DOI:10.11721/cqnuj20210418

基于 CA-ResNet 网络与 nadam 优化的入侵检测算法*

张文泷，魏　延，李媛媛，蒋俊蕊，张　昆，张　杨
(重庆师范大学 计算机与信息科学学院·智能科学学院 教育大数据智能感知与应用重庆工程研究中心，重庆 401331)

摘要:【目的】针对深度学习模型在网络入侵检测中进行参数训练时因梯度消失而导致深度学习模型过拟合在测试集上准确率下降的问题。提出一种结合 LeakyRelu 激活函数与 ResNet 的网络入侵检测算法，即 CA-ResNet，并采用 nadam 优化器对模型进行优化。【方法】该模型在 DNN 的基础上增加了网络的层次，结合了 ResNet 和 LeakyRelu 激活函数。【结果】解决了模型训练时梯度消失的问题，保证了该模型在测试数据集上的表现，使得训练的模型的泛化能力更强，同时通过增加网络的单层维度和总层次的深度，提高了网络的特征提取能力和对尺度的适应性。【结论】使用 KDD Cup99 数据中的 Corrected 数据集对算法进行验证。实验表明，该算法与 CNN 和 CNN-BiLSTM 算法相比具有更高的准确率和 F1-score，准确率能够达到 95.0%，F1-score 能够达到 97.5，时间复杂度为线性时间复杂度。

关键词:网络安全;深度学习;入侵检测;ResNet 残差网络;协同激活函数;CA-ResNet;nadam 优化器
中图分类号:TP309　　**文献标志码**:A　　**文章编号**:1672-6693(2021)04-0097-10

为了有效保障信息和通信技术系统的安全性，入侵检测系统(Intrusion detection systems，IDS)[1-2]应运而生。该系统通常利用 TAP 和 SPAN 端口分析内联流量流的副本，通过先前用流量数据训练好的算法来预测攻击，从而减少了人为的干预，达到高效入侵检测的目的。

目前，机器学习和统计分析的方法已经被广泛用于各种类型的网络入侵检测，基于机器学习和统计分析的入侵检测系统可以协助系统判断是否受到入侵。基于传统机器学习算法的入侵检测包括 K 近邻、SVM[3]、决策树[4]、贝叶斯网络模型[5]等。

近些年来，随着网络设备及相关技术的快速发展，产生了海量的网络数据，传统的机器学习算法越来越难以解决实际网络中海量入侵数据的分类问题。深度学习[6-9](Deep learning，DL)是机器学习领域中一个新的研究方向，它的网络模型含有多个隐藏层的多层感知机构，通过组合低层特征形成更加抽象的高层表示属性类别或特征，从而发现数据的分布式特征表示。文献[8]在 CNN 的基础上运用多尺度的网络进行参数优化，取得了比较理想的结果，但是如果训练参数选用不当则会导致 F1-score[10]偏低。文献[9]将深度神经网络(Deep neural network，DNN)运用到入侵检测中，利用 DNN 进行数据特征提取，相比传统的机器学习算法，在准确率上有明显的提高，但缺点是网络结构简单，误报率较高。

深度学习模型在网络入侵检测中进行参数训练时易造成梯度消失，这会导致深度学习模型过拟合从而在测试集上的准确率下降。为了解决这一问题，本文提出了 CA-ResNet 算法。该算法是一种基于协同激活和 ResNet 原理[11]，将 DNN[12]、ResNet 网络的残差机制、协同激活函数(CA)等运用于网络入侵检测的深度神经网络算法。CA-ResNet 算法在它的神经网络的每一层中使用 Relu 和 LeakyRelu 激活函数协同作用，提取不同的数据特征，来提高算法的泛化能力和准确率，再结合 ResNet 残差层组成 CA-ResNet 层，使用 nadam 优化器[13]进行网络参数的优化，然后利用 CA-ResNet 层来增加网络的深度以提高算法性能，同时解决梯度消失的问题。在对该算法参数和结构进行优化实验时用到了 Hornik 等人[14]所证明的一个理论，即：只需 1 个包含足够多神经元的隐层，多层前馈神经网络就能以任意精度逼近任意复杂度的连续函数。

* **收稿日期**:2020-11-23　**修回日期**:2021-01-08　**网络出版时间**:2021-07-07 09:32
资助项目:重庆市技术创新与应用发展重大主题专项(No. cstc2019jscx-mbdxX0061;No. cstc2019jscx-zdztzx0043)
第一作者简介:张文泷，男，研究领域为计算机视觉、网络安全，E-mail：1148993282@qq.com;**通信作者**：魏延，男，教授，博士，E-mail：942503433@qq.com
网络出版地址:https://kns.cnki.net/kcms/detail/50.1165.N.20210706.1332.002.html

1 基于CA-ResNet的深度神经网络模型

CA-ResNet算法由输入层、全连接层、1个CA-ResNet层、输出层组成。

1.1 CA全连接层

全连接层[15]采用协同激活函数的形式,将每一层的输出平均分为两部分,分别采用Relu[16]激活函数和LeakyRelu激活函数进行非线性化。Relu与LeakyRelu激活函数以及它们对应的导数(图1)。其中:图1a,b分别对应Relu激活函数及导数:$\begin{cases}y=0,x<0\\y=x,x\geqslant 0\end{cases}$,$\begin{cases}y=0,x<0\\y=1,x\geqslant 0\end{cases}$;图1c,d分别对应LeakyRelu激活函数及导数:$\begin{cases}y=0.1x,x<0\\y=x,x\geqslant 0\end{cases}$,$\begin{cases}y=0.1,x<0\\y=1,x\geqslant 0\end{cases}$。

