·网络空间安全·　　文章编号：1000-3428(2022)04-0148-10　　文献标志码：A　　中图分类号：TP309

基于CNN-SIndRNN的恶意TLS流量快速识别方法

李小剑[1]，谢晓尧[1,2]，徐　洋[2]，张思聪[2]

（1.贵州师范大学 数学科学学院 贵阳 550001；2.贵州师范大学 贵州省信息与计算科学重点实验室 贵阳 550001）

摘　要：传统浅层机器学习方法在识别恶意TLS流量时依赖专家经验且流量表征不足，而现有的深度神经网络检测模型因层次结构复杂导致训练时间过长。提出一种基于CNN-SIndRNN端到端的轻量级恶意加密流量识别方法，使用多层一维卷积神经网络提取流量字节序列局部模式特征，并利用全局最大池化降维以减少计算参数。为增强流量表征，设计一种改进的循环神经网络用于捕获流量字节长距离依赖关系。在此基础上，采用独立循环神经网络IndRNN单元代替传统RNN循环单元，使用切片并行计算结构代替传统RNN的串行计算结构，并将两种类型深度神经网络所提取的特征拼接作为恶意TLS流量表征。在CTU-Maluware-Capure公开数据集上的实验结果表明，该方法在二分类实验上F1值高达0.965 7，在多分类实验上整体准确率为0.848 9，相比BotCatcher模型训练时间与检测时间分别节省了98.47%和98.28%。

关键词：恶意TLS流量；独立循环神经网络；切片循环神经网络；一维卷积；全局池化

开放科学（资源服务）标志码（OSID）：

中文引用格式：李小剑，谢晓尧，徐洋，等. 基于CNN-SIndRNN的恶意TLS流量快速识别方法[J]. 计算机工程，2022，48(4)：148-157，164.

英文引用格式：LI X J，XIE X Y，XU Y，et al.Fast identification method of malicious TLS traffic based on CNN-SIndRNN[J]. Computer Engineering，2022，48(4)：148-157，164.

Fast Identification Method of Malicious TLS Traffic Based on CNN-SIndRNN

LI Xiaojian[1]，XIE Xiaoyao[1,2]，XU Yang[2]，ZHANG Sicong[2]

（1.School of Mathematical Science，Guizhou Normal University，Guiyang 550001，China；
2.Key Laboratory of Information and Computing Science Guizhou Province，Guizhou Normal University，Guiyang 550001，China）

【Abstract】Traditional shallow machine learning methods for identifying malicious TLS traffic rely heavily on expert experience，and perform poorly in traffic representation.In addition，the training of the existing deep neural network detection models is time-consuming due to the deepened hierarchical structure.To address the problem，a lightweight end-to-end method for malicious encrypted traffic detection is proposed based on CNN-SIndRNN.The method employs a multi-layer one-dimensional convolutional neural network to extract the local pattern features of a traffic byte sequence，and uses global maximum pooling to reduce dimensions to simplify computational parameters. At the same time，to enhance traffic representation，an improved recurrent neural network is designed in parallel to capture the long-distance dependence of traffic bytes.On this basis，the Independent Recurrent Neural Network（IndRNN）unit is used to replace the traditional Recurrent Neural Network（RNN）unit，and the sliced parallel computing structure is adopted to replace the serial computing structure of the traditional RNN.Then，the features extracted from the two types of deep neural networks are spliced to represent the malicious TLS traffic.The effectiveness of the proposed method is verified on two open datasets.The experimental results show that the method exhibits a F1 score of 0.965 7 in the binary classification experiment.Its overall accuracy rate reaches 84.89% in the multi-classification experiment.Compared with the model of BotCatcher，CNN-SIndRNN model improves the classification performance while reducing the training time by 98.47% and test time by 98.28%.

【Key words】malicious TLS traffic；independently recurrent neural network；sliced recurrent neural network；one dimensional convolution neural network；global pooling

DOI：10. 19678/j. issn. 1000-3428. 0061003

基金项目：中央引导地方科技发展专项资金（黔科中引地[2018]4008）；贵州省科技计划项目（黔科合支撑[2020]2Y013）；贵州省研究生教育创新计划项目（黔教合YJSCXJH[2019]043）。

作者简介：李小剑（1981—），男，博士研究生，主研方向为网络空间安全、深度学习；谢晓尧，教授、博士、博士生导师；徐　洋，教授、博士；张思聪，博士。

收稿日期：2020-03-04　　**修回日期：**2020-04-15　　**E-mail：**lxj@gznu.edu.cn

0　概述

恶意软件利用加密信道和流量加密技术实施恶意行为，该恶意行为隐藏较深且变种频繁，难以被发现。2018年Cisco发布的安全报告指出，为避免攻击行为的暴露，有超过70%的恶意软件通信流量使用了TLS（Transport Layer Security）加密技术[1]。由于勒索病毒、广告木马、挖矿程序等恶意软件与命令控制服务器的通信方法从传统的HTTP协议请求逐渐向加密流量技术转变，因此基于明文的DPI、DFI[2]检测方法已不再适用。解密行为需耗费大量计算资源和时间，在不对流量解密的前提下如何精准识别和快速分类恶意软件加密流量成为当前研究热点之一。

TLS加密协议位于传输层与应用层之间，由于其具有良好的扩展性和兼容性，常用于HTTP、SMTP、POP3等应用层协议中以保护2个通信应用进程间数据的完整性和保密性。加密技术的初衷是保护信息内容安全与用户隐私，但也容易被不法分子用以掩饰其网络违法活动，这给网络安全监管带来了新的挑战。由于加密后流量的上层封装信息不可见，因此研究人员将研究重点转向基于流量行为模式的机器学习技术上。已有研究表明，虽然恶意加密流量也采用标准的TLS协议传输，但其在TLS流协商机制、分组长度、帧到达时间、字节分布等方面与正常流量有明显区别[3]。基于上述关键特征，可以利用随机森林、SVM[4]等经典的机器学习方法区分恶意流量与正常流量，但机器学习方法通常需要人工选择流量特征，要求研究人员具有相关专业背景和丰富的机器学习经验，且特征选择优秀与否将直接影响检测模型最终性能的好坏。近年来，有研究人员使用深度学习技术提高恶意TLS流量检测准确率，1D_CNN[5]、CNN-LSTM[6]、BotCatcher[7]等深度学习模型被陆续提出。这些模型虽然在一定程度上改善了识别效果，但仍存在流量表征不足、模型结构复杂、参数众多等问题，无法同时满足流量检测实时性与效果要求，也难以落实到资源配置有限、对实时性要求较高的应用场景中。

针对上述问题，本文提出一种基于CNN-SIndRNN端到端的恶意TLS流量自动检测方法。利用卷积神经网络（Convolutional Neural Network，CNN）与循环神经网络（Recurrent Neural Network，RNN）各自的优点，充分学习原始流量数据的局部相关特征与时序特征。同时，从具体循环单元结构和网络计算方式2个层面对传统RNN网络模型改进，采用独立循环神经网络（Independently Recurrent Neural Network，IndRNN）单元结构作为循环单元，改善传统RNN存在的长期依赖问题。在此基础上，采用切片循环神经网络（Sliced Recurrent Neural Network，SRNN）并行计算方式代替传统RNN串行计算方式，在保证检测性能的前提下，大幅提高模型训练和检测速度，最终构建完成CNN-SIndRNN检测模型。

