文章编号：2096-1472(2021)-06-06-07
DOI:10.19644/j.cnki.issn2096-1472.2021.06.002

基于注意力机制的病毒软件可视化检测方法

赵晨洁[1,2]，左 羽[1,2]，崔忠伟[2]，李亮亮[1]，吴 恋[2]，王永金[1,2]，韦萍萍[2]

(1.贵州大学计算机科学与技术学院，贵州 贵阳 550025；
2.贵州师范学院数学与大数据学院，贵州 贵阳 550018)
✉495752340@qq.com; bjuzuoyu@163.com; 33374225@qq.com; besruiven@gmail.com;
373201377@qq.com; 1411356083@qq.com; 104018794@qq.com

摘 要：针对当前的病毒软件检测方法难以应对大数据时代下病毒软件快速分类问题，提出一种病毒可视化检测的分类方法。详细阐述了病毒软件可视化过程，并提出一种卷积神经网络结合注意力机制的模型(即CNN_CBAM模型)进行病毒软件家族分类的深度学习方法。病毒软件样本采用BIG2015和Malimg数据集，将其进行可视化，并将CNN_CBAM模型在可视化后的数据集上进行训练。实验结果显示，CNN_CBAM模型能够有效地对病毒软件家族进行分类，且效果优于其他深度学习模型，其准确率比CNN_SVM病毒分析的方法提升16.77%。

关键词：病毒软件；深度学习；灰度图；可视化；注意力机制
中图分类号：TP391.41 **文献标识码**：A

Visual Detection Method of Virus Software based on Attention Mechanism

ZHAO Chenjie[1,2], ZUO Yu[1,2], CUI Zhongwei[2], LI Liangliang[1], WU Lian[2], WANG Yongjin[1,2], WEI Pingping[2]

(1.*College of Computer Science and Technology*, *Guizhou University*, *Guiyang* 550025, *China*;
2.*College of Mathematics and Big Data*, *Guizhou Education University*, *Guiyang* 550018, *China*)
✉495752340@qq.com; bjuzuoyu@163.com; 33374225@qq.com; besruiven@gmail.com;
373201377@qq.com; 1411356083@qq.com; 104018794@qq.com

Abstract: Current virus software detection methods have difficulty in grappling with the rapid classification of virus software in big data era. In view of this issue, this paper proposes a classification method for virus visual detection, which elaborates on the visualization process of virus software. It proposes a deep learning method of convolutional neural network combined with attention mechanism model (ie CNN_CBAM model, Convolutional Neural Network_Convolutional Block Attention Module) to classify virus software families. Virus software samples use the BIG2015 and Malimg datasets, which are visualized in this paper. The proposed CNN_CBAM model is trained on the visualized dataset. The experimental results show that the CNN_CBAM model proposed in this paper can effectively classify the virus software families, and it is better than other deep learning models. Its accuracy rate is 16.77% higher than that of CNN_SVM virus analysis method.

Keywords: virus software; deep learning; grayscale image; visualization; attention mechanism

1 引言(Introduction)

病毒软件是一种破坏计算机系统的软件产品，又称恶意软件[1]。2020年7月McAfee Labs发布的威胁报告[2]显示，在2019年第一季度，病毒软件已经突破9 亿；而在病毒二进制可执行文件方面，从2017年至2019年间，单季度最高新增量高于110 万。计算机病毒的数量一直趋于增长状态，造成的损失也在一直增加。

随着网络的大量智能化软件兴起，病毒软件开发者为了防止被检测系统发现，不断修改现有的病毒软件，使得病毒的特征难以确定。常用的病毒软件检测方法是将检测修改后

基金项目：贵州省省级重点学科“计算机科学与技术”(ZDXK[2018]007号)；贵州省科学技术基金计划(黔科合基础[2016]1116)；贵州省教育厅创新群体研究项目(黔教合KY字[2021]022)；贵州省省级重点支持学科“计算机应用技术”(黔学位合字ZDXK[2016]20号)；贵州省2018年第三批省级服务业发展引导资金项目(黔发改服务[2018]1181号).

