基于软件基因的恶意代码检测与分类方法

何　源[1,2]
HE Yuan

摘　要　随着恶意代码数量和复杂性的不断增加，现有检测方法在应对变种和未知恶意代码方面面临重大挑战。传统的基于特征和行为的检测方法由于对已知特征的依赖和高计算资源的需求，难以满足当前的安全需求。为了解决这些问题，文章提出了一种基于软件基因的恶意代码检测方法。该方法通过从代码中提取独特的基因片段，构建一个全面的恶意代码基因库，并利用 Siamese 网络进行相似性分析，来检测未知和变种的恶意代码。该方法优化了基因库的结构，定义了清晰的基因规则，从而提高检测的准确性和效率。实验结果表明，该方法在公开数据集上的准确率达到 97.8%，不仅在准确性上优于传统方法，还显著降低了计算资源的消耗，为恶意代码检测和家族分类提供了一种高效可靠的解决方案。

关键词　恶意代码分类；软件基因；PowerShell；相似性分析；Siamese Net

doi：10.3969/j.issn.1672-9528.2024.12.037

0 引言

近年来，随着互联网和信息技术的迅猛发展，恶意代码变得越来越复杂和隐蔽。恶意代码不仅种类繁多，且变种频繁，呈现出高度多样性和快速演化的趋势。从早期的病毒和蠕虫到如今的高级持续威胁和勒索软件，恶意代码的发展已经超越了传统防御手段的检测能力。这些恶意代码通常利用社会工程、漏洞利用和混淆技术等手段来逃避传统检测方法的识别，从而对信息安全构成重大挑战。

现有的恶意代码检测方法主要分为基于特征的检测、基于行为的检测和基于机器学习的检测。然而，这些方法各有其局限性。基于特征的检测方法依赖于预定义的特征库，通过匹配已知特征来识别恶意代码，这种方法对于未知或变种恶意代码的检测效果不佳；基于行为的检测方法通过监控代码的运行行为来识别异常，但这种方法需要大量的计算资源，且容易受到误报和漏报的影响；基于机器学习的检测方法近年来得到了广泛关注，虽然其在检测未知恶意代码方面表现出色，但其效果依赖于训练数据的质量和数量，同时，复杂的模型也增加了系统的计算负担和实现难度。

为应对上述挑战，本文提出了一种基于软件基因的恶意代码检测方法（software gene-based malware detection，SGMD）。软件基因是一种从代码中提取的特征片段，能够有效地捕捉恶意代码的本质特征。与传统方法不同，基于软件基因的方法通过构建恶意代码基因库，实现对代码片段的精确匹配和相似性分析，从而能够检测出未知和变种的恶意代码。此外，通过优化基因库和定义清晰的基因规则，本文方法在保证检测准确性的同时，显著降低了计算资源的消耗。图 1 展示了本文提出的基于软件基因的恶意代码检测方法的整体架构。该方法通过基因库构建和高效的特征提取、相似性计算模块进行恶意代码检测，利用 Siamese[1] 网络结构来比较输入样本与基因库中的特征，进行相似性检测和恶意代码识别。

