Automated Android Malware Detection Using Optimal Ensemble Learning Approach for Cybersecurity

基于网络安全最优集成学习方法的自动Android恶意软件检测

Alamro H, Mtouaa W, Aljameel S, et al. Automated android malware detection using optimal ensemble learning approach for cybersecurity[J]. **IEEE Access, 2023.**

南京邮电大学

计算机学院、软件学院、网络空间安全学院

汇报人： 孔茜

目录

1

1.研究背景

南京邮电大学

1.1 什么是恶意代码

1.2 安卓恶意代码

1.3 安卓恶意代码检测

1.4 恶意软件检测技术[1]

[1]Aslan Ö A, Samet R. **A comprehensive review on malware detection approaches**[J]. IEEE access, 2020, 8: 6249-6271.

检测方法	优点	缺点
1.基于特征的	①直接利用已知的恶意软件的特征进行识别，快速准确。 ②有效检测属于同一家族的恶意软件	①无法有效识别未知或变种的恶意软件。 ②特征库需要持续更新，且需大量存储空间。 ③处理特征需要花费时间。
2.基于行为的	①能有效识别未知或变种的恶意软件。 ②有效对抗混淆和多态技术。 ③不需要依赖特征库，比较灵活。	①可能存在误报。 ②不可能监控到所有行为。 ③对于一些隐蔽的恶意行为，可能难以识别。
3.基于启发式的	①能有效识别未知或变种的恶意软件。 ②静态特征和动态特征都能使用。	①规则和训练过程太多。 ②易受变形技术影响。
4.基于机器学习的	①准确率高，速度快 ②可实现自动化	①样本不平衡 ②特征选择存在挑战
5.基于深度学习的	①强大且高效 ②大幅缩小特征空间	①不抵抗规避攻击 ②构造隐藏层需要时间

1.5 本文方法

本文提出了一种使用网络安全最优集成学习方法（AAMD-OELAC）**技术的**自动 Android 恶意软件检测。AAMD-OELAC技术在初步阶段执行数据预处理。对于 Android恶意软件检测过程，AAMD-OELAC 技术遵循**使用三个ML模型的集成学习过程，即最小二乘支持向量机 (LS-SVM)、核极限学习机 (KELM) 和正则化随机向量函数链神经网络 (RRVFLN)。最后，利用猎人猎物优化搜索(HPO)算法对三个ML模型进行最优参数调整**，有助于实现改进的恶意软件检测结果。为了表明AAMD-OELAC方法的优越性，进行了综合实验分析。

1.研究背景

①AAMD-OELAC: 最优集成学习方法的自动化Android恶意软件检测技术(Automated Android Malware Detection Using Optimal Ensemble Learning Approach)
②最优集成学习方法：组合多个基本学习算法以提高整体性能。