的病毒软件的特征进行存储，再通过匹配来检测新的病毒软件，但由于存储病毒的特征价格昂贵并且低效，因此研究如何不使用存储特征识别病毒软件是很有必要的。

2 病毒软件检测算法(Virus software detection algorithm)

2.1 静态方法

静态分析[3]是对病毒软件不执行，直接分析其行为，对该宿主机的操作系统不进行破坏。静态分析常用方式包括语法库调用、操作码频率分布、字节序列、n-gram、字符串签名等。

MA[4]等人提出的语库法调用，与示例恶意软件数据进行语义比较得出分类结果；CHARIKAR[5]等人提出的字符串签名，是将可移植可执行文件(PE)、字符串和字节序列三种不同的静态特性结合起来对病毒软件进行分类；LI[6]等人提出的字符串签名，根据字符串签名的功能长度结合其特征识别病毒软件；SANTOS[7]等人提出操作码频率分布，通过掌握每个操作码的相关性，衡量操作码序列频率来检测恶意软件；KANG[8]等人提出基于n-gram特征，利用机器学习识别病毒软件。

2.2 动态方法

动态分析是在宿主机安全的情况下(如模拟器、沙箱或虚拟机等)对病毒软件进行执行并分析，但对病毒软件的进程监视、安装、激活等需要大量的时间和计算。ZOLKIPLI[9]等人提出的基于行为方法，是将病毒软件进行动态分析，使用HoneyClients和Amun收集恶意软件的行为；ANDERSON[10]等人提出的基于行为检测，是在CWSandbox和Anubis两个虚拟平台上执行每个样本，根据动态收集指令踪迹构建图形的分析识别；LIN[11]等人提出的API调用监控，是应用无监督非负矩阵分解(NMF)进行聚类分析，从大量API调用监控中提取API检测出类似的恶意软件样本；SHAHZAD[12]等人提出的交通流序列模式，是一种基于流特征聚类和序列比对的算法，分析了交通流序列模式之间的序列相似性。

2.3 可视化分析方法

在网络安全空间中，病毒软件样本数据量激增、二进制可执行文件特征量庞大等因素使得大量研究人员尝试将病毒二进制文件进行可视化并用深度学习方法来检测病毒软件。

YOO[13]等人提出的映射，是使用自组织映射来可视化和检测病毒；LORENZO[14]等人提出的动态分析、可视化，是一个可视化地表示程序的总体流程，它依赖以太虚拟机监控程序来基于动态分析秘密地监控程序的执行；RIECK[15]等人提出的API调用监控、可视化，是使用树状图和线程图收集有关API调用的信息和在沙箱中执行的操作；ZHANG[16]等人提出的操作码序列可视化，其操作码是二进制可执行文件分解而成的，他们使用卷积神经网络(CNN)的方法来识别一个二进制可执行文件是否是恶意的；NATARAJ[17]等人，首次将恶意软件的二进制文件可视化为灰度图像；SHANKARPANI[18]等人，采用k近邻算法并融合欧氏距离方法，将可执行文件转化为灰度图像，使用支持向量机(SVMs)对其进行检测。为了解决宿主机安全环境搭建复杂、传统机器学习方法提取高维度特征困难的问题，提供更精准的识别病毒的软件家族，本文提出了使用二进制可视化将二进制文件进行关键信息提取，而后进行灰度图像的转化，并通过CNN_CBAM模型进行病毒软件家族分类。

3 构建CNN_CBAM病毒检测模型(Construction of virus detection model CNN_CBAM)

3.1 注意力机制

注意力机制模块(Convolutional Block Attention Module，简称CBAM模块)[19]，结合空间和通道注意力机制，通过上一层输出的特征作为通道注意力模块的输入，经过通道注意力模块后得到的权值，是空间注意力模块的输入，回溯到原始的灰度图像，分析出病毒存在的具体位置，最后输出相关的特征值。CBAM模块整体流程，如图1所示。

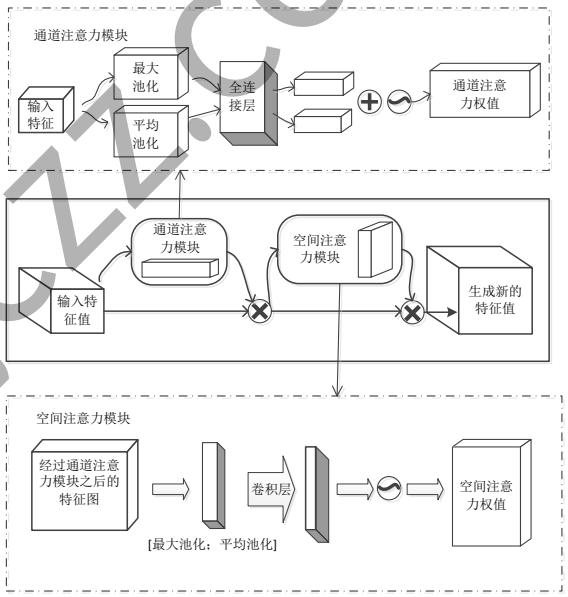


图1 CBAM流程图

Fig.1 CBAM flow chart

通道注意力模块：将四层卷积后的原始病毒特征F作为通道注意力模块的输入$H\times W\times C$，经过平均池化(AvgPool)和最大池化(MaxPool)得到新的特征，如公式(1)所示，输出原始病毒图像中有意义的特征。

$$\begin{aligned}M_c(F)&=\sigma(MLP(\mathrm{AvgPool}(F))+MLP(\mathrm{MaxPool}(F)))\\&=\sigma(W_1(W_0(F^c_{\mathrm{avg}}))+(W_1(W_0(F^c_{\max})))\end{aligned}\tag{1}$$