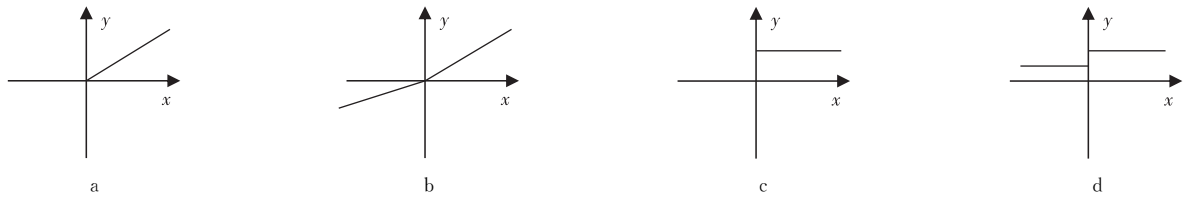

图1 激活函数及对应导数图

Fig. 1 Activation function and corresponding derivative image

在全连接层中,第l层与$l+1$层的一般关系式可以表示为:

$$\delta_i^l=\{\delta_1^{l+1}w_{1i}^{l+1}+\delta_2^{l+1}w_{2i}^{l+1}+\cdots+\delta_n^{l+1}w_{ni}^{l+1}\}a_1'(z_i^l)+\{\delta_{n+1}^{l+1}w_{(n+1)i}^{l+1}+\delta_{n+2}^{l+1}w_{(n+2)i}^{l+1}+\cdots+\delta_m^{l+1}w_{mi}^{l+1}\}a_2'(z_i^l)。$$

其中:$a_1'(z_i^l)$和$a_2'(z_i^l)$分别为Relu和LeakyRelu激活函数的导数,δ_i^l表示第l层的第i个神经单元误差,w表示神经网络中的权重参数。Relu激活函数可以有效减少梯度消失的发生,提高模型泛化能力,但是Relu激活函数不能对$x<0$区间里的特征进行提取,因此再使用LeakyRelu激活函数进行负区间的特征进行提取,以提高模型对数据的特征提取能力。

1.2 CA-ResNet层

为了提高整个算法的入侵检测性能,需要对基础网络模型的深度和宽度进行适当的拓展。如果使用全连接神经网络继续对网络的深度进行拓展,会由于网络层次过深造成梯度消失,使得网络的性能下降。因此,本文结合ResNet的原理提出了CA-ResNet层,可以有效避免梯度消失的问题。

假设CA-ResNet层的输入为x,经过这一段网络的处理之后可以得到期望的输出$H(x)$,现在将输入x传到输出作为下一段网络的初始结果,那么此时需要学习的目标就不再是一个完整的输出$H(x)$,而是输出与输入的差别$F(x)=H(x)-x$。CA-ResNet模型结构见图2,其中两个Dense层均由128个通过Relu激活函数的神经元和128个通过LeakyRelu激活函数的神经元组成,汇聚层由256个上一层直连神经元和间接神经元组合而成。其中神经元个数是根据试错法试验得出的,即在欠拟合和过拟合之间通过二分法寻找到合适的神经元个数。在全连接层中,每一层的后面都加入了LeakyRelu和Relu激活函数,经过试验发现,当一半的神经元使用Relu激活函数,另一半使用LeakyRelu激活函数时能够达到最优的结果。

1.3 CA-ResNet深度神经网络模型结构及算法

在全连接层中,每层后面除了激活函数外,还加入了Dropout层来随机裁剪掉一定比例的神经元,以防止过拟合。于是最终的算法实验模型见图3。算法模型中详细的参数设置及网络结构见表1,表1中的参数及代表的具体含义为:Layer:神经网络当前层次的类型;Output Shape:当前层输出的维度;Param#:当前层的参数数量;Activation:当前层使用的激活函数;Connected to:与当前层相连接的上一层网络的编号。算法对应的伪代码表见表2。

1.4 CA-ResNet深度神经网络模型参数优化

CA-ResNet算法训练流程见图4,具体来说整个流程大致为以下5个步骤:

1) 数据集进行字符转换和归一化等预处理,把数据分为正常和异常两类,分别用0和1进行标识。

2) 将处理好的数据输入到CA-ResNet深度神经网络中,使用全连接神经网络对数据进行特征提取,然后在同一层中分别使用Relu激活函数和LeakyRelu激活函数对当前层输出进行非线性化。

