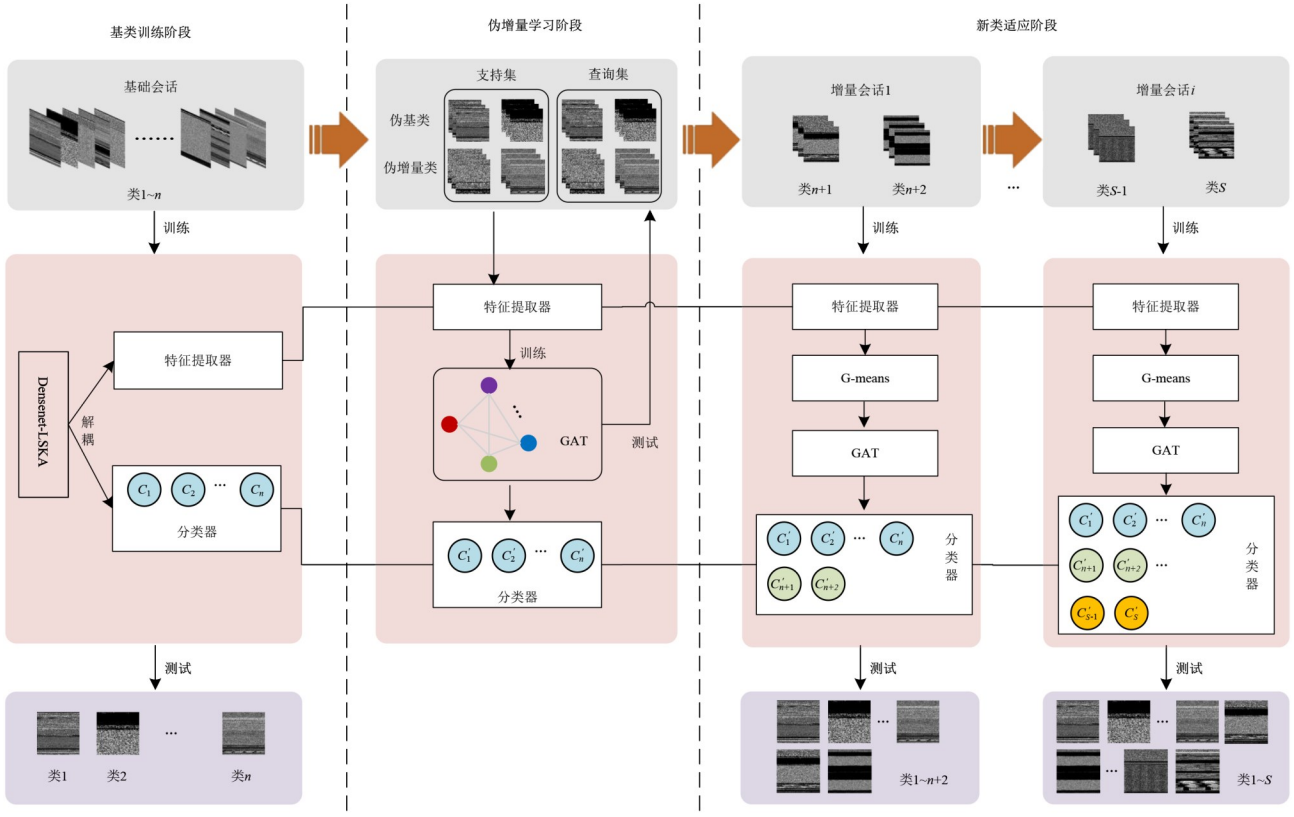


图1　MP-FSCIL模型整体框架

3.1　基类训练阶段

采用基类数据集训练一个改进的Densenet[27]模型,即Densenet-LSKA模型,训练完毕后,对模型进行解耦操作,全连接层之前的所有层构成一个特征提取器,全连接层作为分类器. 在后续阶段,特征提取器的参数将被冻结,只有分类器的参数可以调整.

Densenet-LSKA的设计参考了DenseNet121网络结构,和DenseNet121不同之处在于,DenseNet121将图像输入尺寸统一为224×224,包含四个DenseBlock和三个Transition层,本文结合恶意代码分类任务的实际,参考文献[28],将图像输入尺寸缩小为64×64,有利于在提高恶意代码分类性能的同时减少参数量,防止模型过拟合,由于输入图像尺寸的改变,仅需使用三个DenseBlock和两个Transition层. 此外,在第一个DenseBlock后增加大型可分离核注意力模块[29](LSKA),LSKA具有更大的感受野,能够捕捉更多全局特征,同时通过将二维卷积核分解为级联的水平和垂直一维卷积核,大大减少了参数量和计算量. Densenet-LSKA模型网络结构如图2所示,每一层的具体参数配置如表1所示.

LSKA的结构如图3所示,图中DW-Conv为深度卷

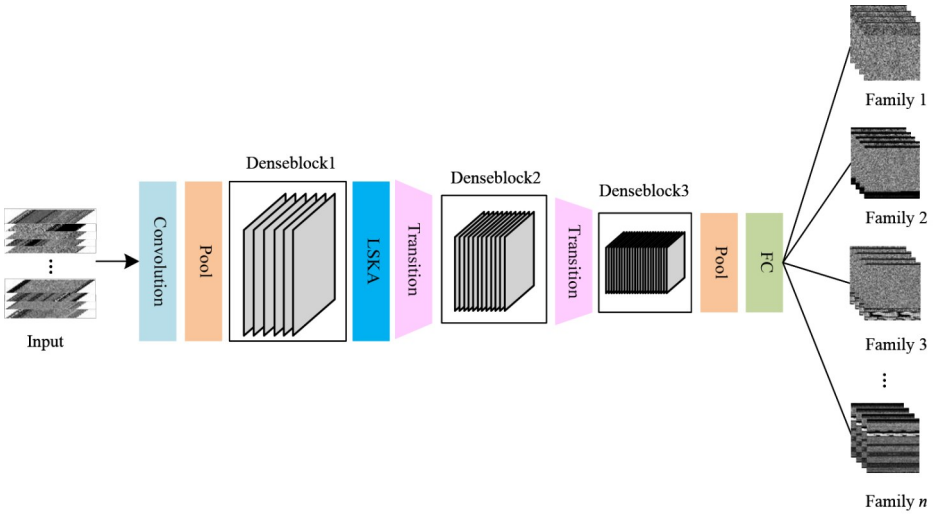


图2 Densenet-LSKA模型网络结构

表1 Densenet-LSKA模型网络结构和参数

层类型	配置参数	输出尺寸
输入层	输入:64×64×3	64×64×3
卷积层	卷积核大小:7×7 步长:2 通道数:64	32×32×64
最大池化层	池化窗口:3×3 步长:2	16×16×64
DenseBlock 1	卷积层数:6 通道增长率:32	16×16×256
LSKA	卷积核大小:11×11 膨胀系数:2	16×16×256
Transition 1	1×1卷积 步长:1 池化窗口:2×2	8×8×128
DenseBlock 2	卷积层数:12 通道增长率:32	8×8×512
Transition 2	1×1卷积 步长:1 池化窗口:2×2	4×4×256
DenseBlock 3	卷积层数:24 通道增长率:32	4×4×1 024
全局平均池化	无	1×1×1 024
全连接层	类别数:n	1×1×n

