DOI:10.19651/j.cnki.emt.2519402

基于动态原型与对比学习的小样本 Android 恶意软件家族分类方法*

朱雪金[1] 李沈洋[1,2] 李 燕[1,2]
(1.无锡学院物联网工程学院 无锡 214105;2.南京信息工程大学计算机学院、网络空间安全学院 南京 210044)

摘 要:为解决小样本条件下安卓恶意软件家族分类中模型泛化性差导致准确率低的问题,本文提出一种监督对比学习驱动的动态原型网络框架 SupProto。该框架采用监督对比学习(SupCon)优化特征嵌入空间,以提升类间分离度和类内紧致性;同时结合基于层次聚类与轮廓系数的动态原型机制,以适应家族内部多模态结构差异。在输入与编码设计方面,本方法通过多源静态特征构建 RGB 图像输入,提供统一且具判别性的恶意软件表示,并利用 DenseNet121 网络结合 CBAM 注意力机制强化特征提取能力。实验结果表明,在 Drebin 和 CIC InvesAndMal2019 两个公开数据集上,SupProto 在 5-way 5-shot 设置下的分类准确率分别达到 90.59% 和 85.64%,在 5-way 1-shot 设置下的准确率分别为 75.56% 和 67.96%。

关键词:恶意软件家族分类;静态特征 RGB 图像;动态原型;对比学习;小样本学习
中图分类号:TN929.5;TP309 **文献标识码**:A **国家标准学科分类代码**:520.4070

Few-shot Android malware family classification based on dynamic prototypes and contrastive learning

Zhu Xuejin[1] Li Shenyang[1,2] Li Yan[1,2]
(1. School of Internet of Things Engineering, Wuxi University, Wuxi 214105, China; 2. School of Computer Science & School of Cyber Science and Engineeing, Nanjing University of Information Science & Technology, Nanjing 210044, China)

Abstract: To tackle low accuracy caused by limited generalization in few-shot Android malware family classification, this paper proposes SupProto, a dynamic prototype network driven by supervised contrastive learning. SupProto uses SupCon to refine the embedding space, improving inter-class separation and intra-class compactness, and adopts a dynamic prototype mechanism based on hierarchical clustering and silhouette coefficients to handle multimodal family structures. In terms of input and encoding design, RGB images are constructed from multi-source static features to provide unified and discriminative representations, while a DenseNet121 combined with a CBAM attention module strengthens feature extraction. Experiments on Drebin and CIC-InvesAndMal2019 show that SupProto achieves 90.59% and 85.64% accuracy in 5-way 5-shot settings, and 75.56% and 67.96% in 5-way 1-shot settings.

Keywords: malware family classification; static feature RGB images; dynamic prototypes; contrastive learning; few-shot learning

0 引 言

智能手机作为现代社会最广泛使用的终端设备,其普及程度已达到前所未有的高度。其中,Android 系统凭借其开源性,已成为市场份额超过 85% 的主流移动平台[1]。然而,这种开放的生态系统和巨大的市场占有率,也使其成为恶意软件攻击的重灾区。恶意软件的泛滥不仅会导致大量用户的个人隐私信息泄露,更有可能构成包括经济损失在内的多种现实威胁,对个人财产乃至社会安全都造成了严重影响[2]。

为了更有效应对 Android 恶意软件的持续威胁,单一样本检测已无法满足需求。恶意软件家族分类作为一种高级分析手段,可以将具有相似特征和行为的样本归为同一类别,帮助安全人员快速识别新变种、追踪传播路径并制定

收稿日期:2025-07-18
* 基金项目:异构工业物联网中恶意软件变体的传播机理及抑制策略研究项目(2024r005)资助

防御策略。近年来,基于机器学习的恶意软件家族分类方法得到了迅速发展,许多研究利用静态或动态特征、深度学习等技术,取得了较高的准确率,普遍超过了 90%,为恶意软件防御提供了重要支持[3-5]。然而,这些方法依赖大量标注样本,目前面临两个问题:一是恶意软件家族迭代频繁,旧模型难以泛化;二是新出现的家族样本稀缺,难以满足传统模型的数据需求。

因此,小样本学习(few-shot learning, FSL)已成为近年来的研究热点,其核心目标是在极少样本条件下保持良好的泛化能力。原型网络[6]、关系网络[7]、匹配网络[8]等方法在图像分类中表现出色,逐步被引入安卓恶意软件检测领域。然而,恶意软件家族通常具有多模态特性,同类样本差异较大,家族边界模糊,导致传统小样本方法难以生成鲁棒的家族原型,泛化能力受限[9]。此外,现有基于元学习的 FSL 方法假设训练与测试样本分布一致,而实际中新家族样本分布未知,进一步限制了泛化能力。

为了解决上述问题,本文提出了一种监督对比学习驱动的动态原型网络框架(supervised contrastive learning-driven dynamic prototype network,SupProto),针对安卓恶意软件家族分类中的小样本、结构复杂和变异频繁等挑战,主要做了以下几方面的工作:

1)本文提出了一种基于静态文件结构的 RGB 图像建模方法,通过提取 Classes.dex 字节流、AndroidManifest.xml 配置文件字节流以及 Classes.dex 文件局部熵特征,生成恶意软件样本的三通道图像,避免了反编译操作,为后续模型训练提供统一、判别性强的输入表达。

2)本文引入基于 DenseNet-121 的主干网络,并结合 CBAM 注意力机制增强模型对恶意软件图像关键特征的感知能力,提升了特征表示的判别性。

3)本文结合监督对比学习策略(supervised contrastive learning, SupCon),通过构造正负样本对,优化了嵌入空间的结构,确保了类间分离和类内紧致性。

4)本文基于层次聚类和轮廓系数优化,提出了自适应的动态原型生成方法,解决了同一恶意软件家族可能包含多个迭代版本导致的类内差异过大的问题,从而避免了传统原型分类方法在多模态类别中的表达不足。

1 相关工作

1.1 基于可视化的恶意软件检测

在恶意软件检测早期,静态分析方法通常依赖人工提取高维特征,工作量大且难以应对快速演化的恶意软件家族。随着深度学习的发展,代码图像化技术通过将恶意代码转化为图像输入,减少了人工干预并提升了检测精度。代表性方法如 Guo 等[10]提出的 MVVDroid,通过将.dex、.xml 和 .so 文件映射为 RGB 图像,并结合 API 图和操作码序列构建多视图输入,在分类效果上取得了一定进展,然而未考虑 .so 文件缺失的情况,可能引入噪声,影响鲁棒性。秦海雪等[11]通过将 .dex 文件转化为三通道图像,结合空间注意力机制提升特征提取效果,然而图像构建缺乏语义关联建模,且固定尺寸输入可能导致信息丢失或特征扭曲。李默等[12]则通过将.dex、.xml 和 .jar 文件合成 RGB 图像,取得了较好结果,然而 .jar 文件与 .dex 存在语义重叠,造成通道冗余。

综上,尽管基于图像可视化的恶意软件检测方法在特征提取和分类性能上取得了显著进展,但仍面临通道冗余、信息丢失等问题。更重要的是,现有方法普遍依赖大规模标注数据,难以应对恶意软件家族数量众多、新种频发、样本稀缺等复杂情境。因此,本文聚焦于在小样本条件下提升恶意软件家族分类的准确性与鲁棒性。

1.2 小样本场景下的恶意软件分类

传统监督学习方法依赖大量标注样本,在面对"冷启动"家族时,模型难以泛化,检测性能下降。为解决这一问题,近年来研究者尝试将小样本学习引入恶意软件检测任务。Ale 等[13]提出的 FSMC 方法通过 API 调用序列结合 Word2Vec 和 Matching Network 实现了 Android 恶意软件的 Few-shot 分类,但其仅基于单一的 API 序列,缺乏多模态静态特征的融合与利用,表达能力有限。Bai 等[14]提出基于孪生网络的嵌入空间方法,通过度量学习优化类间距离,提升了部分小样本任务的分类性能,但在新家族的泛化能力上仍存在局限。Chai 等[15]提出的 DPNSA 方法引入动态卷积网络和双样本激活机制,通过原型网络提升了对少量样本的适应能力,但其灰度图输入和单一特征类型在样本语义多样性较高时,泛化能力较弱。