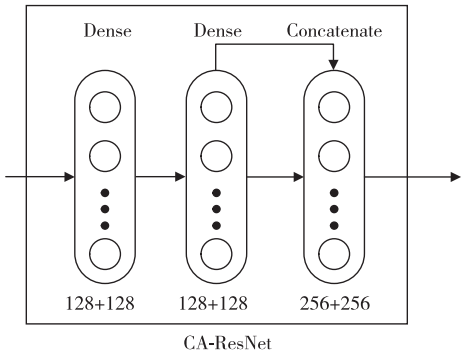

图 2　CA-ResNet 层结构图

Fig. 2　CA-ResNet layer structure diagram

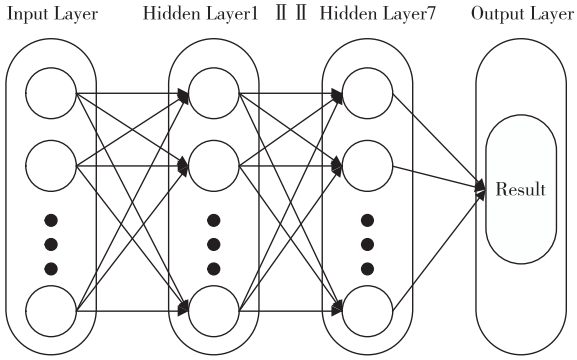

图 3　CA-ResNet 算法模型结构图

Fig. 3　CA-ResNet algorithm model structure diagram

表 1　CA-ResNet 算法结构及参数详细列表

Tab. 1　Detailed list of CA-ResNet algorithm structure and parameters

Layer	Output Shape	Param #	Activation	Connected to
input_1(InputLayer)	(None,41)	0	None	
dense_1(Dense)	(None,128)	5 376	Relu	Input_1[0][0]
dense_2(Dense)	(None,128)	5 376	Leaky_Relu	Input_1[0][0]
concatenate_1(Concatenate)	(None,256)	0	None	dense_1[0][0] dense_2[0][0]
dropout_1(Dropout)	(None,256)	0	None	concatenate_1[0][0]
dense_3(Dense)	(None,768)	197 376	Relu	dropout_1[0][0]
dense_4(Dense)	(None,768)	197 376	Leaky_Relu	dropout_1[0][0]
concatenate_2 (Concatenate)	(None,1 536)	0	None	dense_3[0][0] dense_4[0][0]
dropout_2(Dropout)	(None,1 536)	0	None	concatenate_2[0][0]
dense_5(Dense)	(None,150)	230 550	Relu	dropout_2[0][0]
dense_6(Dense)	(None,150)	230 550	Leaky_Relu	dropout_2[0][0]
concatenate_3 (Concatenate)	(None,300)	0	None	dense_5[0][0] dense_6[0][0]
dropout_3(Dropout)	(None,300)	0	None	concatenate_3[0][0]
dense_7(Dense)	(None,128)	38 528	Relu	dropout_3[0][0]
dense_8(Dense)	(None,128)	38 528	Leaky_Relu	dropout_3[0][0]
concatenate_1 (Concatenate)	(None,256)	0	None	dense_7[0][0] dense_8[0][0]
dense_9(Dense)	(None,128)	32 896	Relu	concatenate_4[0][0]
dense_10(Dense)	(None,128)	32 896	Leaky_Relu	concatenate_4[0][0]
concatenate_5 (Concatenate)	(None,256)	0	None	dense_9[0][0] dense_10[0][0]
concatenate_6 (Concatenate)	(None,512)	0	None	concatenate_5[0][0] concatenate_4[0][0]
dropout_4(Dropout)	(None,512)	0	None	concatenate_6[0][0]
dense_11(Dense)	(None,128)	65 664	Relu	dropout_4[0][0]
dense_12(Dense)	(None,128)	65 664	Leaky_Relu	dropout_4[0][0]
concatenate_7 (Concatenate)	(None,256)	0	None	dense_11[0][0] dense_12[0][0]
dense_13(Dense)	(None,1)	257	Sigmoid	concatenate_7[0][0]

表2 CA-ResNet算法形式化伪代码

Tab.2 CA-ResNet algorithm formalized pseudocode

行号	伪代码
1	function BP(D,η)
2	在(0,1)范围内随机初始化网络中的所有连接权值和阈值
3	repeat
4	for all$(x_k,y_k)\in D$ do
5	计算当前样本输出
6	计算输出神经元的梯度:$g_j=-\frac{\partial E_k}{\partial \hat{y}^k}\cdot\frac{\partial \hat{y}^k}{\partial \beta_j}=-(\hat{y}_j^k-y_j^k)f_1'(\beta_j-\theta_j)-\hat{y}_j^k-y_j^k)f_2'(\beta_j-\theta_j)=\hat{y}_j^k(1-\hat{y}_j^k)\hat{y}_j^k-y_j^k)$;
7	计算隐层的神经元梯度项:$e_h=-\frac{\partial E_k}{\partial b_h}\cdot\frac{\partial b_h}{\alpha_h}-\sum_{j=1}^{\ell}\frac{\partial E_k}{\partial \beta_j}\cdot\frac{\partial \beta_j}{b_h}(f_1'(\alpha_h-\gamma_h)+f_2'(\alpha_h-\gamma_h))=$ $\sum_{j=1}^{\ell}\omega_{hj}g_j(f_1'(\alpha_h-\gamma_h)+f_2'(\alpha_h-\gamma_h))=b_h(1-b_h)\sum_{j=1}^{\ell}\omega_{hj}g_j$;
8	更新权值 $\Delta\omega_{hj}=\eta g_j b_h,\Delta\nu_{ih}=\eta e_h x_i$;
9	更新阈值 $\Delta\theta_i=-\eta g_j,\Delta\gamma_h=-\eta e_h$;
10	end for
11	until 达到停止条件
12	end function