②LS-SVM: 最小二乘支持向量机（Least Square Support Vector Machines）

③KELM：核极限学习机（Kernel Extreme Learning Machine）

④RRVFLN：正则化随机向量函数链神经网络

⑤HPO：猎人猎物优化搜索算法（Hunter–prey optimizer）

⑥F-score：Fisher Score，是特征选择的有效方法之一，其主要思想是鉴别性能较强的特征表现为类内距离尽可能小，类间距离尽可能大。

⑦MCC：马修斯相关系数。一种用于评估二元分类模型性能的指标，特别适用于处理不平衡数据集。

⑧SVM：支持向量机。

⑨P-R曲线：精确率precision vs 召回率recall 曲线。

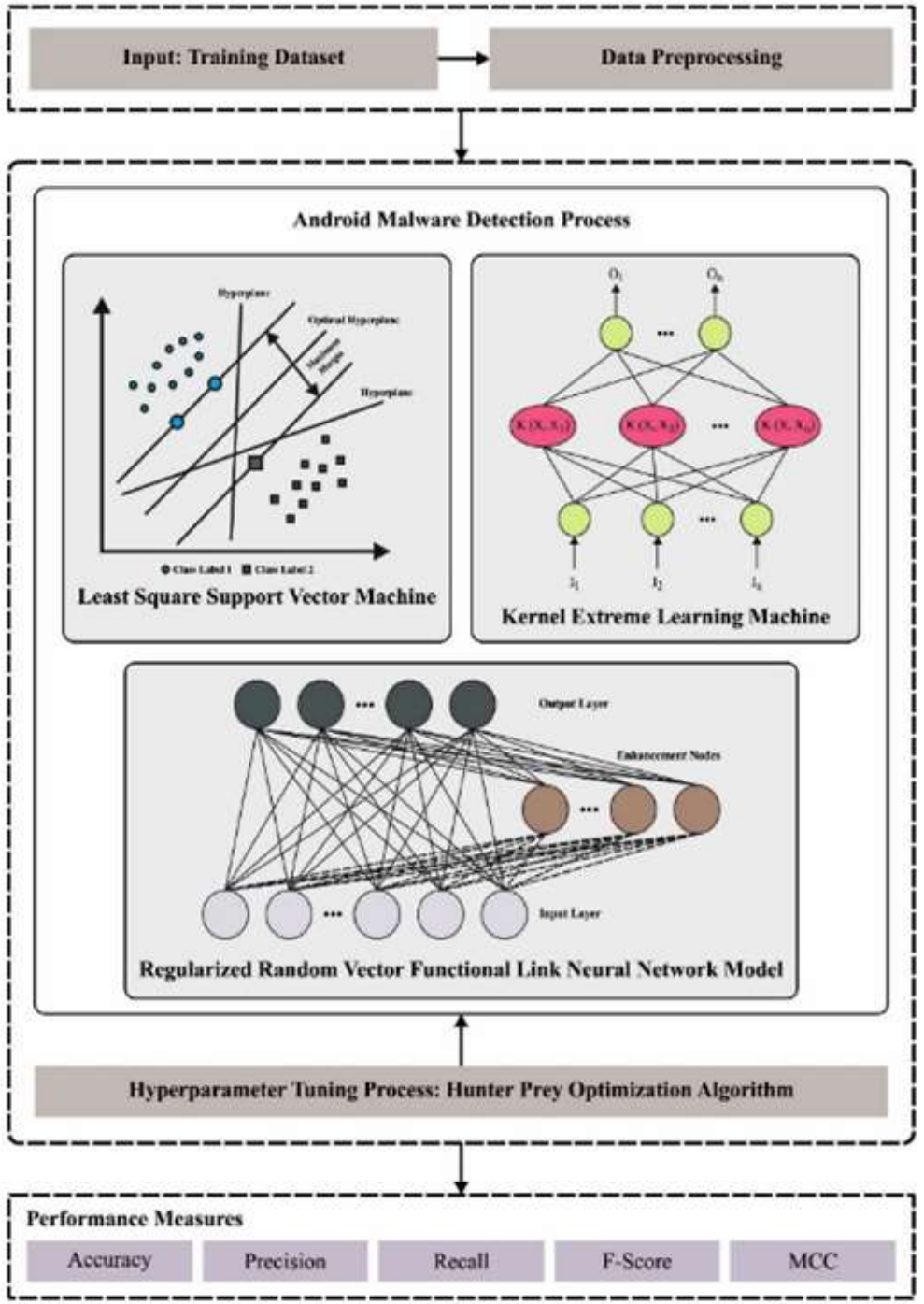
⑩ROC曲线：Receiver Operating Characteristic，受试者工作特征曲线。

2

2.模型介绍

南京邮电大学

2.模型介绍


Input: Training Dataset
Data Preprocessing
Android Malware Detection Process
Least Square Support Vector Machine
Kernel Extreme Learning Machine
Output Layer
Enhancement Nodes
Input Layer
Regularized Random Vector Functional Link Neural Network Model
Hyperparameter Tuning Process: Hunter Prey Optimization Algorithm
Performance Measures
Accuracy
Precision
Recall
F-Score
MCC

2.1 数据预处理

数据预处理主要包括以下步骤：

①使用Androguard工具分析APK文件，提取应用程序的权限和API调用。Androguard是一个专为与Android文件交互而设计的完整包工具，仅限于Python平台。它可以用作单个Android应用程序的反向工程工具。

②Androguard工具可以通过单独移除所有APK文件的DEX文件权限来分析APK文件。因此，生成了一个数据框架，其中包含应用程序（行）和特征（列），所有列都用二进制值表示特定权限或API调用，但行表示恶意软件或良性APK文件。

③对数据进行归一化处理，以便在机器学习模型中使用。

通过这些预处理步骤，可以改善数据质量，为后续的机器学习模型训练做好准备。

2.2 基于集成学习的恶意软件检测分类

2.2.1 LS-SVM模型

LSSVM是Suykens等人提出的一种机器学习算法。LSSVM作为一种基于统计理论的改进型支持向量机，具有先进的完备理论体系，能够将二次优化问题的解转化为线性方程组的求解，从而简化了问题的求解。 LS-SVM 模型的目标函数和约束如下：

$$Minimize: \frac{1}{2}\alpha^T\alpha + \frac{1}{2}\gamma\sum_{t=1}^{M} e_t^2 \qquad (2)$$

$$\text{s.t.}\quad y_t = \alpha^T\varphi(x) + b + e_t$$

在上式中，α 表示权重系数， γ 是用于确定模型复杂度和精度之间权衡的正则化参数，M 表示数据集的数量， e表示误差变量。 b 表示偏差值，φ(x) 表示非线性映射函数。

