基于混合结构深度神经网络的 HTTP 恶意流量检测方法

李佳[1,2]，云晓春[1]，李书豪[1,3]，张永铮[1,2,3]，谢江[1,2]，方方[4]

（1. 中国科学院信息工程研究所，北京 100093；2. 中国科学院大学网络空间安全学院，北京 100049；
3. 中国科学院网络测评技术重点实验室，北京 100195；4. 长安通信科技有限责任公司，北京 102209）

摘 要：针对 HTTP 恶意流量检测问题，提出了一种基于裁剪机制和统计关联的预处理方法，进行流量的统计信息关联及归一化处理。基于原始数据与经验特征工程相结合的思想提出了一种混合结构深度神经网络，结合了卷积神经网络与多层感知机，分别处理文本与统计信息。与传统机器学习算法(如 SVM)相比，所提方法效果提升明显，F_1 值可达 99.38%，且具有更低的时间代价。标注了一套由 45 万余条恶意流量和 2 000 万余条非恶意流量组成的数据集，并依据模型设计了一套原型系统，精确率达到了 98.1%~99.99%，召回率达到了 97.2%~99.5%，应用在真实网络环境中效果优异。

关键词：异常检测；恶意流量数据；卷积神经网络；多层感知机制

中图分类号：TP393

文献标识码：A

doi: 10.11959/j.issn.1000−436x.2019019

HTTP malicious traffic detection method based on hybrid structure deep neural network

LI Jia[1,2], YUN Xiaochun[1], LI Shuhao[1,3], ZHANG Yongzheng[1,2,3], XIE Jiang[1,2], FANG Fang[4]

1. Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

2. School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100093, China

3. Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences, Beijing 100195, China

4. Changan Communication Technology Co., Ltd., Beijing 102209, China

Abstract: In response to the HTTP malicious traffic detection problem, a preprocessing method based on cutting mechanism and statistical association was proposed to perform statistical information correlation as well as normalization processing of traffic. Then, a hybrid neural network was proposed based on the combination of raw data and empirical feature engineering. It combined convolutional neural network (CNN) and multilayer perceptron (MLP) to process text and statistical information. The effect of the model was significantly improved compared with traditional machine learning algorithms (e.g., SVM). The F_1 value reached 99.38% and had a lower time complexity. At the same time, a data set consisting of more than 450 000 malicious traffic and more than 20 million non-malicious traffic was created. In addition, prototype system based on model was designed with detection precision of 98.1%~99.99% and recall rate of 97.2%~99.5%. The application is excellent in real network environment.

Key words: abnormal detection, malicious traffic data, convolutional neural network, multilayer perceptron

收稿日期：2018−07−02；**修回日期**：2018−12−11

基金项目：国家重点研究发展计划（“973”计划）基金资助项目（No.2016YFB0801502）；国家自然科学基金资助项目（No.U1736218）

Foundation Items: The National Basic Research Program of China (973 Program) (No.2016YFB0801502), The National Natural Science Foundation of China (No.U1736218)

1　引言

如今互联网快速发展，人们越来越依赖于网络的各种服务。据统计，截至 2017 年 12 月，我国网民规模达到 7.72 亿人[1]，尤其是近年提出的“互联网+”政策，标志着互联网与传统产业的融合进入新的发展阶段。然而，网络安全形势依然严峻，如仅在 2017 年国内就有 1 256 万余台主机被木马和僵尸恶意程序入侵[2]。本文针对僵尸类、木马类恶意流量进行检测，主要来源为攻击者在受害主机内植入木马僵尸等程序，进行一系列恶意行为，如恶意推广、隐私窃取、漏洞攻击等而产生的基于 HTTP 协议的流量。及时并有效地检测这些流量是维护网络安全的重要手段之一。

目前，恶意流量的检测方法主要有异常检测和误用检测。异常检测可以检测未知恶意类型，但误报率较高；误用检测精度较高，但只能识别已知入侵类型，且特征库内容需要频繁更新。本文提出的检测方法偏向于异常检测，构建的神经网络在遇到未知数据类型时也具有较好的辨识能力。

本文的贡献总结如下。

1) 提出了一种基于语义裁剪、自适应字段划分和统计信息关联的数据预处理方法。通过截断等自适应字段划分方式将流量数据转化为一系列 20×200 维的矩阵数据，然后归一化形成文本信息。同时本文统计了流量字段的相关信息，形成 41 维统计信息。与单独使用文本类信息和统计类信息作为张量化数据相比，本文的这种数据张量化方法使模型 F_1 值分别提高了约 1.4% 和 9%。

2) 基于原始数据与统计数据结合，同时在方法中选择能全面提取数据普适特征的思想，提出了一种混合结构深度神经网络模型。以流量数据的文本类信息和统计类信息作为输入，使用卷积神经网络处理文本类信息，多层感知机处理统计类信息，同时文本类信息和统计类信息在模型判别时所占的比重是可以变化的。所提模型并不是一成不变的，而是可扩展的，只是这个思想是在本文数据上的一个应用。若应用在其他特定数据集，可以对模型结构进行调整。

3) 标注了一套面向 HTTP 恶意流量检测的标注数据集。数据集由两部分组成：① 45 万条以上的 HTTP 恶意流量，包括僵尸类、木马类恶意代码的命令控制通道流量和攻击流量；② 2 000 万条以上的 HTTP 正常流量，涵盖了主流的 HTTP 协议应用类型。据本文了解，这是业界和学术界第一个针对僵尸类、木马类 HTTP 恶意流量检测的规模化深度学习数据集。

4) 设计并实现了一套从 HTTP 正常流量内识别僵尸类、木马类 HTTP 恶意流量的原型系统，自动识别流量异常行为，同时具有可扩展性。将该系统部署在服务器上进行检测，并对训练数据的黑白比例进行了多种调配。比例一致时，系统的 F_1 值达到了 99.38%；比例不一致时，召回率、F_1 值有所下降，在恶意流量占比低至 1.96% 的情况下，系统仍具有 97.2% 的召回率与 98.5%的 F_1 值。这表明该系统应用在数据集上具有良好且稳定的性能。