1　相关工作

目前对基于TLS加密协议的恶意流量检测仍以识别流量行为模式为主。主要有2种检测方法：

1）利用TCP/IP层数据流元数据，包括分组平均长度、分组到达时间间隔、客户端公钥长度、服务器证书有效天数等统计特征进行学习建模，检测恶意TLS流量。文献[8]从流量行为、TLS明文信息、证书3个维度出发，构建在线随机森林模型以实时区分恶意加密流量。文献[9]结合报文负载和流指纹特征，在不依赖五元组信息的条件下，基于逻辑回归模型提高了复杂网络环境下加密恶意流量检测率。上述方法利用统计特征进行机器学习建模。在攻击者对恶意代码进行更新升级后，流量的行为模式在特征集上也随之变化，可能会使部分依据专家经验挑选出的特征失效，导致识别率下降。此外，原始流量数据中隐含的与识别结果强相关的抽象特征难以被特征工程提取。

2）基于恶意TLS加密协议上下文相关流量，如DNS、HTTP等特征检测方法。文献[10]通过检测算法生成DGA域名的方式对失陷主机DNS请求报文中的恶意域名进行分析检测，但该类方法无法检测直接使用IP地址或以P2P形式构造的僵尸网络。

深度学习作为一种端到端的学习框架，在计算视觉任务和自然语言处理等领域有着出色的表现。在恶意流量识别领域，越来越多的研究人员开始构建各种类型结构的神经网络，并采取不同的训练机制以自动提取流量特征。WANG等[11]将恶意软件流量数据转化为灰度图，提出基于CNN的流量分类方法，该项工作是将学习方法应用于恶意软件原始流量分类任务的首次尝试。CHENG等[12]利用Word2vec模型将流量负载转换为句子向量并通过多核一维卷积实现恶意加密C&C流量识别。CNN善于捕捉序列数据的局部模式特征，但却难以获取动态时序信息和进行长距离记忆。ZHOU等[13]对加密会话中包长传输模式与包传输时间序列进行训练，利用长短期记忆（Long Short-Term Memory，LSTM）网络模型进行自动特征提取，并对恶意加密流量和正常流量进行分类，其准确率超过传统的基于流量统计特征的机器学习方法。但由于循环神经网络依赖前一时间步的计算结果，无法充分利用GPU实现大规模并行运算，训练和检测速度较慢。上述检测方法的检测精度和效率仍存在较大的上升空间。

2　恶意TLS流量识别方法

本文所提TLS恶意流量识别方法主要包含3个阶段：1）流量数据预处理；2）网络流量局部相关性与时序特征挖掘；3）流量分类检测。首先从原始网络流量文件中提取网络流量字节，对16进制表示的字

节字符采用Tokenizer分词器进行字节级分词处理后，经过Embedding（词嵌入）转换为字符向量序列；然后利用连续2个卷积层自动提取字符间局部组合信息以增强流量表征，并使用并行的SIndRNN网络捕获会话流时序特征；最后将2个深度神经网络挖掘出的特征进行拼接，输入到softmax/sigmoid分类器中完成恶意TLS流量的识别与分类。模型总体框架如图1所示。

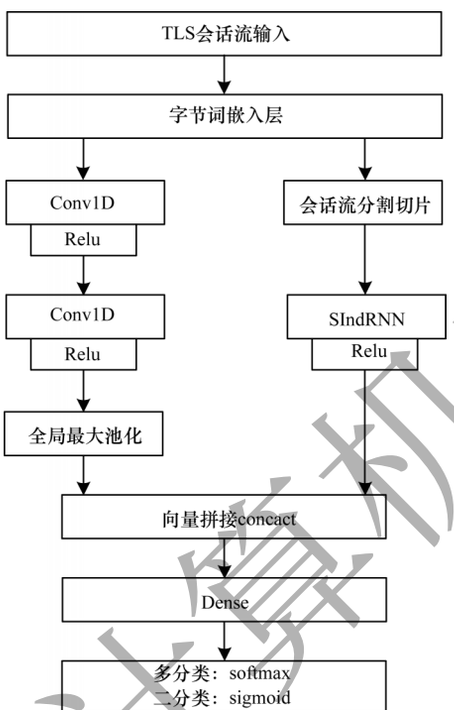

图1 CNN-SIndRNN模型总体框架

Fig.1 Overall framework of CNN-SIndRNN model

2.1 流量预处理与向量化

本文主要关注隐藏于TLS加密协议的恶意软件通信行为流量。以TLS加密数据传递之前的握手协商的数据报文，及与其前向相关的DNS请求响应报文组成TLS会话流，作为特征挖掘空间，主要原因是：1）传输内容已被TLS协议加密，分析流量负载内容难以发现恶意行为；2）TLS握手协议中的ClientHello、ServerHello、Certificate、ClientkeyExchange等消息中所包含加密套件、客户公钥长度、证书有效期、证书是否自签名、前向相关DNS报文中的请求域名、TTL值等信息对识别恶意TLS流量具有重要区分作用[14]。因此，流量预处理主要包含以下步骤：

步骤1 完成原始流量切分与数据包重组，并依据服务器端口和传输层协议从原始网络流量中提取2类通信流，一类是服务器端口号为53的UDP通信流即DNS流，另一类是服务器端口号为443的TCP通信流即TLS通信流。

步骤2 根据2类流中数据帧五元组信息进行关联匹配，最终将TLS握手协商的9个数据包（包含TCP建立连接3次握手）与前向关联2个DNS报文按照通信时间先后顺序合并为一条TLS会话，握手协商过程如图2所示。

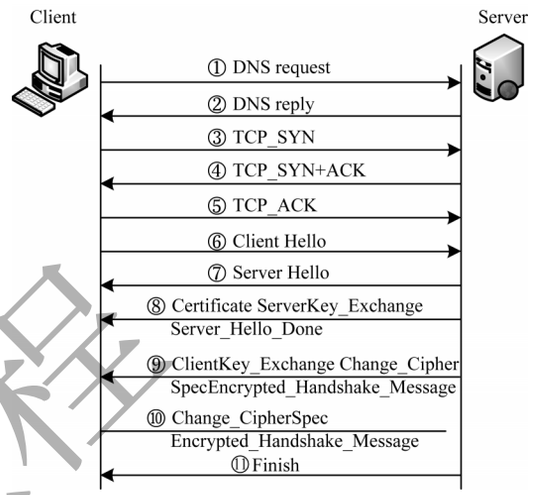

图2 TLS加密会话流握手协商过程

Fig.2 Handshake negotiation process of TLS encrypted session flow

步骤3 删除数据包中特有的MAC地址、IP地址等对分类结果产生干扰的相关字符串。

步骤4 统一输入大小。使用深度神经网络进行训练需固定维度地输入数据，经过对数据分组长度进行统计分析发现，TLS会话流中每个数据包平均字节数约为200。为缩减计算规模并保留分组头部关键信息及方便后续使用切分循环神经网络进行分层切片，提取会话流中每个数据包的前121个字节，超出长度则截断，不足则在末尾补充0X00，并标记其所属类别。