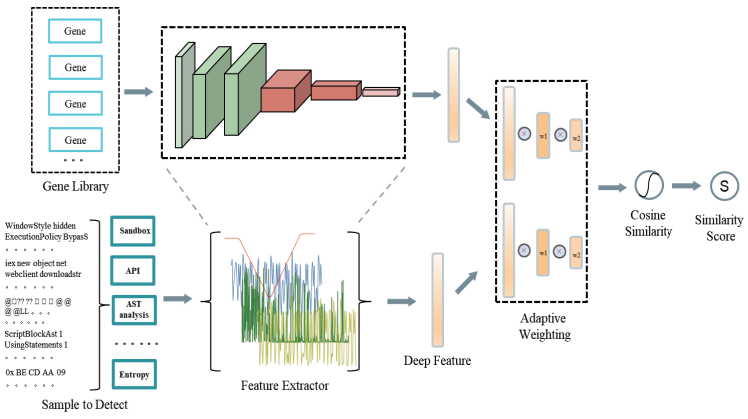

图 1　SGMD 方法结构图

1 相关工作

代码的恶意性检测研究是一个区分代码性质的二分类问题，即判定代码是良性还是恶性。这类研究通过分析代码

1. 湖北省水电工程智能视觉监测重点实验室 湖北宜昌 443002
2. 三峡大学计算机与信息学院 湖北宜昌 443002

的静态特征和动态行为，利用机器学习和深度学习技术建立分类模型。Hendler 等人 [2] 应用 n-gram 技术对命令文本进行预处理并使用四层卷积神经网络模型（convolutional neural network，CNN）对样本进行了恶意性检测；刘岳等人 [3] 提出了一种基于随机森林特征组合的方法，利用随机森林对原始特征进行非线性变化和特征组合，然后使用深度学习神经网络进行分类识别；Fang 等人 [4] 主要通过分析文本字符和抽象语法树（abstract syntax tree，AST）节点的差异，并利用随机森林分类器进行分类；Li 等人 [5] 基于 AST 设计一种基于子树的去混淆方法，并在此基础上创建了首个语义感知的 PowerShell 攻击检测系统。上述方法为之后研究恶意代码家族分类提供了思路，但这些研究主要集中在判定代码是否具有恶意行为，而在识别新型恶意代码方面仍然存在局限。

为了应对不断增长的恶意代码家族变种，研究人员开展了大量关于恶意软件家族识别和分类的工作。Rusak 等人 [6] 通过抽取样本 AST 的高度与节点数量作为家族分类的特征，利用随机森林分类器进行分类；高宇航 [7] 提出了一种基于双向门控循环网络（BiGRU）结合注意力机制的检测模型，利用 PowerShell 语义特征实现恶意代码的家族分类；Yang[8] 通过多模态语义融合和深度学习的应用，结合 Transformer 和 CNN-BiLSTM 的组合模型来实现恶意代码家族分类。尽管现有研究为恶意代码的检测与分类提供了多种解决方案，但许多研究没有充分考虑模型的计算效率和时间性能，这在处理大规模恶意代码数据集时尤为关键。

2 方法

2.1 基因提取

在恶意代码检测中，提取能够代表恶意代码特征的“基因”至关重要。恶意代码样本通常具有复杂和多样的特征，包括静态特征和动态特征。为了全面捕捉恶意代码的行为特征，首先对样本进行反编译和预处理，将其转换为可分析的中间表示，去除噪声和冗余信息。

在基因提取过程中，采用 CNN 模型对预处理后的代码片段进行特征提取，能够有效捕捉代码片段中的时序和空间特征。通过模型的编码层，将每个代码片段转化为高维特征向量，这些向量即为“基因”。这些基因特征向量不仅能够代表代码的局部细节特征，还可以捕捉到全局行为模式。随后，对这些特征向量进行降维处理以减少计算复杂度，并进行标准化处理，以确保不同样本之间的特征具有可比性。最终提取的基因被用于构建基因库，为后续的相似性检测和恶意代码识别提供支持。

2.2 基因库构建

在基因提取之后，需要定义基因库的构建规则，以有效组织和管理提取的基因特征。基因库的构建基于以下规则：基因库按照基因的相似性进行分组，确保同一组中的基因片段具有较高的特征相似性；定义每个基因的唯一标识符和相应的恶意代码家族标签，以便于后续的查询和分类。基因库还需动态更新，以便在新的恶意代码样本出现时及时扩充新的基因特征。基因库构建流程如图 2 所示。

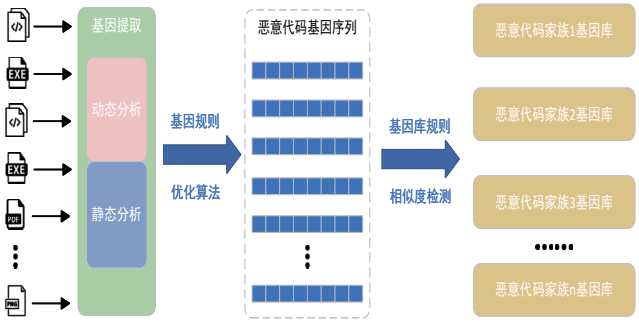

图 2 基因库构建流程

为实现高效的基因库构建和相似性搜索，本文选用 Annoy 向量数据库 [9]。Annoy 库是一种适用于高维向量的近似最近邻搜索工具，通过构建多个随机树来快速查找最相似的基因特征。选择 Annoy 库的优势在于其高效的查询速度和低内存消耗，而且 Annoy 库支持动态增量更新，能够灵活应对不断变化的恶意代码特征需求。