2　相关工作

在网络空间安全领域，恶意流量检测一直是一个热点问题。本文的研究工作是利用深度学习知识检测恶意流量，下面介绍相关的研究现状。

文献[3]建立了检测原型系统 MadTracer，利用 Web 流量检测恶意广告及其相关内容传递路径，记录网络请求、响应、浏览器事件等 HTTP 流量，建立重定向链，判断是否包含恶意广告。实验显示 MadTracer 能够达到 95%的检测准确率。

Gu 等[4]提出了一套启发式算法来区分僵尸网络流量和正常流量，并提出了原型系统 BotSniffer，通过利用僵尸网络在其网络流量中表现出的时空相关性，检测局域网中使用 IRC（internet relay chat）和 HTTP 协议的 C&C 僵尸网络。随后 Gu 等在文献[5]中提出一个通用的检测框架 BotMiner，在检测 IRC、HTTP 和 P2P（peer-to-peer）僵尸网络时可以达到 100%的召回率和 0.4%的误报率。

除上述工作之外，与恶意流量检测方向相关的研究还有恶意 URL（uniform resource locator）、社交网络恶意账户等方面的识别。文献[6]根据 URL 转发行为与传播的关联提出了基于转发的恶意 URL 监测模型，利用从社交网站搜集的信息进行评估。文献[7]针对恶意账户的识别进行了研究，包括垃圾邮件账户、被盗账户和网络钓鱼账户的检测。

但上述研究也存在实验数据集较小，普适性较差等问题，同时成果难以直接应用于恶意流量检测，也无法识别僵尸类、木马类 HTTP 恶意流量。而近年来深度学习在网络空间安全领域的应

用主要在恶意软件识别和入侵检测这两方面。

在恶意软件识别方面，文献[8]使用神经网络进行特征提取，通过利用 RNN（recurrent neural networks）判断函数位置来进行检测。文献[9]提出了一种 Android 恶意软件识别系统 Droid-Sec，使用静态和动态分析提取特征，准确率达到了 96.5%，文献[10]则改进了 Droid-Sec，扩大了训练集以便提取更精确的特征。在入侵检测方面，使用神经网络的研究大多基于 KDD99（KDD cup 1999）和 NSL-KDD。基于 KDD99，文献[11]使用 LSTM(long short-term memory)进行检测分类，取得了 96.93%的准确率。基于 NSL-KDD，文献[12]使用了 DBN+SVM 的方式进行入侵检测，最终取得了比单独方法更高的准确率。

可以看出，深度学习具有强大的特征提取能力。但目前深度学习在恶意流量检测上的研究较少，同时现有检测方法大多基于规则，即偏向误用检测、依赖于特征库。因此，本文提出了利用深度学习来解决 HTTP 恶意流量检测问题。

3 数据预处理

一次完整的 HTTP 交互过程由请求流量和响应流量组成。本文主要研究主机遭受入侵后所产生的恶意行为，而主机主要通过请求体现恶意行为。因此本文选取请求流量进行分析检测。

3.1 面向文本信息的语义裁剪与启发式划分

本文将流量请求进行语义裁剪来获取有效数据，然后将字符转化为 ASCII 值，通过对齐、截断、补足等自适应字段划分生成一系列 20×200 的矩阵数据 $\boldsymbol{A}$，组成文本信息。图 1 直观地展示了矩阵 $\boldsymbol{A}$ 的组成，各个字段行按照原始序列进行排列，可以看出 $\boldsymbol{A}$ 极好地保留了流量数据的文本信息和空间结构。这一过程宜使用卷积神经网络进行特征提取。

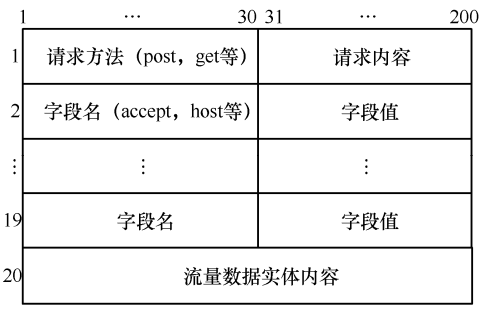

图 1 流量数据文本信息矩阵格式

3.2 关联统计信息

为更全面地提取数据特征，本文获取了流量中各个字段的统计结果作为关联统计信息。统计信息包括目的端口号、HTTP 版本及数据长度信息。本文所使用的数据中 99.9%的请求流量首部行低于 18 行，因此本文仅考虑前 18 个字段行信息，超过则截断，不足则填充 0。但这可扩展的，可根据不同数据集的实际情况进行调整。统计向量 L 如表 1 所示。

表 1 流量数据统计信息向量格式

统计信息	对应向量
端口号	$\boldsymbol{l}_1$
HTTP 版本	$\boldsymbol{l}_2$
首部字段行数	$\boldsymbol{l}_3$
各行字段名长度	$\boldsymbol{l}_4 \cdots \boldsymbol{l}_{2-1}$
URL 长度	$\boldsymbol{l}_{22}$
各行字段值长度	$\boldsymbol{l}_{23} \cdots \boldsymbol{l}_{40}$
实体内容长度	$\boldsymbol{l}_{41}$

3.3 归一化处理

在实验中，数据中分量之间较大的差异性会导致较小分量被掩盖，从而无法提取特征，导致神经网络无法有效收敛。因此需要对数据进行归一化操作，本文对文本信息和统计信息采取不一样的归一化方式，但均归一到(0, 1)。

在文本信息中，为尽可能保留其隐含信息，将其 $\boldsymbol{A}$ 内元素 ASCII 值除以 127，生成矩阵 $\boldsymbol{T}$。$\boldsymbol{T}$ 中各元素采用式(1)进行归一化。

$$t_{ij} = \frac{a_{ij}}{127}, i = 1,2,\cdots,20; j = 1,2,\cdots,200 \tag{1}$$

统计信息取值基本没有上界，因此本文采用了一个极限上边界为 1 的函数来归一化统计信息，最终生成统计向量 $\boldsymbol{X}$。$\boldsymbol{X}$ 采用式(2)进行归一化。

$$x_i = \frac{2}{1+\mathrm{e}^{-l_i}} - 1, i = 1,2,\cdots 41 \tag{2}$$

同时针对数据选择问题，本文分别使用文本信息、统计信息、文本+统计信息作为张量化数据输入模型进行了实验。图 2 和图 3 为模型选择不同数据时在训练和测试过程中的表现。根据实验结果，使用文本+统计信息的数据样本在训练与测试时，其精确率、召回率和 F_1 值均是最优。

