基于改进 CNN-LSTM 融合的僵尸网络识别方法

卢法权 陈丹伟
(南京邮电大学计算机学院、软件学院、网络空间安全学院 江苏 南京 210023)

摘 要 P2P 及 fast-flux 等技术的出现使僵尸网络隐蔽性大大增强。传统人工提取特征的识别方法愈发困难并且识别精度低。该文设计一种新的基于 CNN 及 LSTM 融合网络结构,使用改进激活函数和网络结构的卷积神经网络检测空间特征,并使用长短时记忆网络检测时序特征,将两种特征并联融合用于识别僵尸网络。实验表明,该方法在精度和召回率等方面可满足僵尸网络识别需求。

关键词 僵尸网络 卷积神经网络 长短时记忆网络 特征并联融合 激活函数

中图分类号 TP3 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2024.03.050

BOTNET IDENTIFICATION METHOD BASED ON IMPROVED CNN-LSTM FUSION

Lu Faquan[1] Chen Danwei[2]
(*College of Computer, College of Software, College of Network Space Security, Nanjing University of Posts and Telecommunications, Nanjing 210023, Jiangsu, China*)

Abstract The emergence of P2P and fast-flux technology makes botnet more covert. The traditional recognition method of feature extraction is more and more difficult, and the recognition accuracy is low. In order to solve the above problems, this paper designs a new fusion network structure based on CNN and LSTM. In this method, convolutional neural network with improved activation function and network structure was used to detect spatial features, and LSTM network was used to detect temporal features. Experimental results show that the method can meet the needs of Botnet identification in terms of accuracy and recall.

Keywords Botnet Convolutional Neural Network (CNN) Long and short-term memory (LSTM) Feature parallel fusion Activation function

0 引 言

随着计算机技术的飞速发展,网络技术在社会各领域都有着非常广泛而深入的应用。与此同时,安全问题在网络中日渐凸显。僵尸网络(Botnet)就是不法分子利用网络进行非法牟利的一种常用攻击手段。僵尸网络是指采用一种或多种手段,传播僵尸程序感染大量主机,并通过命令与控制(C&C,Command and Control)信道对僵尸主机进行控制与操作所组成的网络[1]。僵尸网络危害巨大,可以用于执行恶意活动,例如发起分布式拒绝服务(DDoS,Distributed Denial of Service)、发送垃圾邮件、扩散恶意软件、监听用户敏感信息、虚拟货币挖掘等。2012 年产生的 Zeus 僵尸网络及其采用 fast-flux 技术的变种 Zbot 是窃取银行信息的首选木马,通过浏览器中的键盘记录器攻击工作,接收所有的网络银行和信用卡详细信息,在其活动的峰值时间段,Zeus 危及了美国银行、NASA、思科和亚马逊等公司[2]。2016 年的 Mirai 僵尸网络通过不断扫描互联网,获取物联网设备的 IP 地址,对 OVH、Dyn 等公司发起大规模 DDoS 攻击[3]。2017 年爆发了 PBot 恶意软件攻击。国家互联网应急中心的《CNCERT 互联网安全威胁报告》指出,2019 年 12 月,我国境内就有近 129 万个 IP 地址所对应的主机被木

收稿日期:2021-01-13。**卢法权**,硕士生,主研领域:信息安全。**陈丹伟**,教授。

马或僵尸程序控制[4]。僵尸网络正不断挑战网络的安防能力。

1　相关工作

目前来说,僵尸网络检测分为基于特征和基于异常的僵尸网络检测方法。

基于特征的僵尸网络检测方法也称为误用检测。该方法检测与已知异常行为之间的匹配程度。搜集异常流量的行为特征,创建相干的特征库,当监测的网络通信行为与库中的记录相匹配时,系统就判断这种行为是入侵。文献[5]分析僵尸网络感染过程的三个必不可少的行为(启动、准备、攻击),通过虚拟机技术统一监听可疑进程的系统活动和网络流量,并与三个行为进行匹配与关联,从而实现僵尸网络的检测。文献[6]借助 Snort 进行自定义规则设置,提出了 Bothunter 僵尸网络检测系统,通过状态匹配进行僵尸网络的检测。这种方法对于已知的攻击识别率很高,但是无法识别未知的攻击。同时,随着恶意流量的不断增加和更新,维护攻击特征库也是一个耗时耗力的困难工作。

基于异常的恶意流量分类方法利用机器学习,根据所采集数据的样本特征与标签之间的关系,建立恶意流量识别模型。传统的机器学习方法主要有决策树、朴素贝叶斯、支持向量机等算法。文献[7]提出了利用 C4.5 决策树、朴素贝叶斯和贝叶斯网络算法设计分类器,针对基于 IRC(实时网络聊天)协议僵尸网络具有较低的误报率。文献[8]采用支持向量机(SVM)算法来区分正常网络域名访问和僵尸网络域名访问,从而进行识别与检测。文献[9]对比了 C4.5、Random-Forest 等不同机器学习算法对僵尸网络流量的检测性能。