深度学习网络模型输入要求为数值型张量，需将TLS会话流基于字节向量化。文献[15]对每个数据分组向量进行one-hot编码，并将每个字节编码为256维向量，但这种编码方式容易造成生成的二维矩阵数据过于稀疏，且不能表达2个字节间的有效关联。本文利用Tokenizer分词器对每条会话流进行分词编号，统计流量中出现的所有字节字符并生成字典，并对任意给定的会话流输入 $\boldsymbol{s}_i=[b_1^i, b_2^i, \cdots, b_l^i]$，转换为长度为 n 的向量表示，n 为会话流最大截断长度，向量中的每一位元素表示字节在字典中的索引值。这种编码方式简单高效，同时也保存了字节间的前后顺序信息。在此基础上，利用Word Embedding词嵌入技术转换为一个字节向量序列 $\boldsymbol{s}_{1:n}=w_1 \oplus w_2 \oplus \cdots \oplus w_n$。其中：$w_i \in R^k$；$k$ 为嵌入层向量的维度；$\oplus$ 表示拼接操作。这样任一TLS会话流 $\boldsymbol{s}$ 可表示成一个 $n \times k$ 二维的稠密矩阵向量。

2.2 基于CNN、SIndRNN的特征提取

经2.1节预处理后的网络流量数据可视为按层次结构组织的字节流序列。字节、数据分组、会话流与自然语言处理领域中的单词、句子和文档结构非常相似。数据分组中特定的位置代表特定的语义信息，如Client Hello分组的第55、56字节0X0303表示

客户端所支持TLS协议版本号为1.2。在NLP任务中，CNN可以从句子中提取局部单词组合信息。本文将一维卷积看作自然语言模型中的n元语法器（n-gram），自动从TLS会话流中提取字节组合信息。由于数据包内部分字节之间具有前后相关性，联系较为紧密，若采用传统CNN结构中的卷积操作后紧接池化操作，有可能会过滤掉相关特征，造成信息损失。为此，连续采用2个卷积层进行特征提取，并从元语法的角度分析，多层卷积可以扩大感受野，获得更大值的语法信息[16]。在进行第一层卷积操作时，卷积核的宽度必须设置与会话流中字节向量表示的维度一致，卷积核只在流量矩阵$\boldsymbol{s}\in\mathbb{R}^{nxk}$垂直方向上自上而下滑动遍历。2次卷积操作可表示为：

$$c_i=f(\boldsymbol{W}_1\times\boldsymbol{s}[i*\text{step}:i*\text{step}+l_1-1]+b_1) \tag{1}$$

$$c_i'=f(\boldsymbol{W}_2\times C[i*\text{step}:i*\text{step}+l_2-1]+b_2) \tag{2}$$

其中：l_1、l_2为卷积核的尺寸；step为卷积步长；$\boldsymbol{W}_1\in\mathbb{R}^{l_1\times k}$，$\boldsymbol{W}_2\in\mathbb{R}^{l_2\times 1}$为卷积核权值矩阵；$f$为非线性函数，这里采用Relu函数。2层卷积计算后，使用全局最大池化（Global Max Pooling，GMP）代替全连接层以减少参数量。在传统的卷积神经网络中，卷积层经过下采样操作后，通常会将结果打平并送入多个全连接层，这样做会造成整个网络模型参数量大，参数更新困难，且容易造成过拟合。假设经过最后一次卷积操作后得到$W\times H\times C$个特征图，全连接层有M个神经元，仅全连接层需要更新的参数就达$W\times H\times C\times M$个。全局池化是将池化的窗口大小设置成与特征图大小一致，一张特征图输出一个特征值。相比全连接层，其参数量为$1\times1\times C=C$。由于没有参数设置且不需要优化算法进行调参，全局池化能够降低模型过拟合风险。本文选择全局最大池化来提取每张特征图中的最重要区域。

CNN善于捕捉序列型数据的局部模式特征，却难以提取链式结构输入集中的的的各单元之间依赖关系。在网络流量层次结构中，最底层为依时间顺序排列的字节序列，字节序列又按照网络协议类型聚合为上层的分组序列。为更深入挖掘TLS数据流在时间序列上的特性以加强流量表征，本文在提取序列局部组合特征的同时，并行设计了一种改进的循环神经网络进行时间特征学习。经预处理步骤得到的单个TLS会话流字节序列长度达11×121=1 331，若采用经典的RNN网络学习，容易出现长期依赖关系能力差和训练时间过长2大问题。LSTM和GRU网络通过增加门控机制对RNN循环单元进行改进，虽在一定程度上改善了梯度消失和梯度爆炸问题，但由于使用tanh和sigmoid作为激活函数，层与层之间梯度易衰减，因此构建和训练深层循环神经网络存在困难。为解决恶意TLS会话流字节序列输入过长造成训练时间过长等问题，本文从具体循环单元结构和网络计算方式2个层面对传统RNN进行改进，提出一种基于SIndRNN的网络模型挖掘字节序列长距离依赖关系。此外，本文选择独立循环神经网络的单元结构代替传统RNN循环单元，IndRNN[17]单元结构如图3所示。

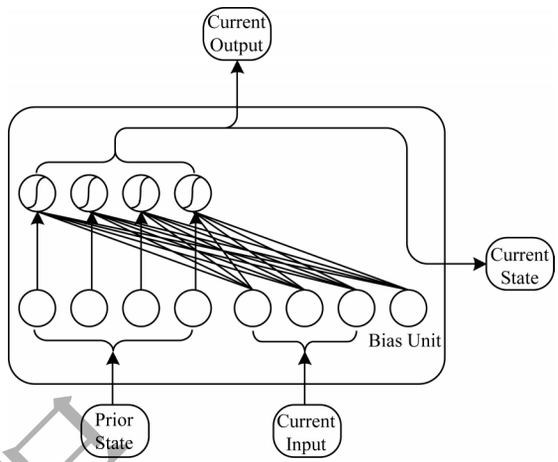

图3　IndRNN单元结构

Fig.3　IndRNN unit structure

与传统RNN网络最大区别在于，IndRNN算法中同一层的各神经元之间相互独立，每个神经元仅接受当前时刻的输入信息和上一时间步的隐层状态信息，以避免被同一时刻特征间关系所影响，同时消除同一时刻的冗余特征。当堆叠2层或多层神经元时，下一层每个神经元独立处理前一层所有神经元的输出，每个神经元独立处理一种类型的时间模式，增强了对更长序列的建模能力。IndRNN在t时刻第n个神经元状态如式（3）所示：

$$h_{t,n}=\sigma(W_n x_t+U_n\odot h_{t-1,n}+b_n) \tag{3}$$

其中：x_t、$h_{t-1,n}$分别是t时刻的输入和$t-1$时刻的隐层状态；W_n表示输入层到隐层权值；U_n表示上一时刻隐层到当前时刻隐层权值；⊙表示哈达马积（hadamard product）；σ为非饱和激活函数Relu。

使用IndRNN循环单元的另一个原因是其可以进一步精简神经网络的训练参数量。假设输入序列向量表示维度为M，对于只有N个隐层单元的单层循环神经网络而言，RNN参数量为$N(M+N)$，LSTM引入了4个门控单元，参数量为$4N(M+N)$，而IndRNN由于结构上同层各神经元间彼此独立，没有与上一时间步所有神经元相连，参数量只有$N(M+1)$。使用切片循环神经网络SRNN[18]并行计算结构代替经典RNN串行结构。SRNN将输入序列切分为若干个等长子序列，每个子序列内的循环单元并列工作，这样可以减少同层相邻循环单元之间的计算依赖时间。本文基于SRNN网络结构建立SIndRNN网络模型。假设输入序列$\boldsymbol{X}=[x_1,x_2,\cdots,x_T]$，其中：$x_i\in\mathbb{R}^k$表示每一时刻的输入；$T$表示序列的总长度。对长度为$T$的序列$X$进行$k$次分割，可分为若干个长度为$n$的子序列，此时有：

$$T=n^{k+1} \tag{4}$$

对于原始网络流量数据而言，数据分组与分组之间，分组内字节之间存在前后依赖的时序关系。本文提取的TLS会话流序列长度为1 331，选择先从会话层进行切分，再从数据分组层进行切分，如此可