注:CA-ResNet算法的输入为训练集 $D=\{(x_k,y_k)\}_{k=1}^m$ 和学习率 η,输出为连接权值或阈值确定的多层前馈神经网络

3) 经过多层的全连接神经网络后,为防止深度神经网络由于梯度消失造成准确率下降,再使用ResNet对全连接层神经网络进行特征保留。

4) 最后一层使用单个神经元并通过sigmoid激活函数对输出进行二分类,再使用二分类交叉熵损失函数进行误差计算。

5) 使用nadam优化器对CA-ResNet深度神经网络进行参数优化。

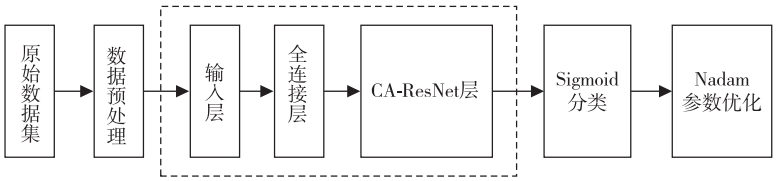

图4 CA-ResNet算法训练流程图

Fig.4 Flow chart of CA-ResNet algorithm training

2 基于CA-ResNet深度神经网络的入侵检测算法

目前,入侵检测算法中事件分析器主要采用的是 K 近邻、SVM、决策树、贝叶斯网络模型以及深度神经网络等分类算法。不同的算法针对不同的数据集表现有5%以内的准确率差异,针对KDD Cup99[17]入侵检测数据集,采用CA-ResNet深度神经网络作为事件分析器的入侵检测算法取得最优的效果。

通过前文对模型设计和参数的优化,将使用数据集优化完成的CA-ResNet深度神经网络模型嵌入到整个网络入侵检测系统中。

CA-ResNet模型作为入侵检测算法中的核心部分,为整个入侵检测系统提供了识别功能,可以判断出当前连接是否为异常连接。整个入侵检测算法的流程(图5)主要有5个步骤:

1）将训练好的 CA-ResNet 深度神经网络模型部署到入侵检测系统中的检测策略库；

2）开启入侵检测系统，对整个局域网或个体主机进行入侵检测；

3）事件发生器负责监视网络连接请求和网络通信状态并采集原始数据，将收集到的原始数据转换为事件，向系统的其他部分提供此事件；

4）事件分析器接收来自事件数据库的网络连接事件，并使用课题组部署在检测策略库中的算法进行入侵检测判断；

5）如果由 4）判断结果为网络入侵事件，那么将由响应单元采取与响应策略库相对应的响应动作。

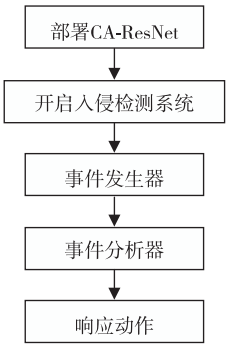

图 5　基于 CA-ResNet 算法的入侵检测流程

Fig. 5　Intrusion detection process based on CA-ResNet algorithm

3 实验与分析

3.1 实验环境

CA-ResNet 入侵检测算法实验为了更加高效和稳定地进行模型搭建和模型参数的训练，采用了基于 TensorFlow 的 Keras 深度学习框架。实验具体的硬件环境和软件环境如表 3 所示。

入侵检测实验结果的评价指标采用准确率（Accuracy，P_{Accuracy}）、精确率（Precision，$P_{\text{Precision}}$）、召回率（Recall，P_{Recall}）、F1-score（召回率和精确率的加权调和平均数，V_{F1}）等 4 个指标。这 4 个评价指标一般依赖于一个 2×2 的混淆矩阵（表 4）。在混淆矩阵中，定义 V_{TN} 为数据正常且预测也正常的数据量，V_{TP} 为数据异常且预测也异常的数据量，V_{FN} 为数据异常而预测正常的数据量，V_{FP} 为数据正常而预测异常的数据量。于是关于这 4 个指标的具体计算公式为：$P_{\text{Accuracy}}=\dfrac{V_{\text{TP}}+V_{\text{TN}}}{V_{\text{TP}}+V_{\text{TN}}+V_{\text{FP}}+V_{\text{FN}}}$，$P_{\text{Recall}}=\dfrac{V_{\text{TP}}}{V_{\text{TP}}+V_{\text{FN}}}$，$P_{\text{Precision}}=\dfrac{V_{\text{TP}}}{V_{\text{TP}}+V_{\text{FP}}}$，$V_{\text{F1}}=2\times\dfrac{P_{\text{Precision}}\times P_{\text{Recall}}}{P_{\text{Precision}}+P_{\text{Recall}}}$。

表 3　实验环境

Tab. 3　Experimental environment

名称	配置
CPU	型号：i7-7700，主频：3.6 GHz，4 核 8 线程 64 位
GPU	型号：NVIDIA GeForce GTX 1060，显存：6 GB
内存	8 GB
操作系统	Windows 10 专业版
编程语言	Python 3.7

表 4　混淆矩阵

Tab. 4　Confusion matrix

	预测正确	预测错误
预测为正例	V_{TP}	V_{FP}
预测为反例	V_{TN}	V_{FN}

3.2 入侵检测数据集分析及预处理

3.2.1 *数据集*　CA-ResNet 入侵检测实验采用 KDD Cup99 数据集，该数据集中的每个网络连接数据为 42 维，41 维特征数据加上 1 维标记位。KDD Cup99 数据集中的测试数据和训练数据有着不同的概率分布，测试数据包含了一些从未出现在训练数据中的攻击类型，这使得入侵检测更具有现实性，可以更加客观地评价算法在入侵检测中的泛化能力，因此自 KDD Cup99 数据集发布以来，它就成为了 IDS 评估使用最广泛的数据集。本文也选用 KDD Cup99 作为本实验的数据集。该数据集由 500 万个包含 41 个特征的网络连接记录组合而成，其中的模拟攻击可以分为以下 4 大类：