传统的机器学习方法难以表示复杂的非线性函数关系,处理复杂问题的泛化能力有限,导致对于上述僵尸网络识别效率不高,同时这些算法需要有很强的先验知识,人工提取特征也需要花费大量的时间。而深度学习能够通过多层神经网络处理后自动提取特征,并且能发现人工无法识别的细微特征差异,用简单模型即可完成复杂的分类任务。

因此,针对上述问题,本文提出一种基于深度学习的僵尸网络识别方法,首先对卷积神经网络结构和激活函数进行改进,提取细粒度空间特征,再利用长短时记忆网络提取时序特征,并检测域名、IP 地址的变换,最终将特征进行并联融合使模型具有较高的识别率。

2　相关原理

2.1　P2P 僵尸网络

近年来,僵尸网络不断演变进化,传统的基于 IRC 和 HTTP 协议的僵尸网络需要依靠中心控制节点通过命令与控制信道发送指令与相互通信,容易被安全人员检测到中心服务器,当服务器被销毁后,整个僵尸网络就会中断,无法运转。因此出现了基于 P2P 协议的僵尸网络。P2P 僵尸网络各节点之间成为了点对点对等模式,使得当某些僵尸主机被查杀后并不会影响整个网络,会立即有新的节点作为中心控制节点顶替上来,由此克服了传统僵尸网络单点失效的问题,加大了检测的难度。但是由于僵尸网络在通信行为上的特点与正常网络流量有所区别,即僵尸网络系统中各节点间需要不断进行通信,所以其具有网络流数据包较小且周期较短等特点,导致数据包数量小于正常流量,数据包字节数小于正常流量,每秒字节数也要小于正常流量,这是较为显著的区别。部分特征如表 1 所示。

表 1　部分网络流特征

特征
数据包数量
数据包字节数
数据包到达时间
字节速率
持续时间

2.2　Fast-flux 技术

Fast-flux 技术是指不断改变域名和 IP 地址映射关系的一种技术,即不停地变换与域名相关的 IP 地址来让访问的同一域名,但其实是不同的服务器主机。攻击者通过控制底层域名服务器返回具有成百上千条 IP 地址的 IP 地址池来实现,从而极大地增强了隐蔽性,提高了追溯的难度。但是由于其域名和 IP 地址映射关系是不断改变的,当前控制服务器需通知网络内的其他僵尸主机 IP 地址发生了变化,这就导致此类僵尸网络的通信流量会出现明显的上下文关系。同时,一般会使用依序循环方法,即 IP 地址循环变化也有一

定的规律,产生的通信流量也是上下文有关联的,不像正常访问的流量,上下文没有明显关联,也没有那么频繁的 IP 地址变换。因为 IP 地址的不停变换以及随机感染的特性,在域名中的 A 记录数更多、ASN 自治系统分布更广。因为要查询底层服务器,其响应时间要更长,部分特征如表 2 所示。

表 2 部分域名特征

特征
A 记录数
ASN 数
响应包大小
请求响应时间

3 基于深度学习的僵尸网络识别方法

3.1 设计思路与方法有效性

本文鉴于深度学习在图像识别领域和自然语言处理领域优秀的能力,将其迁移到僵尸网络检测上来。根据第 2 节分析的特征,僵尸网络在通信行为的特征如时间、交换数据包数等,与正常网络流量有所区别。这就使得其特征向量转化为图片后与正常流量有较大区别,可将其转化为空间特征上的差异性进行检测。本文采用 CNN 提取流量的空间特征,省去人工提取的繁琐,改进 3×3 卷积核能够有效减少参数量,让网络模型更加轻量化。这样既更换随机正则化 GELU 激活函数保留了神经元的概率性,同时也保留了对输入的依赖性,不会随意丢弃特征,保证了特征的细节性,使其对 P2P 等隐蔽性较强的僵尸网络有更好的识别效果。

针对采用 fast-flux 技术的域名变换僵尸网络,在域名中的 A 记录数、ASN 数等特征要多于正常流量,使用 LSTM 对网络流的时序特征进行提取。LSTM 对于长距离依赖关系有优势,具有长程记忆能力,对于时序特征的前后文关系有优势。

最终将两者特征进行并联融合,不同于用 CNN 提取的空间特征再送入 LSTM 提取时序特征的串联模式,可能造成空间特征丢失以及时序特征提取的不全面等问题。并联融合能够保证空间特征和时序特征的完整性,以此提高整体的检测效果。方法主要包括四个阶段:数据预处理、网络模型构建、网络模型训练、僵尸网络识别。整体框架如图 1 所示。