综上,现有小样本恶意软件检测方法在一定程度上缓解了样本稀缺问题,但仍面临泛化能力不足和模型稳定性差等挑战,难以应对恶意软件家族复杂多变的分布特性。为此,本文基于现有研究,提出一种更具鲁棒性和适应性的小样本安卓恶意软件家族分类框架 SupProto。

2 SupProto 框架

SupProto 框架的主要思想通过监督对比学习训练嵌入网络,并结合动态原型生成方法,提升安卓恶意软件在小样本条件下的分类能力。框架总体结构图如图 1 所示,首先,数据集经过数据可视化模块,从每个 APK 文件提取 Classes.dex 文件、AndroidManifest.xml 文件以及 Classes.dex 文件的局部熵信息,并映射到 RGB 三个通道,生成特征图像。完成数据可视化后,数据集根据每个类别实际样本数量分为三个不相交部分:Base、Val 和 Test。

对于 Val 和 Test 部分,采用 N-way K-shot 方法进行测试。每个任务随机抽取 N 个类别每类别选取 K 个样本作为支持集 S_i,剩余样本作为查询集 Q_i,并合并成一个任务单元。这样,网络通过支持集和查询集结合进行验证与测试,评估新样本的泛化能力。在训练阶段,SupProto 通过监督对比学习优化嵌入网络,增强同类样本间的一致性,

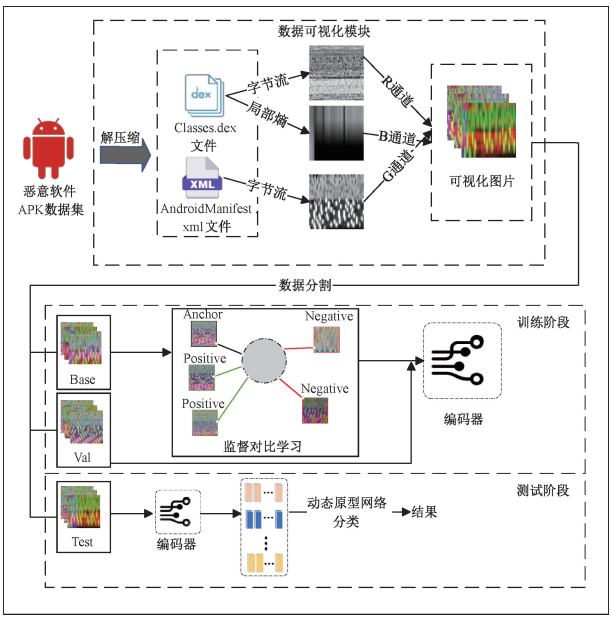


图 1　SupProto 框架总体结构图

Fig. 1　The architecture of the SupProto framework

并拉大不同类别之间的差异。训练过程中，嵌入网络不断优化，提升泛化能力。同时，使用划分好的 Val 子集进行 episodic 任务测试，与测试阶段一致，通过支持集和查询集组合评估模型在新类别上的泛化能力，并选取性能最优的编码器参数。

在测试阶段，训练好的特征编码器将样本转化为高质量特征表示，并输入到动态原型网络进行分类。动态原型网络通过层次聚类和轮廓系数法选择最优聚类数，并生成簇中心。查询样本根据与簇中心的距离，融合簇中心生成代表性原型，最终选择最近的原型进行分类。

2.1　数据可视化

为实现对 APK 文件的结构信息和潜在恶意特征的全面表达，本文提出一种无需反编译的 APK 可视化方法，综合利用 Classes.dex 字节码、AndroidManifest.xml 配置文件以及基于 Classes.dex 的局部熵信息，将三者分别映射至 RGB 图像的 3 个通道，生成具有判别能力的彩色图像用于训练图像识别模型。

对于 R 通道和 G 通道，本方法分别从 APK 文件中提取了 Classes.dex 字节码和 AndroidManifest.xml 文件的数据。首先依据数据长度确定矩阵宽度。其中 Classes.dex 文件因其长度变化大(从几 KB 到 4 MB 左右不等)，本方法采用动态自适应策略，通过文件字节数估算最合适的矩阵宽度 w：

$$w=\begin{cases}256, & n<512\,\mathrm{KB}\\ 512, & 512\,\mathrm{KB}\leqslant n<2\,048\,\mathrm{KB}\\ 1\,024, & 2\,048\,\mathrm{KB}\leqslant n\end{cases}\tag{1}$$

需要说明的是，该策略并非为并非还原文件尺寸本身，而是在图像构建阶段保持结构感与形态平衡，避免因比例失衡带来的语义扰动，从而增强后续插值过程中的图像表达质量与模型感知能力。具体宽度 w 设定参考了图像处理中常用的标准分辨率，并结合 Classes.dex 文件的大小分布，划分合理区间，以兼顾结构完整性与处理一致性。

对于 AndroidManifest.xml 文件，其长度通常在 1～32 KB 区间，变化范围较小，所以统一采用 $w=256$ 固定宽度构造矩阵，以简化结构并提高通道间的一致性。随后，对所有输入字节流按所选宽度填充并重构为二维矩阵，填充操作根据其余数 p 进行补零处理：

$$p=w-(n \bmod w)\tag{2}$$

得到的字节矩阵随后通过双线性插值缩放至固定尺寸 128×128。对于目标像素 (x,y)，双线性插值公式为：

$$\boldsymbol{I}_{\mathrm{resized}}(x,y)=(1-\alpha)((1-\beta)\boldsymbol{I}(x_1,y_1)+\beta\boldsymbol{I}(x_1,y_2))+\alpha(1-\beta)\boldsymbol{I}(x_2,y_1)+\beta\boldsymbol{I}(x_2,y_2))\tag{3}$$

其中，$\alpha=x-\mathrm{floor}(x)$，$\beta=y-\mathrm{floor}(y)$ 为目标像素 (x,y) 相对于其周围 4 个邻近像素的相对偏移。最后，对生成的图像应用 *CLAHE* 和 *Gamma* 校正，提升图像的对比度和亮度，得到最终的红色和绿色通道图像。

对于 B 通道，用于刻画 Classes.dex 文件在空间上的复杂性特征，以增强模型对代码结构扰动(如混淆、加壳等)的鲁棒性。本方法采用滑动窗口方法提取字节级局部熵特征，设定窗口大小为 256 字节，步长为 128 字节，对字节流进行滑动计算。对每个窗口 S 计算其香农熵 $H(S)$：

$$H(S)=-\sum_{i=0}^{255}p_i\log_2(p_i),\ p_i=\frac{n_i}{256}\tag{4}$$

其中，n_i 表示滑动窗口中字节值为 i 的字节数量，p_i 是该值的概率分布。所得熵序列的长度视 Classes.dex 文件大小而定，通常集中在 2 000～3 000 个熵值点，为统一处理流程，同样固定采用宽度 $w=256$ 将其重构为二维矩阵 $\boldsymbol{H}(i,j)$ 同时进行零填充，并将熵值归一化至 [0,255]：

$$\boldsymbol{I}(i,j)=\frac{\boldsymbol{H}(i,j)}{8}\times 255\tag{5}$$

随后，B 通道图像与 R，G 通道一样，最后使用双线性插值方法缩放至 128×128 的统一尺寸，以确保三通道图像在空间对齐上的一致性。数据可视化算法具体流程如算法 1 所示。

不同恶意软件家族的图像样本如图 2 所示。如图 2(a)所示，同一家族内部图像具有一致的纹理与结构特征；而如图 2(b)与(c)所示，不同家族之间差异显著，从而验证了本文可视化方法的有效性。