其中，$M_c(F)$的大小是F的通道个数，即C；F特征的大小为$C\times H\times W$；$\sigma()$表示Sigmoid激活函数；W_0为$\frac{c}{r}\times C$的矩阵；W_1为$C\times\frac{c}{r}$的矩阵。

空间注意力模块：该模块主要是获取通道模块特征作为输入，而后在平均池化(AvgPool)和最大池化(MaxPool)中选择合适的进行池化后再进行卷积，得到的值就是用来说明病毒灰度图像中哪些区域特征是值得关注的，如公式(2)所示。

$$\begin{aligned}M_s(F)&=s(f^{7\times7}([\mathrm{AvgPool}(F);\mathrm{MaxPool}(F)]))\\&=s(f^{7\times7}([F^s_{\mathrm{avg}};F^s_{\max}]))\end{aligned}\tag{2}$$

其中，$s()$表示Sigmoid激活函数，$f^{7\times7}()$表示卷积操作且7×7是卷积核大小。

本文的注意力机制算法模块是将第四个卷积层输出的特征F，放入注意力机制模块$M_c(F)$中，计算出原始病毒图像的有意义的特征权重$\hat{c}$；然后传送到下一层$M_s(F)$，进行计算得到每个区域的注意力权重值$\hat{s}$；再将两个权重值进行公式(3)计算，最终得到注意力权重$\hat{F}$。

$$\hat{F}=\sum_{i=1}^{w}\sum_{i=1}^{h}\hat{c}\,\hat{s} \tag{3}$$

其中，w代表病毒图像的宽，h代表病毒图像的高。

3.2 模型搭建

本文在卷积神经网络中添加CBAM模块(注意力机制)，对病毒的图像关注通道的同时也关注了空间，对病毒图像进行更深层的学习，从而识别出病毒图像种族。整体的病毒训练模型是将原始病毒文件进行可视化转化，将转化后的图像放入本文设计的CNN_CBAM模型中进行训练识别。病毒检测整体框架，如图2所示。

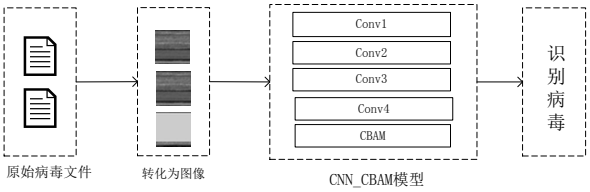


图2 病毒训练流程图

Fig.2 Flow chart of virus training

将病毒文件转化为灰度图像作为CNN_CBAM模型的输入，经过四层卷积核大小为3×3、padding=“same”的卷积层。为了病毒图像特征更好地展示，便于识别，加入CBAM模块得到哪些病毒特征需要关注和需要关注病毒图像位置的权重$\hat{F}$。将权重$\hat{F}$与第四层卷积后的特征相乘，得到的病毒特征更加有效，最后通过Softmax函数进行病毒种族分类，采用测试集数据验证模型。

3.3 模型训练

本文病毒检测模型主要是将病毒软件转化为图像后，传入本文设计的CNN_CBAM模型中。CNN_CBAM模型由四层卷积、一个CBAM和一个全连接层构成。

卷积层的深度越大，它输出的特征图越小，因此设计一个四层卷积能充分学习到病毒图像的特征值，并且为了解决病毒图像边缘信息丢失问题，在卷积层采用padding=“same”进行填充。激活函数采用Relu使得网络训练更快，防止梯度消失。选交叉熵为损失函数，如公式(4)所示，输出层分类函数选Softmax。

$$loss=-\sum_{i=1}^{n}\hat{y}_{i1}\log y_{i1}+\hat{y}_{i2}\log y_{i2}+\cdots+\hat{y}_{in}\log y_{im} \tag{4}$$

其中，n是病毒的样本数量，m是病毒种族的个数。

病毒灰度图像作为数据集源，为卷积神经网络提供学习的数据，对其进行特征的提取，提取后的特征作为CBAM模块的输入；再进一步对病毒灰度图像进行特征的关注，即哪些特征和哪些区域特征，进入全连接共享特征，通过Softmax对病毒软件进行家族分类；最后用待检测样本对模型进行检测。本文网络的模型详细参数如表1所示。