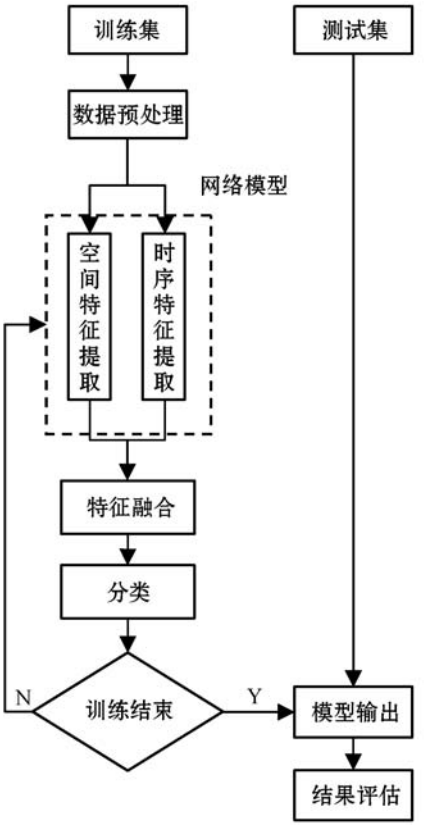

图 1 整体框架流程

3.2 空间特征提取

3.2.1 数据预处理

数据预处理是进行网络流量识别的基础工作,其目的是将网络原始流量提取有效部分并转化为满足 CNN 输入格式的数据。数据预处理可分为

步骤 1(流量重组与切分):首先读取 pcap 文件数据,分析每一条流量的结构,若有数据流负载信息,则提取出有效载荷字节,并保存数据。本文对网络流量按照 1 024 字节进行统一长度进行处理,即长度超过 1 024 字节的流量进行截取,长度小于 1 024 字节的流量在其末尾进行补 0 操作,补齐 1 024 字节长度。这样做能将多个文献中提到有效载荷最关键的部分集中在前面部分,因为前面部分包含了建立连接和前段数据包,更能体现流量类型特征,后面部分则更多的是数据内容。因此可以考虑将其直接截掉,当长度超过 1 024(32×32)字节时,精度不会再有显著提升。[16-17,21] 另一方面,卷积神经网络的输入大小必须是相同的,截取固定长度的网络流量也符合卷积神经网络的输入要求。同时,由于仅使用前 1 000 字节,使得该方法相比传统的机器学习分类方法要更加轻量化。

步骤 2(数据归一化):数据归一化也是卷积神经网络数据预处理环节的基础工作之一。步骤 1 提取的长度为 1 024 字节的网络流量,每个字节的取值范围都在[0,255],将其构造为一个向量,每个字节对应一

个分量,并将每个分量的数值除以 255,让其取值范围归一到[0,1]区间。这样做的优势在于,原不同特征数据的范围可能会有很大的差别,归一化可以使数据有相同的分布,网络学习、收敛就会越快,不容易发生不收敛或者梯度消失的情况。

步骤 3(标签标注):经过前面的处理后,需要为样本打上标签,标注其属于哪种类型的网络流量。对于网络流量这种离散型数据而言,一般采用 one-hot 编码方式进行标签标注,主要是采用 N 位状态寄存器来对 N 个状态进行编码,比如说当 $N=3$ 时,即有三种网络类型,$[1,0,0]^{T}$ 就代表该网络流属于第 1 种类型。使用 one-hot 编码的优势在于在分类任务中,特征之间距离的计算或相似度的计算是非常重要的,使用 one-hot 编码,将离散特征的取值扩展到了欧氏空间,会让特征之间的距离计算更加合理。

3.2.2　卷积神经网络结构设计

在第 2 节中已经介绍了卷积神经网络基础知识,本节具体介绍各层所使用的具体方法设计。

1) 卷积层。卷积的具体操作是自上而下、自左向右的顺序扫描整张图片,目的是为了提取图像的局部特征,并且能够在高层将信息整合。一般而言,很多文章通常采用 3×3 或 5×5 大小的卷积核,且仅简单堆叠二至三层,这样网络的性能是有限的。本文仅使用 3×3 大小的卷积核,采用模块化堆叠方式,将两个 3×3 大小的卷积核作为一个模块堆叠三层,并在中间穿插池化层。这样做的理由是,3×3 是最小的能够捕获像素八邻域信息的尺寸,采用 3×3 的小卷积核连续卷积两次就可达到 5×5 卷积核一次提取特征图的能力,保证感受野大小不变的同时还减少了 28% 的参数量,增强网络的性能。与此同时,多个 3×3 的卷积层比一个大尺寸卷积层有更多的非线性(更多层的激活函数),更丰富的特征,更强的判别能力,对于复杂任务就有更强的拟合能力。