1）拒绝服务攻击（DoS）：网络入侵者通过向目标主机注入大量请求，从而导致目标主机资源的匮乏，从而短暂或永久地中断服务。

2）扫描攻击（Probe）：网络入侵者试图收集有关网络计算机的信息类型，然后通过目标主机防火墙并获得根访问。

3）来自远程主机的未授权访问（R2L）：攻击者可以向目标发送数据包，但在该主机本身没有用户账户的入侵，试图利用一个漏洞获得本地访问，伪装成目标主机现有的用户。

4) 未授权的本地超级用户特权访问(U2R):一种网络入侵常用的方法,入侵者试图利用用户先前存在的访问权限,并利用漏洞获得根控制。

实验数据训练集采用kddcup.data_10_percent,测试集采用Corrected,都是由KDD99数据集提供的一个10%的训练子集和测试子集(表5)。在训练集中共出现了22个攻击类型,而剩下的17种攻击类型只在测试集中出现,这样的设计可以更加有效地检验分类模型的泛化能力。

表5 数据集数据分布表

Tab.5 Data set data distribution table

KDD Cup99	正常访问	DoS	Probe	U2R	R2L	样本总数
训练集	97 278	391 458	4 107	152	1 126	494 021
测试集	60 593	229 853	4 166	228	16 189	311 049

3.2.2 *数据预处理* 1) 符号特征的数字化。为了提高深度学习模型的泛化能力和算法的运算效率,本文对KDD Cup99数据集进行统一处理,将字符型数据转化为数值型数据。转化数值型数据时,采用函数映射的方法,将每一种类型的字符形式都对应唯一确定的二进制编码,即:f: $A\rightarrow B(10)$,该式中:A 为网络流数据特征中原始的字符串,B 为二进制编码格式的数据;f 为映射关系。其中4种字符型数据映射关系的集合见表6。

表6 数据集字符映射关系

Tab.6 Character mapping relation of data set

字符类型	映射关系
Protocol_type (Protocol type)	A:{TCP,UDP,ICMP} B:{1,2,3}
Service (Target host network service type)	A:{'aol','auth',…,'Z39_50'} B:{1,2,…,70}
Flag (Connection normal or wrong)	A:{'OTH','REJ','RSTO','RSTOS0','RSTR','S0','S1','S2','S3','SF','SH'} B:{1,2,…,11}
Label (Network intrusion flag)	A:{'NORMAL','DOS','U2R','R2L','PROBE'} B:{0,1}

2) 数值特征的标准化。该数据集的取值范围较大且数据分布不均衡,在网络中出现收敛速度慢和精度不准确等问题,为了解决该问题,本文将数据进行标准化处理,即:$m_j=\frac{1}{n}(x_{1j}+x_{2j}+\cdots+x_{nj})$,$s_j=\frac{1}{n}(|x_{1j}-m_j|+|x_{2j}-m_j|+\cdots+|x_{nj}-m_j|)$,$x'_{ij}=\frac{x_{ij}-m_j}{s_j}$。其中:$m_j$ 为平均值,s_j 表示绝对偏差之和,x_{ij} 为预处理后的数值特征,标准化后的值为 x'_{ij}。

3) 归一化处理[18]。为了消除因为量纲导致的不同特征之间具有的巨大差异,需要对剩下的38个特征向量进行归一化,采用最大最小归一化方法,具体计算公式为:$x''=\frac{x-x_{\min}}{x_{\max}-x_{\min}}$。其中:$x_{\min}$,$x_{\max}$ 分别为每一个数据项的最小值和最大值,x 为原始数据的值,x'' 为归一化后的数据值。

3.3 实验结果分析

3.3.1 *实验评价指标与超参数分析* 在CA-ResNet入侵检测算法的实验中通过算法的实验结果在评价指标上的表现,从3个方面采用控制变量法分析不同的超参数对算法检测效果的影响:

1) 不同优化器对检测性能的影响。由于目前深度神经网络中参数的求解都是以梯度下降法为基础的,所以算法陷入局部最优解是无法避免的问题。但是,可以通过使用不同的优化器来尽可能地降低陷入局部最优解的概率。为了研究不同优化器对模型的检测率的影响,本文使用几种不同的优化器对模型进行优化,实验结果见表7,其中 α 表示LeakyRelu激活函数的参数取值。通过实验对比发现,nadam优化器和sgd优化器可以在CA-ResNet算法上取得较好的结果,本文使用这两种优化器进行进一步的实验,来确定参数 α 以及Batch size大小

给算法带来的不同影响和测试效果。

2）LeakyRelu 激活函数的参数 α 对检测性能的影响。为了对比 LeakyRelu 激活函数中的参数 α 对入侵检测算法性能的影响，课题组选用 nadam 和 sgd 两种优化性能最好的优化器进行对比试验，以 0.05 为步长，从 $\alpha=0$ 开始对参数 α 进行大小调整。通过两组对比实验（表 8～9）发现当 $\alpha=0.1$ 时，在使用 nadam 优化器和 sgd 优化器对算法参数进行优化时，均能取得同组最优的结果。使用 nadam 优化器时的最优结果准确率为 0.939，F1-score 能达到 0.961。同时，使用 sgd 优化器时的最优结果略低于使用 nadam 优化器时的最优结果，准确率为 0.936，F1-score 能达到 0.959。

表 7　优化器选择实验对照表

Tab. 7　Optimizer selection experiment comparison table

α	优化器	Batch size	准确率	F1-score
0.1	nadam	200	0.935	0.958
0.1	adamax	200	0.931	0.955
0.1	adadelta	200	0.927	0.954
0.1	adagrad	200	0.929	0.954
0.1	rmsprop	200	0.928	0.954
0.1	sgd	200	0.936	0.959