为了解决上述优化问题，构造了相应的拉格朗日函数：

$$L_{LSSVM} = \frac{1}{2}||\alpha||^2 + \frac{1}{2}r\sum_{t=1}^{M} e_t^2 - \sum_{t=1}^{M} \beta_t\{a^T\varphi(x) + b + e_t - y_t\}$$

其中， β表示拉格朗日乘子。

对上式中的α、b、 e_t、 β_t分别求偏导并令导数等于0，即可得到线性方程组，解之可得LSSVM算法的计算公式 (5)。

$$f(x) = \sum_{t=1}^{M} \beta_t K(x, x_t) + b \qquad (5)$$

其中，$K(x，x_t)$为核函数。

2.2.2 KELM模型

核极限学习机（Kernel Based Extreme Learning Machine，KELM）是基于极限学习机（Extreme Learning Machine，ELM）并结合核函数所提出的改进算法，KELM 能够在保留 ELM 优点的基础上提高模型的预测性能。

ELM 是一种单隐含层前馈神经网络，其学习目标函数g(f)可用矩阵表示为：

$$g(f) = h(f) \times \beta = H \times \beta = \hat{O}$$

其中f为输入向量，h(f)、H为隐层节点输出，β为输出权重，$\hat{O}$表示目标输出。为增强神经网络的稳定性，引入正则化系数 C和单位矩阵II，则输出权值的最小二乘解为：

$$H^T \left(\frac{II}{C} + HH^T \right)^{-1} \hat{O}, \qquad (6)$$

其中，II/C表示内核参数。

接下来，ELM 的核函数确定如下：

$$K\tilde{E}LM = HH^T, \tag{7}$$

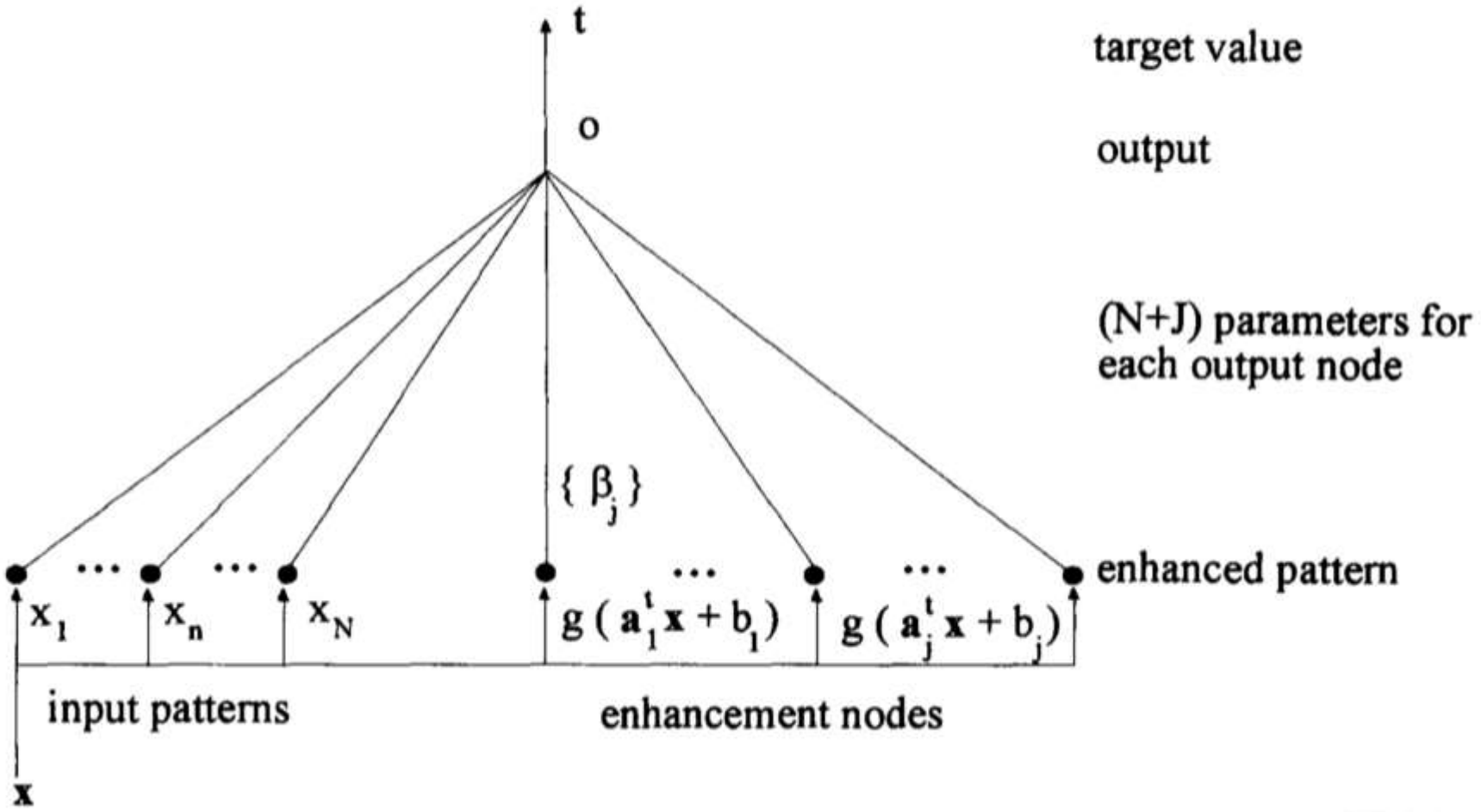
$$K\tilde{E}LM = h(f_i)\,h(f_j) = K(f_i, f_j), \tag{8}$$