得到3个隐藏层。由式(4)可知,当$T=1\ 331$,$k=2$时,可得各隐藏层子序列长度n=11。第一次切分得到长度相等的11个子序列,每个子序列长度为121,恰好对应TLS会话流中的11个数据分组。再对每个子序列执行相同的切分操作,得到最底层的最小子序列数$s_0=n^k=11^2=121$,每个子序列长度$l_0=\frac{T}{n^k}=\frac{1\ 331}{11^2}=11$,SIndRNN网络结构如图4所示。

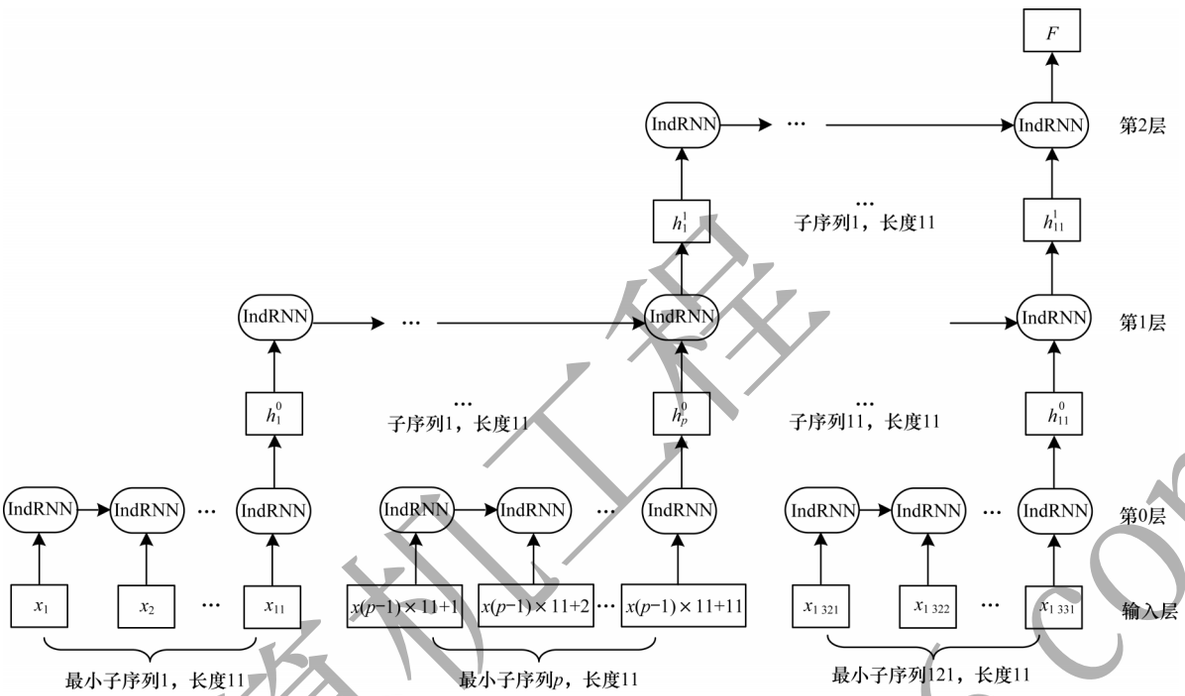

图4 SIndRNN网络结构

Fig.4 SIndRNN network structure

对第0隐藏层121个最小子序列进行并行训练,将得到的121个最小子序列训练结果作为第1隐藏层网络的输入。第1隐藏层子序列的数量为11,同样,将得到的11个子序列并行训练的结果作为第2隐藏层网络的输入。第2隐藏层子序列的数量为1,通过3层隐藏层计算后最终得到隐状态F,以此作为整个SIndRNN网络的输出。这种切片计算方式,既保留了子序列内部时序关系,也保留了子序列之间的时序信息,即网络流中的数组分组间和数据分组内字节间的时序特征。SIndRNN网络训练算法如下:

输入 会话流序列向量表示$\boldsymbol{X}$_seqs=$[x_1,x_2,\cdots,x_T]$,$x_i\in\mathbb{R}^k$, units, n

输出 SIndRNN隐状态F

```
x_seqs_split=[ ]
for i in range(X_seqs.shape[0]):
split_1=Split(X_seqs[i],n)
min_seqs=[ ]
for j in range(n):
split_2=Splif(split_1[j],n)
min_seqs.append(split_2)
end for
X_seqs_split.append(min_seqs)
end for
Input_1=X_seqs_split
hidden_layer1=IndRNN(units(Input_1))
Input_2=TimeDistibuted(hidden_layer1,shape=(n,T/n^2))
hidden_layer2=IndRNN(units)(Input_2)
Input_3=TimeDistibuted(hidden_layer2,shape=(n,n,T/n^2))
F=IndRNN(units)(Input_3)
```

SIndRNN训练速度优势分析:假设每个IndRNN单元处理数据的时间为t,输入序列的长度为T,则IndRNN网络训练所花费的时间$t_{\text{IndRNN}}=T\times t$,SIndRNN层子网训练所花费的时间分别为$n^2\times t$、$n\times t$、$t$,因此SIndRNN网络总训练时间$t_{\text{SIndRNN}}=n^2\times t+n\times t+t=(n^2+n+1)\times t$,相比于IndRNN,理论上可节约的训练时间$t_{\text{reduce}}=t_{\text{IndRNN}}-t_{\text{SIndRNN}}=T\times t-(n^2+n+1)\times t=(n^{k+1}-n^2-n-1)\times t$(已知$T=n^{k+1}$)。在本文$T$=1331,$k$=2,$n$=11的情况下,SIndRNN与IndRNN相比,理论上训练时间下降了$\frac{t_{\text{reduce}}}{t_{\text{IndRNN}}}\times 100\%=\frac{11^3-(11^2+11+1)}{11^3}\times 100\%=\frac{1\ 198}{1\ 331}\times 100\%=88.24\%$。再加上IndRNN相比于LSTM、RNN训练参数量更少,因此SIndRNN的训练速度具有较大的优势。

2.3 CNN-SIndRNN分类算法流程

在进行分类之前,需要将CNN网络与SIndRNN网络输出的2种类型表征进行聚合,为丰富原始流量表征,本文采用拼接方式将2种类型的特征向量进行串联,再输入到分类函数中。对于二分类任务,基于数据流特征,使用sigmoid分类器判断输入数据流为正常加密流量还是恶意加密流量。代价函数选

择更适用于二分类场景的二元交叉熵损失函数，函数公式如式（5）和式（6）所示：

$$\text{Sigmoid: } f(x)=\frac{1}{1+e^{x}} \tag{5}$$

$$\text{Loss}(y_{\text{pre}},y_{\text{true}})=-[y_{\text{true}}\log_a y_{\text{pre}}+(1-y_{\text{true}})\log_a(1-y_{\text{pre}})] \tag{6}$$