表 8　sgd 优化器下 α 选择实验对照表

Tab. 8　Comparison table of α selection experiment under SGD optimizer

α	优化器	Batch size	准确率	F1-score
0.00	sgd	200	0.928	0.954
0.05	sgd	200	0.936	0.959
0.10	sgd	200	0.936	0.959
0.15	sgd	200	0.928	0.955
0.20	sgd	200	0.930	0.955
0.25	sgd	200	0.927	0.954

表 9　nadam 优化器下 α 选择实验对照表

Tab. 9　Comparison table of α selection experiment under nadam optimizer

α	优化器	Batch size	准确率	F1-score
0.00	nadam	200	0.928	0.954
0.05	nadam	200	0.934	0.958
0.10	nadam	200	0.939	0.961
0.15	nadam	200	0.933	0.957
0.20	nadam	200	0.929	0.955
0.25	nadam	200	0.926	0.953

3）训练 Batch size 对检测性能的影响。在确定了优化器类型和 LeakyRelu 激活函数的 α 值之后，课题组还研究了训练 Batch size 对算法性能的影响。5 种不同的 Batch size 在 nadam 和 sgd 优化器上的实验结果见表 10 和表 11，从这两个表可以看出，不同的 Batch size 对算法的准确率和 F1-score 影响较大。当使用 nadam 优化器，Batch size 为 400 时取得最佳实验结果，准确率和 F1-score 分别为 0.950，0.975。

表 10　sgd 优化器下 Batch size 选择实验对照表

Tab. 10　Comparison table of Batch size selection experiment under SGD optimizer

α	优化器	Batch size	准确率	F1-score
0.1	sgd	100	0.929	0.955
0.1	sgd	200	0.924	0.952
0.1	sgd	300	0.935	0.959
0.1	sgd	400	0.939	0.961
0.1	sgd	500	0.936	0.959

表 11　nadam 优化器下 Batch size 选择实验对照表

Tab. 11　Comparison table of Batch size selection experiment under nadam optimizer

α	优化器	Batch size	准确率	F1-score
0.1	nadam	100	0.934	0.958
0.1	nadam	200	0.939	0.959
0.1	nadam	300	0.940	0.961
0.1	nadam	400	0.950	0.975
0.1	nadam	500	0.934	0.959

此外，再分别对 nadam 优化器和 sgd 器优化在准确率和 F1-score 两个指标上进行了直观的统计分析。通过图 6 可以看出 nadam 优化器在准确率指标上的表现要优于 sgd 优化器，通过图 7 可以看出 nadam 优化器在 F1-score 指标上的表现要优于 sgd 优化器。

3.3.2 *与类似入侵检测算法的对比分析*　实验完成后，将最终确定的 CA-ResNet 算法与其它经典的机器学习算法以及最近提出的深度学习算法进行对比（表 12）。课题组使用所有模型在测试数据集上的准确率、精确率、召回率、F1-score 以及时间复杂度 O 进行详细对比，其中的 k 表示算法的深度。

为了更加直观的体现出 CA-ResNet 算法的优点，图 8 使用直方图的形式与表 12 中表现最优的 Naive Bayes 算法、DNN 算法和 CNN 算法在 4 个不同的评价指标上进行直观的比较。

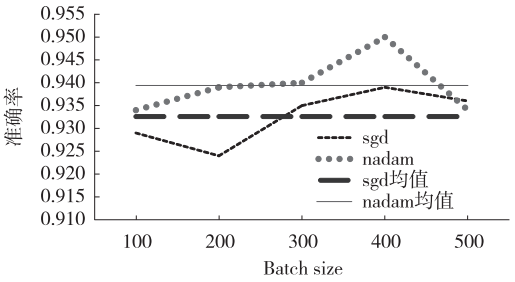

图6 nadam优化器与sgd优化器的准确率对比图

Fig. 6 Accuracy comparison diagram of the nadam optimizer and the SGD optimizer

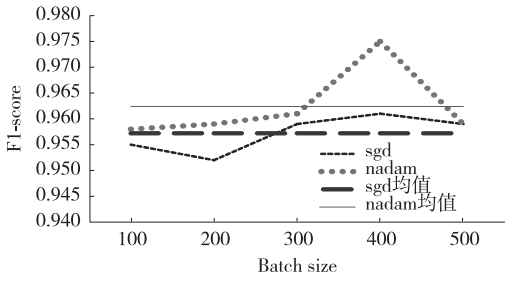

图7 nadam优化器与sgd优化器的F1-score对比图

Fig. 7 F1-score comparison diagram of the nadam optimizer and the SGD optimizer

表12 CA-ResNet算法与其他算法的效果对比

Tab. 12 Comparison of CA-ResNet algorithm with other algorithms

算法	准确率	精确率	召回率	F1-score	O
SVM	0.811	0.992	0.772	0.868	1
LR	0.848	0.989	0.821	0.897	1
SVM-KNN[3]	0.911	0.990	0.898	0.943	$k\times n$
RF	0.927	0.999	0.910	0.953	1
Decision Tree	0.928	0.999	0.912	0.953	$\log_2 n$
KNN	0.929	0.998	0.913	0.954	$k\times n$
Naive Bayes	0.929	0.988	0.923	0.955	1
DNN[11]	0.930	0.988	0.923	0.955	k
CNN[8]	0.937	0.990	0.930	0.960	k
CA-ResNet	0.950	0.988	0.945	0.975	k