2) 激活函数。激活函数对于深度学习模型非常重要,若不使用激活函数的话,网络的每层都只能做线性变换,这就导致了模型的表达能力通常不够。而激活函数则带来了非线性变化,使得神经网络几乎可以拟合任意函数,这就极大地增强了网络模型的表达能力。下面介绍本文将比较的三种激活函数。

(1) tanh。tanh 函数的值域为(-1,1),它的表达式如式所示:

$$\tanh(x)=\frac{1-e^{-2x}}{1+e^{-2x}} \tag{5}$$

其图像如图 2 所示。

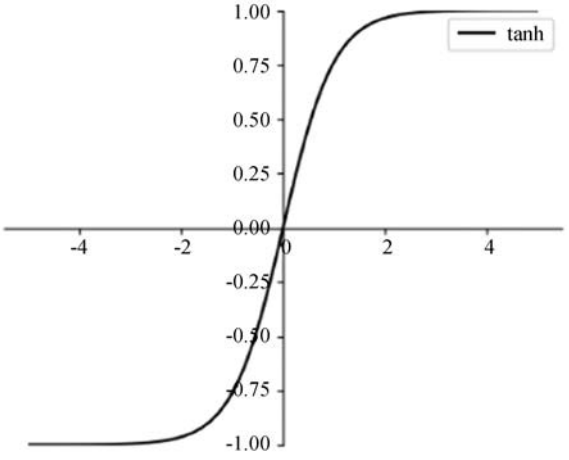

图 2　tanh 图像示意图

可以看到,tanh 实际上是 Sigmoid 函数的变形,但是 tanh 的输出是“零为中心”的,因此,在实际应用中,会比 sigmoid 要更好一些,但是在饱和神经元的情况下,它仍然没有解决梯度消失的问题,同时它进行的是指数运算,在速度上会有一定的影响。

(2) ReLU。近年来,ReLU 函数变得越来越受欢迎,它的值域为[0,+∞],它的表达式如式所示:

$$\mathrm{ReLU}(x)=\begin{cases}x & x\geqslant 0\\ 0 & x<0\end{cases} \tag{6}$$

其图像如图 3 所示。

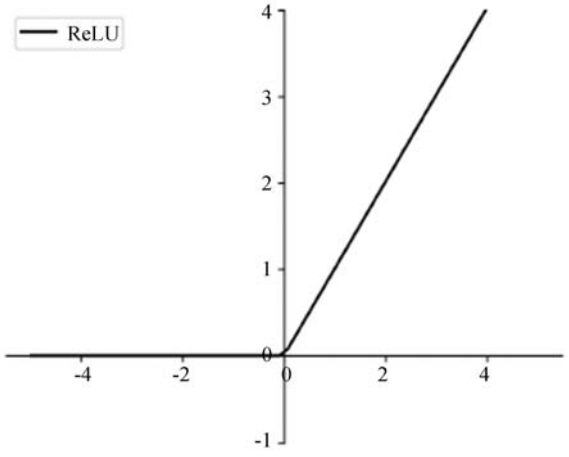

图 3　ReLU 图像示意图

ReLU 函数有效缓解梯度消失的问题,在训练时也比较稳定,对于输入只需判断是否大于 0,计算速度非常快。

但是 ReLU 函数可能会出现神经元死亡(Dead ReLU Problem)的情况。由于负半轴的输出为 0,那么反向传播的梯度也为 0,权重就不会更新,导致神经元不再学习,即神经元死亡,就会导致特征学习的丢失。在实际训练中,如果学习率设置太高,就会导致较多的神经元死亡,因此必须设置一个合理的学习率进行训练。

(3) GELU。高斯误差线性单元激活函数(GELU)

在 2016 年被提出,是一种高性能的神经网络激活函数,其非线性变化是一种符合预期的随机正则变换方式。它在激活中引入了随机正则的思想,是一种对神经元输入的概率描述,直观上更符合自然的认知。其公式如式所示:

$$GELU(x) = xP(X \leqslant x) = x\phi(x) \tag{7}$$

该公式的意义为,对于每一个输入 x,它会乘上一个伯努利分布 Bernoulli($\Phi(x)$)。这么选择是因为神经元的输入趋向于正态分布,这么设定使得当输入 x 减小时,输入会有一个更高的概率被归零,也就是 dropout 掉。这样不同于 ReLU 输入少于零就会被归零,使得 GELU 不仅保留了概率性,同时也保留了对输入的依赖性。在实际使用时,其使用的公式如式(8)所示:

$$GELU(x) = 0.5x(1 + \tanh[\sqrt{2/\pi}(x + 0.044\ 715x^3)]) \tag{8}$$

其图像如图 4 所示。

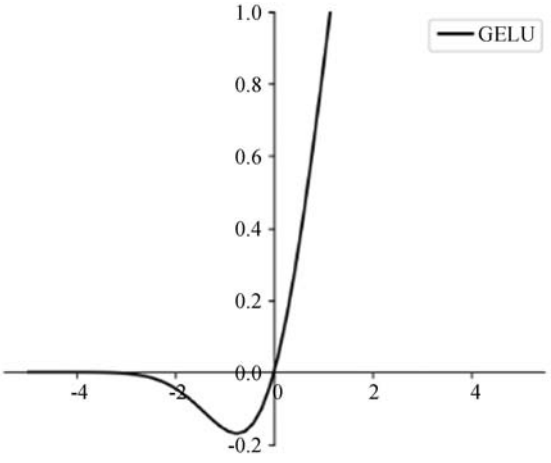

图 4 GELU 图像示意图

3)池化层。池化层是周期性地插入卷积层之间,且一般紧跟在卷积层之后,本文使用最大值池化(Max-Pooling),即取局部感受野中值最大的点。Max-Pooling 能够减小卷积层参数误差造成估计均值的偏移,更多地保留纹理信息。这符合网络流量灰度图的特征,我们关注更多的是不同流量的不同纹理特征,因此,Max-Pooling 更适合我们的网络。

3.3 时序特征提取

为了进一步提升对僵尸网络的检测效果,还要采用 LSTM 对网络流量进行时序特征的提取,这是由于新型僵尸网络的特征决定的。

3.3.1 数据预处理

数据预处理与 3.2.1 节大体相同,LSTM 本身就用来处理自然语言任务,而网络流结构与之大体相同,故不需要图像化这一步骤。字节组成数据包,数据包又组成网络流。本文对每条网络流截取前 8 个数据包,每个数据包取 100 个字节,若长度不够,则在末尾补 0×00。接下来还需要对数据包向量进行编码,每个数据包为 100 维向量 $\boldsymbol{\alpha} = (a_1, a_2, \cdots, a_i)$, $i = 1, 2, \cdots, 100$, $0 \leqslant a_i \leqslant 255$,采用 one-hot 编码将每个字节编码为 256 维的向量。这样做是防止在训练时,网络模型将字节这种离散型值当作连续型数值,从而影响参数更新,降低识别率。

3.3.2 LSTM 结构设计

LSTM 结构相较 CNN 更为简单,只需设计 LSTM 内细胞(单元)个数即可。但是 LSTM 训练时间较长,作为辅助的域名特征判断,不适宜设计太复杂,否则训练时长大大增加,导致整个网络的时间性能大幅减弱。结构设计如下:

LSTM 层 L1:由 100 个单元构成,输出为 100 个 256 维向量。

全连接层 FC1:由 256 个神经元组成,输出为 256 维向量。

LSTM 层 L2:由 8 个单元构成,输出为 8 个 128 维向量。

全连接层 FC2:由 128 个神经元组成,输出为 128 维向量。

3.4 网络整体构建

整体网络结构及每层输出如图 5 所示。

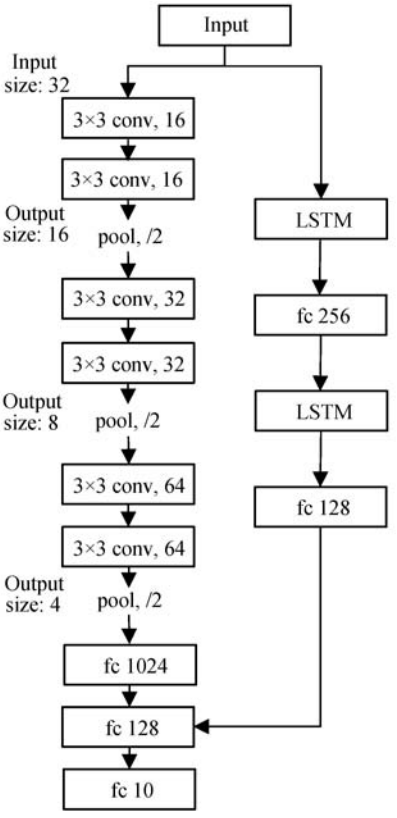

图 5 网络整体结构示意图

卷积神经网络结构:

1) 卷积层 C1:两个 3 ×3 卷积层串联,通道数为 16,输出为 32 ×32 ×16。

2) 池化层 S1:进行一次 2 ×2 的最大值池化操作,输出为 16 ×16 ×16。

3) 卷积层 C2:两个 3 ×3 卷积层串联,通道数为 32,输出为 16 ×16 ×32。

4) 池化层 S2:进行一次 2 ×2 的最大值池化操作,输出为 8 ×8 ×32。

5) 卷积层 C3:两个 3 ×3 卷积层串联,通道数为 64,输出为 8 ×8 ×64。

6) 池化层 S3:进行一次 2 ×2 的最大值池化操作,输出为 4 ×4 ×64。

7) 全连接层 D1:由 1 024 个神经元构成,与 S3 层全连接,输出为 1 024 维向量。

8) 全连接层 D2:由 128 个神经元构成,与 D1 层全连接,输出为 128 维向量。

LSTM 结构在 3.3.2 节已经说明,此处对时序特征的 FC2 层 128 维向量与空间特征的 D2 层 128 维向量进行并联融合,组成 256 维向量,输出到最后一层全连接层,输出 10 维向量。