2.2　编码器结构设计

1) 编码器总体结构

为提升特征提取能力并增强对关键区域的关注，本文提出的编码器结合了 DenseNet-121 网络[16]和卷积块注意力模块 (convolutional block attention module，CBAM)[17]。该编码器整体结构如图 3 所示，主要由三部分组成：(1) DenseNet-121 主干网络，用于高效特征提取；

算法 1　APK 数据可视化算法

输入：APK 文件路径 *apk_path*

输出：RGB 图像 *img_rgb*，尺寸 128×128×3

初始化：滑动窗口 ***window***←[] 步长 *stride* ←128 熵序列 ***entropy_seq***←[] 中间矩阵 $\boldsymbol{W}_1$ $\boldsymbol{W}_2$ $\boldsymbol{W}_3$←[],[],[]

1：提取 Classes.dex 和 AndroidManifest.xml 文件字节流分别为 D 和 X

2：**for** all$F\in\{D,X\}$**do**

3：　**if** $F==D$ **then**

4：　　n←*len*(F)

5：　　w ← *Adaptive_width*(n)

6：　　$\boldsymbol{W}_1$← *Reshape_pad*(F,w)

7：　　R 通道←*Enhance* (*BilinearResize* ($\boldsymbol{W}_1$, 128×128))

8：　　**for** i=0 to$n$− 256 **step** *stride* **do**：

9：　　***window***← F [i : i + 256]

10：　　p← 计算 ***window*** 中字节的频率分布

11：　　H← 根据频率分布计算熵值

12：　　将 H 加入 ***entropy_seq***

13：　　**end for**

14：　　$\boldsymbol{W}_2$←*Normalization*(*Reshape_pad*(***entropy_seq***,256))

15：　B 通道←*BilinearResize* ($\boldsymbol{W}_2$,128×128)

16：　**if** $F==X$ **then**

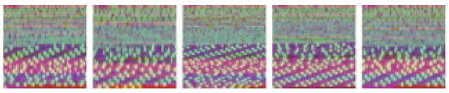
17：　　$\boldsymbol{W}_3$← *Reshape_pad*(F,256)

18：　G 通道 ← *Enhance*(*BilinearResize* ($\boldsymbol{W}_3$,128×128))

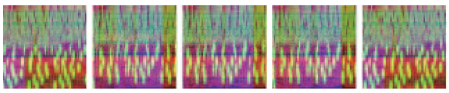
19：**end for**

20：合并三个通道 R，G，B，返回 *img_rgb*

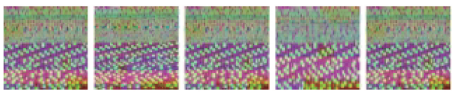
(2) CBAM 注意力机制模块，用于强化特征图中的关键区域；(3) 特征投影层(Projection Head)，将增强后的特征映射到低维嵌入空间并进行归一化处理。



(a) Plankton家族
(a) The Plankton family



(b) Mobiletx家族
(b) The Mobiletx family



(c) Fakedoc家族
(c) The Fakedoc family

图 2　不同恶意软件家族 RGB 图像样例对比

Fig. 2　RGB image samples of different malware families

2) DenseNet 主干网络

近年来，DenseNet 在图像识别领域展现了显著优势，

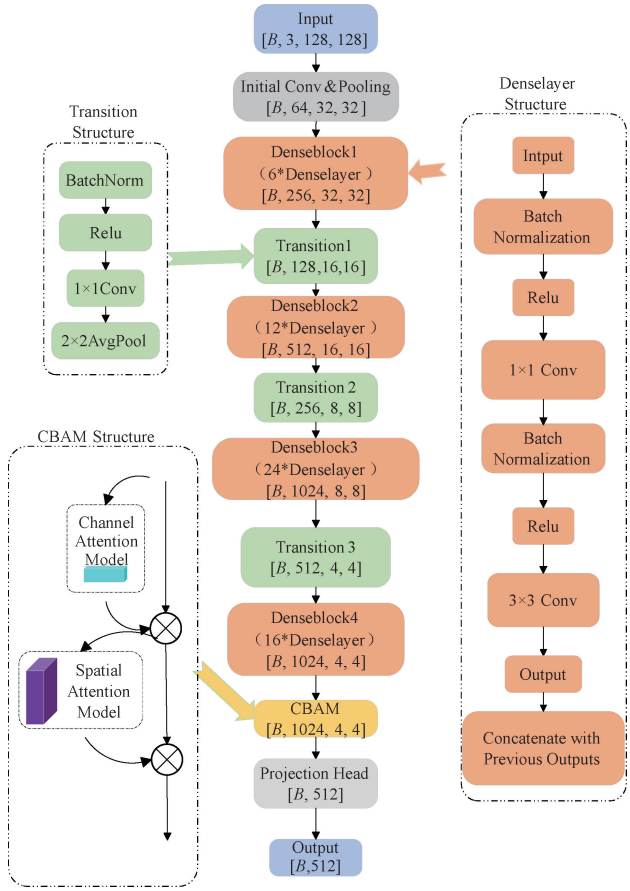


图 3　编码器结构图

Fig. 3　Encoder architecture

尤其在缺陷识别[18]和病理图像识别[19]中表现突出。其稠密连接结构通过将每一层的输出与后续所有层的输入连接，有效复用特征并增强信息传递。在小样本学习任务中，DenseNet 通过这种结构缓解了梯度消失问题，并提升了语义建模能力。特别是在恶意软件家族分类任务中，DenseNet 能够高效地学习和提取复杂特征，有助于提高分类性能和泛化能力。

3) CBAM 注意力机制

CBAM 是一种在计算机视觉领域被广泛验证的高效模块，例如在医学图像的病理分类[20]和管道的焊缝缺陷检测[21]中均取得了良好效果。受其在视觉任务中捕获关键特征能力的启发，本研究尝试将其引入到恶意软件分类领域，旨在利用其通道和空间注意力来增强对恶意软件的表征能力。如图 4 与 5 所示，该机制通过结合通道注意力(channel attention module，CAM)和空间注意力(spatial attention module，SAM)两个子模块分别从通道和空间两个维度引导模型关注更具判别性的区域特征。CAM 模块通过全局平均池化和最大池化提取通道级别的全局信息，并通过 MLP 网络进行加权融合，生成通道注意力图；SAM 模块则在空间维度上执行池化操作并通过卷积层生成空间注意力图。最终，两个注意力图与原始特征图逐通

道和逐像素相乘，强化了重要区域的特征，抑制了冗余背景信息，进而提升了特征表达的判别性。这使得模型在处理复杂恶意软件变种时，能够更好地聚焦关键区域，提高鲁棒性。

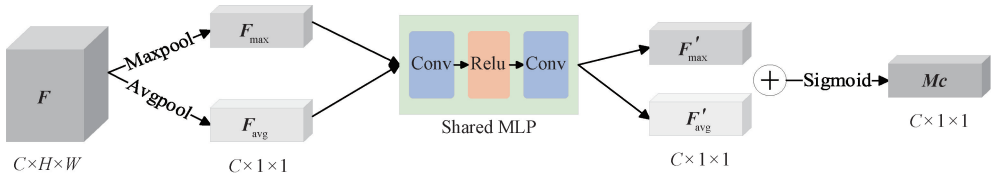


图 4　通道注意力结构图

Fig. 4　Channel attention architecture

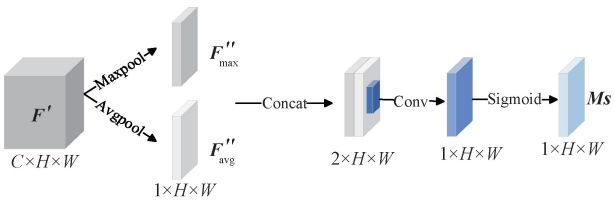


图 5　空间注意力结构图

Fig. 5　Spatial attention architecture

2.3　监督对比学习策略

在小样本场景下，恶意软件图像分类面临提取器泛化能力不足的问题。为增强模型对同类样本的一致性建模能力，本文引入监督对比学习[22]作为嵌入网络的训练策略，以显式增强模型对同类样本的一致性建模能力。该方法能够在嵌入空间中拉近同类样本之间的距离，同时有效区分异类样本，从而构建更具判别性和结构性的特征表示空间。