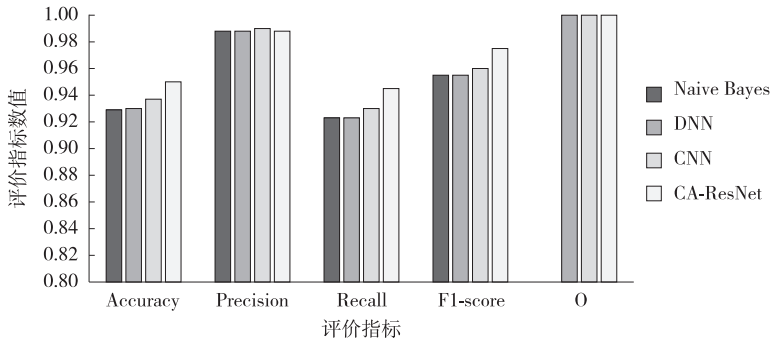

图8 CA-ResNet算法与其他算法的直方图对比

Fig. 8 Histogram comparison between CA-ResNet algorithm and other algorithms

由表12和图8可见,CA-ResNet算法与传统的机器学习算法相比较,在准确率、召回率和F1-score上有一定的提高,准确率的最大差距能达到13.9%,最小差距为2.1%,F1-score的最大差距为10.2%,最小差距为1.5%,从总体上比较,本文提出的算法比传统的机器学习算法有较大的提升,主要体现在时间复杂度上。CA-ResNet与文献[8,11]提出的深度学习算法比较,准确率和F1-score也有一定的提高,精确率略有不足,召回率有较大的提高。与文献[19]中最新提出的基于KNN离群点检测和随机森林的多层入侵检测方法相比在准确率上也略有提升,在时间复杂度上有明显的降低,提升了算法的检测速度。

3.3.3 CA-ResNet算法在UNSW-NB15数据集中的实验分析 不失一般性,将使用在KDD Cup99数据集上的本文算法,用相同的步骤在UNSW-NB15数据集上进行实验并与同类算法进行对比分析。课题组使用所有模型在测试数据集上的准确率、精确率、召回率以及F1-score进行详细对比。

从表 13 可以看出，CA-ResNet 算法在 UNSW-NB15 数据集中仍然有着较好的表现，在精确度这一决定性的指标上也能体现出该算法的检测性能。与 Decision Tree 相比较准确率提高了 2%，F1-score 提高了 2%。

表 13　CA-ResNet 算法与其他算法的效果对比

Tab. 13　Comparison of CA-ResNet algorithm with other algorithms

算法	准确率	精确率	召回率	F1-score
ANN	0.813	0.989	0.772	0.868
Naive Bayes	0.821	0.999	0.801	0.890
LR	0.832	0.988	0.812	0.903
Decision Tree	0.856	0.990	0.824	0.907
CA-ResNet	0.875	0.988	0.858	0.926

4 结束语

本文提出一种结合了协同激活函数和 ResNet 全连接神经网络的深度神经网络入侵检测算法 CA-ResNet。使用全连接神经网络作为基础网络结构，在各层网络之间加入 Relu 激活函数和 LeakyRelu 激活函数协同使用，来更好的提取数据特征。在 ResNet 理论的基础上融合协同激活函数的使用，提出了 CA-ResNet 层网络结构。CA-ResNet 层可以有效的防止由于网络层数过多而造成的梯度消失问题，保证了模型的检测效果，同时提高了网络对数据的特征提取能力。CA-ResNet 算法较 K 近邻、SVM、决策树、贝叶斯网络模型等机器学习算法，在准确率和 F1-score 方面都有明显提高，较 CNN、CNN-BiLSTM 等深度学习算法[20]也有一定的提升。

本文提出的 CA-ResNet 算法在入侵检测识别的准确率和 F1-score 上有一定的提升，但在精确率这一指标上表现欠佳、在使用 nadam 优化器训练过程中的稳定性较差。如何解决这两个问题需要进一步研究，下一步研究将针对 CA-ResNet 入侵检测算法的泛化能力展开，将该算法运用到最新的入侵检测数据集中，同时研究如何实现该算法在防火墙以及交换机上的物理部署。

参考文献：

[1] 黎佳玥，赵波，李想，等. 基于深度学习的网络流量异常预测方法[J]. 计算机工程与应用，2020，56(6)：39-50.
LI J Y，ZHAO B，LI X，et al. Network traffic anomaly prediction method based on deep learning[J]. Computer Engineering and Applications，2020，56(6)：39-50.

[2] SANTORO D，ESCUDERO-ANDREU G，KYRIAKOPOULOS K G，et al. A hybrid intrusion detection system for virtual jamming attacks on wireless networks[J]. Measurement，2017，109：79-87.

[3] 付子燨，徐洋，吴招娣，等. 基于增量学习的 SVM-KNN 网络入侵检测方法[J]. 计算机工程，2020，46(4)：115-122.
FU Z X，XU Y，WU Z D，et al. SVM-KNN network intrusion detection method based on incremental learning[J]. Computer Engineering，2020，46(4)：115-122.

[4] 郭慧，刘忠宝，柳欣. 基于云模型与决策树的入侵检测方法[J]. 计算机工程，2019，45(4)：142-147.
GUO H，LIU Z B，LIU X. Intrusion detection method based on cloud model and decision tree[J]. Computer Engineering，2019，45(4)：142-147.

[5] 赵会群，刘金銮. 基于贝叶斯网络的复杂事件大数据处理系统测试数据生成方法研究[J]. 计算机应用研究，2018，322(8)：155-158.
ZHAO H Q，LIU J L. Research on complex event big data processing system test data generation method based on Bayesian network[J]. Application Research of Computers，2018，35(8)：155-158.