$$g(f) = \begin{bmatrix} K(f, f_i) \\ \vdots \\ K(f, f_N) \end{bmatrix} \left(\left(\frac{II}{C} + K\tilde{E}LM \right)^{-1} \hat{O} \right), \tag{9}$$

K(f , fi) 表示 KELM 的核函数。

2.2.3 RRVFLN模型

函数链神经网络（Functional-link neural network，简称FLNN）的结构如下图。



$$\sum_{j=1}^{L} \beta_j g\left(w_j x_i + b_j\right) + \sum_{j=L+1}^{L+d} \beta_j x_{ij} = 0_i, i = 1, 2, \ldots\ldots, N \tag{11}$$

其中wj表示输入权重，g(x) 表示激活函数，bj 表示隐藏层增强节点的偏差，βj表示结果权重。wj 和 bj 通常随机定义。

有L个增强节点。

与传统的 RVFLN 网络相比，标准化的 RVFLN 可以有效地防止过拟合问题并提高网络的归一化容量。大多数情况下，找到输出权重和最小训练误差：

$$L_{RVFL} = \frac{1}{2}CE_2^2 + \frac{1}{2}\beta_2^2, \qquad (16)$$

在式(16)中，输出误差E = O−Y, C为正则化因子，可用于权衡模型的复杂性和训练误差的影响。为了确定 LRVFL 的最小值，通过将 RVFLN 与 β 的梯度拟合到 0 来获得 β 的封闭形式解：

$$\beta = \begin{cases} \left(H^T H + \frac{I}{C}\right)^{-1} H^T T & if\ N > L + n \\ H^T \left(\frac{I}{C} + H^T H\right)^{-1} T & if\ N < L + n' \end{cases} \qquad (17)$$

在等式(17)中，I 表示维度为 L +n 的单位矩阵。

2.3 使用 HPO 算法调整参数

在这项工作中，HPO 方法被用于 ML 模型的参数调整。这种方法是一种启发式优化算法，灵感来自于猎人和猎物之间的相互作用。该方法模拟了猎人追捕猎物的过程，通过调整猎人的行为来搜索问题的最优解。HPO的流程如下图所示。