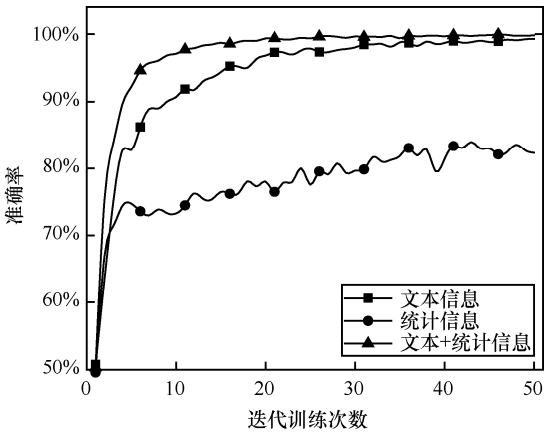

图 2　使用文本信息、统计信息和文本+统计信息的模型训练过程

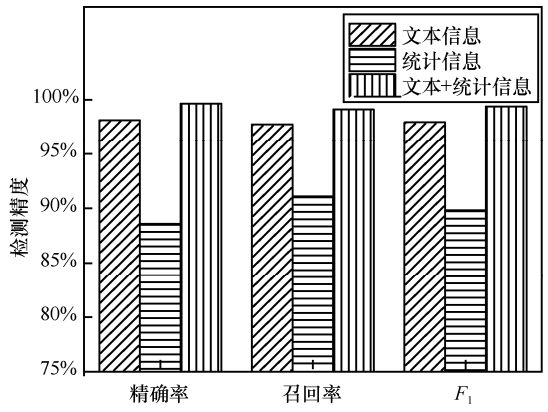

图 3　使用文本信息、统计信息和文本+统计信息的模型测试结果

4　混合结构深度神经网络模型

本文基于将原始数据和统计数据共同处理，选择能更全面提取 HTTP 流量数据结构与字符信息的卷积神经网络组件的思想，提出了一种对应的混合结构深度神经网络模型，并在特定数据集上建立一个较优的模型组合。

4.1　神经网络结构

神经网络结构如图 4 所示，文本信息矩阵 $\boldsymbol{T}$ 通过卷积层及池化层，最终生成张量 $\boldsymbol{M}_1$；统计信息 $\boldsymbol{X}$ 通过 2 个隐含层生成张量 $\boldsymbol{X}_2$。然后这 2 个张量拼接成 40 维张量 $\boldsymbol{Z}$ 进入输出层，最终通过 softmax regression 将模型输出转化为概率分布。

4.1.1　卷积池化层

1984 年，Fukushima[13]基于感受野概念提出了神经认知机（neocognitron），这是卷积神经网络的首次实现。卷积可以共享特征单元即卷积核从而降低复杂度。而每个卷积核处理部分区域，完成特征抽象的同时也提高了训练效率。本文的卷积网络使用了一个卷积层，利用 2 个卷积核，经过最大池化后将 $\boldsymbol{T}$ 转化成 10×100×2 的张量 $\boldsymbol{T}_2$。卷积核每次扫过一个字符的长度，并在 $\boldsymbol{T}$ 周围补 0 以保持卷积后的维度不变。最终 $\boldsymbol{T}_2$ 展开成一维张量进入后续隐含层。

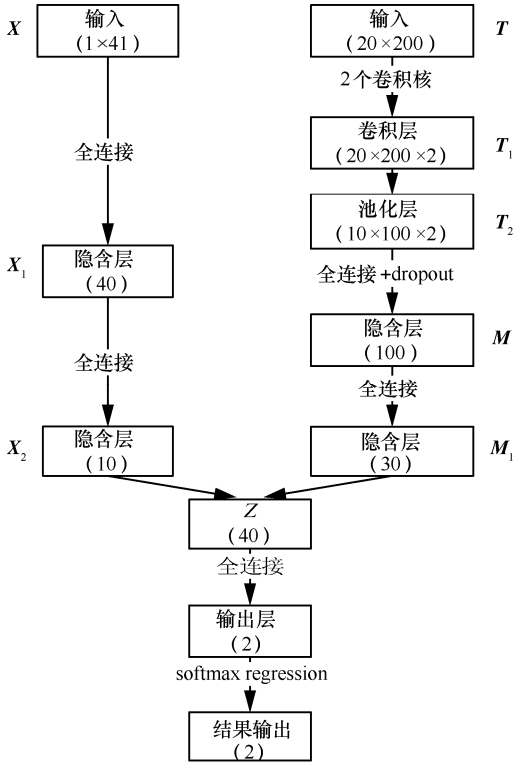

图 4　混合神经网络结构

4.1.2　多层感知机

1958 年，Rosenblatt[14]提出了感知机。感知机是最简单的神经网络，只有输入层和输出层。多层感知机可以认为是加入了隐含层的神经网络。理论上引入非线性隐含层后，神经网络可以拟合任意函数，且层数越深，拟合函数越复杂。但使用较深的网络也会遇到如梯度弥散等问题，因此在实际使用时需要权衡层数和问题复杂度之间的关系。

本文使用多层感知机处理统计信息，在提取合适的统计特征和参数更新问题之间的权衡下，经过实验调试确立了如图 4 左侧的结构，包含有 2 个隐含层，形成 10 维的张量输出。

4.1.3　分类层——softmax regression

神经网络在处理分类问题时，输出层通常使用 softmax regression，对类别特征求指数函数，然后进行标准化，使得其和为 1，特征的值越小的类，输出概率也越小。softmax regression 保证模型输出结果形成一种概率分布，如式(3)所示。

$$\mathrm{softmax}(x) = \mathrm{normalize}(\mathrm{e}^{x}) \tag{3}$$

其中，判定为第 i 类别的概率如式(4)所示。

$$\mathrm{softmax}(x)_i=\frac{\mathrm{e}^{x_i}}{\sum_j \mathrm{e}^{x_j}},i=1,2 \tag{4}$$