对于多分类任务，输出层神经元激活函数选择softmax函数，它将多个神经元的输出映射到（0，1）区间内，各个输出之和为1，对应于划分到各个类别的概率。代价函数选择适用于多分类场景的多元交叉熵损失函数Categorical Cross Entroy，函数式如式（7）和式（8）所示：

$$\text{softmax: } S(y_i)=\frac{e^{y_i}}{\sum_{j}^{c}e^{y_j}} \tag{7}$$

$$\text{Loss}(y_{\text{pre}},y_{\text{true}})=-\sum_{j}^{c}y_{\text{true}}\times\ln y_{\text{pre}} \tag{8}$$

其中：c表示分类类别数。

模型训练采用Adam优化器，通过计算梯度一阶矩估计和二阶矩估计，自适应调整学习率，并根据损失函数计算偏导数以更新模型权值W与偏置b。设网络模型第l层权值与偏置分别为$W^{(l)}$、$b^{(l)}$，参数更新过程如式（9）和式（10）所示：

$$W^{(l)}\leftarrow W^{(l)}-\eta\frac{\partial\text{Loss}(y_{\text{true}},y_{\text{pre}})}{\partial W^{(l)}} \tag{9}$$

$$b^{(l)}\leftarrow b^{(l)}-\eta\frac{\partial\text{Loss}(y_{\text{true}},y_{\text{pre}})}{\partial b^{(l)}} \tag{10}$$

其中：η为学习率。

本文提出基于CNN_SIndRN的恶意TLS加密流量识别模型的算法流程分为4个步骤：

输入　经预处理后得到的TLS会话流$s=\{s_seq_1, s_seq_2,\cdots,s_seq_n\}$

输出　TLS会话流类别

步骤1　TLS会话流编码向量表示。

1. tokenizer = Tokenizer(num_word=255)//构造分词器

2. tokenizer.fit_on_texts(s_seq_i)

3. vocab = tokenizer.word_index//构造词汇表，用于将字//节字符映射为字典索引值

4. s_seq_vec= tokenizer.texts_to_sequence(s_seq_i)//对会//话流中所有字节进行编号

5. s_encoder=Embedding(len(vocab+1, embed_size=128, input_length=1331)(s_seq_vec)//字节序列向量化，利用词嵌//入技术，将每条会话流编码为1331×128二维矩阵向量

步骤2　TLS会话流特征提取。

1. C = Conv1D(s_encoder)

2. C' = Conv1D(C)

3. feature_localcorre = GlobalMaxPooling(C')//全局最大//池化，提取局部模式特征

4. feature_temporal=SIndRNN(s_encoder)//调用SIndRNN//算法提取时序特征

5. input = concact(feature_localcorre, feature_temporal)//将两种类型表征进行拼接

步骤3　模型训练与验证。

1. train_data, val_data=StratifiedKFold(n_splits=5, shuffle=True)(s)//采用五折交叉验证

2.K=0

3.While(K< n_splits)：

4.do

5.while(not satisfy Early Stopping or epoch<预设值)

6.do

7.for (i=0;i<len(train_data)//batch_size;i++)

8.from train_data select batch_size sampls randomly,

9. output=Softmax(input), y_{pre} =argmax (output)//前向计//算，输出预测值

10. 利用式（6）和式（8）计算代价函数，式（9）和式（10）反向更新模型权值与偏置

11.end for

12. 使用val_data对模型进行验证

13.end while

14.K++

15.end While

步骤4　模型测试与评估。使用测试数据对训练好的模型进行预测，并使用真实标签对模型性能进行评估。

3　实验

3.1　实验环境与设置

本文实验环境运行在Windows10操作系统上，CPU为Intel酷睿i5-9400 F/2.9 GH/6 cores，内存16 GB，显卡为英伟达GeForceGTX1660。网络模型创建以及训练参数调优采用深度学习平台keras2.25，后端调用tensorflow-GPU1.8完成并行加速运算，开发环境为python3.7.3。IndRNN算法实现部分参考链接：https://github.com/titu1994/Keras-IndRNN/blob/master/ind_rnn.py。

CNN_SIndRNN模型网络结构如表1所示。超参数设置经过多次实验调整，最终选择结果如表2所示。实验结果采用五折交叉验证，将样本平均分为5份，轮流将其中4份作为训练集，剩余1份作为测试集，最终结果取5次测试均值，训练集、验证集和测试集比例设置为8∶1∶1。

表1　CNN_SIndRNN模型网络结构

Table 1　CNN_SIndRNN model network structure

层(类型)	输出尺寸	参数量
input_1(InputLayer)	(None,1 331)	0
embedding_1(Embedding)	(None,1 331,128)	37 888
conv1d_1(Conv1D)	(None,1 327,64)	41 024
input_4(InputLayer)	(None,11,11,11)	0
conv1d_2(Conv1D)	(None,1 325,128)	24 704
time_distributed_2(TimeDistributed)	(None,11,128)	71 168
global_max_pooling1d_1(GlobalMax)	(None,128)	0
Ind_rnn_3(IndRNN)	(None,128)	16 640
concatenate_1(Concatenate)	(None,256)	0
dense_1(Dense)	(None,8/1)	2 056/257
Total_param/Trainable_param	193 480/191 681	

表2 模型参数设置

Table 2 Model parameter setting

层	参数设置
Embedding	Output_dim:128,Dropout:0.5
Conv1D_1	Kernel_size:5,num_filters:64,strides:1,activation:relu,padding:valid
Conv1D_2	Kernel_size:3,num_filters:128,strides:1,activation:relu,padding:valid
SIndRNN	Units:128,return_squences:False,activation:relu,n:11,k:2
Full_connect	Units:1/8,activation:Sigmoid/Softmax

3.2 实验数据

本文实验数据中恶意样本来源为CTU-Malware-Capture[19]和Malware-Traffic-Analysis[20]2个公开的实验项目,样本是通过在沙盒隔离环境内运行多款恶意软件,并捕获其在运行过程中发送给远端控制服务器的网络流量生成。正常流量使用CTU-Normal-Capture[21]数据集。原始流量文件格式为pcap,但并非完全都属于TLS类型流量。由于TLS握手协商报文使用明文传输,因此在数据预处理阶段,通过调用scapy库对捕获的数据报头部信息进行解析,对于一个完整的TLS会话,其通信过程一定包含ClientHello、Server Hello、Certificate、ServerKeyExchange等特定类型消息。如果某个数据流中没有检测到以上消息,则可将其判定为非TLS流。若数据流中检测到以上部分消息,但由于网络环境不稳定,出现TCP、TLS握手过程不完整、丢包等现象导致TLS连接建立失败,这类流量也被判定为非TLS流。经过滤后得到的实验数据具体分布如表3所示。

表3 数据集样本分布

Table 3 Data set sample distribution

数据集	恶意样本家族/应用	样本数
CTU-Malware-Capture	Malware	5 359
	CTU-Normal-Capture	4 978
Malware-Traffic-Analysis	Trickbot	502
	Adware	1 196
	Zbot	820
	WannaCry	100
	Dridex	4 289
	EITest	515
	Emote	1 487
	CTU-Normal-Capture	4 978

3.3 评估指标

实验采用精确率(Precision),召回率(Recall)、调和平均数(F1值)、整体准确率(Overall_ACC)及ROC曲线来评估模型性能的指标。计算公式如式(11)~式(14)所示:

$$P_{\text{Precision}}=\frac{T_{\text{TP}_i}}{T_{\text{TP}_i}+F_{\text{FP}_i}} \tag{11}$$

$$R_{\text{Recall}}=\frac{T_{\text{TP}_i}}{T_{\text{TP}_i}+F_{\text{FN}_i}} \tag{12}$$

$$\text{F1}=2\times\frac{P_{\text{Precision}}\times R_{\text{Recall}}}{P_{\text{Precision}}+R_{\text{Recall}}} \tag{13}$$

$$O_{\text{Overall\_ACC}}=\frac{\sum_{i=1}^{c}T_{\text{TP}_i}}{\sum_{i=1}^{c}(T_{\text{TP}_i}+F_{\text{FN}_i})} \tag{14}$$