表1 模型参数

Tab.1 Model parameters

层	卷积核	激活函数
输入层		Relu
卷积层1(Conv1)	3×3	Relu
卷积层2(Conv2)	3×3	Relu
卷积层3(Conv3)	3×3	Relu
卷积层4(Conv4)	3×3	Relu
注意力层(CBAM)		Relu
全连接层(Fc1)		Softmax

4 实验结果与分析(Results and analysis)

4.1 数据

由于病毒软件的特殊性，有专门的网站提供大量的恶意软件样本供其学习研究，如VirusShare(https://virusshare.com)、VirusTotal(https://www.virustotal.com)、MalShare(https://malshare.com)等。本文使用了2015年微软提供的上万数量的带标签病毒软件样本集Microsoft Malware Classification Challenge(BIG2015)[20]，每个PE文件具有所属恶意软件家族的标签，如图3所示。为了验证本文可视化的效果，同时使用了Malimg数据集，如图4所示。

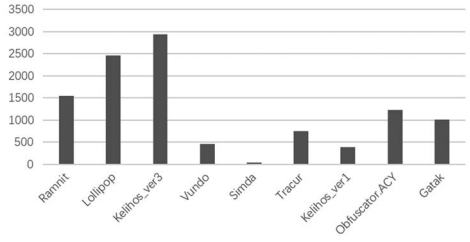


图3 BIG2015数据集种类

Fig.3 BIG2015 dataset types

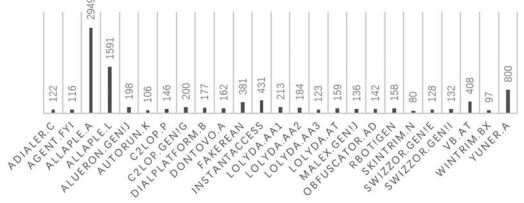


图4 Malimg数据集种类

Fig.4 Malimg dataset types

4.2 数据处理、可视化

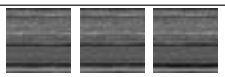
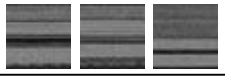
Malimg数据集：将图像调整为一个$n\times n$的二维矩阵，用对应的病毒软件种类标记每个特征数组，使用公式(5)对特征进行标准化。

$$z=\frac{X-\mu}{\sigma} \tag{5}$$

其中，原病毒灰度图像：X；均值：μ；标准差：σ。使用scikit-learn[21]实现标准化处理后示例，如表2所示。

表2 Malimg数据集标准化后示例

Tab.2 Example after normalization of the Malimg dataset

病毒种族	样例
Adialer.C	
Dontovo.A	
Fakerean	
Lolyda.AA3	
Rbot!gen	
Swizzor.gen!I	

Kaggle Microsoft恶意软件数据集：该数据集提供的是病毒软件汇编代码，因此要先将病毒软件的二进制代码转化为可视化的灰度图像。对每一个数据集BIG2015中病毒软件的样本，都将其按8位无符号整数向量读取，并存储为一组二维数组，其值的范围为0—255，0为黑色像素，255为白色像素。可视化流程图如图5所示。

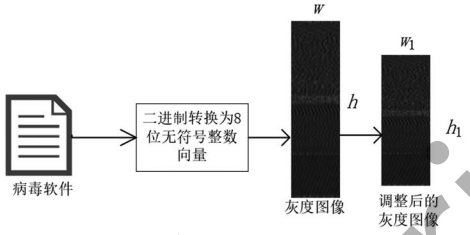


图5 病毒软件可视化过程

Fig.5 Virus software visualization process

图5中宽度w是固定的，高度h根据每个病毒文件的大小而定。表3是各类病毒软件转化为灰度图像的示例。

表3 BIG2015数据集可视化样本

Tab.3 BIG2015 dataset visualization sample

病毒种类	可视化样本
Ramnit	
Simda	
Tracur	

4.3 评价方法

(1)真正样本：检测该病毒正确的个数，用*TP*表示；

(2)真负样本：对非某一病毒种族样本，被判定为非某一病毒种族样本的个数，用*TN*表示；

(3)假正样本：对实际不是A病毒种族，被判定为A病毒种族样本的个数，用*FP*表示；

(4)假负样本：没有检测到该病毒的个数，用*FN*表示。

本文采用的评估标准为准确率(*ACC*)、精确率(*Precision*)、召回率(*Recall*)以及$F1-Score$。计算公式如下：

$$ACC=\frac{TP+TN}{TP+TN+FN+FP} \tag{6}$$

$$Precision=\frac{TP}{TP+FP} \tag{7}$$

$$Recall=\frac{TP}{TP+FN} \tag{8}$$

$$F1-Score=2\times\frac{Precision\times Recall}{Precision+Recall} \tag{9}$$

4.4 实验结果

本文将处理好的BIG2015数据集和Malimg数据集分别进行训练，发现无论是BIG2015数据集还是Malimg数据集都有较高的精确度，说明在本文模型的训练上有较好效果。