从式(3)和式(4)可以看出，原始神经网络的输出被用作置信度输入之后变成了一种概率分布，从而可以使用经典损失函数进行模型优化。

本文在输出层也使用了 softmax regression，张量 $\boldsymbol{Z}$ 可以认为是 2 种信息经过变换后生成的一种自编码，在输出层进行估算，得到各分类的概率分布。

4.1.4 损失函数——交叉熵

神经网络需要定义一个适当的损失函数，作为参数优化的方向。目前，处理分类问题的神经网络一般使用交叉熵作为损失函数，因为交叉熵刻画的是 2 个概率分布之间的距离，而神经网络输出可变为概率分布，数据标签 Y 用 one-hot 码表示后也可以作为一个概率分布。交叉熵如式(5)所示，表示通过概率分布 q 来表达 p 的困难程度，因此，q 代表预测值概率分布，p 代表正确的概率分布。

$$H(p,q)=-\sum_x p(x)\log(q(x)) \tag{5}$$

本文研究的是二分类问题，选择交叉熵作为损失函数，每条流量数据的标签 Y =[y_1 y_2]使用式(6)进行表示。

$$Y=\begin{cases}[1\quad 0],\text{恶意流量}\\ [0\quad 1],\text{正常流量}\end{cases} \tag{6}$$

损失函数通过式(7)进行计算。

$$L=-\sum_i y_i\log(y_i^*),i=0,1 \tag{7}$$

其中，y_i^* 为神经网络输出的二分类概率值。定义了损失函数之后，每次训练神经网络经过前向计算后回溯更新网络参数，采用 BP(back propagation)算法[15]。

4.2 过拟合的解决方法——DropOut

过拟合是深度学习中一个常见的问题，是指神经网络在训练集表现极好，在测试集表现不好的情况。降低过拟合的解决方法有数据集扩增、正则化、DropOut 等。

本文选择扩增数据和 DropOut 来降低过拟合，这些方法使模型精确率提高了 1.5%，召回率提高了 1%，F_1 值提高了 1.2%。扩增数据很好理解，即增大训练集。而 DropOut 主要原理则是在训练时将神经网络部署 DropOut 处的输出节点数据随机丢弃一部分，使相应神经元失效，而模型用于测试时再使用全部神经元。本文将 DropOut 部署在池化层之后，训练时每次掩藏 40%的神经元。

4.3 模型参数调优

为了优化模型，本文在激活函数等方面尝试了一些方案，从中选取较优的结构和参数。同时为了简化实验，本文在基于理论和前期实验效果的基础上制定了参数选择如下：网络权重初始化采用高斯正态随机分布，标准差设置为 0.1，偏置均设置为常量值 0.1；卷积核每次移动步长为 1；在训练网络参数的过程中使用 ReLU(rectified linear unit)作为激活函数；使用 Adam 作为优化器，学习率设置为 0.001。

4.3.1 激活函数选择——ReLU

激活函数是网络节点进行信号传递的工具，将网络中当前节点的输出转为后续节点的输入。目前，主流激活函数有 ReLU 函数、sigmoid 函数、tanh 函数。理论上，当数据信号到来时，sigmoid 函数会激活约一半的神经元，对输入信号很敏感；tanh 函数具有单侧抑制性，但存在梯度消失问题；而 ReLU 函数较为简单但最符合实际神经元模型。这 3 种函数表达式如式(8)~式(10)所示。

$$\text{sigmoid 函数：}\quad R=\frac{1}{1+\mathrm{e}^{-y}} \tag{8}$$

$$\tanh\text{ 函数：}\quad R=\frac{\mathrm{e}^{y}-\mathrm{e}^{-y}}{\mathrm{e}^{y}+\mathrm{e}^{-y}} \tag{9}$$

$$\text{ReLU 函数：}\quad R=\max(0,y) \tag{10}$$

模型使用 3 种不同激活函数的训练效果如图 5 和图 6 所示。使用 ReLU 函数的模型收敛的速度最快，且最终的测试效果也最优，F_1 值达到了 99.38%。

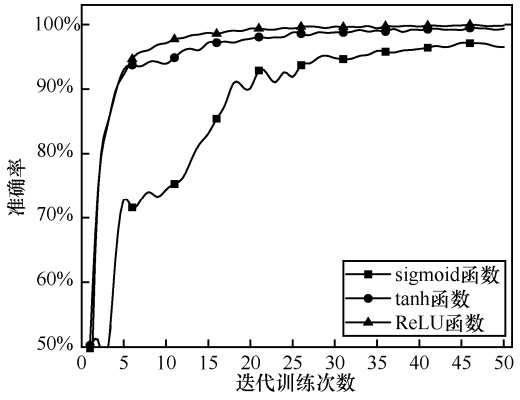

图 5 使用 sigmoid 函数、tanh 函数、ReLU 函数的模型训练过程

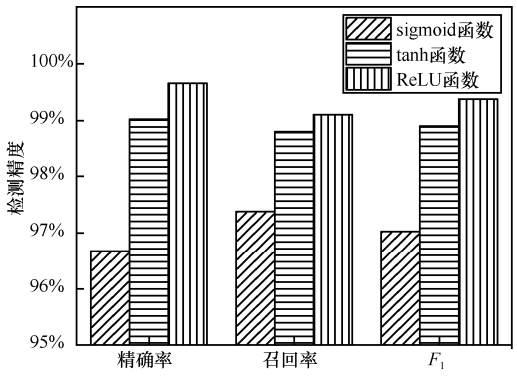

图 6　使用 sigmoid 函数、tanh 函数、ReLU 函数的模型测试结果

4.3.2　优化器选择——Adam

定义模型的损失函数后，需要选择优化损失函数的方式，也称为梯度下降或优化器。

神经网络中常用的优化器是随机梯度下降法（SGD，stochastic gradient descent），该方法随机抽取一个或多个样本计算误差，然后更新权值。本文先采用了 SGD，但在实验中模型极易发生振荡。因此在经过理论研究和实际测试后，本文选取优化 Adam 算法。