2.3 相似度检测

Siamese 网络是一种在相似性学习中广泛应用的深度学习架构，尤其适用于需要对输入对进行相似性度量的任务。该网络由两个共享权重的子网络组成，分别接收成对的输入，通过提取特征并使用相似性度量函数来计算输入对之间的相似性。通过这种结构，Siamese 网络能够有效学习到输入数据在高维特征空间中的相似性和差异性，为相似性检测提供了强大的支持。

在恶意代码检测方法中，采用 Siamese 网络结合自适应加权模块对提取的代码基因片段进行相似性对比，其结构如图 3 所示。

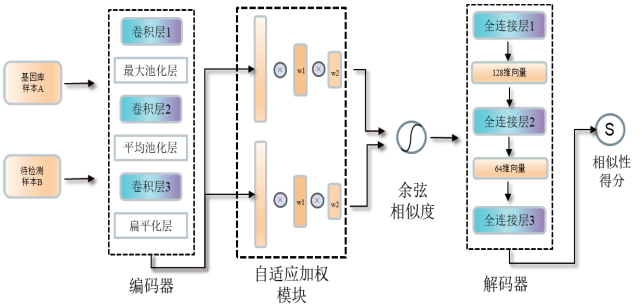

图 3 Siamese 网络结构图

Siamese 网络通过共享参数的双子网络确保了对两个输入片段的处理过程一致，减少了模型的偏差，从而提高了检

测的鲁棒性。通过相似性度量函数结合自适应加权机制，Siamese 网络能够精确判断基因片段在特征空间中的距离，并动态调整各特征的权重，这使得模型可以更有效识别未知和变种的恶意代码。 在本文方法中，为了实现有效的恶意代码相似性检测，采用了对比损失函数来优化 Siamese 网络，并通过自适应加权模块进一步增强了模型在不同特征权重下的灵活性和准确性。对比损失函数为：

$$L(W,Y,D)=\frac{1}{N}\sum_{i=1}^{N}\left[Y_i\cdot D_i^2+(1-Y_i)\cdot\max(M-D_i,0)^2\right] \quad (1)$$

式中：$L(W, Y, D)$ 是对比损失函数的值，反映了模型预测的相似性与实际标签之间的误差；N 是样本对的总数；Y_i 为标签变量；D_i 表示样本对在特征空间中的距离；M 是一个超参数，确保不同类别的样本对在特征空间中保持足够的距离。

3 实验

3.1 数据集和实验环境

本文通过采用的数据集为 White[10] 在 Github 上的开源数据集，该数据集中的样本已被分类不同的家族，去除该数据集中家族样本数量不到 10 个和被分为 Unknown 家族的样本，样本总计 3878 个，被分为 Downloader、Shellcode Inject、Unicorn、Powerfun、SET 和 PowerShell Empire 六个恶意代码家族，各个家族的样本数量分布如图 4 所示，由于各家族样本数量分布不均，所以在划分数据集时，从每个家族中按照一定比例取出一部分作为测试集。