3.5　网络训练过程

(1) 分类函数选择:本文使用 softmax( )作为激活函数。softmax( )函数通常在多分类任务中使用,作为最后的"分类器",其公式如下:

$$a_i = \frac{e^{z_i}}{\sum_k e^{z_k}} \tag{9}$$

式中:z_i 表示网络的第 i 个输出;a_i 代表 softmax 的第 i 个输出值。通俗地说,softmax( )函数能够将网络的输出映射成为(0,1)之间的值,并且这些值的累和为 1(满足概率的性质),那么概率值最大的(也就是值对应最大的)节点,就是网络最后的预测目标。

(2) 损失函数选择:本文采用交叉熵损失函数 C(CrossEntropyLoss),其公式如式(10)所示。

$$C = -\sum_{k=1}^{N}(p_k \times \log q_k) \tag{10}$$

(3) 优化函数选择:本文采用 Adam 优化器,其本质上是带有动量项的 RMSprop,它利用梯度的一阶矩估计和二阶矩估计动态调整每个参数的学习率。它的优点主要在于经过偏置校正后,每一次迭代学习率都有个确定的范围,使得在训练过程中参数更新比较平稳。

一轮网络的训练由一次前向传导过程和一次反向传播过程组成。首先是经过整个模型逐层传递学习的特征值,然后给出网络模型的预测。再通过损失函数计算出预测值与真实值之间的损失,这是一轮前向传导。反向传播过程(BP,back propagation)将根据损失值,通过优化函数对整个网络中的参数进行更新。

4　实验与分析

4.1　数据集

本文实验采用的数据集由正常网络流量和僵尸网络流量两部分组成。这样组成的数据集更加贴合日常实际使用的情况。其中,正常网络流量为使用 Wireshark 流量采集对本地的主机路由端口进行日常流量的采集,包含了 5 种常用的正常流量类型。僵尸网络流量则从 CTU[23] 数据集中选取。该数据集包含恶意流量捕获、正常流量捕获和混合捕获。恶意流量捕获则是由 Stratosphere IPS 项目的一个名为 Malware Capture Facility 的项目长期进行的,著名的 CTU-13 数据集便出自其手。本文从中选取了 5 种具有代表性的不同类型僵尸网络流量,包含有两种 P2P 类型的僵尸网络流量,两类 IRC 和一类 HTTP 类型的僵尸网络流量。包括第 1 节提到的 Zeus 银行木马,通过攻击银行键盘记录器来窃取客户的银行卡信息等。其中 Zeus 和 Virut 是同时使用了 fast-flux 技术的僵尸网络。原始流量保存为 pcap 格式文件,具体的数据集类型如表 3 所示。

表 3　数据集组成表

类别	名称	种类
僵尸网络	Neris	IRC
	Rbot	IRC
	Virut	HTTP
	Nsis	P2P
	Zeus	P2P
正常流量	Gmail	电子邮件
	Weibo	社交网络
	WOW	游戏
	MySQL	数据库
	FaceTime	即时通信

还要对 pcap 文件进行预处理,本文基于会话对 pcap 文件进行切分,数据流是具有相同 < 源 IP,源端

口、目的 IP、目的端口、协议 > 五元组的数据包集合，会话则是双向数据流，以此进行切分。当数据体量较大时，训练较为费时，故切分完对每种网络流量随机选择 5 000 条，僵尸网络流量和正常流量各 25 000 条，整个数据集总共 50 000 条。选取 80% 作为训练集，剩余 20% 作为测试集。后续对流量进行符合神经网络输入形式的预处理在 3.2.1 节和 3.3.1 节已经分别介绍。

4.2 实验配置

实验使用 Python 语言进行编程，采用 PyTorch 框架进行整个网络模型的搭建与训练。实验硬件方面，CPU 为 8 核 Intel 9700k 3.6 GHz，内存 16 GB。GPU 方面，使用了一块 Nvidia RTX 2070 作为训练加速器。本实验采用 CUDA 和 Nvdia GPU 进行训练，相比于 CPU 极大地缩短了训练时间。