4.4.1 模型准确率变化

为了更好地比较本文模型，本文用相同的数据集在Inception V3模型上进行对比，发现本文CNN_CBAM模型能更好地训练识别出病毒的种类。以下是数据集Malimg和数据集BIG2015在不同网络训练的损失率(Loss)和精确率(ACC)。为了便于表示，将数据集Malimg用A来代替，数据集BIG2015用B来代替，如图6至图13所示。

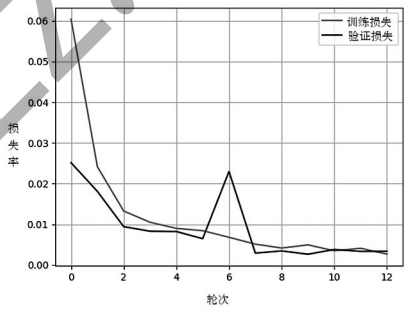


图6 数据集A在本文模型的损失率

Fig.6 Loss rate of data A in text model

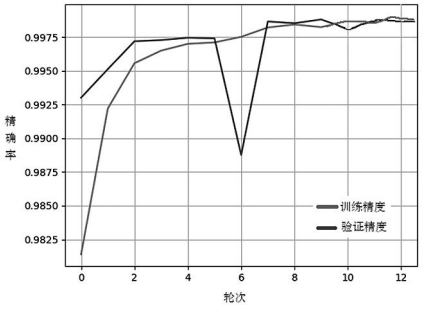


图7 数据集A在本文模型的精确率

Fig.7 Accuracy of data A in text model

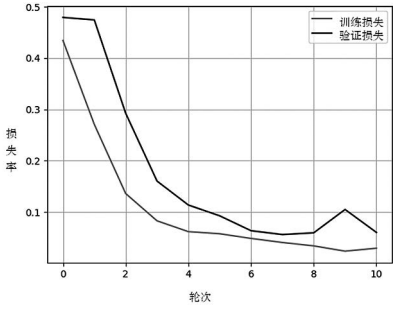


图8 数据集B在本文模型的损失率

Fig.8 Loss rate of data B in text model

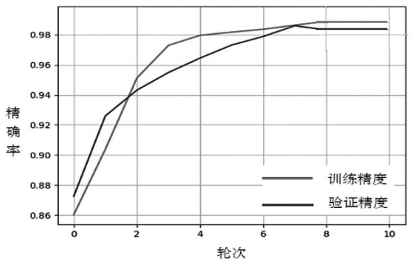


图9 数据集B在本文模型的精确率

Fig.9 Accuracy of data B in text model

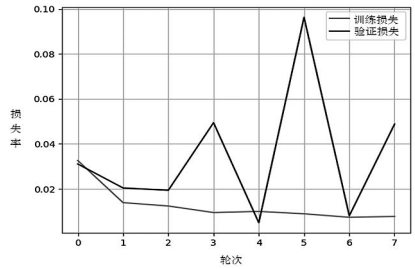


图10 数据集A在Inception V3的损失率

Fig.10 The loss rate of A in Inception V3

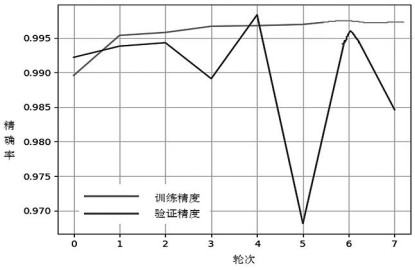


图11数据集A在Inception V3的准确率

Fig.11 Accuracy of A in Inception V3

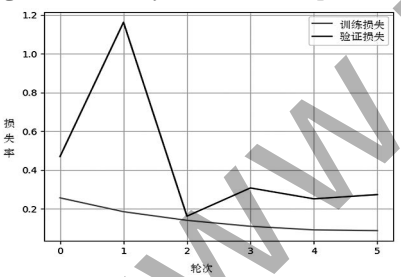


图12 数据集B在Inception V3的损失率

Fig.12 The loss rate of B in Inception V3

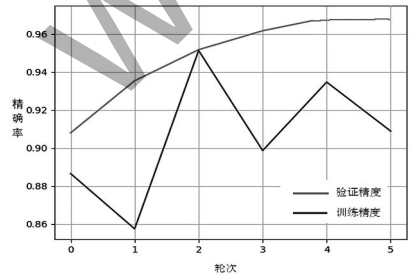


图13 数据集B在Inception V3的准确率

Fig.13 Accuracy of B in Inception V3

图6和图8是不同数据集在CNN_CBAM中的损失，能明显看出训练集的损失一直在降低，验证集的损失与训练集的损失高度契合。图7和图9是Malimg数据集和BIG2015数据集在CNN_CBAM中的准确率，发现在BIG2015数据集中第8轮epoch之后开始趋于稳定在98%左右。在Malimg数据集训练中第6轮的epoch有低谷出现，主要是测试样本有一些病毒软件的图像比较特殊，在训练集里没有包含测试集中的样本，导致无法准确学习到该样本的特征值。