积,DW-D-Conv为深度膨胀卷积,Conv为逐点卷积,LSKA的输出可由式(1)~(4)获得.

$$\overline{Z}^{C}=\sum_{H,W}W^{C}_{(2d-1)\times 1}*\left(\sum_{H,W}W^{C}_{1\times(2d-1)}*F^{C}\right) \tag{1}$$

$$Z^{C}=\sum_{H,W}W^{C}_{\lfloor k/d\rfloor\times 1}*\left(\sum_{H,W}W^{C}_{1\times\lfloor k/d\rfloor}*\overline{Z}^{C}\right) \tag{2}$$

$$A^{C}=W_{1\times 1}*Z^{C} \tag{3}$$

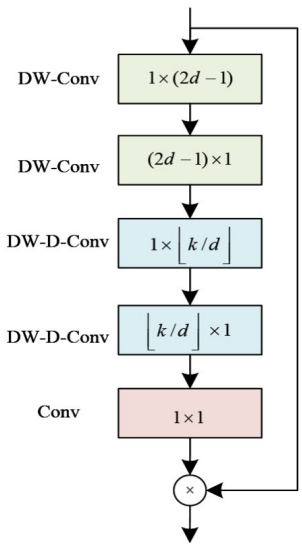


图3 LSKA结构

$$\overline{F}^{C}=A^{C}\otimes F^{C} \tag{4}$$

其中,$F\in\mathbb{R}^{C\times H\times W}$为输入特征图;$C$为输入通道的数量;$H$和$W$分别表示特征图的高度和宽度;*和$\otimes$分别表示卷积和哈达玛积;$d$为膨胀率;$\lfloor\cdot\rfloor$为向下取整操作;$W^{C}_{x\times y}$表示$C$个尺寸为$x\times y$的卷积核. 本文$k$值取11,$d$值取2,则LSKA模块对数据的处理如图4所示,对于输入尺寸为$H\times W\times C$的特征图,深度卷积的卷积核尺寸为1×3和3×1,分别提取水平和垂直方向的局部特征,深度膨胀卷积的卷积核尺寸为1×5和5×1,分别提取水平和垂直方向的全局特征,逐点卷积的卷积核尺寸为1×1×C,对各通道的特征进行整合,这一过程既实现了大核卷积的效果,同时和直接采用11×11的大核卷积相比,卷积核的参数量从121减小至16,有利于提高模型计算效率.

3.2 伪增量学习阶段

在学习新类的过程中,通常需要动态调整分类边界

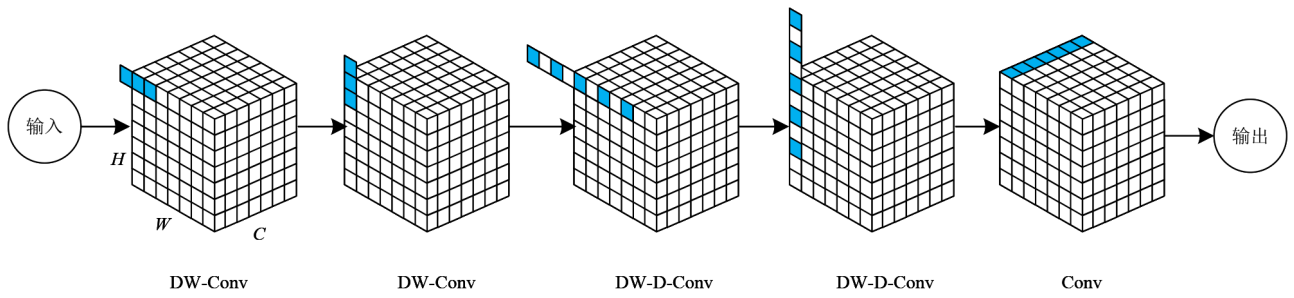


图4　LSKA模块数据处理流程

来容纳新增类别,但随着新类别的持续增加,特征空间的分类边界可能变得模糊不清,导致分类准确率下降.为解决这一问题,参考文献[26]设计伪增量学习算法,该算法采用元学习方法在基类数据集采样多个N-way K-shot任务,区分伪基类和伪增量类来模拟小样本类增量学习的场景,以提高对小样本任务的泛化能力,并将特征提取器提取的每个类别对应的原型作为结点,输入到图注意力网络(GAT),利用GAT对可区分特征的注意力机制对这些原型进行微调,得到增强后的原型表示,从而增强模型决策边界,提高分类性能.具体流程如算法1所示,其中N_i是伪增量类的数目,y_q和$\hat{y}_q$分别表示真实标签和预测值,$\mathcal{L}(\cdot)$为交叉熵损失函数.

算法1　伪增量学习算法

输入: 基类数据集$\mathcal{D}_b$,特征提取器$\mathcal{F}$,随机初始化的GAT模型$\mathcal{G}$,伪增量任务个数n

输出: 训练好的GAT模型$\mathcal{G}'$

FOR $i \leftarrow 1$ TO n DO
　从$\mathcal{D}_b$中采样伪基类的支持集$\mathcal{S}_b$和查询集$\mathcal{Q}_b$
　通过$\mathcal{F}$得到$\mathcal{S}_b$中每个类的原型表示W_b
　从$\mathcal{D}_b$中采样伪增量类的支持集$\mathcal{S}_i$和查询集$\mathcal{Q}_i$
　FOR c IN N_i DO
　　$\alpha \leftarrow \mathrm{rand}(90°, 180°, 270°)$
　　将$\{\mathcal{S}_i, \mathcal{Q}_i\}$中类$c$的图像旋转$\alpha$得到$\{\mathcal{S}_i', \mathcal{Q}_i'\}$
　END FOR
　通过$\mathcal{F}$得到$\mathcal{S}_i'$中每个类的原型表示W_i
　用$\mathcal{G}$更新$\{W_b, W_i\}$得到$\{W_b', W_i'\}$
　用$\left[\mathcal{F}, (W_b', W_i')\right]$得到$\{\mathcal{Q}_b, \mathcal{Q}_i'\}$的预测值$\hat{y}_q$
　$\mathrm{loss} \leftarrow \mathcal{L}(y_q, \hat{y}_q)$
　反向传播,优化$\mathcal{G}$
END FOR

需要说明的是,以上伪增量学习的过程,并不能模拟真实网络环境中恶意代码的演化,但这一过程能够充分利用已知的恶意代码类别,提高特征提取的鲁棒性,从而使模型更好地泛化到未知类别.