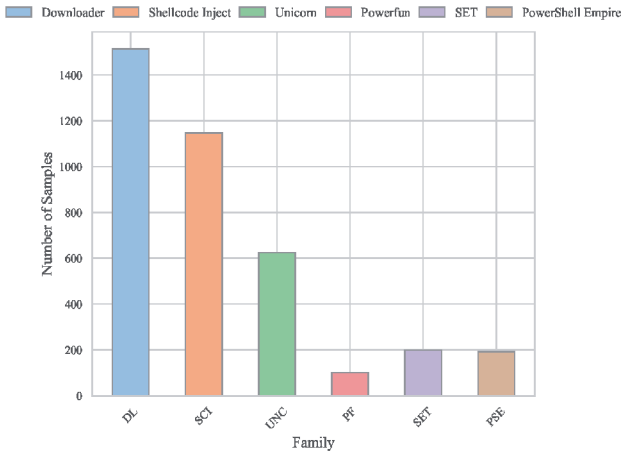

图 4 家族分布情况

3.2 实验参数与评估指标

实验中的模型迭代次数统一选择为 100 次，模型的输入数据的特征维度为 512，隐藏层维度为 128，Siamese 网络由两个共享权重的子网络组成，每个子网络包含 3 层全连接层，激活函数选择 ReLU，以增强模型的非线性表达能力。优化算法选择 Adam，学习率设定为 0.000 1，批大小设置为 64。

本文对恶意代码分类领域的模型性能进行评价时，采用准确率（Accuracy）、精确率（Precision）、召回率（Recall）和 F_1 值（F_1-score）4 个指标进行评价，它们的计算公式基于真正例（TP）、假正例（FP）、真反例（TN）和假反例（FN）的概念。这些指标的计算如公式为：

$$A=\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \quad (2)$$

$$P=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \quad (3)$$

$$R=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \quad (4)$$

$$F_1=\frac{2PR}{P+R} \quad (5)$$

3.3 对比实验分析

通过设置对比实验，将本方法与现有恶意代码分类方法进行对比。表 1 展示了本文提出的基于软件基因的恶意代码检测方法与近年来几种主流恶意代码检测方法及市面上常见的反病毒引擎的性能对比。相比于传统的基于特征匹配的方法和近年来的深度学习模型，本文方法在准确率和召回率上均有显著提升，特别是在识别未知和变种恶意代码方面显示出更高的鲁棒性。

表 1 模型关键组件的消融实验结果

Method	Years	ACC	PRE	Recall	F_1
SGMD (Ours)	2024	0.977 9	0.977 2	0.937 4	0.952 8
DNN	2023	0.959 3	0.929 0	0.893 1	0.904 6
Transformer-CNN	2022	0.960 9	0.954 0	0.904 4	0.920 1
BiGRUA	2018	0.923 4	0.874 3	0.834 3	0.883 2
Windows Defender	2024	0.953 2	0.963 1	0.893 2	0.913 4
Huorong	2024	0.963 2	0.981 2	0.912 3	0.923 8

图 5 为本文的方法与文献 [7] 的实验结果对比图。图（a）为原始数据集中不同类别的数据分布的散点图，尽管图中不同类别的数据点的分界相对明显，但仍有一定程度的混杂；图（b）和图（d）是通过软件基因方法和处理后的数据的聚类效果，这些数据是高维数据经过降维后到二维平面时的展示；图（c）和图（e）通过折线图比较了 Transformer-CNN 模型和 SGMD 方法的预测值与真实标签之间的拟合情况。结果表明，SGMD 方法的预测折线与真实标签的折线更为接近，显示出较小的偏差，这证实了 SGMD 方法在准确性和泛化能力上的优越性。