训练时，基本框架与流程在第三章已经详细介绍过，具体的参数方面，在空间特征学习中，数据流选取长度为 1 024 字节，即输入图片大小为 32 ×32，时序特征学习中，每条数据流取前 8 个数据分组，每个数据分组取前 100 字节。多篇文献指出，当字节数超过 1 024 (32 ×32)时，模型准确率不再有显著提高；时序特征方面，选取 8 和 100 两个值具有较高的准确率和较低的训练时长，并且基本涵盖了网络流的主要内容，若取值再往上增加，准确率没有显著提升同时训练时间成本大幅增加。mini-batch 大小选择 64，损失函数为交叉熵损失函数 CrossEntropyLoss，优化器为 Adam 优化器，初始学习率设置为 0.002，总训练回合数为 50 epochs。为了验证本文方法的有效性，共设置两组实验如下：

1）不同激活函数对于识别性能的影响。

2）与其他传统机器学习方法的识别性能比较。

4.3 实验指标

为了能够对本文方法的优劣进行评价，采用目前常用的评价标准：精度 A(Accuracy)、误报率 F(FPR)、召回率 R(Recall)。精度 A 用来衡量模型对于数据集的整体识别效果，值越高，识别性能越好；误报率 F、召回率 R 则是用来衡量单一类别的识别效果。相关公式定义如下：

$$A = \frac{T_P + T_N}{T_P + F_P + F_N + T_N} \tag{11}$$

$$F = \frac{F_P}{F_P + T_N} \tag{12}$$

$$R = \frac{T_P}{T_P + F_N} \tag{13}$$

4.4 结果分析

在比较激活函数识别性能实验部分，使用介绍的网络结构，保持整体结构不变，只更换激活函数，共进行 3 次实验。实验结果见图 6、图 7 所示。可以看出，tanh()这种饱和激活函数的精确度最低，识别性能最差，且收敛速度也很慢。它无法很好地解决梯度消失的问题，在网络训练轮次到达一定程度的时候，梯度传播的逐渐消失导致参数的更新优化不到位，因此无法获得良好的性能。对于 ReLU 这个深度学习普遍使用的激活函数，精确度已经达到一个不错的标准，然而 ReLU 在负半轴会导致神经元死亡的缺陷，使其精度无法达到更高的层次。而 GELU 激活函数识别性能最好，符合之前分析的加入了统计的特性，比 ReLU 要高出 1 个点左右，且收敛速度相比之也更快。

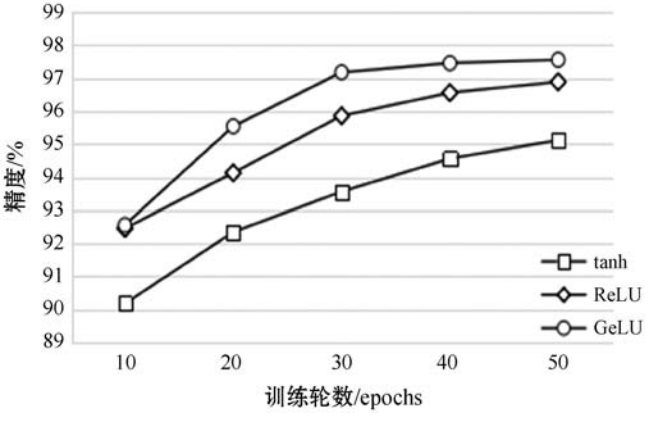

图 6 三种激活函数 Acc 值

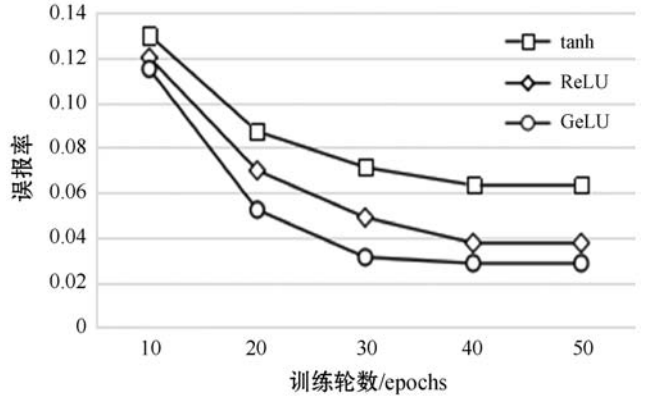

图 7 三种激活函数 FPR 值

对于模型整体的识别性能实验如表 4 所示。其中，SVM 使用机器学习库 scikit-learn 搭建，文献[21]为未经改进的普通 CNN 结构，CNN 和 LSTM 为本文框架提取出来的单一框架。可以看出，传统的机器学习方法对于隐蔽性较强的僵尸网络而言，性能较差。对于使用单一特征而言，CNN 的效果要比 LSTM 更好，说明空间特征相比时序特征识别效率更高，能更好地反映网络流量特性，起主导作用。

表 4　整体模型效果对比(%)

方法	精度	召回率
SVM	93.82	94.12
文献[21]	96.65	97.08
CNN	97.67	97.83
LSTM	95.76	95.88
本文	98.56	99.19