3.3　新类适应阶段

对增量会话中每个类别的训练集,采用特征提取器得到该类别每个样本的高维特征表示,随后采用G-means算法[30]对这些特征进行聚类.聚类分析时,将同一类别内的全部样本视为无标签数据,通过无监督学习策略挖掘类别内部的潜在结构.

G-means算法是基于传统k-means的改进方法,其核心思想是通过统计假设检验动态确定最佳聚类数量(k值).与传统k-means需预先指定k不同,G-means通过递归分裂初始聚类,验证数据分布是否符合正态性假设,从而自动推断最优聚类结构.

从$k=1$开始,执行传统k-means算法,获得初始聚类中心,随后对每个簇执行$k=2$的k-means算法,得到两个子簇中心$\boldsymbol{c}_1$和$\boldsymbol{c}_2$,定义投影向量:

$$\boldsymbol{v}=\boldsymbol{c}_1-\boldsymbol{c}_2 \tag{5}$$

计算数据点在$\boldsymbol{v}$上的投影并归一化:

$$x_i'=\frac{\langle \boldsymbol{x}_i, \boldsymbol{v}\rangle}{\|\boldsymbol{v}\|^2} \tag{6}$$

利用Anderson-Darling检验验证投影数据是否服从正态分布:

$$A^2(Z)=-\frac{1}{n}\sum_{j=1}^{n}(2i-1)\left[\log F(x_i')+\log\left(1-F(x_{n+1-i}')\right)\right]-n \tag{7}$$

$$A_*^2(Z)=A^2(Z)\left(1+\frac{4}{n}-\frac{25}{n^2}\right) \tag{8}$$

其中,F为标准正态分布函数,设定一个显著性水平α,若$A_*^2(Z)$大于α,则拒绝正态性假设,认为该聚类需进一步分裂,否则终止迭代.

重复以上步骤,直到所有聚类均通过正态性检验或达到预设的最大聚类数$k_{\max}$,最终k值即为最优聚类数量.

得到增量会话中每个类别的最佳聚类数量和多个簇中心后,将簇中心作为该类的原型表示,输入到GAT网络,GAT中的结点将被更新,更新后的结点包含基类和增量类中的所有类别的原型信息.随后用特征提取器得到待测试样本的特征表示,便可通过原型匹配进行分类.分类过程如图5所示,依次计算待测样本的特征向量与每一类原型的余弦相似度,余弦相似度最大值所在类别为预测类别.基类的每个类别只有一个原

型，而增量类的原型数量由 G-means 算法自动得到，随着增量会话的增加，原型数量也不断累积.

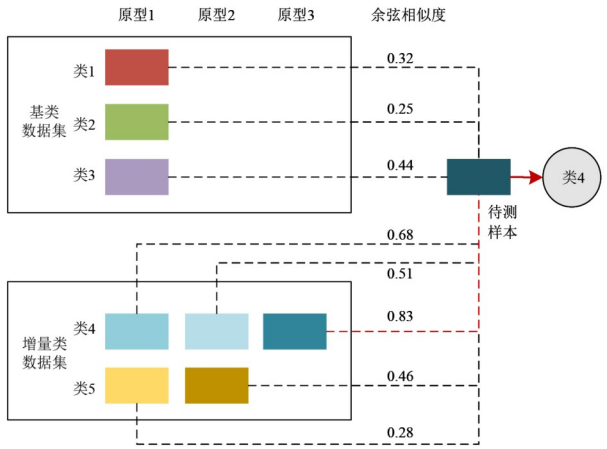


图5 待测样本在基类和增量类上的测试流程

4 实验与分析

4.1 实验设置

4.1.1 数据集

本文选取文献[31]提出 Fusion 数据集进行实验，该数据集对 Malimg、Microsoft Big 2015 和 MaleVis 三个数据集中所有恶意代码类别进行整合，三个数据集的发布具有时间上的先后关系，分别为 2011、2015 和 2019 年，且彼此之间没有类别上的交集. 整合后的 Fusion 数据集包含 59 个不同家族的恶意代码图像共 32 601 个，所有图像均处理为 224 × 224 像素，其中 9 339 个样本来自 Malimg 数据集，包含 25 个类别，10 868 个样本来自 Microsoft Big 2015 数据集，包含 9 个类别，12 394 个样本来自 MaleVis 数据集，包含 25 个类别，每个类别按照 7.0∶1.5∶1.5 的比例划分为训练集、验证集和测试集. Fusion 数据集的样本种类和数量分布如图 6 所示.