图10和图11是Malimg数据集在Inception V3模型中的训练结果，从图中能明显发现在第5轮epoch中Loss和ACC大幅度升降，并且验证集的ACC不稳定，虽然最后训练达到了98.5%。同样，在BIG2015数据集中也是类似的情况，在Inception V3中BIG2015的准确率只达到了91.32%，但还是证明Inception V3对于处理病毒图像有一定效果。综上所述，无论是在Malimg数据集还是BIG2015数据集中，本文CNN_CBAM模型的训练精确度高于Inception V3，病毒识别效果有较大的提升。

4.4.2 不同数据集对比

本实验采用Malimg数据集、BIG2015数据集分别在Inception V3、本文CNN_CBAM模型中进行对比，最终结果为混淆矩阵，对角线上的数字在该行数字的值越大说明效果越好。图14至图17是各个模型的结果。

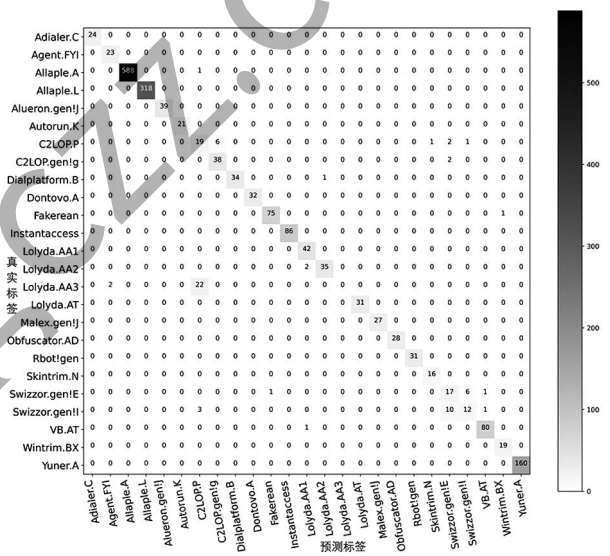


图14 数据集A在模型Inception V3的混淆矩阵

Fig.14 Confusion matrix of dataset A in model Inception V3

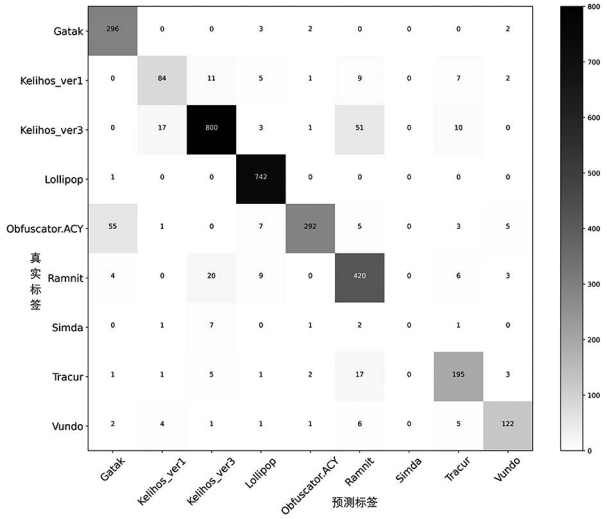


图15 数据集B在模型Inception V3的混淆矩阵

Fig.15 Confusion matrix of dataset B in model Inception V3

从图14和图15中可以看出，模型Inception V3在病毒图像Malimg数据集和BIG2015数据集中预测效果不佳，无法准确识别Malimg数据集类型为Lolyda.AA3和BIG2015数据集类型为Simda。这是因为Lolyda.AA3和Simda病毒种族的样本数量小，在训练时无法对其中的特征进行提取，造成无法识别，同时也说明了现在流行的深度学习模型不能通用于病毒图像的识别。