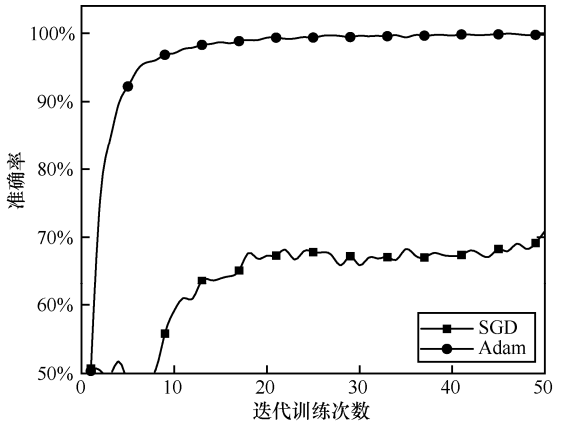

图 7　使用 Adam 算法和 SGD 算法的模型训练过程

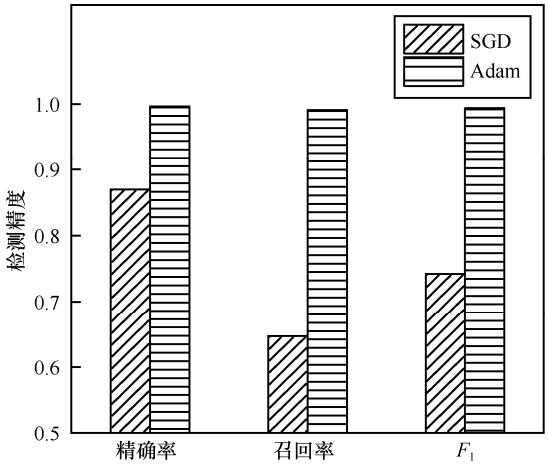

图 8　使用 Adam 算法和 SGD 算法的模型测试结果

使用 8SGD 和 Adam 的模型训练过程和测试结果如图 7 和图 8 所示，可以看出不论是模型收敛过程还是测试结果，使用 Adam 的效果都远优于 SGD。

4.3.3　卷积核大小选择——2×3

卷积核的尺寸可以表示它所提取的局部特征的范围。尺寸越大代表卷积核提取特征的能力越强，但不相关数据的干扰性和计算资源的消耗也越大。本文采用了 6 种尺寸的卷积核进行实验，卷积核的高度分别为 1、2、3，长度分别为 3、4。

实验结果如图 9 和图 10 所示，采用 2×3 卷积核的网络具有最佳的整体指标，精确率达到 99.6%，召回率达到 99.105%，F_1 值达到 99.38%。这是因为 2×3 的卷积核每次进行卷积操作时不仅能考虑流量行内的信息，而且能兼顾行间的信息，同时也考虑了距离因素。故本文中采用 2×3 大小的卷积核对于 HTTP 流量的分类具有最佳效果。

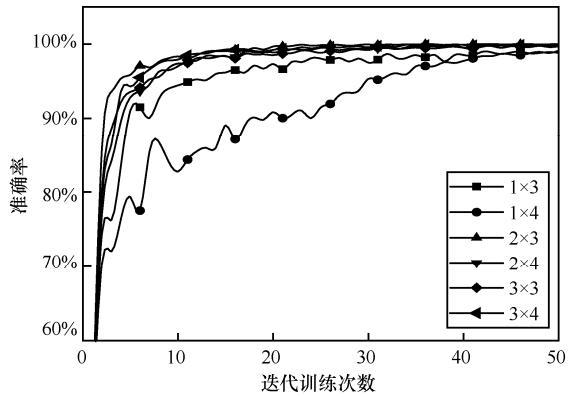

图 9　采用不同卷积核的模型训练过程

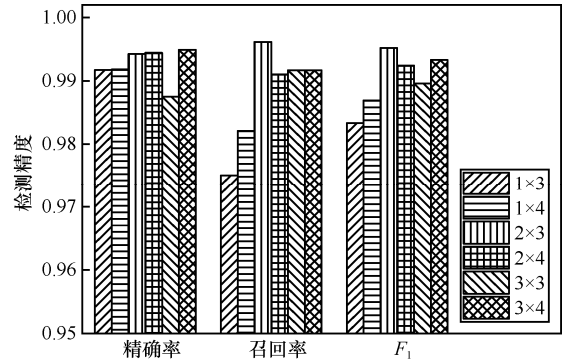

图 10　采用不同卷积核的模型测试结果

5　实验验证

本文设计并实现了一个针对 HTTP 恶意流量检测的原型系统，并利用数据集进行了实验和评估。

5.1 原型数据集

本文标注了一套基于 HTTP 协议的流量数据集。数据集内包含 45 万条以上的恶意流量，2 000 万条以上的非恶意流量。

恶意流量经木马养殖而产生。在沙箱内运行病毒样本，模拟用户主机被植入病毒木马后的恶意行为，最终提取出带有恶意行为特征的流量。本文从 200 万个恶意样本中得到 10 000 个基于 HTTP 协议的数据流，并从中筛选出 45 万个以上确定具有恶意行为特征的数据分组。本文的恶意流量中，恶意推广、下载型恶意软件产生的流量约 50%~60%，后门控制、盗号、隐私窃取等约 40%。另外，请求方法为“get”的流量数据占比为 64.66%，请求方法为“post”的流量数据占比为 35.33%。

非恶意流量来源于多个服务器和主机的正常通信。在骨干网内获取镜像流量，通过白名单抓取流量数据，经筛选后得到 2 000 万余条 HTTP 流量数据，基本涵盖了主流的 HTTP 正常通信，其中绝大部分是请求标识为“get”和“post”的交互流量数据，占比分别为 84.08%和为 15.9%。

5.2 系统框架

原型系统使用 Python 实现，开发及训练测试工作在 2 台服务器上进行，每台服务器为 8 块 GeForce GTX TITAN X 系列 GPU、主频 2 061 MHz 的 64 核 CPU、128 GB 内存。

系统框架如图 11 所示，数据经过数据张量化模块后批量输入神经网络模块进行自动训练与测试。最终进入结果统计模块进行统计并输出。

5.2.1 数据张量化模块

在数据张量化模块中，训练数据和测试数据的预处理方式一致。系统将 HTTP 流量数据中的请求部分提取出来，采用第 3 节的数据预处理方法将流量张量化，每条流量产生一条 20×200 和 1×41 的神经网络张量化数据。