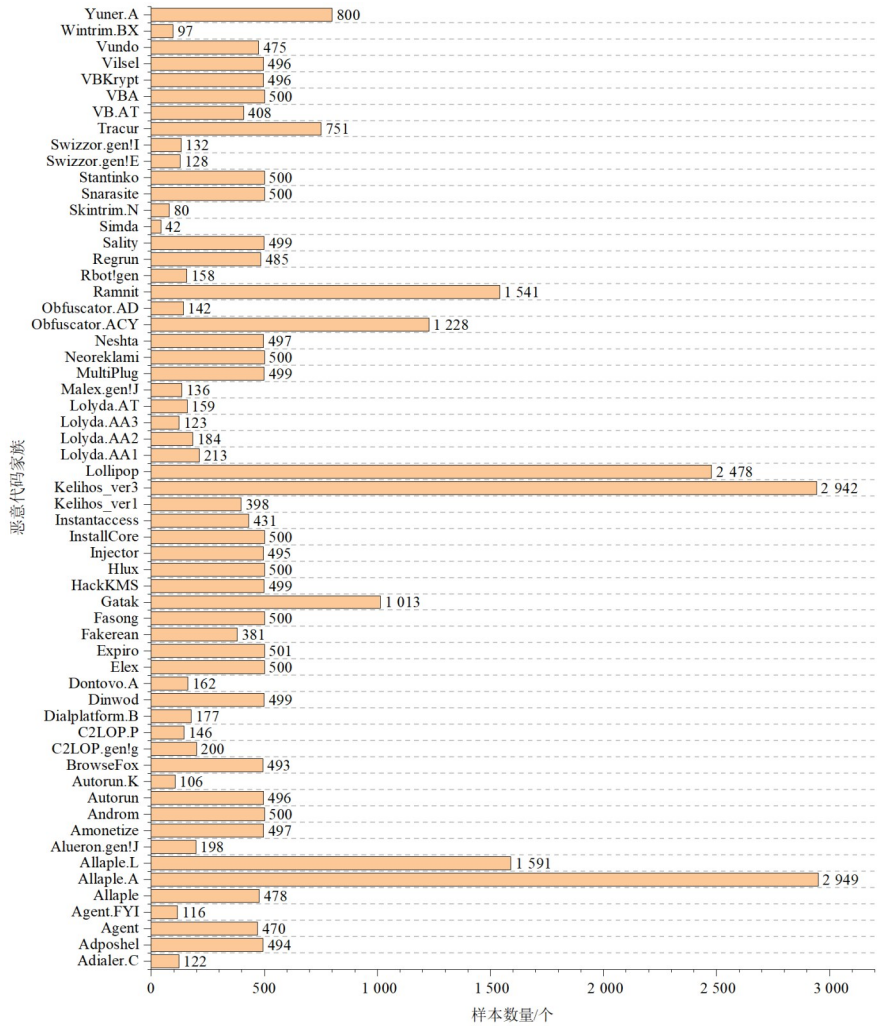


图6 Fusion 数据集样本分布

为模拟现实网络环境中未知或新型恶意软件攻击不断到来的情况，将 Fusion 数据集中来自 Malimg 数据集的 25 类作为基类，来自 Microsoft Big 2015 数据集的 9 类作为增量类，分为 3 个增量会话，每个增量会话包

含3个类别. 基类的训练集中所有样本均可参与训练,而增量类的训练集中仅选取10个样本用于训练. 未采用MaleVis数据集的原因是,该数据集通过三字节映射生成RGB彩色图像,与其他两个数据集直接将单字节转换为灰度图的处理方式存在根本性差异.

4.1.2　评价指标

为评估模型精度,在每个会话结束后,使用已知所有类别的测试集进行测试,将预测标签与真实标签进行比较,记录当前会话的准确率(Accuracy),并采用宏平均方法在最后一个会话上计算召回率(Recall)和误报率(FPR),以上三个指标分别定义为

$$\text{Accuracy}=\frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \tag{9}$$

$$\text{Recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \tag{10}$$

$$\text{FPR}=\frac{\text{FP}}{\text{FP}+\text{TN}} \tag{11}$$

其中,TP和FP分别表示对正样本的正确预测和错误预测,TN和FN分别表示对负样本的正确预测和错误预测. 为评估模型抗遗忘性能,使用性能下降率PD和平均遗忘率F_{avg}两个指标,分别定义为

$$\text{PD}=A_0-A_{\text{b}} \tag{12}$$

$$F_{\text{avg}}=\frac{1}{T-1}\sum_{t=1}^{T-1}\left(\text{acc}_t^{\text{initial}}-\text{acc}_t^{\text{final}}\right) \tag{13}$$

其中,A_0表示基础会话上的准确率;A_{b}表示最后一个增量会话上的准确率;T为总会话数;$\text{acc}_t^{\text{initial}}$和$\text{acc}_t^{\text{final}}$分别表示第$t$个会话中新增类别在第$t$个会话和最后一个会话上的准确率. PD和$F_{\text{avg}}$的值越小,表明模型抗遗忘性能越强.

为评估模型复杂度,使用参数量、训练时间和推理时间三个指标,参数量为模型各阶段需要学习的参数总量,训练时间为各阶段的训练时间总和,推理时间为预测一个样本所需要的时间.

4.1.3　实验环境和超参数设置

实验环境如表2所示. 各阶段使用的超参数如表3所示. 其中,学习率、权重衰减参数、损失函数和图像归一化尺寸的设置参考了文献[28],批量大小的设置在GPU(Graphics Processing Unit)性能允许范围内尽可能增大,最终确定为128,训练轮次的设置依据模型收敛情况在基类训练和伪增量学习阶段分别设置为100和50,保证模型在该范围内既能达到收敛,又不造成过多时间上的开销,其余参数均为对比实验的结果.

表2　实验环境配置

实验环境	具体配置
操作系统	Windows 11
CPU	Intel(R) Core(TM) i7-13620H CPU @ 2.40 GHz
内存	16 GB
硬盘	1 TB
显卡	NVIDIA GeForce RTX 4050
开发框架	Pytorch
开发语言	Python 3.10

表3　超参数设置

阶段	超参数	设置
基类训练阶段	学习率	0.002
	权重衰减参数	0.001
	批量大小	128
	训练轮次	100
	损失函数	交叉熵损失函数
	优化器	AdamW
	图像归一化尺寸	(64,64)
伪增量学习阶段	伪增量任务类型	10-way 10 shot
	采样任务个数	100
	训练轮次	50
	学习率	0.002
	权重衰减参数	0.001
	损失函数	交叉熵损失函数
	优化器	AdamW
新类适应阶段	新类数量	9
	增量会话个数	3
	增量任务类型	3-way 10-shot

4.2　MP-FSCIL性能实验

为系统评估MP-FSCIL方法的有效性,以少量样本微调(Finetune)作为基线方案,并选取小样本类增量学习领域近年来的四种先进方法开展对比实验,实验结果如表4所示,模型在每个会话上的准确率动态变化曲线如图7示. 其中,基线Finetune方法基于ResNet18网络实现,首先在基类数据集上完成100轮训练,待模型收敛后,再利用增量数据集中每一类别的全部训练样本,对模型参数进行微调更新. 对于CEC[26]、FACT[32]、C-FSCIL[33]和BiDistFSCIL[34]四种对比方法,由于其均已公开源代码,为保证复现结果的可靠性,本研究在复现过程中仅根据实验环境的GPU性能,对批量大小进行了适应性调整,未对源代码及其他关键超参数(如学习率、训练轮次、损失函数等)作任何修改.