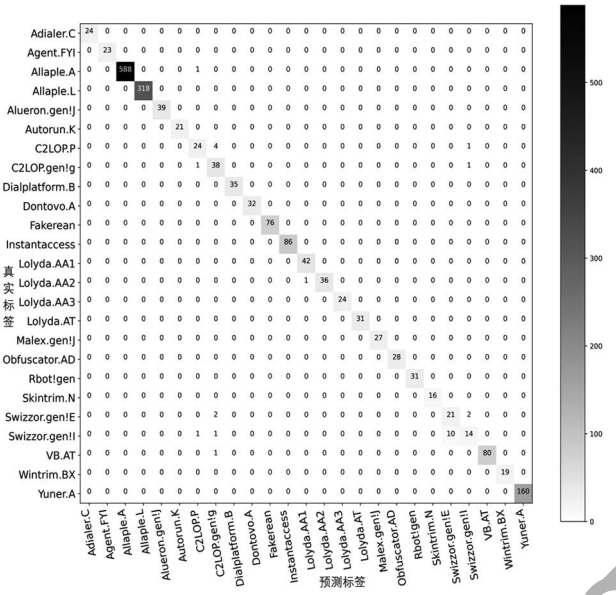


图16 数据集A在本文模型的混淆矩阵

Fig.16 Confusion matrix of dataset A in text model

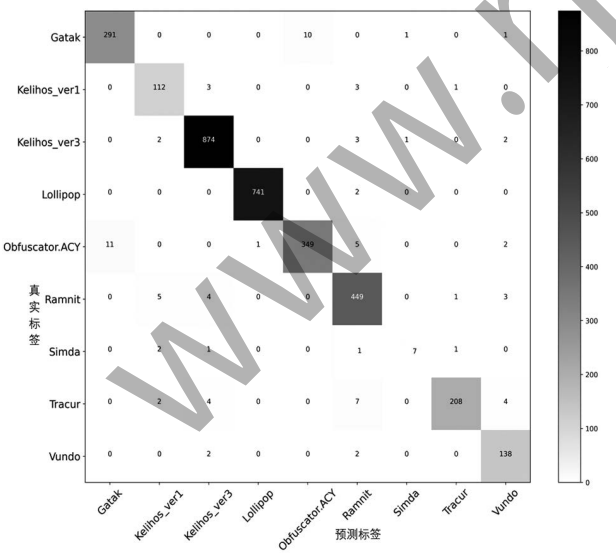


图17 数据集B在本文模型的混淆矩阵

Fig.17 Confusion matrix of dataset B in text model

从图16分析可知，本文设计的模型对Malimg数据集的分类有较高的准确率，在样本较少的Lolyda.AA3种族中也可以进行准确的识别，说明本文在卷积神经网络后加入的注意力机制可以准确地提取病毒灰度图像的特征，无论样本大小，能够准确地对需要特别关注的特征进行提取。

从图17中可以看出，本文设计的CNN_CBAM模型在BIG2015数据集中的效果比Inception V3模型的效果较好，只是无法准确预测到Simda病毒种类。由于该病毒种类的样本较少，因此训练时无法达到精确度非常高的水平，但本文模型对其他病毒种类的识别精确度相对较高，较Inception V3模型有很大的提升。

为了更好地说明本文提出的CNN_CBAM模型的检测效果，找到与本文相对应的BIG2015数据集，文献[22]用CNN_SVM对病毒软件进行检测与之对比，如表4所示。

表4 BIG2015在不同模型的评估标准

Tab.4 Evaluation criteria of BIG2015 in different models

模型	准确率	精确率	召回率	*F*1−*Score*
文献[22] CNN−SVM	0.7723	0.84	0.77	0.79
Inception V3	0.75	0.80	0.69	0.68
本文CNN_CBAM	0.94	0.89	0.85	0.86

从表4中可以发现，本文CNN_CBAM模型在BIG2015数据集中的准确率较CNN_SVM模型提升0.1677；精确率较Inception V3模型提升0.09，较CNN_SVM模型提升0.05；召回率提升至0.85；*F*1−*Score*提升至0.86，充分说明将病毒特征放入CNN_CBAM模型中能够更加准确地识别细微的特征，使得识别病毒更加准确。

5 结论(Conclusion)

本次实验说明了深度学习对病毒识别是很有帮助的，通过将病毒转化为可视化图像，能够让网络学习到更多人工无法提取的特征，并且本文加入了CNN_CBAM模型，通过将病毒特征放入通道注意力模块中，确定具体哪些病毒特征值得关注，再经过空间注意力机制，确定具体哪些位置的病毒特征需要加强学习，使得病毒特征能更好地被学习到。

但有些病毒软件的无效信息太多，直接转化为图像使得图像的信息量过大，在卷积神经网络中学习会影响学习的特征，以后在病毒软件检测可视化中可以考虑将病毒文件的特征提取后再进行可视化对其进行分类。

参考文献(References)

[1] YASSINE L, LANET J L, SOUIDI E M. A behavioural in−depth analysis of ransomware infection[J]. IET Information Security, 2020, 15(1):38−58.

[2] 瑞星.2020年上半年中国网络安全报告[EB/OL].[2020−04−28]. http://it.rising.com.cn/d/file/it/dongtai/20210113/2020.pdf.