如图 6 所示，训练过程中，首先对每个原始图像样本应用两种不同的数据增强方式（几何增强与颜色增强），以构造两个视图作为正样本对。经过增强后，一个 batch 中共包含 N 个原始样本，将被扩展为 $2N$ 个增强图像实例。所有图像被输入编码器中提取特征向量，得到一组嵌入表示$\{\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_{2N}\}$其中同一类样本在颜色和结构上具有一致性。

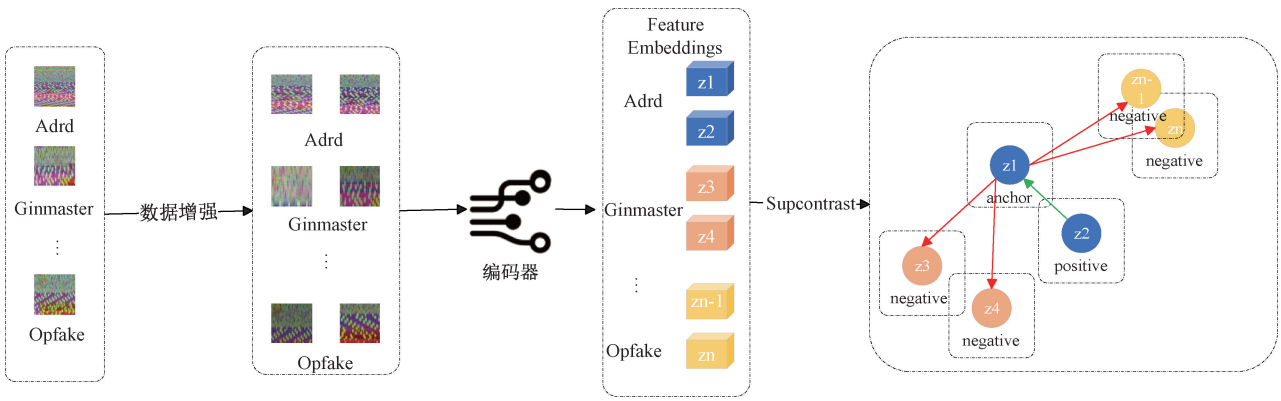


图 6　监督对比学习总体流程图

Fig. 6　Supervised contrastive learning flowchart

对于每一个嵌入特征 $\boldsymbol{z}_i$，被视为 anchor，监督对比损失函数会寻找与其标签相同的其他样本作为正样本集合 $P(i)$，其余不同类样本作为负样本构成对比集合 $A(i)$。训练目标是在保持 anchor 与正样本距离尽可能接近的同时，将其与所有负样本之间的距离最大化，从而增强类间区分度。其损失函数定义如下：

$$\mathcal{L}=\frac{1}{|I|}\sum_{i\in I}\frac{-1}{|P(i)|}\sum_{p\in P(i)}\log\frac{\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_p/\tau)}{\sum_{a\in A(i)}\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_a/\tau)} \tag{6}$$

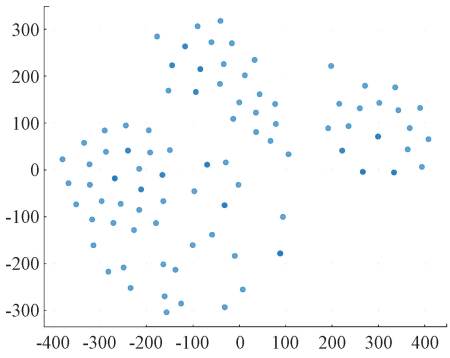
其中，I 是 batch 中所有 anchor 的索引集合，τ 是温度参数，用于调节对相似度的敏感程度。

2.4　动态原型分类

在小样本恶意软件家族检测任务中，同一恶意软件家族常包含多个迭代版本，若直接将支持样本简单平均构建原型，容易因类内差异过大导致原型失真。为解决这一问题，本文提出了一种基于层次聚类与轮廓系数的动态原型融合机制。该机制通过类内聚类划分家族内潜在子结构，并利用轮廓系数（Silhouette Score）评估不同聚类层级，从而自动选择最优划分方式。此策略能够更真实地反映恶意软件家族在演化过程中的结构差异，确保生成的聚类子簇更具代表性。接着，模型根据查询样本与各子簇中心的距离，采用 softmax 加权融合生成动态原型表示，进一步增强了分类器在复杂变种下的适应能力与鲁棒性。

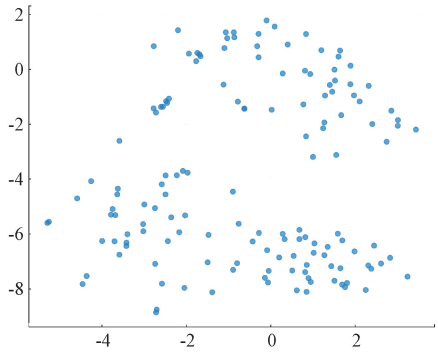
为探究家族内部的真实分布特性,本文利用预训练的编码器,对数据集中主要家族的特征分布进行了抽样和 t-SNE 降维可视化[23]。分析结果揭示了一个普遍现象:即多数家族内部并非单一的紧凑簇,而是存在明显的多模态结构。为直观地展示这一发现,本文以图 7(a) Opfake 家族和图 7(b) Plankton 家族为例,其内部便分别呈现出三个和两个独立的子簇。考虑到恶意软件家族的演化路径通常较为有限,并为覆盖这些观察到的以及未来可能出现的更复杂的变体,本文将最大聚类数设为 4,以确保原型划分覆盖常见演化情况,同时避免在支持样本较少(如 5-shot)时因过度划分导致聚类退化。为获得有效的原型划分,本文采用自底向上的层次聚类策略,如图 8 所示,以每个样本作为独立簇,通过 Ward 链接准则计算簇间距离,逐步合并距离最小的簇,构建聚类树。在不同剪枝层级(如 $k=2,3,4$)上进行切割,并通过轮廓系数评估每种划分的聚类质量,选择最佳聚类方案。轮廓系数可衡量簇内紧致性与簇间可分性,定义如下:

$$s=\frac{1}{T}\sum_{j=1}^{T}\frac{b_j-a_j}{\max(a_j,b_j)} \tag{7}$$



(a) Opfake 家族内部特征分布图

(a) Opfake intra-family feature distribution plot



(b) Plankton 家族内部特征分布图

(b) Plankton intra-family feature distribution plot

图 7　代表性家族内部特征分布图

Fig. 7　Intra-family feature distribution of representative families

其中,a_j 表示第 j 个样本与其所在簇内其他样本的平均距离(即类内紧致性),b_j 表示该样本与最近的其他簇中所有样本的平均距离(即类间分离度),T 为当前类别中支持样本的总数。该指标的值范围为 $[-1,1]$,值越大表示当前聚类结构越合理,既具备良好的类内紧密性,又具备足够的类间可分性。本文采用轮廓系数最高的一次聚类结果作为最终划分方案,并将该次聚类得到的所有簇中心表示为:

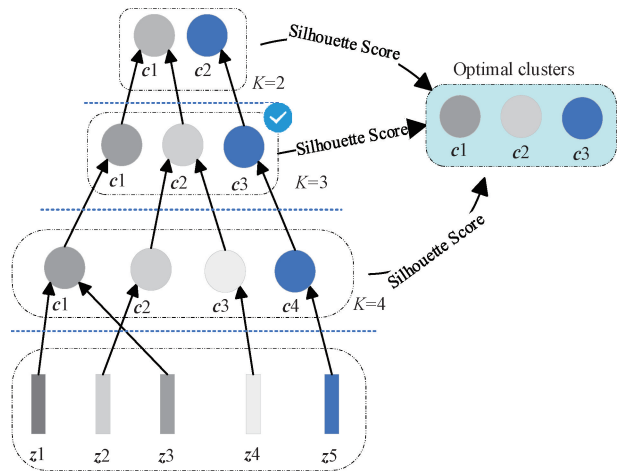


图 8　层次聚类流程示意图

Fig. 8　Hierarchical clustering process

$$\mathcal{C}_i=\{\boldsymbol{c}_i^{(j)}\}_{j=1}^{t_i} \tag{8}$$

其中 $\boldsymbol{c}_i^{(j)}$ 表示家族 i 中第 j 个子簇的中心向量,t_i 为该类的最终聚类数。该集合 $\mathcal{C}_i$ 表征了家族 i 在支持集中所呈现的潜在子结构分布。在推理阶段,针对任意查询样本 $\boldsymbol{q}_m$,首先需计算其与各子簇中心之间的距离 d_j:

$$d_j=\| f(\boldsymbol{q}_m)-\boldsymbol{c}_i^{(j)} \|^2 \tag{9}$$

然后通过 softmax 函数获得每个中心 $\boldsymbol{c}_i^{(j)}$ 对该查询样本的响应权重 α_j:

$$\alpha_j=\frac{\exp(-d_j/\tau)}{\sum_{k=1}^{t_i}\exp(-d_k/\tau)} \tag{10}$$

其中,τ 为温度参数,用于调节距离的敏感度,最终,查询样本 $\boldsymbol{q}_m$ 在第 i 类下的动态原型 $\boldsymbol{r}_i^m$ 表示为:

$$\boldsymbol{r}_i^m=\sum_{j=1}^{t_i}\alpha_j\cdot\boldsymbol{c}_i^{(j)} \tag{11}$$

模型将该样本分配至与其动态原型距离最小的类别:

$$y(\boldsymbol{q}_m)=\operatorname{argmin}_i\| f(\boldsymbol{q}_m)-\boldsymbol{r}_i^m \|^2 \tag{12}$$

3　实验结果与分析

3.1　数据集与实验设计

1) 安卓恶意软件数据集

本文使用安卓恶意软件检测领域中两个广泛使用的公开数据集构建实验数据,分别为 Drebin 数据集[24]和 CIC-InvesAndMal2019 数据集[25]。Drebin 数据集共包含 5 560 个恶意样本,覆盖 179 个恶意软件家族,其中每个家族的样本数量差异较大,存在大量长尾类。CIC-InvesAndMal2019 数据集包含 426 个恶意样本和 5 065 个

良性样本，恶意样本来自 42 个不同的家族，每个家族的样本数大致在 10 左右，分布较为均衡。

2）实验设置

本次实验均在本地工作环境下进行，具体软硬件配置如表 1 所示。

表 1　实验环境配置

Table 1　Experimental setup

实验环境	详细信息
操作系统	Windows 11
CPU	Intel i7-14700HX
内存	16 GB
GPU	NVIDIA GeForce RTX 4070 Laptop GPU
编程语言	Python 3.8
编程工具	PyCharm 2024.1.1
机器学习框架	PyTorch 2.4.1

3）评价指标

为全面评估本文所提出方法在少样本恶意软件家族分类任务中的性能，本文在实验中采用准确率(Accuracy)、精确率(Precision)、召回率(Recall)和 F1-score 等指标评估方法有效性。由于任务为多分类，本文采用宏平均(Macro-Averaging)计算各项指标，其计算公式具体如下：

$$Accuracy = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i + TN_i}{TP_i + TN_i + FN_i + FP_i} \tag{13}$$

$$Precision = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FP_i} \tag{14}$$

$$Recall = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i} \tag{15}$$

$$F1\text{-}score = \frac{1}{C}\sum_{i=1}^{C}\frac{2 \times Precision(i) \times Recall(i)}{Precision(i) + Recall(i)} \tag{16}$$

其中，TP_i 表示模型预测属于第 i 个家族且实际也属于第 i 家族的样本，TN_i 表示模型预测不属于第 i 个家族且实际也不属于第 i 个家族的样本，FP_i 表示模型预测属于第 i 个家族但实际并不属于该家族的样本，FN_i 表示模型预测不属于第 i 个家族但实际属于该家族的样本。

3.2　与其他模型实验对比

为全面验证所提出模型在小样本恶意软件家族分类任务中的性能，本文在 5-way 5-shot 与 5-way 1-shot 设置下，选取当前主流的少样本学习方法进行对比实验，并引入 FSMC[13] 作为安卓恶意软件领域内的代表性工作进行横向对比。对比方法涵盖以下 3 类：1）Fine-tuning 类方法：RFS[26]、SKD[27]；2）Meta-learning 方法：MAML[28]、MTL[29]、ANIL[30]；3）Metric-learning 方法：Prototypical Network[6]、Relation Network[7]、Matching Network[8]。

所有方法均在 Drebin 与 CIC-InvesAndMal2019 两个公开恶意软件数据集同时采用相同方法，并基于 1 000 个随机 episode 的平均准确率进行比较，实验结果汇总如表 2 所示。

表 2　不同模型实验准确率对比

Table 2　Accuracy comparison of different models　%

模型	Drebin		CIC-InvesAndMal2019	
	5way 5shot	5way 1shot	5way 5shot	5way 1shot
Prototypical Network[6]	80.70	69.38	79.01	66.62
Relation Network[7]	74.86	69.08	74.13	61.34
Matching Network[8]	78.97	66.70	76.38	65.02
MAML[28]	71.90	63.62	65.72	57.82
MTL[29]	80.56	64.13	77.06	59.52
Anil[30]	84.30	69.95	78.96	66.49
RFS[26]	86.63	73.83	76.12	63.80
SKD[27]	84.53	71.54	81.57	65.42
FSMC[13]	71.27	62.54	61.28	56.79
SupProto	**90.59**	**75.56**	**85.64**	**67.96**

实验结果显示，本文提出的 SupProto 模型在 Drebin 与 CIC-InvesAndMal2019 两个数据集上均取得了优于现有主流方法的分类性能。在 Drebin 数据集上，SupProto 在 5-way 5-shot 设置下的准确率为 90.59%，相比优化类方法、度量类方法以及元学习类方法平均提升了 5.01%、12.41%和 11.67%。在 5-way 1-shot 场景下，SupProto 也达到了 75.56% 的准确率，分别比上述 3 类方法最高者提高了 1.73%、6.18%和 5.61%。

CIC-InvesAndMal2019 数据集上，SupProto 在 5-shot 场景下准确率为 85.64%，分别比优化类方法、度量类方法和元学习类最高者提升了 4.07%、6.63%和 6.68%；而在 1-shot 任务中，仅达到 67.96%，与 ProtoNet 相比略高 1.34%，与 SKD 和 ANIL 等方法也差距较小。

值得注意的是，SupProto 在 1-shot 场景下的分类性能未能达到预期，尤其在 CIC 数据集上表现较弱。其原因在于 SupProto 的原型生成依赖于类内聚类建模来增强判别力，但在 1-shot 设置下无法进行聚类，策略退化为传统的 Prototypical Network，失去了结构性建模的优势。此外，CIC 数据集样本较少，限制了监督对比学习在特征空间的优化，导致模型在极小样本下难以学习到有效地区分特征，进而影响泛化能力。

3.3　不同可视化方法对比

为验证本文所提出的多源静态特征融合可视化方法在特征判别性上的优势，本文设计了一组综合性的对比实验，实验结果如图 9 所示。所有实验均在 Drebin 数据集上，采用本文所提出的 SupProto 编码器及完全相同的训练参数，唯一的变量是输入端的可视化图像生成方式。外部对比方面，本文选取了两种无需反编译的代表性可视化

方法作为基线:Dex-RGB[11]方法和 Markov[31]方法。内部消融方面,实现了两个简化版本以分析多源融合策略中每个组件的贡献,分别为仅使用 classes.dex 字节流生成的"Dex 单通道"灰度图,以及使用 classes.dex 和 AndroidManifest.xml 生成的"Dex+XML"双通道图像。