[6] 朱虎明，李佩，焦李成，等. 深度神经网络并行化研究综述[J]. 计算机学报，2018，41(8)：1861-1881.
ZHU H M，LI P，JIAO L C，et al. Review of parallel deep neural network[J]. Chinese Journal of Computers，2018，41(8)：1861-1881.

[7] GU G X，CHEN C T，BUEHLER M J. De novo composite design based on machine learning algorithm[J]. Extreme Mechanics Letters，2017：19-28.

[8] 刘月峰，王成，张亚斌，等. 用于网络入侵检测的多尺度卷积 CNN 模型[J]. 计算机工程与应用，2019，55(3)：90-95.
LIU Y F，WANG C，ZHANG Y B，et al. Multi-scale convolution CNN model for network intrusion detection[J]. Computer Engineering and Applications，2019，55(3)：90-95.

[9] JAVAID A Y，NIYAZ Q，SUN W，et al. A deep learning approach for network intrusion detection system[EB/OL]. (2015-12-03)[2021-01-08]. https://dl.acm.org/doi/epdf/10.4108/eai.3-12-2015.2262516.

[10] HUANG H，XU H，WANG X，et al. Maximum F1-Score discriminative training criterion for automatic mispronunciation detection[J]. IEEE/ACM Transactions on Audio，Speech，and Language Processing，2015，23(4)：787-797.

[11] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, USA: IEEE, 2016: 770-778.

[12] VIGNESWARAN R K, VINAYAKUMAR R, SOMAN K P, et al. Evaluating shallow and deep neural networks for network intrusion detection systems in cyber security[C]//2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT). Bengaluru, Indian: IEEE, 2018: 18166942.

[13] 杨观赐,杨静,李少波,等. 基于 Dopout 与 ADAM 优化器的改进 CNN 算法[J]. 华中科技大学学报(自然科学版),2018,46(7):122-127.
YANG G S, YANG J, LI S B, et al. Improved CNN algorithm based on Dopout and ADAM Optimizer[J]. Journal of Huazhong University of Science and Technology (Natural Science), 2018, 46(7): 122-127.

[14] HORNIK K, STINCHCOMBE M, WHITE H. Multilayer feedforward networks are universal approximators[J]. Neural Networks, 1989, 2(5): 359-366.

[15] CHEN T, LU S, FAN J. SS-HCNN: semi-supervised hierarchical convolutional neural network for image classification[J]. IEEE Transactions on Image Processing, 2019, 28(5): 2389-2398.

[16] 张涛,杨剑,宋文爱,等. 关于改进的激活函数 TReLU 的研究[J]. 小型微型计算机系统,2019,40(1):60-65.
ZHANG T, YANG J, SONG W A, et al. Research on improved activation function TReLU[J]. Minicomputer Systems, 2019, 40(1): 60-65.

[17] SIDDIQUE K, AKHTAR Z, KHAN F A, et al. KDD Cup99 data sets: a perspective on the role of data sets in network intrusion detection research[J]. Computer, 2019, 52(2): 41-51.

[18] RUOTI S, HEIDBRINK S, ONEILL M, et al. Intrusion detection with unsupervised heterogeneous ensembles using cluster-based normalization[C]//2017 IEEE International Conference on Web Services. Honolulu, USA: IEEE, 2017: 17188381.

[19] 任家东,刘新倩,王倩,等. 基于 KNN 离群点检测和随机森林的多层入侵检测方法[J]. 计算机研究与发展,2019,56(3):116-125.
REN J D, LIU X Q, WANG Q, et al. Multilayer intrusion detection method based on KNN outliers detection and random forest [J]. Journal of Computer Research and Development, 2019, 56(3): 116-125.

[20] 张玉清,董颖,柳彩云,等. 深度学习应用于网络空间安全的现状、趋势与展望[J]. 计算机研究与发展,2018,55(6):3-28.
ZHANG Y Q, DONG Y, LIU C Y, et al. Situation, trends and prospects of deep learning applied to cyberspace security[J]. Journal of Computer Research and Development, 2018, 55(6): 3-28.

Intrusion Detection Algorithm Based on CA-ResNet Network and Nadam Optimizer

ZHANG Wenlong, WEI Yan, LI Yuanyuan, JIANG Junrui, ZHANG Kun, ZHANG Yang

(Intelligent Perception and Application of Big Data in Education Chongqing Engineering Research Center, College of Computer and Information Science, Chongqing Normal University, Chongqing 401331, China)

Abstract: [**Purposes**] For deep learning model in network intrusion detection parameter training is easy to cause the gradient disappeared and further cause deep learning model fitting of the decline of the accuracy on the test set. A CA-ResNet intrusion detection algorithm based on Residual Network (ResNet) incorporating LeakyRelu activation function was proposed, and the model is optimized by nadam optimizer. [**Methods**] This model adds the layer of network to DNN, through the combination of residual network and LeakyRelu activation function. [**Findings**] The problem of gradient disappearance during model training is solved. The performance of the model on the test data set is ensured, and the generalization ability of the trained model is enhanced. At the same time, the feature extraction ability and adaptability to scale of the network are improved by increasing the single-layer dimension and the depth of the total layer of the network. [**Conclusions**] The algorithm is verified by the corrected dataset in KDD Cup99 data. Experimental results show that the algorithm has higher accuracy and F1-score than CNN and CNN-BiLSTM algorithm, the accuracy rate can reach 95.0%, F1-score can reach 97.5%, the time complexity is linear time complexity.

Keywords: network security; deep learning; intrusion detection; ResNet residual network; cooperative activation function; CA-ResNet; nadam optimizer

(责任编辑 许 甲)