2.模型介绍

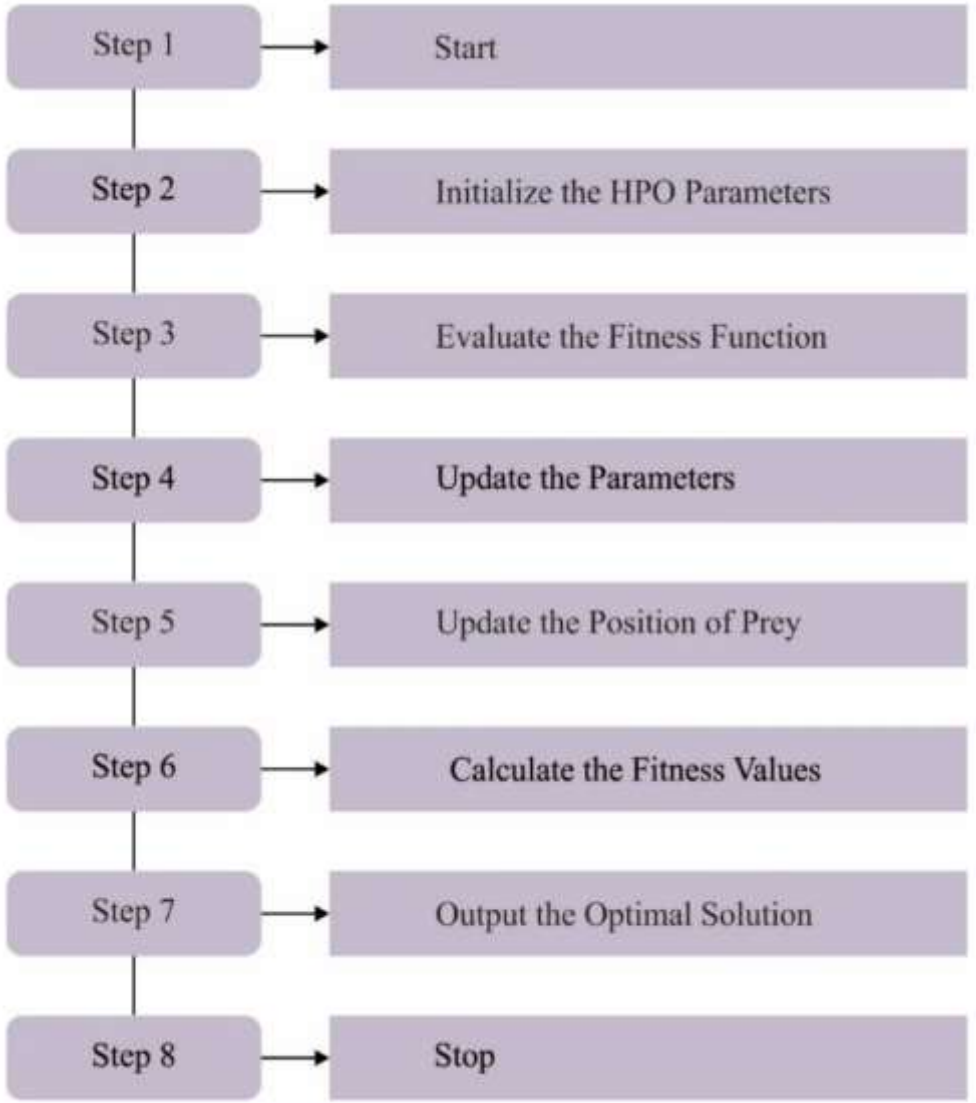


FIGURE 2. Flowchart of HPO algorithm.

总结猎人和猎物的选择方式如下：

$$x(t+1) = \begin{cases} x(t) + 0.5[(2CZP_{pos} - x(t)) + (2(1-C)Z\mu - x(t)] & R_4 < \beta \\ T_{pos} + CZ\cos(2\pi R_3) \times (T_{pos} - x(t)) & R_4 \geq \beta \end{cases} \tag{23}$$

x(t)和x(t+1)表示猎人的现有位置和下一个位置，Ppos表示猎物的位置，Tpos是全局最优位置，μ表示每个位置的均值，C表示开发和探索之间的平衡变量，其值从1下降到0.02，Z表示自适应参数，R3是范围[−1,1]的随机数， R4是[0,1]范围内的随机数，β是一个调节参数。

如果R4的值小于β，搜索代理将被视为猎人，搜索代理的下一个位置将用第一个式子更新；如果R4的值大于β，搜索代理将被视为猎物，搜索代理的下一个位置将用第二个式子更新。

3

3.性能验证

在这项研究中，AAMD-OELAC 技术的恶意软件检测结果在 AnroAutoPsy 数据集上进行了研究。它保存了 7500 个样本，有两个类，如表 1 所示。

TABLE 1. Details of database.