5.2.2 神经网络模块

神经网络模块采用第 4 节得出的最优网络结构。模型流量数据标注 one-hot 码，模型训练时根据训练结果的反馈更新结构参数。模型收敛后，系统等待输入测试数据。

5.2.3 结果统计模块

系统输入测试数据后，结果统计模块统计所有测试数据中判断正确的数量、恶意与非恶意流量数据各自的准确率以及其他一些相关信息，将结果以图表形式展示，同时将误判的流量数据以文件形式提取出来，便于后期进行分析。

5.3 实验设计

实验主要检验系统对恶意流量的检测能力，以及在不同场景下系统是否能保持较好的辨识能力。

5.3.1 实验数据配置

在数据预处理和模型调优阶段，训练数据中恶意与非恶意流量比例为 1:1；在原型系统实验阶段，训练数据中恶意与非恶意流量比例为 1:1、1:4、1:8、1:12、1:16、1:24、1:32、1:50 等共 8 种实验场景。不同阶段恶意流量均为 50 000 条。

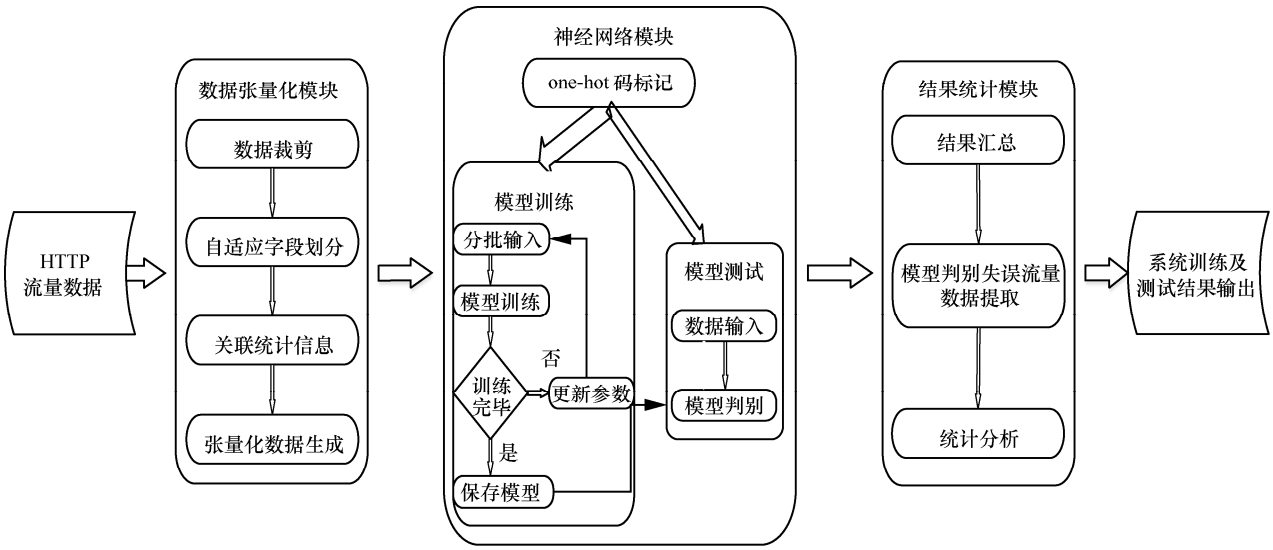

图 11 系统基本结构

数据预处理阶段、模型调优和原型系统实验阶段使用相同测试数据，共 20 000 条恶意和 160 000 条非恶意流量。测试集和训练集没有交集。

5.3.2　度量参数

为对实验进行有效的量化分析，本文引入如下基本指标：*t_b* (true black)表示恶意流量被正确识别为恶意流量的数目；*t_w*(true white)表示正常流量被正确识别为正常流量的数目；*f_b*(false black)表示正常流量被错误识别为恶意流量的数目，即误报数；*f_w*(false white)表示恶意流量被错误识别为正常流量的数目，即漏报数。

在训练阶段，主要使用准确率（ACC, accuracy）表示模型辨识能力的提升，综合展示模型在识别恶意与正常流量数据时的性能提升。因为模型在训练过程中不仅需要关注于对恶意流量的辨识，也需要对正常流量建立辨识能力，两者是同等重要的。准确率依据式(11)计算，表示综合训练准确率。

$$\mathrm{ACC}=\frac{t\_b+t\_w}{t\_b+f\_b+t\_w+f\_w} \tag{11}$$

实验测试阶段结果通过精确率 P、召回率 R 和综合评价指标 F_1 进行评价。式(12)为精确率计算方法，式(13)为召回率的计算方法。

$$P=\frac{t\_b}{t\_b+f\_b} \tag{12}$$

$$R=\frac{t\_b}{t\_b+f\_w} \tag{13}$$

评价指标由交叉验证产生各指标的平均值，包括精确率 P、召回率 R 和综合评价指标 F_1，F_1 值与精确率和召回率的关系如式(14)所示。

$$F_1=\frac{2PR}{P+R} \tag{14}$$

5.4　实验结果分析与评价

本文将模型与传统机器学习方法进行了对比实验，同时测试了相应系统在不同环境下的检测效果。实验将从收敛过程、统计结果等方面进行阐述总结及分析。

5.4.1　传统机器学习方法实验效果

本文使用 SVM 和决策树（decision tree）与本文模型进行对比。两者与决策树及其相关方法在恶意流量检测方面应用较广，如文献[16]在检测恶意软件和 DDoS（distributed denial of service）时使用了 SVM，文献[17]使用了决策树相关的方法进行网络系统传播故障的可靠性分析，均获得了不错的结果。

实验结果如图 12 所示，SVM 在分类时分别使用线性核函数和高斯核函数将数据特征向高维空间映射，F_1 值分别具有 88.587%和 85.379%，决策树使用了 CART（classification and regression tree）算法作为分类算法，F_1 达到了 84.758%。可以看出，SVM 和决策树相关方法应用在数据集上时，精确率、召回率和 F_1 值均远低于本文所用方法。图 12 中为恶意流量与非恶意流量比例 1:1 时 SVM、决策树和本文所用方法的实验结果，linear_SVM 表示使用线性核函数的 SVM 模型、RBF_SVM 表示使用高斯核函数的 SVM 模型、CART_DecTree 表示使用 CART 算法的决策树模型。