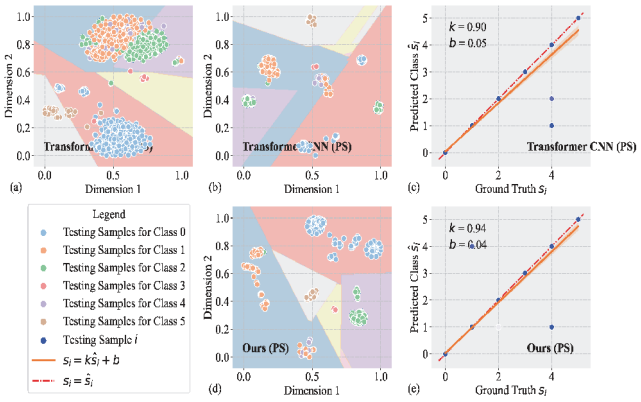

图 5 实验结果

3.4 消融实验分析

为了论证本文提出的基于软件基因的恶意代码检测方法的有效性，在公开的恶意代码数据集上进行了多次消融实验，结果如表 2 所示。首先，从包含所有关键模块的完整方法开始，然后逐步移除或替换各个模块，以评估每个模块对方法整体性能的影响。具体来说，当移除深度特征提取模块（deep feature extraction，DFE）后，检测方法在提取和识别关键特征方面的能力显著下降，表明 DFE 对提取恶意代码的核心特征至关重要。同时，剔除相似性对比模块（similarity comparison module，SCM），发现方法在识别未知和变种恶意代码时的准确率大幅降低，验证了 SCM 在捕捉代码片段之间相似性方面的核心作用。最后，当移除基因库优化模块（gene library optimization，GLO）时，分类的精度和召回率均有所下降，说明 GLO 有效地增强了恶意代码家族的分类效果。这些消融实验不仅验证了本文所提出的各个模块的重要性，还揭示了其在提升整体检测性能中的互补作用。

表 2 模型关键组件的消融实验结果

IDS	ACC	PRE	Recall	F_1
SGMD (Ours)	0.977 9	0.977 2	0.937 4	0.952 8
DFE (W/O)	0.968 8	0.941 2	0.942 8	0.942 0
SCM (W/O)	0.966 1	0.982 9	0.889 0	0.908 6
GLO (W/O)	0.963 5	0.970 4	0.886 8	0.904 7

4 结语

本文提出了一种基于软件基因的恶意代码检测方法，通过引入基因特征提取和相似性检测技术，提升了方法在处理未知和变种恶意代码时的检测准确性。实验结果证明了此方法的有效性，与现有技术相比，SGMD 在恶意代码检测和家族分类任务中表现出了高精度和较低的计算资源消耗优势。未来工作中，将计划探索更高级的基因特征优化策略，进一步提高模型的检测准确性和鲁棒性。同时，考虑到恶意代码不断变化的攻击模式，也将致力于开发更具适应性的检测方法，以应对新兴的恶意代码变种和复杂的攻击手段。

参考文献：

[1] CHICCO D. Siamese neural networks: An overview[J]. Artificial neural networks, 2020,2190(8): 73-94.

[2] HENDLER D, KELS S, RUBIN A. Detecting malicious powershell commands using deep neural networks[C]//Proceedings of the 2018 on Asia conference on computer and communications security. NewYork:ACM,2018: 187-197.

[3] 刘岳，刘宝旭，赵子豪，等．基于特征组合的 Powershell 恶意代码检测方法 [J]. 信息安全学报 ,2021,6(1):40-53.

[4] FANG Y, ZHOU X Y, HUANG C. Effective method for detecting malicious PowerShell scripts based on hybrid features[J]. Neurocomputing, 2021, 448(8): 30-39.

[5] LI Z Y, CHEN Q A, XIONG C L, et al. Effective and light-weight deobfuscation and semantic-aware attack detection for powershell scripts[C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. NewYork:ACM,2019: 1831-1847.

[6] RUSAK G, AL-DUJAILI A, O'REILLY U M. Ast-based deep learning for detecting malicious powershell [DB/OL].(2018-10-03)[2023-10-19].https://doi.org/10.48550/arXiv.1810.09230 .

[7] 高宇航，彭国军，杨秀璋，等．基于深度学习的 PowerShell 恶意代码家族分类研究 [J]. 武汉大学学报 (理学版), 2022, 68(1): 8-16.

[8] YANG X Z, PENG G J, ZHANG D N, et al. PowerDetector: Malicious powerShell script family classification based on multi-modal semantic fusion and deep learning[J]. China communications, 2023, 20(11): 202-224.

[9] SPOTIFY. Annoy: approximate nearest neighbors in C++/Python optimized for memory usage and loading speed [EB/OL] (2013-05-07) [2024-09-04]. https://github.com/spotify/annoy.

[10] WHITE J. Pulling back the curtains on encoded command powerShell attacks[EB/OL]. (2017-03-10) [2024-04-08]. https://unit42.paloaltonetworks.com/unit42-pulling-back-the-curtains-on-encodedcommand-powershell-attacks/.

【作者简介】

何源（2000—），男，湖北黄冈人，硕士研究生，研究方向：恶意代码分析、数据挖掘。

（收稿日期：2024-09-19）