结合表4的详细数据和图7的可视化结果可以得出,在模型精度方面,所有方法在初始会话均达到98%以上的准确率,MP-FSCIL以99.36%的准确率略优于其他方法,随着增量会话的推进,所有方法准确率均呈下降趋势,在最后一个会话上MP-FSCIL的准确率和召回率最高,分别为87.15%和90.04%,且FPR值最低,仅为0.4%,表明MP-FSCIL的精度优于其他方法;模型在会话3上的准确率明显低于其他会话,表明类别数量的增加导致分类

表4 不同方法性能指标对比

方法	准确率/%				PD/%	召回率/%	FPR/%	F_{avg}/%	参数量/M	训练时间/min	推理时间/ms
	会话0	会话1	会话2	会话3							
Finetune	98.51	89.8	72.84	64.97	33.54	66.75	1.17	12.01	**11.17**	**96**	**6.4**
CEC	98.51	96.66	82.38	76.62	21.89	80.95	0.72	8.23	11.18	142	8.3
FACT	98.29	95.27	79.65	74.91	23.38	79.81	0.84	7.96	11.18	128	9.7
C-FSCIL	98.43	93.4	86.02	78.26	20.17	82.54	0.65	8.63	12.75	139	8.5
BiDistFSCIL	98.29	94.25	85.73	75.43	22.86	78.48	0.74	8.08	15.84	315	11.4
MP-FSCIL	**99.36**	**97.87**	**94.24**	**87.15**	**12.21**	**90.04**	**0.40**	**2.31**	16.18	207	12.6

注：表中加粗数字表示该列性能指标最优值.

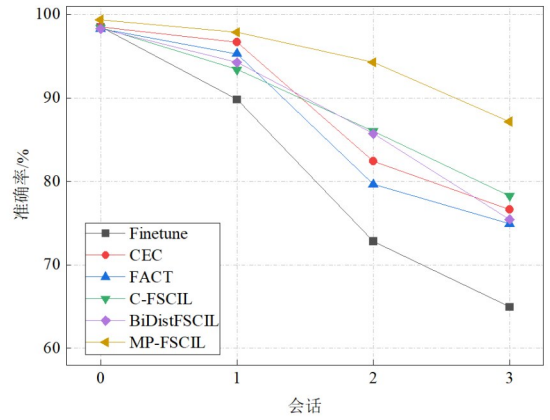


图7 不同方法在每个会话上的准确率

难度上升，这一现象可归结于增量过程中持续增加的分类复杂性：随着迭代次数的累积，类原型的渐进式扩张致使特征空间区分度降低，最终削弱了模型对新旧类别样本的判别能力. 在模型抗遗忘性能方面，MP-FSCIL的PD值为12.21%，F_{avg}为2.31%，均为最低，表明MP-FSCIL的抗遗忘性能最佳. 在模型复杂度方面，MP-FSCIL参数量为16.18 M，比其他模型高出0.34~5.01 M，训练时间为207 min，和其他模型相比处于中等水平，推理时间为12.6 ms，虽然和其他模型相比有所增加，但毫秒级别的推理速度不影响检测的实时性，可见MP-FSCIL在带来性能提升的同时，并未显著增加计算复杂度和资源开销.

4.3 消融实验

4.3.1 多原型方法消融实验

为探究本文提出的多原型学习方法对实验结果的影响，在新类适应阶段，去除G-means模块，即采用单原型进行实验，和多原型方法进行效果对比，结果如图8和表5所示. 结合图8的可视化结果和表5的详细数据可以得出，在模型精度方面，随着增量会话的推进，多原型方法相较于单原型方法准确率提升愈加明显，在最后一个会话上多原型方法和单原型方法相比具有更高的召回率和更低的FPR值，表明采用多原型方法能够带来模型精度的显著提升；在模型抗遗忘性能方面，和单原型方法相比，多原型方法的PD和F_{avg}显著降低，表明其抗遗忘性能更佳；在模型复杂度方面，多原型方法和单原型方法相比，参数量保持不变，训练时间和推理时间略有增加，究其原因在于引入了G-means模块，从而导致模型整体计算量小幅上升.

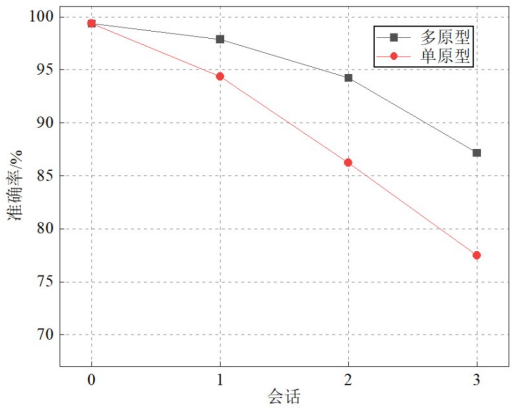


图8 多原型和单原型方法在每个会话上的准确率

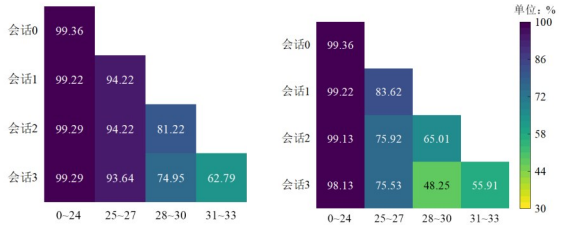
为了更细致地观察模型的分类细节，在每个会话上区分旧类和新类，根据准确率绘制热力图如图9所示，横轴的0~24表示基类，25~27、28~30和31~33分别表示三个增量会话中依次出现的新类别，热力图的颜色越深，表示准确率越高. 此外，在最后一个会话上绘制所有类别的混淆矩阵如图10所示，其中基类和增量类用红线分割.

表5 多原型和单原型方法性能指标对比

方法	准确率/%				PD/%	召回率/%	FPR/%	F_{avg}/%	参数量/M	训练时间/min	推理时间/ms
	会话0	会话1	会话2	会话3							
单原型	99.36	94.34	86.22	77.46	21.90	81.43	0.68	8.69	16.18	**188**	**9.5**
多原型	**99.36**	**97.87**	**94.24**	**87.15**	**12.21**	**90.04**	**0.40**	**2.31**	**16.18**	207	12.6

注：表中加粗数字表示该列性能指标最优值.