其中:T_{TP_i}表示属于类别i的加密流量被正确识别为该类别数量;F_{FP_i}表示不属于类别i加密流量被识别为类别i的数量;F_{FN_i}表示属于类别i的加密流量被识别为非该类别的数量。ROC曲线的横坐标、纵坐标分别由假阳率(FPR)和真阳率(TPR)绘制而成,对于二分类实验,ROC与坐标轴围成的面积(AUC)越大,表示分类模型性能越好。

3.4 实验结果与分析

为评估CNN_SIndRNN模型各模块设计合理性,在CTU-Malware-Capture数据集上对训练损失函数的收敛性进行对比。实验时,序列长度T设置为1 331,时序特征提取子模块分别采用RNN、GRU、LSTM与IndRNN循环单元。结果如图5所示,各循环单元采用相同参数设置与相同的切片分层方式。IndRNN在4种循环单元中收敛速度最快且趋势较为平稳,GRU及LSTM均有小幅度振荡,RNN则出现收敛困难,性能最差,这说明IndRNN在多层循环神经网络捕获长序列数据特征的优势。图6为消融实验中单独使用CNN、SIndRNN及同时使用CNN与SIndRNN模块训练准确率对比,可以看出组合局部模式特征与时序特征在识别恶意TLS流量时,比单独使用CNN或SIndRNN更有效。

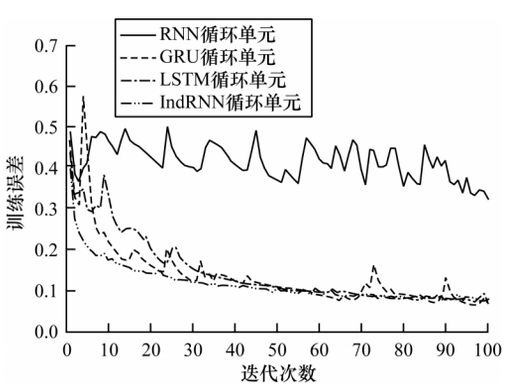

图5 训练误差迭代对比

Fig.5 Comparison of training error iterations

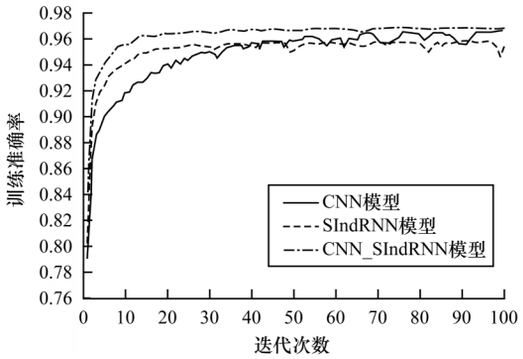

图6　训练准确率对比

Fig.6　Comparison of training accuracy

3.4.1　二分类结果对比

为验证本文所提CNN_SIndRNN模型在识别恶意TLS流量时的效果，在CTU-Malware-Capture数据集上与已有的自动挖掘特征深度学习基准模型1D-CNN、BotCatcher和经典机器学习模型SVM、随机森林进行实验对比。在基于1D_CNN的分类实验中，采用文献[5]提出的1D_CNN分类结构，提取TLS会话流前784个字节，输入2层一维卷积网络提取字节序列局部特征组合。BotCatcher模型采用文献[7]提出的分类模型，分别截取TLS会话流前1 024个字节和前10个数据包（不包含上关联2个DNS数据包，每个数据包取前80个字节），分别输入CNN与LSTM网络，其中LSTM采用2层双向网络结构。SVM、随机森林模型采用文献[4]提出的基于TCP、UDP连接、TLS、X509证书日志三个维度共15个统计特征构建分类器，其中SVM核函数采用线性核，随机森林基分类器n_estimators设置为100，对恶意样本的实验结果如表4所示，表中加粗数据表示该组数据的最大值。

表4　二分类实验结果对比

Table 4　Comparison of two classification experiment results

分类模型	精确率	召回率	F1值
SVM	0.927 5	0.877 5	0.901 8
1D-CNN	0.926 6	0.928 7	0.927 6
RandomForest	0.953 0	0.963 8	0.958 4
BotCatcher	**0.964 8**	0.960 4	0.962 6
CNN-SIndRNN	0.959 8	**0.971 7**	**0.965 7**

由表4可知，CNN-SIndRNN模型的召回率、F1值分别为0.971 7、0.965 7，在5种模型中最高，检测出恶意样本的效果最好。虽然经典机器学习算法也能取得较好的检测结果，但需要依据专家经验构建具有强分辨力的特征以区分恶意样本与良性样本，特征工程耗时复杂。而深度学习方法能够自主选择特征，不仅减少了工作量而且提升了分类效果。为更直观评估本文模型在二分类实验性能，根据测试结果中的假阳率（FPR），真阳率（TPR）绘制ROC曲线图，结果如图7所示。CNN-SIndRNN模型的ROC_curve面积为0.994 5，在5种模型中面积最大，说明本文所提出模型对二分类测试数据集的ROC拟合程度最好。

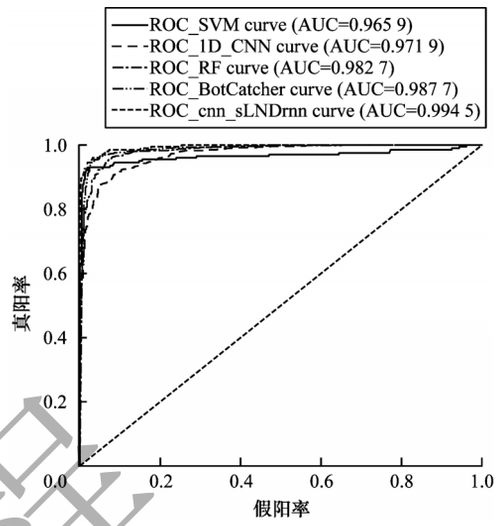

图7　二分类ROC

Fig.7　ROC of two classification

3.4.2　多分类结果对比

在实际应用中不仅需要通过通信流量鉴别恶意软件，还需要对其所属类别进行标记，并划分至相应家族以供安全人员研究。为此，在Malware-Traffic-Analysis数据集上对1D_CNN、BotCatcher、CNN-SIndRNN模型进行多分类实验，以验证模型的泛化能力。对3种模型采用早停（Early Stopping）策略动态控制模型训练迭代次数，容忍度统一设置为20，即训练模型过程中连续20次迭代后验证集损失函数值仍在上升则停止训练。3种模型训练迭代轮数平均值分别为21、23、22，取模型前20轮的训练准确率变化进行对比，结果如图8所示。BotCatcher、CNN-SIndRNN模型的初始验证集准确率高于1D_CNN模型，对训练数据的拟合能力更强，且随着训练轮数增加，CNN-SIndRNN模型的准确率达到最高。