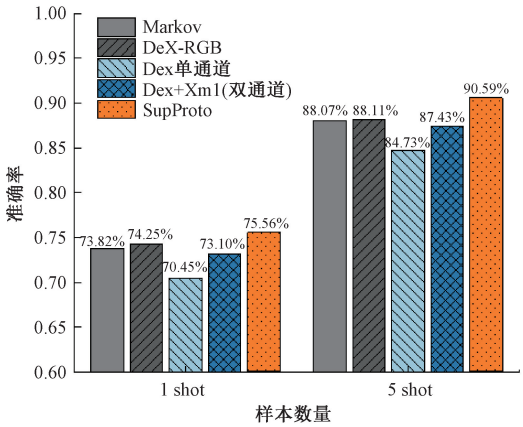


图 9　不同可视化方法在 1-shot 和 5-shot 设置下的性能对比

Fig. 9　Accuracy comparison of different visualization methods

在与外部基线的对比中,本文的完整方法表现出显著优势。在 5-shot 场景下,其准确率达到 90.59%,比次优的 Markov 方法高出 2.48%;即便在更具挑战性的 1-shot 场景下,也保持了 1.31% 的领先。这证明了本文方法生成的特征表示具有更强的判别性。

更重要的是,内部消融实验揭示了本文多源融合策略的内在价值。在 5-shot 设置下,从仅使用"Dex 单通道"(84.73%),到加入 XML 信息后的"Dex + XML"(87.43%),再到融合了 Dex 熵信息的完整三通道方法(90.59%),这一结果有力地证明了,本文所选的每一种信息源都为最终的性能增益做出了不可或缺的贡献,从而验证了本文可视化方法的合理性。

3.4　不同嵌入网络对比

为验证 SupProto 模型的特征嵌入网络有效性,本文选取了 ResNet18、MobileNetV3、EfficientNetB0 和 DenseNet121 作为对比模型。其中,ResNet18 和 MobileNetV3 为经典轻量化网络,EfficientNetB0 和 DenseNet121 为近年来在图像分类中表现优异的主流模型。此外,为了深入分析 SupProto 模型中注意力机制的协同效应,本文还基于其主干网络 DenseNet121 构建了两个变体模型:一个仅集成通道注意力模块(DenseNet121+CAM),另一个仅集成空间注意力模块(DenseNet121+SAM),并将它们纳入消融实验进行对比。

在 Drebin 数据集上,本文进行了 1 000 次测试以确保结果可靠。每个实验在 5-way 1-shot 和 5-way 5-shot 设置下,采用相同的训练和测试框架进行评估,评估结果如表 3 和表 4 所示。

表 3　5-way 1-shot 设置下不同编码器性能对比

Table 3　Performance comparison of different encoders under the 5-way 1-shot setting　%

模型	准确率	精确率	召回率	F1 值
Resenet18	73.06	75.04	73.06	71.02
MobileNetV3	72.78	73.97	72.78	70.27
EfficiententB0	72.97	74.48	72.97	71.11
Densenet121	73.15	75.51	73.15	71.13
Densenet121+CAM	72.12	73.81	72.12	69.80
Densenet121+SAM	71.57	73.18	71.57	69.36
SupProto	**75.56**	**77.35**	**75.56**	**73.41**

表 4　5-way 5-shot 设置下不同编码器性能对比

Table 4　Performance comparison of different encoders under the 5-way 5-shot setting　%

模型	准确率	精确率	召回率	F1 值
Resenet18	87.83	89.64	87.83	87.20
MobileNetV3	88.10	89.78	88.10	87.56
EfficientnetB0	88.06	89.76	88.06	87.53
Densenet121	88.17	90.02	88.17	87.71
Densenet121+CAM	88.53	90.27	88.53	88.04
Densenet121+SAM	87.78	89.45	87.78	87.24
SupProto	**90.59**	**92.02**	**90.59**	**90.22**

如表 3 所示,在 1-shot 设置下,在 DenseNet121 基线(73.15%)上单独集成 CAM 或 SAM 后,准确率反而分别下降至 72.12% 和 71.57%。这表明,在仅有一个支持样本的极端条件下,单一注意力机制所生成的权重并不可靠,容易因特征误判而损害性能。然而,集成了完整 CBAM 模块的 SupProto 模型则表现出更强的鲁棒性,准确率达到了 75.56%,相比基线提升了 2.41%,并显著高于 ResNet18 等其他对比模型约 2.5%,充分证明了其嵌入网络设计的有效性。

如表 4 所示,随着样本数增加到 5-shot,各模型性能均有显著提升,注意力模块的作用也变得更清晰:单独使用 CAM 带来了微小的性能增益(88.53%),表明模型已能初步识别"哪些"特征通道更重要;而 SAM 性能仍略低于基线(87.78%),说明学习"哪里"是关键空间位置的任务更为复杂,需要更多样本。与之形成鲜明对比的是,集成了完整 CBAM 模块的 SupProto 模型表现出强大的协同效应,准确率达到 90.59%,相比原始 DenseNet 显著提升了 2.42%。这有力地证明了 CBAM 通过"先筛选通道、再定位空间"的机制,实现了最有效的特征增强,并领先于其他所有对比模型。

3.5　动态聚类与固定层次 k 聚类的性能对比

为验证基于轮廓系数与层次聚类方案的有效性,本文对比了不同剪枝层级($k = 2,3,4$)下的固定簇数层次聚

类，对比结果如图 10 所示。具体来说，k 决定了最大簇数，当实际聚类数达到 k 时，选择 k 作为最终簇数。由于层次聚类是基于距离逐步合并簇，剪枝操作可能受数据结构影响，导致实际聚类数小于 k 值，此时聚类数将调整为实际值。为进一步评估聚类效果，另引入传统的原型网络方法作为对比基线，去分析聚类方案的提升效果。

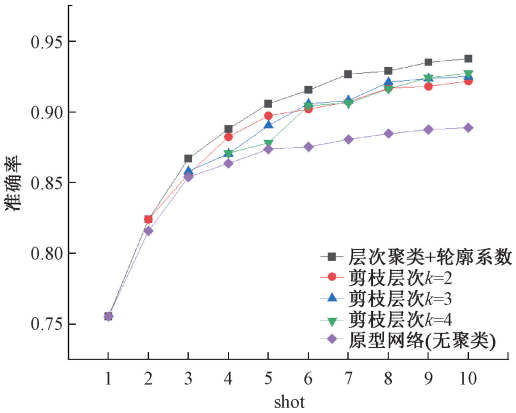


图 10　动态聚类与不同层次 k 聚类对比图

Fig. 10　Comparison of dynamic clustering and fixed-k hierarchical clustering

实验结果表明，结合层次聚类与轮廓系数的方案在 5-way 分类任务中，无论在不同 shot 数下，均保持了最优性能，验证了该方案的有效性。与传统原型网络方法相比，该方案在 5-way 分类任务中平均提高了约 3%的准确率，表现出显著的优势。与固定剪枝层级的层次聚类方法相比，尤其在 7-shot 场景下，本方案比最优的 $k=3$ 层次聚类方案提升了 1.86%，进一步证明了该方案在选择最佳聚类数时的灵活性和准确性。

总体来看，层次聚类与轮廓系数结合的方案在 5-way 分类任务中平均提升了 1.1%，在确保高准确率的同时，能够动态调整聚类簇数，适应不同数据结构，提供更稳定优越的性能。

3.6　不同数据增强方法对比

监督对比学习的性能在很大程度上依赖于数据增强所构造的正样本对的质量。为系统性地验证本文所采用的增强策略的有效性，并剖析其中不同组件的贡献，本文设计了一组消融实验，去测试 4 种不同的增强策略：

1）无增强：不进行任何随机数据增强，作为性能基准。

2）颜色增强：在“无增强”基础上，仅加入颜色变换，包括 ColorJitter(颜色抖动)和 RandomGrayscale(随机灰度化)。

3）几何增强：在“无增强”基础上，仅加入几何变换，包括随机裁剪缩(random resized crop)和随机水平翻转(random horizontal flip)。