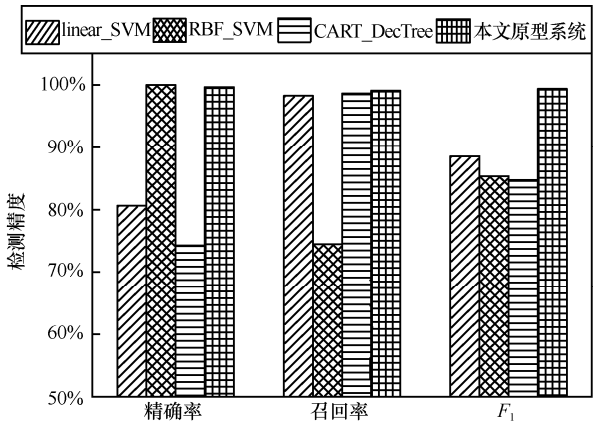

图 12　实验结果

同时从时间代价上分析，实验中本文方法的时间复杂度约 $O(n_{\mathrm{features}}^2\times n_{\mathrm{samples}}\times n_{\mathrm{iteration}})$，而 SVM 系列算法的时间复杂度在 $O(n_{\mathrm{features}}\times n_{\mathrm{samples}}^2)$ 和 $O(n_{\mathrm{features}}\times n_{\mathrm{samples}}^3)$ 之间，决策树时间复杂度为 $O(n_{\mathrm{features}}\times n_{\mathrm{samples}}^2\log(n_{\mathrm{samples}}))$。针对数据集特征样本量大、特征量固定的特点，本文方法的时间代价与样本量呈线性关系，虽然样本量的增加会导致训练迭代次数的增加，但综合增加的时间代价肯定小于样本量的平方，而且随着样本量大幅增大，模型的能力越来越趋于完备，迭代次数增加会越来越慢。因此，本文方法因为样本量增加而带来的时间复杂度低于 SVM 系列算法和决策树算法。

因此，本文方法相比于 SVM、决策树等机器学习方法更适合于数据集中 HTTP 恶意流量的检测。

5.4.2　训练阶段的模型收敛过程分析

如图 13 所示，模型使用训练集训练的过程中，虽然在不同场景下均到达了收敛，但训练数据中恶意与非恶意流量比例差别越大，准确率反而收敛越

快，如在 1:1、1:4 的实验场景下，模型收敛的速度明显慢于其他场景。这是因为随着非恶意流量比例的增大，模型在训练中更多学习的是非恶意流量，从而更易于将流量判断为非恶意流量，而比例的增加会增加检测准确率，因此训练数据比例相差越大，相应的模型在收敛过程中的整体准确率越好。

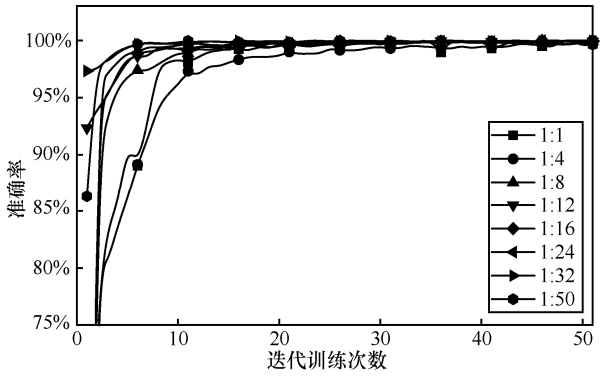

图 13　模型在不同比例恶意流量的训练数据下达到收敛的过程

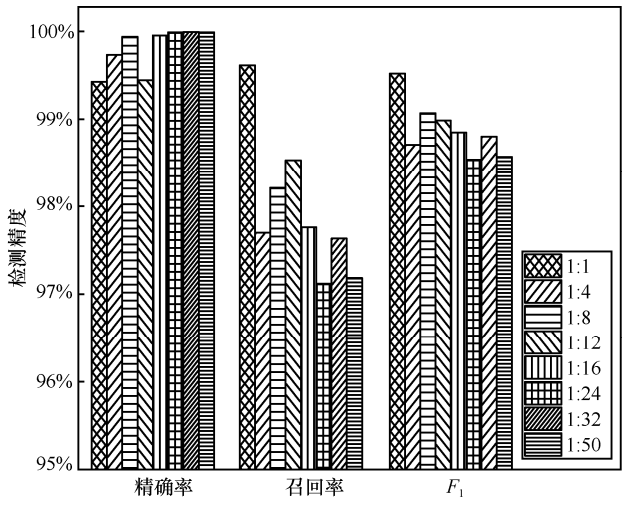

图 14　模型在不同比例恶意流量的训练数据下达到收敛后的测试结果

但需要注意的是，训练数据中恶意流量不能占比太小，否则模型需要花费更多的时空代价进行训练，同时更难以提取特征。图 15 展示了在 1:1、1:4、1:24 场景时模型对恶意流量的召回率收敛过程。1:1 场景时，召回率达到很好的收敛，几乎没有振荡；1:4 场景时，召回率达到收敛的速度变慢了，且收敛时发生的振荡也变大了；而 1:24 场景时，同样振荡，且最终的收敛效果也变得更差。可以判断的是，随着训练数据中恶意流量的占比不断下降，模型相应的辨识能力会不断减弱，最终完全丧失。因此，在保证模型检测率的前提下，需要将训练数据内恶意与非恶意流量比例控制在一定范围内。

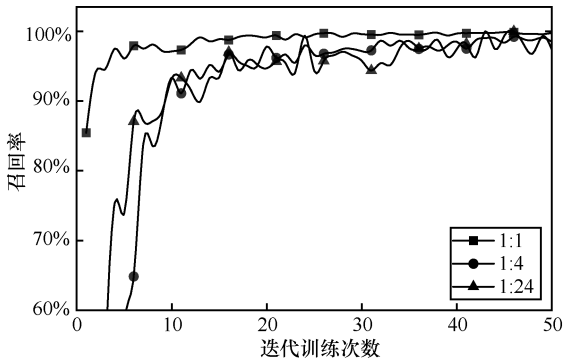

图 15　恶意流量与非恶意流量比例 1:1、1:4 和 1:24 时的召回率

5.4.3　测试阶段准确率分析

实验结果表明系统具有非常好的检测性能，如图 14 所示，随着训练数据中非恶意流量占比的增大，模型越来越偏向于将流量判断为非恶意，这也导致了系统的测试准确率越来越接近于 1，在 1:32、1:50 的实验场景中，模型精确率均达到了 99.99% 以上。但同时恶意流量占比相对下降，导致模型对恶意流量的辨识能力减弱，1:1 时，召回率为 99.105%，而 1:50 时，召回率下降到 97.188%。因为模型的测试准确率已经极为接近 1，无法再提高，但召回率降低导致模型的 F_1 值也在不断下降。但从图 15 可以看出，虽然召回率指标等随着训练数据中恶意流量占比的不断降低而降低，但即便是 1:50，即恶意流量占比仅 1.96% 时，模型的召回率、F_1 值也达到了 97.188%、98.57%，这表明系统仍然具有非常高的辨识能力，具有较强的稳定性。