而本文方法对 CNN 结构细节方面进行改进,更换小卷积核和高性能激活函数,使其能够有更细粒度的特征识别,可以看出相较文献[21]只有两层结构的普通 CNN 效果要更好。本文最终整体模型将空间特征再融合了 LSTM 提取的时序特征,最终取得了良好的性能。

5　结　语

本文提出一种将空间和时序特征融合进行识别新型僵尸网络的方法,并且对 CNN 的结构和激活函数进行改进,采用小卷积核提取更细粒度的特征,并采用高性能 GeLU 激活函数,具有更强的非线性拟合能力,再辅以 LSTM 进行时序特征提取,最后进行特征并联融合,取得了 98.56% 的整体精度。但还有一些不足需要改进,整体框架模型较大,在时间性能上还有提升空间,主要表现为 LSTM 的训练较为耗时,以及僵尸网络数据难以采集,都是使用数据集的数据进行训练、测试,接下来要继续考虑小样本的模型学习。

参　考　文　献

[1] 金双民,郑辉,段海新. 僵尸网络研究系列文章之—僵尸网络研究概述[J]. 中国教育网络,2006(6):51 – 54.

[2] 崔丽娟,马卫国,赵巍,等. 僵尸网络综述[J]. 信息安全研究,2017,3(7):589 – 600.

[3] Kolias C, Kambourakis G, Stavrou A, et al. DDoS in the IoT: Mirai and other botnets[J]. Computer,2017,50(7): 80 – 84.

[4] 温森浩. 2019 年 12 月网络安全监测数据分析[J]. 互联网天地,2020(1):55 – 58.

[5] Liu L, Chen S Q, Yan G H, et al. Bottracer: Execution-based bot-like malware detection[C]//11th International Conference on Information Security,2008:97 – 113.

[6] Gu G F, Porras P, Yegneswaran V, et al. BotHunter: Detecting malware infection through IDS-driven dialog correlation[C]//16th USENIX Security Symposium,2007:1 – 16.

[7] Livadas C, Walsh R, Lapsley D, et al. Using machine learning techniques to identify botnet traffic[C]//31st IEEE Conference on Local Computer Networks,2006:967 – 974.

[8] Alieyan K, Anbar M, Almomani A, et al. Botnets detecting attack based on DNS features[C]//International Arab Conference on Information Technology,2018:1 – 4.

[9] Stevanovic M, Pedersen J M. An efficient flow-based botnet detection using supervised machine learning[C]//International Conference on Computing, Networking and Communications,2014:797 – 801.

[10] 周志华. 机器学习[M]. 北京:清华大学出版社,2015:114 – 115.

[11] 崔鹏飞,裘玥,孙瑞. 面向网络内容安全的图像识别技术研究[J]. 信息网络安全,2015,15(9):154 – 157.

[12] Huang J, Wang P, Zang T N, et al. Detecting domain generation algorithms with convolutional neural language models[C]//17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 2018: 1360 – 1367.

[13] Zhao L X, Cai L J, Yu A, et al. A novel network traffic classification approach via discriminative feature learning[C]//35th Annual ACM Symposium on Applied Computing, 2020:1026 – 1033.

[14] Aceto G, Ciuonzo D, Montieri A, et al. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges[J]. IEEE Transactions on Network and Service Management,2019,16(2):445 – 458.

[15] 陈雪娇,王攀,俞家辉. 基于卷积神经网络的加密流量识别方法[J]. 南京邮电大学学报(自然科学版),2018,38(6):36 – 41.

[16] 吴迪,方滨兴,崔翔,等. BotCatcher:基于深度学习的僵尸网络检测系统[J]. 通信学报,2018,39(8):18 – 28.

[17] 牛伟纳,蒋天宇,张小松,等. 基于流量时空特征的 fast-flux 僵尸网络检测方法[J]. 电子与信息学报,2020,42(8):1872 – 1880.

[18] 王伟. 基于深度学习的网络流量分类及异常检测方法研究[D]. 合肥:中国科学技术大学,2018.

[19] Hendrycks D, Gimpel K. Bridging nonlinearities and stochastic regularizes with Gaussian error linear units[EB]. arXiv:1606.08415v2,2016.

[20] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[EB]. arXiv:1409.1556, 2015.

[21] 冯文博,洪征,吴礼发,等. 基于卷积神经网络的应用层协议识别方法[J]. 计算机应用,2019,39(12):3615 – 3621.

[22] Zhauniarovich Y, Khalil I, Yu T, et al. A survey on malicious domains detection through DNS data analysis[J]. ACM Computing Surveys,2018,51(4):1 – 36.

[23] MCFP Dataset-Malware Capture facility project[EB/OL]. [2022 – 12 – 22]. https://mcfp.felk.cvut.cz/publicDatasets/datasets.html.