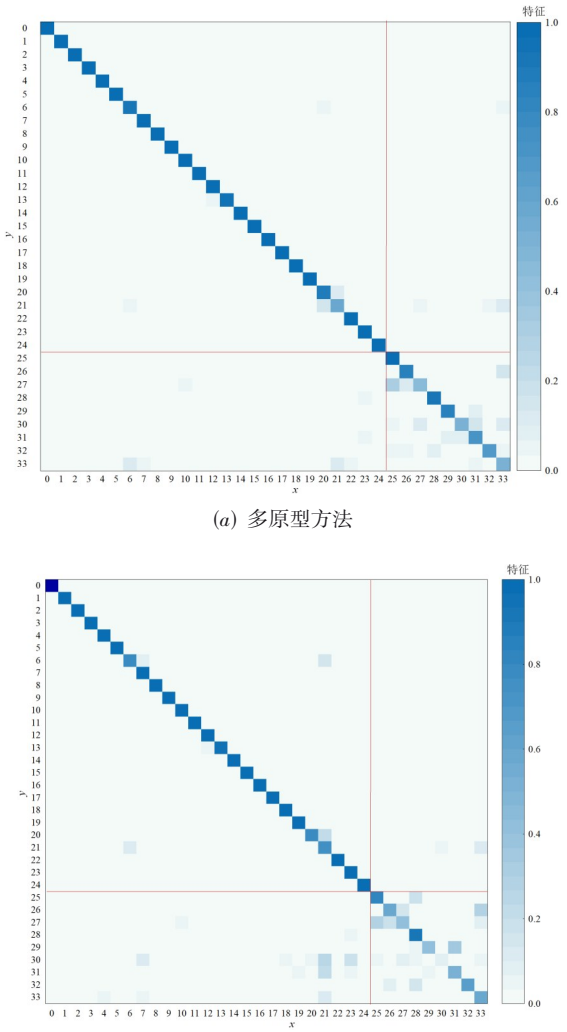
从图9热力图的结果可以得出,在基类上,多原型方法和单原型方法性能相近,均达到了较好的分类效果,说明模型对基类的记忆保持较好,不存在灾难性遗忘问题. 在新类上,多原型方法准确率明显高于单原型方法,随着增量会话的增加,两种方法的分类准确率均呈下降趋势,多原型方法在25~27三个类上下降0.58个百分点,在28~30三个类上下降6.27个百分点,单原型方法在25~27三个类上下降8.09个百分点,在28~30三个类上下降16.76个百分点,说明多原型方法准确率下降幅度较小,能够有效缓解灾难性遗忘问题,而单原型方法准确率下降幅度较大,对旧知识遗忘较快.



(a) 多原型方法　　(b) 单原型方法

图9　多原型和单原型方法在新旧类别上准确率对比

从图10混淆矩阵的结果可以得出,在基类上,多原型方法和单原型方法对每个类别均有较好的分类效果,说明模型能够有效提取并区分每个类别的特征. 在增量类上,多原型方法性能下降幅度较小,而单原型方法性能下降明显,错误分类结果更多,更容易将新类错误地识别为旧类,说明采用多原型方法能够有效防范新旧类别特征之间的混淆,从而提高模型对新类的识别能力. 究其原因,当面对样本稀缺的新类别时,由于样本间可能存在显著的特征异质性,传统的单原型建模方法本质上是对高维特征向量进行算术平均,这种过度聚合操作容易导致特征信息的损失,难以有效表征类内存在的差异性特征分布. 相比之下,多原型范式通过构建多个原型向量,可实现对类内不同特征子空间的精细化建模,从而在数学本质上更契合小样本场景下的数据分布特性.



(a) 多原型方法

(b) 单原型方法

图10　多原型和单原型方法混淆矩阵对比

4.3.2　Densenet-LSKA模型消融实验

为探究Densenet-LSKA模型对实验结果的影响,选择不同的模型作为特征提取器,和Densenet-LSKA进行效果对比,结果如图11和表6所示.

结合图11的可视化结果和表6的详细数据可以得出,模型的选择对实验结果也有较大影响,在模型精度

表6　不同模型作为特征提取器时性能指标对比

方法	准确率/%				PD/%	召回率/%	FPR/%	F_{avg}/%	参数量/M	训练时间/min	推理时间/ms
	会话0	会话1	会话2	会话3							
MobileNetV4[35]	97.86	95.63	88.08	82.17	15.69	83.60	0.62	6.08	**7.85**	**156**	**8.2**
Resnet18[36]	98.29	97.56	91.69	82.89	15.4	84.61	0.58	5.91	15.23	179	8.5
EfficientNetV2[37]	98.36	97.47	92.7	84.58	13.78	85.75	0.51	5.47	27.99	252	13.6
ViT[38]	98.22	97.25	94.29	85.16	13.06	87.46	0.49	4.28	86.44	396	15.8
Densenet121[27]	98.43	97.28	94.04	85.45	12.98	88.69	0.47	2.89	12.03	183	10.4
Densenet-LSKA	**99.36**	**97.87**	**94.24**	**87.15**	**12.21**	**90.04**	**0.40**	**2.31**	16.18	207	12.6

注:表中加粗数字表示该列性能指标最优值.

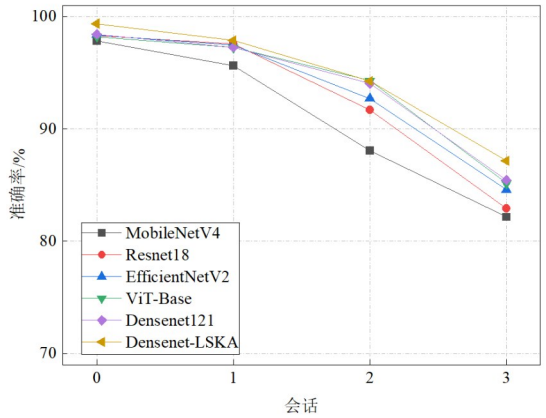


图 11 不同模型作为特征提取器时每个会话上的准确率

方面，Densenet-LSKA 的准确率最高，表明 Densenet-LSKA 模型对恶意软件图像特征的提取最为有效；在模型抗遗忘性能方面，Densenet-LSKA 的 PD 和 F_{avg} 值最低，表明其抗遗忘能力最强；在模型复杂度方面，Densenet-LSKA 的参数量、训练时间和推理时间均属于适中水平.

4.3.3 伪增量学习模块消融实验

为探究伪增量学习模块对实验结果的影响，在基类训练阶段结束后，跳过伪增量学习阶段，直接进入新类适应阶段，测试模型在新类上的性能，实验结果如图 12、图 13 和表 7 所示.

结合图 12、图 13 的可视化结果和表 7 的详细数据可以得出，在模型精度方面，采用伪增量学习时，模型在第二和第三个增量会话上准确率分别提高 5.79 个百分点和 5.57 个百分点，召回率提高 4.46 个百分点，FPR 降低 0.21 个百分点，且模型在新类上性能比较稳定，尤其是在第 28~30 三个类上准确率改善较为明显，表明采用伪增量学习能够有效提高模型对新类的识别能力，从而带来总体分类准确率的提升；在模型抗遗忘性能方面，采用伪增量学习时，PD 和 F_{avg} 分别降低 5.57 个百分点和 3.82 个百分点，表明伪增量学习模块能够提高模型抗遗忘性能；在模型复杂度方面，采用伪增量学习时参数量和训练时间都有所增加，究其原因在于引入了 GAT 模块，从而带来额外的计算和资源开销.