Class	No. of Samples
Benign	5000
Malware	2500
Total Samples	7500

4.性能验证

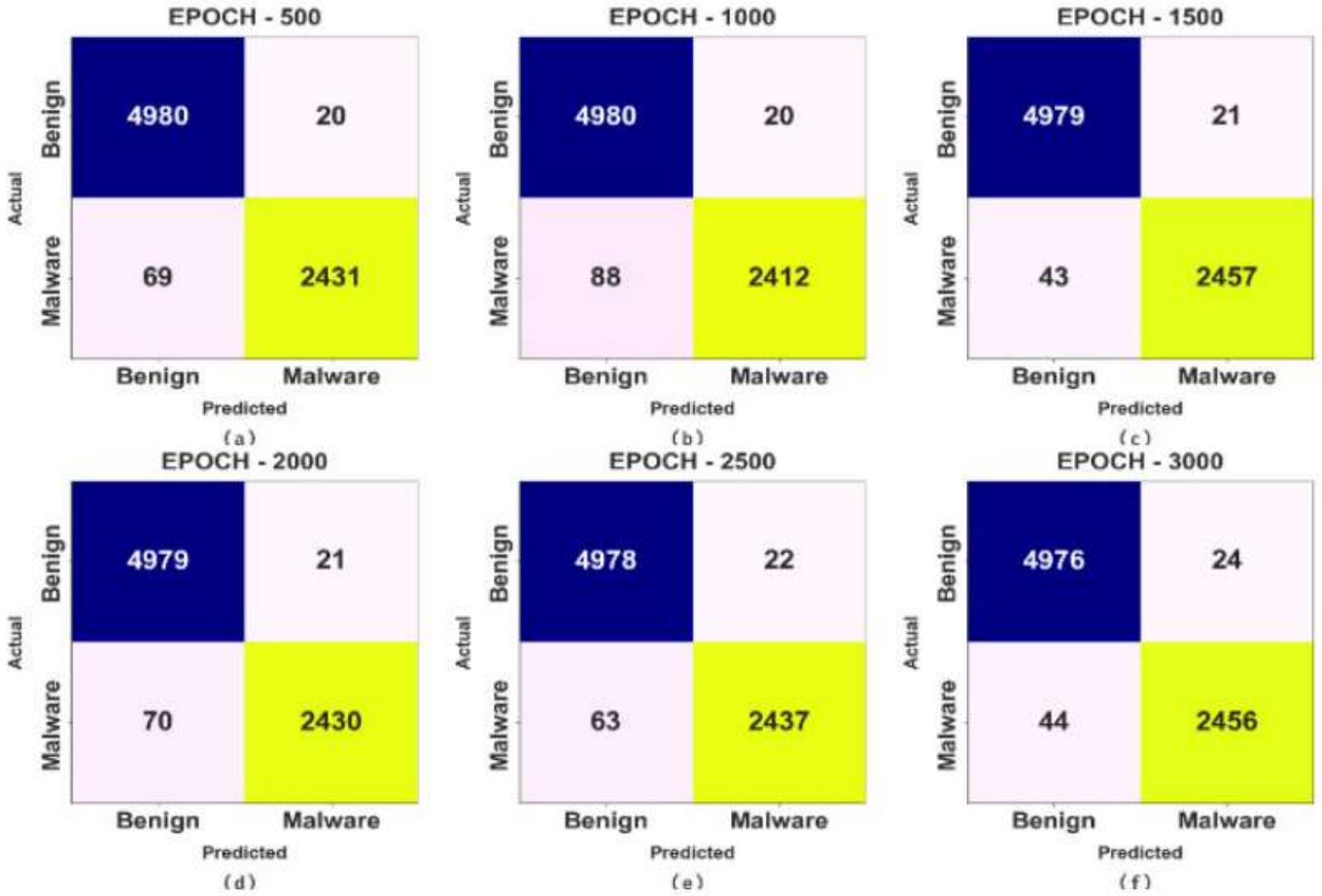
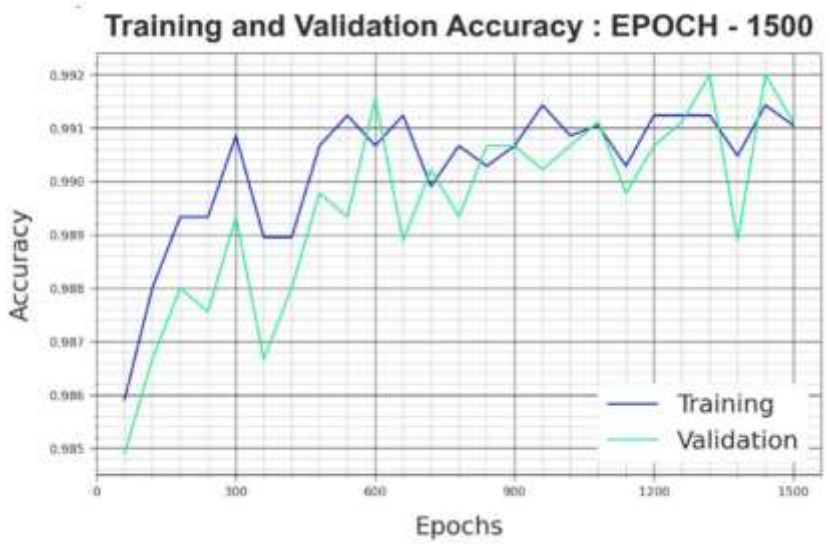


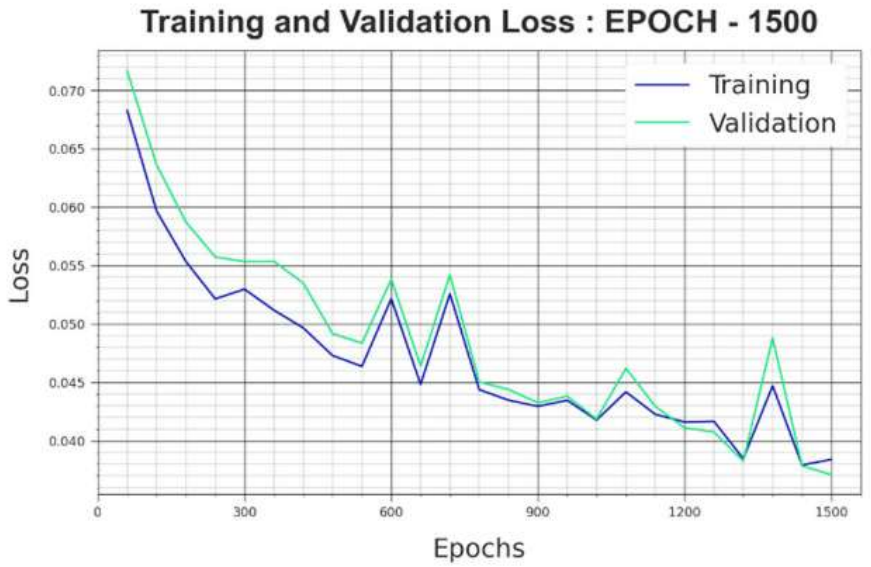
FIGURE 3. Confusion matrices of AAMD-OELAC system (a-f) Epochs 500-3000.