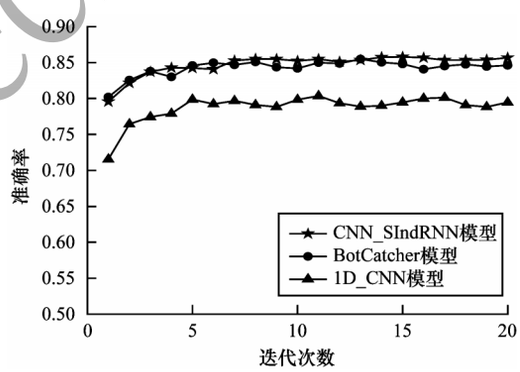

图8　3种模型前20次训练过程验证集准确率对比

Fig.8　Comparison of accuracy rate of validation set in first 20 training processes of three models

图9是3种模型在测试集上进行分类后输出的8维混淆矩阵，混淆矩阵主对角线上的数值为各类别被预测正确的比重（召回率，精确到小数点后2位），恶意软件家族识别结果F1值及各模型整体准确率如表5所示，表中加粗数据表示该组数

据的最大值。1D_CNN模型由于缺乏对流量层次结构和上下文时序信息特征表达，因此各类家族样本的召回率和F1值均低于BotCatcher与CNN_SIndRNN模型，整体准确率也只有0.795 8。本文所提模型整体准确率达0.848 9，比1D_CNN、BotCatcher模型分别高出5.31%、0.32%，说明本文提出的组合模型在构建的流量特征空间中，自动挖掘恶意TLS流量特征方面更有效。3种深度学习模型对WannaCry恶意样本家族的识别率均较低，这是因为WannaCry和ZBot木马变种在与C&C服务器建立连接时通信行为较为相似，相当数量的WannaCry样本被误判为ZBot，且该类别样本数量较少，深度学习模型无法充分学习其隐含的高级抽象特征，导致识别效果欠佳。

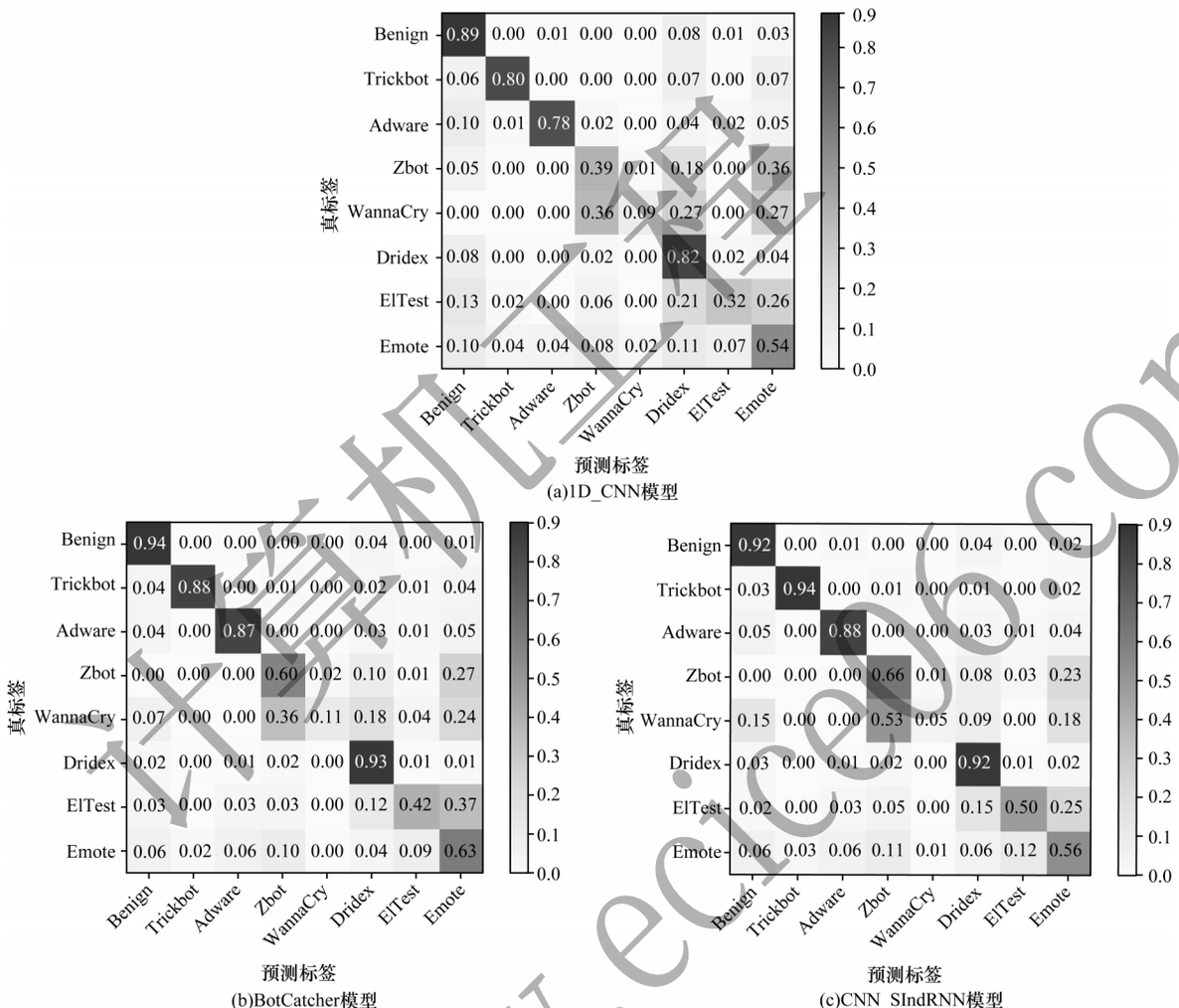

图9 3种模型测试集预测结果混淆矩阵

Fig.9 Confusion matrix of prediction results of three models on test data

表5 多分类实验F1值与整体准确率

Table 5 F1 value and overall accuracy of multi classification experiment

分类模型	F1值								整体准确率
	Benign	Trickbot	Adware	Zbot	WannaCry	Dridex	EITest	Emote	
1D_CNN	0.895 7	0.849 2	0.839 4	0.545 2	0.097 7	0.842 2	0.384 5	0.565 4	0.795 8
BotCatcher	0.931 7	**0.918 3**	0.865 9	**0.630 9**	0.083 8	0.906 6	**0.511 3**	0.581 2	0.845 7
CNN_SIndRNN	**0.942 0**	0.891 4	**0.875 7**	0.606 7	**0.132 0**	**0.909 6**	0.458 9	**0.610 4**	**0.848 9**

表6为3种模型参数量、训练、测试时间和模型大小对比结果，表中加粗数据表示该组数据的最大值。训练时间是在约1万条训练样本上单次迭代时间的平均值，测试时间表示在所有测试样本(1 389条)单次测试平均值。1D_CNN算法所用时间最少，CNN_SIndRNN训练时间和测试时间分别为7.499 s、0.453 s，比1D_CNN算法耗时稍长，说明影响模型整体运行速度的主要因素为时序特征提取模块，但本文的SIndRNN模块对CNN并行处理速度影响较小。BotCatcher模型训练耗时达489.902 s，在3种模型中耗时最长，这是因为其LSTM