4）完整策略(本文采用)：同时使用几何增强与颜色增强。

所有实验均在 Drebin 数据集上，采用完全相同的输入、模型、参数和训练流程进行，唯一的变量是训练时所用的数据增强策略，实验结果如图 11 所示。

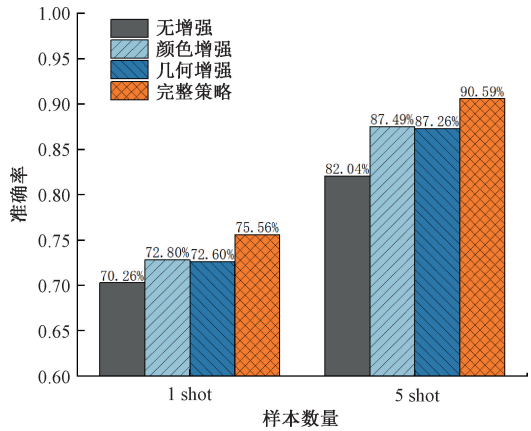


图 11　不同数据增强方法在 1-shot 和 5-shot 设置下的性能对比

Fig. 11　Comparison of different data augmentation strategies in 1-shot and 5-shot scenarios

该消融实验清晰地展示了数据增强策略在不同小样本设置下的影响。

在更具挑战性的 1-shot 场景下，所有数据增强策略均带来了性能提升。其中，“无增强”的基线准确率为 70.26%，在分别加入颜色增强和几何增强后，准确率提升至相似的水平(72.80% 和 72.60%)。而同时使用两种增强的完整策略则取得了 75.56% 的最佳性能，相比基线提升了 5.3%，证明了即便在单一样本条件下，复合增强策略也能有效提升模型的泛化能力。

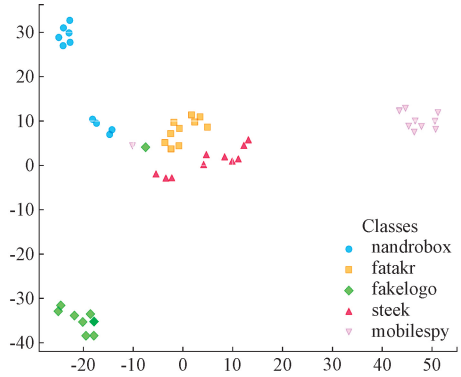
随着样本数量增加到 5-shot，数据增强带来的优势变得更为显著。此时，“无增强”基线的准确率为 82.04%，而“仅颜色增强”和“仅几何增强”策略均将其大幅提升至约 87%。最终，完整策略再次达到最优性能，准确率高达 90.59%，相比基线提升了 8.55%。这一结果表明，在样本量相对增多时，复合增强策略能更充分地挖掘数据潜力，从而获得了最佳的分类效果。

3.7　不同嵌入网络训练方法比较

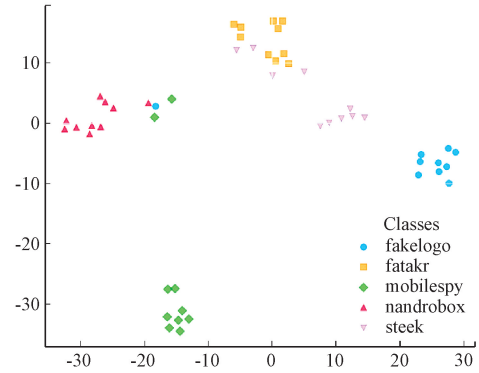
为研究不同训练方法对嵌入特征表示效果的影响，本实验选取了 3 种具有代表性的训练策略进行对比：基于 Softmax 的传统监督训练、SimCLR 无监督对比学习[32]、Episodic training 和本文的 SupCon 方法。Softmax 训练通过分类损失强调类别区分；SimCLR 采用数据增强配对与对比损失进行特征学习，不依赖标签信息；Episodic training 模拟 few-shot 场景，通过支持集与查询集进行任务级训练。

如图 12 所示，实验使用 Drebin 数据集的可视化图像，选取 5 个恶意软件家族(fakelogo、fatakr、mobilespy、nandrobox 和 steek)，从每个家族随机抽取 10 个样本，确

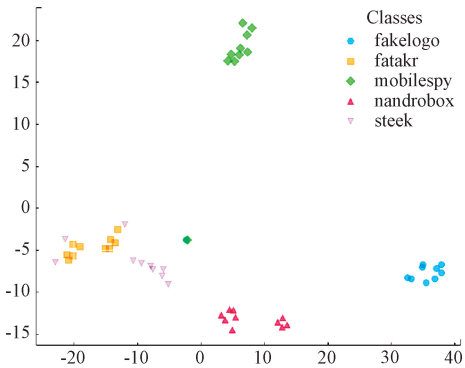
保类别均衡。所有实验均采用 ResNet18 作为特征编码器,以消除网络结构差异的影响。在完成高维特征提取后,使用 t-SNE 对嵌入向量进行二维降维可视化,直观展示不同训练方法对编码器表征能力的影响。



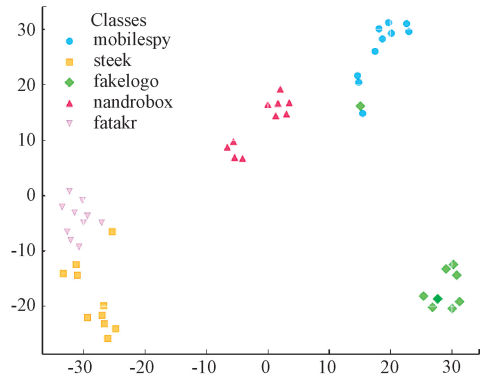
(a) SimCLR特征分布图
(a) SimCLR feature distribution plot



(b) Episodic training特征分布图
(b) Episodic training feature distribution plot



(c) SoftMax特征分布图
(c) SoftMax feature distribution plot



(d) SupCon特征分布图
(d) SupCon feature distribution plot

图 12　不同训练方法下的特征分布图

Fig. 12　Feature distribution plots from different training methods

如图 12(d)所示,SupCon 方法在嵌入空间中展现出最优的聚类结构:同类样本距离显著小于异类样本,各类簇结构紧凑、边界清晰。如图 12(a)所示,SimCLR 在无标签监督下也能实现较好的类内聚合,但由于缺乏标签语义约束,部分类别的类间分离度较差。如图 12(b)所示,Episodic training 通过支持集与查询集构建任务,间接优化特征结构,但由于未显式优化对比损失,其部分类别聚类效果较弱。相比之下,如图 12(c)所示,Softmax 分类器虽然能构建基本的类别判别边界,但未优化嵌入空间,导致类间边界模糊。综合来看,SupCon 作为嵌入网络的训练范式,符合任务需求并具有显著优势。

4　结　　论

针对小样本安卓恶意软件家族分类泛化性差导致分类准确率低的问题,本文提出了 SupProto 框架。该框架通过 RGB 图像建模融合多源静态特征,利用 DenseNet-121 网络与 CBAM 注意力机制增强关键特征的提取能力。其核心在于结合监督对比学习(SupCon)优化特征嵌入空间,并设计基于层次聚类与轮廓系数的动态原型生成机制以适应家族内部的多模态结构。

实验在 Drebin 和 CIC-InvesAndMal2019 数据集上验证了 SupProto 的有效性。结果表明,该框架在典型的 5-way 5-shot 和 5-way 1-shot 设置下,分类准确率显著优于多种主流小样本学习方法,有效提升了小样本条件下的恶意软件家族识别能力。然而,在 5-way 1-shot 等极低样本场景下,模型的鲁棒性和泛化能力仍有提升空间。未来工作将重点突破 1-shot 性能瓶颈,着力于开发更鲁棒的单样本表示与原型学习策略,并进一步引入对抗性训练与特征增强方法,以提升模型对极稀缺样本的适应能力。此外,考虑到本文使用的数据集时效性有限,后续工作还将致力于在一个更新、更具挑战性的数据集上对 SupProto 框架进行验证,以评估其在应对持续演化的恶意软件威胁时的真实效能。

参考文献

[1]　HOU Q, DIAO W, WANG Y, et al. Can we trust the phone vendors? Comprehensive security measurements on the Android firmware ecosystem[J]. IEEE Transactions on Software Engineering, 2023, 49(7): 3901-3921.