6　结束语

本文引入了深度学习理论，利用原始数据和经验特征工程相结合的思想进行 HTTP 恶意流量的检测研究。采用基于裁剪机制和统计信息关联的预处理方法进行流量数据张量化处理，提出了一种卷积神经网络和多层感知机相结合的混合结构深度神经网络模型，F_1 值可以达到 99.38%。而对照实验的结果表明不论是时间代价还是检测精度，本文所提模型都远优于 SVM、决策树等传统机器学习方法。基于所提模型，本文设计并实现了一套针对 HTTP 恶意流量检测的原型系统，对于多种不同场景中的真实流量，均能有效检测恶意流量，平均精确率为 99.5%以上，召回率为 97.7%以上。此外，本文提出了一套面向 HTTP 恶意流量检测的数据集，包含 45 万条以上恶意流量和 2 000 万条以上非恶意流量，为本领域相关研究工作的实验验证提供了有力支撑。

未来将在以下 2 个方面开展进一步工作：1) 考虑恶意行为的时序性和关联性，针对包括但不限于 HTTP 的整体流量进行建模，设计更加合理的深度学习模型，提升并泛化发现隐藏恶意行为特征的能力；2) 基于真实恶意事件及相应网络流量，补充完善本文所标注数据集，使其覆盖更多种类的恶意行为。

参考文献：

[1] 中国互联网络信息中心. 中国互联网络发展状况统计报告[R]. 中国互联网络信息中心. 2018.
INIC. The statistical report on internet development in China[R]. China Internet Network Information Center. 2018.

[2] 国家互联网应急中心. 2016 年中国互联网网络安全报告[R]. 国家互联网应急中心.2017.
NIEC. A survey of china's internet security situation[R]. National Internet Emergency Center. 2017

[3] LI Z, ZHANG K, XIE Y, et al. Knowing your enemy: understanding and detecting malicious web advertising[C]//The 2012 ACM Conference on Computer and Communications Security. 2012: 674-686.

[4] GU G, ZHANG J, LEE W. BotSniffer: detecting botnet command and control channels in network traffic[C]//The Network and Distributed System Security Symposium. 2008.

[5] GU G, PERDISCI R, ZHANG J, et al. BotMiner: clustering analysis of network traffic for protocol-and structure-independent botnet detection[C]//The 17th USENIX Security Symposium. 2018: 139-154.

[6] CAO J, LI Q, Y JI, et al. Detection of forwarding-based malicious URLs in online social networks[J]. International Journal of Parallel Programming, 2016, 44(1): 163-180.

[7] ADEWOLE K S, ANUAR N B, et al. Malicious accounts: dark of the social networks[J]. Journal of Network and Computer Applications, 2017, 79: 41-67.

[8] SHIN E C R, SONG D, MOAZZEZI R. Recognizing functions in binaries with neural networks[C]//USENIX Security Symposium. 2015: 611-626.

[9] YUAN Z, LU Y, WANG Z, et al. Droid-sec: deep learning in android malware detection[C]//ACM SIGCOMM Computer Communication Review. 2014, 44(4): 371-372.

[10] YUAN Z, LU Y, XUE Y. Droiddetector: android malware characterization and detection using deep learning[J]. Tsinghua Science and Technology, 2016, 21(1): 114-123.

[11] KIM J, KIM J, THU H L T, et al. Long short term memory recurrent neural network classifier for intrusion detection[C]//Platform Technology and Service (PlatCon), 2016 International Conference on. IEEE, 2016: 1-5.

[12] SALAMA M A, EID H F, RAMADAN R A, et al. Hybrid intelligent intrusion detection scheme[M]. Soft Berlin Computing in Industrial Applications. 2011: 293-303.

[13] NASRABADI N M. Pattern recognition and machine learning[J]. Journal of Electronic Imaging, 2007, 16(4): 049901.

[14] ROSENBLATT F. The perceptron: a probabilistic model for information storage and organization in the brain[J]. Psychological Review, 1958, 65(6): 386.

[15] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors[J]. Nature, 1986, 323(6088): 533.

[16] WATSON M R, MARNERIDES A K, MAUTHE A, et al. Malware detection in cloud computing infrastructures[J]. IEEE Transactions on Dependable and Secure Computing, 2016, 13(2): 192-205.

[17] MO Y, XING L, ZHONG F, et al. Reliability evaluation of network systems with dependent propagated failures using decision diagrams[J]. IEEE Transactions on Dependable and Secure Computing, 2016, 13(6): 672-683.

[作者简介]

李佳（1983− ），男，河北邢台人，中国科学院信息工程研究所博士生，主要研究方向为网络信息安全。

云晓春（1971− ），男，黑龙江哈尔滨人，博士，中国科学院信息工程研究所客座研究员、博士生导师，主要研究方向为网络信息安全、网络恶意事件感知。

李书豪（1983− ），男，山西吕梁人，博士，中国科学院信息工程研究所副教授级高工，主要研究方向为网络信息安全、恶意代码分析与防范。

张永铮（1978− ），男，黑龙江哈尔滨人，博士，中国科学院信息工程研究所研究员、博士生导师，主要研究方向为网络信息安全、网络态势感知与处理。

谢江（1996− ），男，四川宣汉人，中国科学院信息工程研究所硕士生，主要研究方向为网络信息安全。

方方（1985− ），男，河南郑州人，长安通信科技有限责任公司研究员，主要研究方向为主机及网络信息安全。