采用循环输入方式，对流量字节逐个处理，每步计算依赖上一时间步计算结果。而SIndRNN采用的IndRNN循环单元在每个子序列内是并行工作的，在保证获取长距离依赖关系前提下可大幅提高运行速度，减少计算等待时间，与BotCatcher相比，训练时间与测试时间分别节省了98.47%和98.28%。此外，本文还设计了变体模型CNN_IndRNN，即将流量字节视为整体处理，采用分层切片方式逐个输入到IndRNN模块中，训练时间约为49.99 s。与之相比，本文模型训练时间可节省约85%，这也与2.2节理论推算基本相符。本文设计的轻量级模型CNN_SIndRNN模型 参数共有193 480个，模型大小仅为785 K，远低于1D_CNN与BotCatcher模型，在保证分类性能的前提下，缩短了模型训练、检测时间，并简化了模型结构，有效地做到了分类精度和运行时间的均衡。CNN_SIndRNN模型在计算实验环境中每秒可处理3 066条TLS会话流，可以满足校园网级别的恶意TLS流量实时检测要求。

表6　模型效果对比

Table 6　Comparison of effect of models

模型	参数量	训练时间/s	测试时间/s	模型大小
1D_CNN	5 828 488	**3.769**	**0.281**	22.7M
BotCatcher	2 121 916	489.902	26.391	49.5M
CNN_SIndRNN	**193 480**	7.499	0.453	**785.0K**

4　结束语

本文基于CNN_SIndRNN模型提出一种恶意TLS流量快速识别方法，使用多层一维卷积在流量字节序列中提取局部模式特征，在不同层采用不同卷积核值以获取更大局部感受野。为增强流量表征，并行设计一个IndRNN网络层用以挖掘流量字节间长距离依赖关系。此外，采用切片循环神经网络并行计算结构代替经典RNN串行结构，从而大幅提高模型训练和检测速度。实验结果表明，与BotCatcher相比，该方法在保证恶意TLS识别效果的前提下，训练时间和检测时间分别节省了98.47%和98.28%，并降低了模型复杂度。由于目前专门用于恶意软件加密流量数据集相对较少，本文方法对少数类恶意加密流量的识别率及泛化能力仍有待提升。下一步将从数据层面处理样本不均衡问题，对于少数类样本采用上采样方法并结合SMOTE算法，根据Tomek link距离合成新样本。此外，尝试在其他数据集上进行测试，以提高模型的泛化能力。

参考文献

[1] CISCO. Encrypted traffic analytics white paper[EB/OL]. 2020-05-07]. https://www. cisco. com/c/dam/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encryted-traf-anlytcs-wp-cte-en. pdf.

[2] 陈良臣，高曙，刘宝旭，等. 网络加密流量研究进展及发展趋势[J]. 信息安全网络，2019，19(3)：19-25.
CHEN L C, GAO S, LIU B X, et al. Research status and development trends on network encrypted traffic identification[J]. Netinfo Security, 2019, 19(3): 19-25. (in Chinese)

[3] 徐国天. 基于异常加密流量标注的Android恶意进程识别方法研究[J]. 信息安全网络，2020，20(7)：30-41.
XU G T. Android malicious process identification method based on abnormal encrypted traffic annotation[J]. Netinfo Security, 2020, 20(7): 30-41. (in Chinese)

[4] ANISH S, SHEKHA W, FABIO D, et al. Feature analysis of encrypted malicious traffic[J]. Expert Systems with applications, 2019, 125: 130-141.

[5] WANG W, ZHU M, WANG J, et al. End-to-end encrypted traffic classification with one-dimensional convolution neural networks[C]//Proceedings of 2017 IEEE International Conference on Intelligence and Security Informatics. Washington D. C. , USA: IEEE Press, 2017: 43-52.

[6] ZOU Z, GE J G, ZHENG H B, et al. Encrypted traffic classification with a convolutional long short-term memory neural network[C]//Proceedings of 2018 IEEE International Conference on High Performance Computing and Communications. Washington D. C. , USA: IEEE Press, 2018: 329-334.

[7] 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.
WU D, FANG B X, CUI X, et al. BotCatcher: botnet detection system based on deep learning[J]. Journal on Communications, 2018, 39(8): 18-28. (in Chinese)

[8] ANDERSON B, PAUL S, MCGREW D. Deciphering malware's user of TLS (without decryption)[J]. Journal of Computer Virology and Hacking Techniques, 2018, 14(3): 195-211.

[9] 胡斌，周志洪，姚立红，等. 结合报文负载与流指纹特征的TLS恶意流量检测[J]. 计算机工程，2020，46(11)：157-163.
HU B, ZHOU Z H, YAO L H, et al. TLS malicious traffic detection combining features of packet payload and stream fingerprint[J]. Computer Engineering, 2020, 46(11): 157-163. (in Chinese)

[10] SCHUPPEN S, TEUBERT D, HERRMANN P, et al. FANCI: feature-based automated nxdomain classifica- tion and intelligence[C]//Proceedings of the 27th USENIX Conference on Security Symposium. New York, USA: ACM Press, 2018: 1165-1181.

[11] WANG W, ZHU M, ZENG X, et al. Malware traffic classification using convolutional neural network for representation learning[C]//Proceedings of 2017 International Conference on Information Networking. Washington D. C. , USA: IEEE Press, 2017: 712-717.

[12] 程华，谢金鑫，陈立皇. 基于CNN的加密C&C通信流量识别方法[J]. 计算机工程，2019，45(8)：31-34，41.
CHENG H, XIE J X, CHEN L H. CNN-based encrypted C&C communication traffic identification method[J]. Computer Engineering, 2019, 45(8): 31-34, 41. (in Chinese)

（下转第164页）

（上接第157页）

[13] 邹源，张甲，江滨. 基于LSTM循环神经网络的恶意加密流量检测[J]. 计算机应用与软件，2020，37(2)：308-312.
ZOU Y，ZHANG J，JIANG B. Detection of malicious encrypted traffic based on LSTM recurrent neural network[J]. Computer Applications and Software，2020，37(2)：308-312.（in Chinese）

[14] CONSTANTINOS P，FRAN C，VASILIOS K. Encrypted and covert DNS queries for botnets：challenges and counter measures[EB/OL]. [2020-01-17]. https://www.researchgate.net/publication/335854377_Encrypted_and_Covert_DNS_Queries_for_Botnets_Challenges_and_Countermeasures.

[15] WANG W，SHENG Y，WANG J，et al. HAST-IDS：learning hierarchical spatial-temporal features using deep neural networks to Improve Intrusion Detection[J]. IEEE Access，2018，6：1792-1806.

[16] 刘洋，赵科军，葛连升，等. 一种基于深度学习的快速DGA域名分类算法[J]. 山东大学学报(理学版)，2019，54(7)：1-8.
LIU Y，ZHAO K J，GE L S，et al. A fast DGA domain detection algorithm based on deep learning [J]. Journal of Shang dong University(Natural Science)，2019，54(7)：1-8.（in Chinese）

[17] LI S，LI W，COOK C，et al. Independently Recurrent Neural Network (indRNN)：building a longer and deeper RNN [C]//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C.，USA：IEEE Press，2018：5457-5466.

[18] YU Z P，LIU G S. Sliced recurrent ceural networks[C]// Proceedings of the 27th International Conference on Computational Linguistics. Washington D. C.，USA：IEEE Press，2018：59-69.

[19] STRATOSPHERE L. Malware capture facility project[EB/OL]. [2020-01-17]. https://www.stratosphereips.org/datasets-malware/normal.

[20] BRAD A. Malware traffic analysis[EB/OL]. [2020-01-17]. https://www.malware-traffic-analysis.net.

[21] STRATOSPHERE L. Malware capture facility project [EB/OL]. [2020-01-17]. https://mcfp.felk.cvut.cz/publicDatasets/CTU-13-Dataset/.

编辑　赖玉玲