[3] FIRDAUS A, ANUAR N B, KARIM A, et al. Discovering optimal features using static analysis and a genetic search

based method for Android malware detection[J]. Frontiers of Information Technology & Electronic Engineering, 2018, 19(06):712-737.

[4] MA X, GUO S, BAI W, et al. An API semantics-aware malware detection method based on deep learning[J]. Security and Communication Networks, 2019(1):1-9.

[5] CHARIKAR M S. Similarity estimation techniques from rounding algorithms[J]. Applied and Computational Harmonic Analysis, 2016, 2(3):380-388.

[6] LI D, LI Q, YE Y, et al. A framework for enhancing deep neural networks against adversarial malware[J]. IJCSA, 2020, 2(3):315-321.

[7] SANTOS I, BREZO F, NIEVES J, et al. Idea: Opcode-sequence-based malware detection[C]// Engineering Secure Software and Systems. Computer Science. Berlin, Heidelberg: Springer, 2010:6-13.

[8] KANG B J, YERIMA S Y, SEZER S, et al. N-gram opcode analysis for android malware detection[J]. IJCSA, 2016, 1(1): 231-255.

[9] ZOLKIPLI M F, JANTAN A. A framework for defining malware behavior using run time analysis and resource monitoring[C]// Software Engineering and Computer Systems. Communications in Computer and Information Science. Berlin, Heidelberg: Springer, 2011:199-209.

[10] ANDERSON B, QUIST D, NEIL J, et al. Graph-based malware detection using dynamic analysis[J]. Journal in Computer Virology, 2011, 7(4):247-258.

[11] LIN Q G, LI N, QI Q, et al. Using API call sequences for IoT malware classification based on convolutional neural networks[J]. International Journal of Software Engineering and Knowledge Engineering, 2021, 31(04):587-612.

[12] SHAHZAD F, SHAHZAD M, FAROOQ M. In-execution dynamic malware analysis and detection by mining information in process control blocks of Linux OS[J]. Information Sciences, 2013, 231(9):45-63.

[13] YOO S, KIM S, KIM S, et al. AI-HydRa: Advanced hybrid approach using random forest and deep learning for malware classification[J]. Information Sciences, 2020, 546(2):420-435.

[14] LORENZO A D, MARTINELLI F, MEDVET E, et al. Visualizing the outcome of dynamic analysis of android malware with VizMal[J]. Information Security Technical Report, 2020, 50(2):1-9.

[15] RIECK K, TRINIUS P, Willems C, et al. Automatic analysis of malware behavior using machine learning[J]. Journal of Computer Security, 2011, 19(4):639-668.

[16] ZHANG T R, YANG L X, YANG X F, et al. Dynamic malware containment under an epidemic model with alert[J]. Physica A: Statistical Mechanics and its Applications, 2017, 3(470):249-260.

[17] NATARAJ L, KARTHIKEYAN S, JACOB G, et al. Malware images: Visualization and automatic classification[C]// International Conference Proceeding Series (ICPS). 2011 International Symposium on Visualization for Cyber Security. New York, USA: ACM, 2011:11-20.

[18] SHANKARPANI M K, KANCHERLA K, MOVVA R, et al. Computational intelligent techniques and similarity measures for malware classification[J]. Computational Intelligence for Privacy and Security, 2012, 394(1):215-236.

[19] WOO S, PARK J, LEE J Y, et al. CBAM: Convolutional block attention module[J]. European Conference on Computer Vision, 2018, 10(8):168-203.

[20] KAGGLE. Microsoft malware classification challenge(BIG2015) [EB/OL]. [2015-3-19]. https://www.kaggle.com/c/malware-classification.

[21] SWAMI A, JAIN R. Scikit-learn: Machine learning in python[J]. Journal of Machine Learning Research, 2013, 12(10):2825-2830.

[22] AGARAP A F. Towards building an intelligent anti-malware system: A deep learning approach using support vector machine (svm) for malware classification[J]. ArXiv Preprint, 2017, 8(1):45-52.

作者简介：

赵晨洁(1995-)，女，硕士生.研究领域：图像处理，深度学习.

左　羽(1962-)，男，硕士，教授.研究领域：网络安全，机器学习，图像处理，深度学习.

崔忠伟(1980-)，男，博士，副教授.研究领域：物联网技术，机器学习.

李亮亮(1994-)，男，硕士生.研究领域：图像处理，深度学习.

吴　恋(1986-)，女，博士，副教授.研究领域：通信与信息系统，机器学习，图像处理.

王永金(1994-)，男，硕士生.研究领域：图像处理，深度学习.

韦萍萍(1975-)，女，博士，副教授.研究领域：图像处理，深度学习.