表 7 伪增量学习对各项性能指标的影响

是否采用伪增量学习	准确率/%				PD/%	召回率/%	FPR/%	F_{avg}/%	参数量/M	训练时间/min	推理时间/ms
	会话 0	会话 1	会话 2	会话 3							
×	99.36	97.61	88.45	81.58	17.78	85.58	0.61	6.13	**12.13**	**128**	12.6
√	**99.36**	**97.87**	**94.24**	**87.15**	**12.21**	**90.04**	**0.40**	**2.31**	16.18	207	**12.6**

注：表中加粗数字表示该列性能指标最优值.

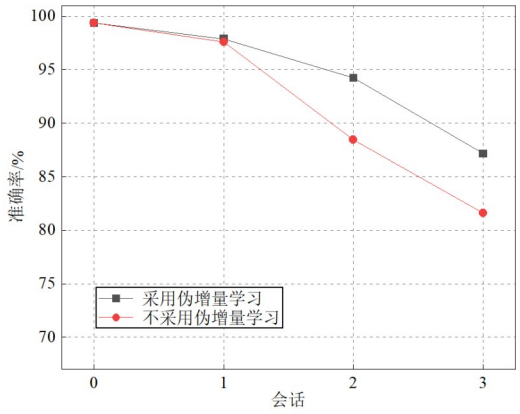
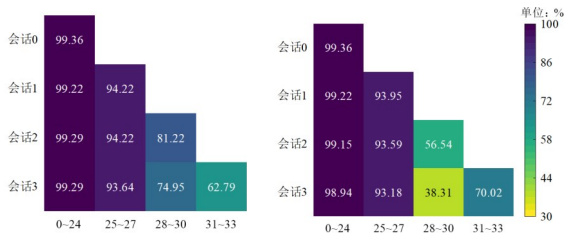


图 12 伪增量学习对每个会话准确率的影响



(a) 采用伪增量学习 (b) 不采用伪增量学习

图 13 伪增量学习对新旧类别分类准确率的影响

5 结语

本文针对恶意代码检测领域面临的挑战，提出了基于多原型小样本类增量学习(MP-FSCIL)的恶意代码分类方法，通过融合改进的 DenseNet 骨干网络、自适应原型聚类与跨数据集增量验证机制，显著提升了模型在开放动态环境中的鲁棒性与泛化能力. 在基类训练阶段，结合 LSKA 模块设计了改进的 DenseNet 模型，通过局部-全局注意力协同优化，提升了模型特征提取的稳定性和鲁棒性；在伪增量学习阶段，通过区分伪基类与伪增量类构建小样本类增量场景，并利用图注意力网络(GAT)对原型节点进行注意力微调，增强特征表示与决策边界，从而提升分类性能；在新类适应阶段，通过 G-means 算法引入多原型学习策略，显著提升了模型在新类小样本场景下的准确率. 然而，本文方法仍存在以下局限性：其一，MP-FSCIL 框架基于图像特征构建，而当前恶意代码检测领域缺乏高质量的图像化数据集支撑，现有公开数据集普遍存在样本时效性滞后、恶意代码变种覆盖不全等问题，导致模型难以充分捕捉真实网络环境中代码样本的多样性与复杂性特征，这在一定程度上限制了 MP-FSCIL 的实际部署效能；其二，尽管 MP-FSCIL 在提升新类别识别准确率方面取得进展，但现有性能指标

与网络安全实战中恶意代码检测的高精度要求仍存在差距;其三,当前大模型技术的快速发展催生了新的安全挑战,利用大模型复制、生成恶意代码的现象日益增多,这类生成式恶意代码往往具有特征隐蔽性强、变种迭代速度快、与合法代码相似度高的特点,而本文方法尚未针对该类新型恶意代码的检测需求进行适配,难以有效应对大模型驱动下恶意代码演化的新趋势. 针对上述问题,后续研究将重点开展三个方面工作:一是持续跟踪恶意代码检测领域的数据集建设进展,在更多数据集上验证模型性能;二是探索多模态特征融合机制,通过整合静态特征和动态特征等多维度信息,着力提升分类精度与鲁棒性;三是针对大模型生成式恶意代码的检测需求,研究生成式恶意代码的特征演化规律,提高MP-FSCIL框架对真实网络环境中新形态恶意代码的检测能力,为大模型时代的恶意代码防御提供技术支撑.

致谢　感谢王鹏、吴暄,以及张丹丹博士给本文提出的参考意见.

参考文献

[1] Kaspersky Security Network. Mobile malware evolutionin 2024[EB/OL]. (2025-03-03)[2025-5-1]. https://securelist.com/mobile-threat-report-2024/115494/.

[2] SONG Y F, ZHANG D D, WANG J, et al. Application of deep learning in malware detection: A review[J]. Journal of Big Data, 2025, 12(1): 99.

[3] BROSOLO M, PUTHUVATH V, ASMITHA K A, et al. SoK: Visualization-based malware detection techniques[C]//Proceedings of the 19th International Conference on Availability, Reliability and Security. New York: ACM, 2024: 1-13.

[4] FANG W B, HE J J, LI W S, et al. Comprehensive Android malware detection based on federated learning architecture[J]. IEEE Transactions on Information Forensics and Security, 2023, 18: 3977-3990.

[5] CHAI Y H, CHEN X M, QIU J, et al. MalFSCIL: A few-shot class-incremental learning approach for malware detection[J]. IEEE Transactions on Information Forensics and Security, 2024, 20: 2999-3014.

[6] TAO X Y, HONG X P, CHANG X Y, et al. Few-shot class-incremental learning[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2020: 12180-12189.

[7] CUI Y W, XIONG W T, TAVAKOLIAN M, et al. Semi-supervised few-shot class-incremental learning[C]//2021 IEEE International Conference on Image Processing. Piscataway: IEEE, 2021: 1239-1243.

[8] YANG B Y, LIN M B, ZHANG Y X, et al. Dynamic support network for few-shot class incremental learning[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023, 45(3): 2945-2951.

[9] SONG Z Y, ZHAO Y F, SHI Y J, et al. Learning with fantasy: Semantic-aware virtual contrastive constraint for few-shot class-incremental learning[C]//2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2023: 24183-24192.

[10] LIU J R, JI Z, PANG Y W, et al. NTK-guided few-shot class incremental learning[J]. IEEE Transactions on Image Processing, 2024, 33: 6029-6044.