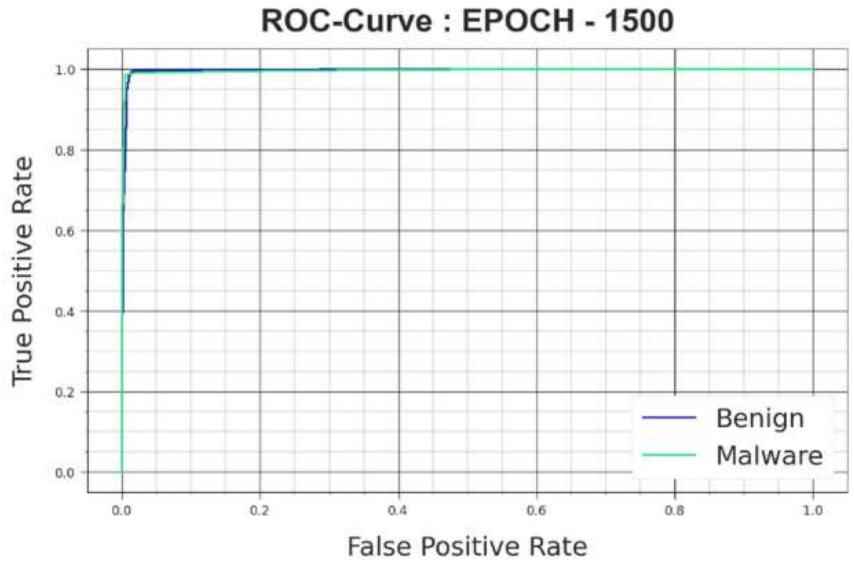
4.性能验证

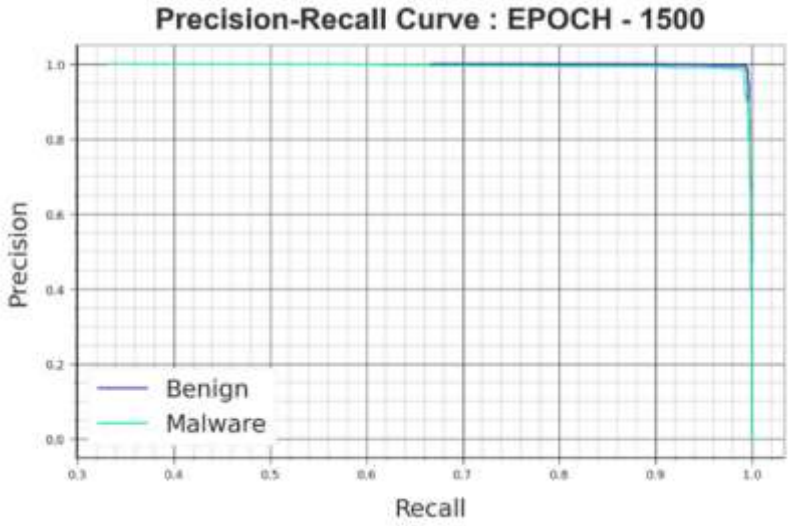
Class	$Accu_y$	$Prec_n$	$Reca_l$	F_{score}	MCC
Epoch - 500					
Benign	99.60	98.63	99.60	99.11	97.33
Malware	97.24	99.18	97.24	98.20	97.33
Average	98.42	98.91	98.42	98.66	97.33
Epoch - 1000					
Benign	99.60	98.26	99.60	98.93	96.76
Malware	96.48	99.18	96.48	97.81	96.76
Average	98.04	98.72	98.04	98.37	96.76
Epoch - 1500					
Benign	99.58	99.14	99.58	99.36	98.08
Malware	98.28	99.15	98.28	98.71	98.08
Average	98.93	99.15	98.93	99.04	98.08
Epoch - 2000					
Benign	99.58	98.61	99.58	99.09	97.27
Malware	97.20	99.14	97.20	98.16	97.27
Average	98.39	98.88	98.39	98.63	97.27
Epoch - 2500					
Benign	99.56	98.75	99.56	99.15	97.45
Malware	97.48	99.11	97.48	98.29	97.45
Average	98.52	98.93	98.52	98.72	97.45
Epoch - 3000					
Benign	99.52	99.12	99.52	99.32	97.96
Malware	98.24	99.03	98.24	98.63	97.96
Average	98.88	99.08	98.88	98.98	97.96
Malware	98.41	99.07	98.41	98.74	98.11
Average	98.97	99.13	98.97	99.05	98.11