[2] KHAN S, YUSUF A, HAIDER M, et al. A review of Android and iOS operating system security[C]. 2022 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS), 2022: 67-72.

[3] AMENOVA S, TURAN C, ZHARKYNBEK D. Android malware classification by CNN-LSTM[C]. 2022 International Conference on Smart Information Systems and Technologies(SIST), 2022: 1-4.

[4] DUY P N, CAM N T. Multi-feature image analysis for android malware classification using convolutional neural networks [C]. 2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC), 2024: 1425-1430.

[5] NGUYEN C D, KHOA N H, DOAN K N D, et al. Android malware category and family classification using static analysis [C]. 2023 International Conference on Information Networking (ICOIN), 2023: 162-167.

[6] SNELL J, SWERSKY K, ZEMEL R. Prototypical networks for few-shot learning[J]. Advances in neural information processing systems, 2017, 30: 4077-4087.

[7] SUNG F, YANG Y, ZHANG L, et al. Learning to compare: Relation network for few-shot learning[C]. IEEE Conference on Computer Vision and Pattern Recognition, 2018: 1199-1208.

[8] VINYALS O, BLUNDELL C, LILLICRAP T, et al. Matching networks for one shot learning [J]. Advances in Neural Information Processing Systems, 2016, 29:3630-3638.

[9] ZHOU F, WANG D, XIONG Y, et al. FAMCF: A few-shot android malware family classification framework [J]. Computers & Security, 2024, 146: 104027.

[10] GUO J, MENG Z, ZHANG Q, et al. MVVDroid: Android malware detection based on multi-view visualization[C]. 2023 9th International Conference on Big Data Computing and Communications(BigCom), 2023: 96-102.

[11] 秦海雪,王勇.结合注意力与双线性网络的 Android 恶意软件检测[J].计算机工程与设计,2023,44(11):3290-3297.
QIN H X, WANG Y. Android malware detection combining attention and bilinear networks [J]. Computer Engineering and Design, 2023, 44(11): 3290-3297.

[12] 李默,芦天亮,谢子恒.基于代码图像合成的 Android 恶意软件家族分类方法[J].计算机应用,2022,42(5):1490-1499.
LI M, LU T L, XIE Z H. Android malware family classification method based on code image synthesis[J]. Journal of Computer Applications, 2022, 42(5): 1490-1499.

[13] ALE L, LI L, KAR D, et al. Few-shot learning to classify Android malwares [C]. 2020 IEEE 5th International Conference on Signal and Image Processing(ICSIP), 2020: 1001-1007.

[14] BAI Y, XING Z, LI X, et al. Unsuccessful story about few shot malware family classification and siamese network to the rescue[C]. ACM/IEEE 42nd International Conference on Software Engineering, 2020: 1560-1571.

[15] CHAI Y, DU L, QIU J, et al. Dynamic prototype network based on sample adaptation for few-shot malware detection [J]. IEEE Transactions on Knowledge and Data Engineering, 2022, 35(5): 4754-4766.

[16] HUANG G, LIU Z, LAURENS V D M, et al. Densely connected convolutional networks[C]. IEEE Conference on Computer Vision and Pattern Recognition, 2017: 4700-4708.

[17] WOO S, PARK J, LEE J Y, et al. Cbam: Convolutional block attention module[C]. European Conference on Computer Vision(ECCV), 2018: 3-19.

[18] 赵国威,曾静.基于 EMD-GAF 和改进的 SERE-DenseNet 的滚动轴承故障诊断方法[J].电子测量技术,2023,46(20):170-176.
ZHAO G W, ZENG J. A rolling bearing fault diagnosis method based on EMD-GAF and improved SERE-DenseNet [J]. Electronic Measurement Technology, 2023, 46(20): 170-176.

[19] 赵琪玉,张俊华,张剑青,等.基于双重注意力机制的间质性肺病高分辨率 CT 图像分类方法[J].国外电子测量技术,2024,43(6):1-11.
ZHAO Q Y, ZHANG J H, ZHANG J Q, et al. A classification method for interstitial lung disease HRCT images based on dual attention mechanism[J]. Foreign Electronic Measurement Technology, 2024, 43(6): 1-11.

[20] ZHUANG J, WU X, MENG D, et al. A swin transformer and residualnetwork combined model for breast cancer disease multi-classification using histopathological images [J]. Instrumentation, 2024, 11(1): 112-120.

[21] 凌晓,刘露,孙宝财,等. 基于 DSG-ResNet34 的聚乙烯燃气管道电熔焊接缺陷检测[J]. 仪器仪表学报,2025,46(6):228-240.
LING X, LIU L, SUN B C, et al. Defect detection of polyethylene gas pipeline electrofusion welding based on DSG-ResNet34[J]. Chinese Journal of Scientific Instrument, 2025, 46(6): 228-240.

[22] KHOSLA P, TETERWAK P, WANG C, et al. Supervised contrastive learning[J]. Advances in Neural Information Processing Systems, 2020, 33: 18661-18673.

[23] KANG B, GARCIA D G, LIJFFIJT J, et al. Conditional t-SNE: More informative t-SNE embeddings[J]. Machine Learning, 2021, 110: 2905-2940.

[24] ARP D, SPREITZENBARTH M, HUBNER M, et al. Drebin: Effective and explainable detection of android malware in your pocket[C]. The Network and Distributed System Security Symposium, 2014, 14(1): 23-26.

[25] TAHERI L, KADIR A F A, LASHKARI A H. Extensible android malware detection and family classification using network-flows and API-calls[C]. 2019 International Carnahan Conference on Security Technology(ICCST). IEEE, 2019: 1-8.

[26] TIAN Y, WANG Y, KRISHNAN D, et al. Rethinking few-shot image classification: A good embedding is all you need? [C]. Computer Vision-ECCV 2020: 16th European Conference, 2020: 266-282.

[27] RAJASEGARAN J, KHAN S, HAYAT M, et al. Self-supervised knowledge distillation for few-shot learning[J]. ArXiv preprint arXiv:2006.09785, 2020.

[28] FINN C, ABBEEL P, LEVINE S. Model-agnostic meta-learning for fast adaptation of deep networks[C]. International Conference on Machine Learning. PMLR, 2017: 1126-1135.

[29] SUN Q, LIU Y, CHUA T S, et al. Meta-transfer learning for few-shot learning[C]. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019: 403-412.

[30] RAGHU A, RAGHU M, BENGIO S, et al. Rapid learning or feature reuse? Towards understanding the effectiveness of MAML[J]. ArXiv preprint arXiv: 1909.09157, 2019.

[31] 孙世淼,刘亚姝,严寒冰. 基于多字节频率域可视化和深度学习的恶意软件检测[J]. 计算机工程与设计,2024,45(8):2272-2280.
SUN SH M, LIU Y SH, YAN H B. Malware detection based on multi-byte frequency domain visualization and deep learning[J]. Computer Engineering and Design, 2024, 45(8): 2272-2280.

[32] CHEN T, KORNBLITH S, NOROUZI M, et al. A simple framework for contrastive learning of visual representations[C]. International Conference on Machine Learning. PMLR, 2020: 1597-1607.

作者简介

朱雪金,博士,讲师,主要研究方向为恶意软件分析与检测、深度学习。

E-mail:zxj@cwxu.edu.cn

李沈洋,硕士研究生,主要研究方向为计算机视觉,恶意软件检测。

E-mail:lishenyang@nuist.edu.cn

李燕(通信作者),博士,教授,主要研究方向为机器学习、计算机视觉。

E-mail:002200@nuist.edu.cn