[11] GABER M G, AHMED M, JANICKE H. Malware detection with artificial intelligence: A systematic literature review[J]. ACM Computing Surveys, 2024, 56(6): 1-33.

[12] 任卓君, 陈光, 卢文科. 恶意软件的操作码可视化方法研究[J]. 计算机工程与应用, 2021, 57(18): 130-134.
REN Z J, CHEN G, LU W K. Research on visualization method of malware opcodes[J]. Computer Engineering and Applications, 2021, 57(18): 130-134. (in Chinese)

[13] LEE H, KIM S, BAEK D, et al. Robust IoT malware detection and classification using opcode category features on machine learning[J]. IEEE Access, 2023, 11: 18855-18867.

[14] 王硕, 王坚, 王亚男, 等. 一种基于特征融合的恶意代码快速检测方法[J]. 电子学报, 2023, 51(1): 57-66.
WANG S, WANG J, WANG Y N, et al. A fast malicious code detection method based on feature fusion[J]. Acta Electronica Sinica, 2023, 51(1): 57-66. (in Chinese)

[15] 李思聪, 王坚, 宋亚飞, 等. TriCh-LKRepNet: 融合三通道映射与结构重参数化的大核卷积恶意代码分类网络[J]. 电子学报, 2024, 52(7): 2331-2340.
LI S C, WANG J, SONG Y F, et al. TriCh-LKRepNet: A large kernel convolutional malicious code classification network for structure reparameterisation and triple-channel mapping[J]. Acta Electronica Sinica, 2024, 52(7): 2331-2340. (in Chinese)

[16] LING X, WU L F, DENG W, et al. MalGraph: Hierarchical graph neural networks for robust windows malware detection[C]//IEEE INFOCOM 2022 - IEEE Conference on Computer Communications. New York: ACM, 2022: 1998-2007.

[17] DO XUAN C, HUONG D. A new approach for APT malware detection based on deep graph network for endpoint systems[J]. Applied Intelligence, 2022, 52(12): 14005-14024.

[18] ZHANG S F, WU J H, ZHANG M Z, et al. Dynamic malware analysis based on API sequence semantic fusion[J]. Applied Sciences, 2023, 13(11): 6526.

[19] TRIZNA D, DEMETRIO L, BIGGIO B, et al. Nebula: Self-attention for dynamic malware analysis[J]. IEEE Transactions on Information Forensics and Security, 2024, 19: 6155-6167.

[20] LI C, CHENG Z J, ZHU H, et al. DMalNet: Dynamic malware analysis based on API feature engineering and graph learning[J]. Computers & Security, 2022, 122: 102872.

[21] HAN W J, XUE J F, WANG Y, et al. MalDAE: Detect-

ing and explaining malware based on correlation and fusion of static and dynamic characteristics[J]. Computers & Security, 2019, 83: 208-233.
[22] ALHASHMI A A, DAREM A A, ALASHJAEE A M, et al. Similarity-based hybrid malware detection model using API calls[J]. Mathematics, 2023, 11(13): 2944.
[23] HUANG X, MA L, YANG W Y, et al. A method for windows malware detection based on deep learning[J]. Journal of Signal Processing Systems, 2021, 93(2): 265-273.
[24] JEON J, JEONG B, BAEK S, et al. Hybrid malware detection based on Bi-LSTM and SPP-net for smart IoT[J]. IEEE Transactions on Industrial Informatics, 2022, 18(7): 4830-4837.
[25] QIANG Q, CHENG M, HU Y, et al. An incremental malware classification approach based on few-shot learning[C]//ICC 2022 - IEEE International Conference on Communications. Piscataway: IEEE, 2022: 2682-2687.
[26] ZHANG C, SONG N, LIN G S, et al. Few-shot incremental learning with continually evolved classifiers[C]//2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2021: 12450-12459.
[27] HUANG G, LIU Z, VAN DER MAATEN L, et al. Densely connected convolutional networks[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2017: 2261-2269.
[28] 张丹丹, 宋亚飞, 刘曙. MalMKNet: 一种用于恶意代码分类的多尺度卷积神经网络[J]. 电子学报, 2023, 51(5): 1359-1369.
ZHANG D D, SONG Y F, LIU S. MalMKNet: A multi-scale convolutional neural network used for malware classification[J]. Acta Electronica Sinica, 2023, 51(5): 1359-1369. (in Chinese)
[29] LAU K W, PO L M, REHMAN Y A U. Large separable kernel attention: Rethinking the large kernel attention design in CNN[J]. Expert Systems with Applications, 2024, 236: 121352.
[30] FENG Y, HAMERLY G. PG-means: Learning the number of clusters in data[M]//Advances in Neural Information Processing Systems 19. Cambridge: MIT Press, 2007: 393-400.
[31] SALAS M P, DE GEUS P L. Deep learning applied to imbalanced malware datasets classification[J]. Journal of Internet Services and Applications, 2024, 15(1): 342-359.
[32] ZHOU D W, WANG F Y, YE H J, et al. Forward compatible few-shot class-incremental learning[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2022: 9036-9046.
[33] HERSCHE M, KARUNARATNE G, CHERUBINI G, et al. Constrained few-shot class-incremental learning[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2022: 9047-9057.
[34] ZHAO L L, LU J, XU Y L, et al. Few-shot class-incremental learning via class-aware bilateral distillation[C]//2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2023: 11838-11847.
[35] QIN D F, LEICHNER C, DELAKIS M, et al. MobileNetV4: Universal models for the mobile ecosystem[C]//Computer Vision - ECCV 2024. Cham: Springer, 2025: 78-96.
[36] 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. Piscataway: IEEE, 2016: 770-778.
[37] TAN M X, LE Q V. EfficientNetV2: Smaller models and faster training[EB/OL]. (2021-06-23)[2025-09-09]. https://arXiv.org/abs/2104.00298.
[38] DOSOVITSKIY A, BEYER L, KOLESNIKOV A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[EB/OL]. (2021-06-03)[2025-09-09]. https://arXiv.org/abs/2010.11929.

作者简介

王 坚 男,1982年2月出生于陕西省渭南市. 现为空军工程大学防空反导学院副教授. 主要研究方向为智能信息处理和恶意软件检测.
E-mail: 26471375@qq.com

刘 强 男,1993年11月出生于陕西省渭南市. 现为空军工程大学硕士研究生. 主要研究方向为智能信息处理和恶意软件检测.
E-mail: dugugongsui@163.com

王 蕾 女,1983年7月出生于陕西省商洛市. 现为空军工程大学空管领航学院讲师. 主要研究方向为信号与信息处理.
E-mail: xiaoci2000@sina.com