4.性能验证


Training and Validation Accuracy : EPOCH - 1500
Accuracy
Epochs
Training
Validation


Training and Validation Loss : EPOCH - 1500
Loss
Epochs
Training
Validation


ROC-Curve : EPOCH - 1500
True Positive Rate
False Positive Rate
Benign
Malware


Precision-Recall Curve : EPOCH - 1500
Precision
Recall
Benign
Malware

4.性能验证

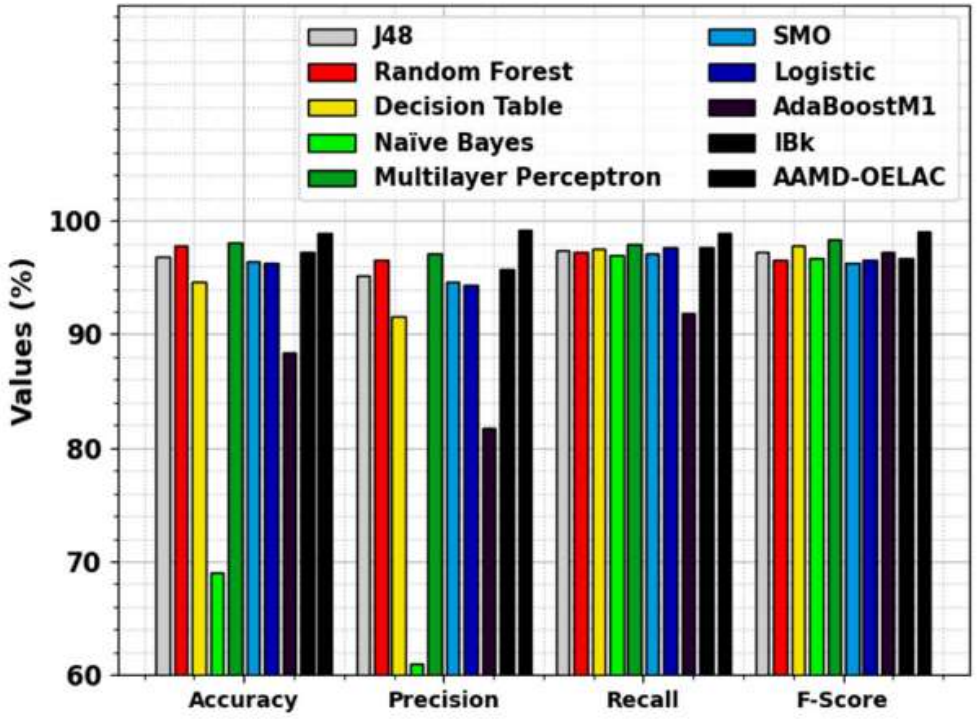


FIGURE 9. Comparative outcome of AAMD-OELAC approach with other methods.

TABLE 3. Comparative result of the AAMD-OELAC method with other approaches.

Algorithm	$Accu_y$	$Prec_n$	$Reca_l$	F_{score}
J48	96.80	95.20	97.42	97.24
Random Forest	97.80	96.60	97.22	96.62
Decision Table	94.60	91.60	97.52	97.77
Naïve Bayes	69.10	61.00	96.92	96.71
Multilayer Perceptron	98.10	97.10	97.90	98.35
SMO	96.40	94.60	97.07	96.26
Logistic	96.30	94.40	97.74	96.58
AdaBoostM1	88.40	81.70	91.83	94.25
IBk	97.20	95.70	97.66	96.68
AAMD-OELAC	98.93	99.15	98.93	99.04

4

4.总结与思考

本文的贡献：

①提出了一种智能AAMD-OELAC技术，包括数据预处理、集成学习和基于HPO的超参数调优，用于Android恶意软件检测。

②执行基于集成学习的分类过程，包括LS-SVM、KELM和RRVFLN模型，用于Android恶意软件检测。

③HPO算法和集成学习过程的结合提高了Android恶意软件的检测精度。通过利用多个分类器和优化策略，该模型可以有效地识别Android应用程序中的恶意模式和行为。

思考：

①本文虽然说用了集成学习的方法，但并没有介绍具体用了什么样的集成学习的方法，感觉只是三个单独的ML的模型的使用，这方面可以自己进行进一步研究。

②目前机器学习方法的不是主流了，所以不知道该模型的性能是否有他所说的那么好。

恳请各位老师批评指正

南京邮电大学

计算机学院、软件学院、网络空间安全学院
汇报人：孔茜