BotCatcher：基于深度学习的僵尸网络检测系统

吴迪[1,2]，方滨兴[3,4,5]，崔翔[1,3]，刘奇旭[1,2]

（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. 广州大学网络空间先进技术研究院，广东 广州 510006；4. 电子科技大学广东电子信息工程研究院，广东 东莞 523808；
5. 北京邮电大学网络空间安全学院，北京 100876）

摘 要：机器学习技术在僵尸网络检测领域具有广泛应用，但随着僵尸网络形态和命令控制机制逐渐变化，人工特征选取变得越来越困难。为此，提出基于深度学习的僵尸网络检测系统——BotCatcher，从时间和空间这 2 个维度自动化提取网络流量特征，通过结合多种深层神经网络结构建立分类器。BotCatcher 不依赖于任何有关协议和拓扑的先验知识，不需要人工选取特征。实验结果表明，该模型性能良好，能够对僵尸网络流量进行准确识别。

关键词：僵尸网络；深度学习；检测；特征

中图分类号：TP309.5

文献标识码：A

doi: 10.11959/j.issn.1000−436x.2018135

BotCatcher: botnet detection system based on deep learning

WU Di[1,2], FANG Binxing[3,4,5], CUI Xiang[1,3], LIU Qixu[1,2]

1. Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
2. School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
3. Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China
4. Institute of Electronic and Information Engineering of UESTC in Guangdong, Dongguan 523808, China
5. School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: Machine learning technology has wide application in botnet detection. However, with the changes of the forms and command and control mechanisms of botnets, selecting features manually becomes increasingly difficult. To solve this problem, a botnet detection system called BotCatcher based on deep learning was proposed. It automatically extracted features from time and space dimension, and established classifier through multiple neural network constructions. BotCatcher does not depend on any prior knowledge which about the protocol and the topology, and works without manually selecting features. The experimental results show that the proposed model has good performance in botnet detection and has ability to accurately identify botnet traffic .

Key words: botnet, deep learning, detection, feature

1 引言

僵尸网络（botnet）[1]是指一群可被攻击者远程控制的非合作用户终端。其中，被感染的终端称为僵尸主机（bot），控制者（botmaster）可以通过命令与控制（C&C, command and control）信道对僵尸主机进行一对多的操控。作为一种大规模攻击平台，攻击者可以利用僵尸网络发起分布式拒绝服务

收稿日期：2018−03−13；**修回日期**：2018−07−10

通信作者：崔翔，cuixiang@iie.ac.cn

基金项目：国家重点研发计划基金资助项目（No.2016YFB0801604）；东莞市引进创新科研团队计划基金资助项目（No.201636000100038）；中国科学院网络测评技术重点实验室和网络安全防护技术北京市重点实验室基金资助项目

Foundation Items: The National Key Research and Development Program of China (No.2016YFB0801604), Dongguan Innovative Research Team Program (No.201636000100038), The Key Laboratory of Network Assessment Technology at Chinese Academy of Sciences and Beijing Key Laboratory of Network Security and Protection Technology

（DDoS, distributed denial of service）、垃圾邮件、钓鱼攻击、恶意软件分发、加密勒索、虚拟货币挖掘等大规模攻击活动，对互联网造成了极大的安全威胁。2016 年，Mirai 僵尸网络通过控制物联网智能设备对 OVH、Dyn 等公司发起多次大规模 DDoS 攻击，并引发了美国东海岸断网事件和德国电信用户访问网络异常事件[2]。2017 年，WannaCry 通过 MS17-010 漏洞在全球范围内爆发，影响近百个国家上千家企业及公共组织，该程序感染计算机后会植入敲诈者病毒，导致电脑大量文件被加密[3]。安天和电信云堤发布的《2017 全球僵尸网络 DDoS 攻击威胁态势报告》中指出，2017 年，受到黑客 DDoS 攻击的国家共 130 个，其中，我国被攻击总次数高达 12 200 万次，占全球受攻击总数的 84.79%。

在僵尸网络检测领域，机器学习技术目前得到了广泛应用。尤其在异常检测方面，研究人员利用分类（如朴素贝叶斯[4]、支持向量机[5]、随机森林[6]）或聚类（如 DBSCAN[7]、*X*-means[8]）算法，依据多种特征建立模型，识别恶意网络流量。这些检测模型在论文实验中都具有较低的漏报率和误报率，但是都面临一个相同的问题：依赖人工选取的特征。特征通常在模型建立前由研究者通过经验设定，常见的角度包括网络流属性（如数据分组数量、数据分组平均字节）、时间（如相邻 2 条数据流平均间隔时间）、行为（如是否访问相同服务器）等。合理的特征可以有效地提高模型的性能，但是一方面，人工选取对设计者的先验知识有着较高要求，另一方面，固定的特征也为攻击者提供了可乘之机。攻击者可以利用对抗机器学习思想，针对性地改变僵尸网络流量相关特征，借此逃避模型的检测。文献[9]指出攻击者可以通过向僵尸网络流量中注入特定数据分组和数据流噪声的方法消除空间相似性以及在通信中加入随机时延来消除时间相似性。文献[10]指出攻击者可以通过使僵尸主机随机访问正常域名从而逃避防御人员对 C&C 服务器的聚类。

目前，深度学习技术在图像分类和文本识别领域有着广泛的应用[11-12]，其通过多层神经网络结构与大量参数的调节，可以对样本的特征进行逐层抽象和提取。因此，为了解决特征选取困难和容易被攻击者针对的问题，本文围绕基于深度学习的僵尸网络检测系统展开讨论。本文的主要贡献如下。

1) 提出一种新型检测模型——BotCatcher，利用深度学习技术自动化学习网络流量时间和空间这 2 个维度的特征，将特征提取与模型训练过程结合起来，从全局的角度识别僵尸网络流量。该检测模型不依赖于任何有关协议和拓扑的先验知识，也不需要人工选择特征。

2) 提出 2 种深层神经网络结构用于时空特征的提取。空间维度方面，借鉴卷积神经网络（CNN, convolutional neural network）在图像识别领域的应用方法，将网络流量映射为灰度图像，并利用多层 CNN 从中逐步抽取特征。时间维度方面，先对每条数据流进行处理得到对应的数据分组序列与字节序列，再分别送入长短期记忆（LSTM, long short-term memory）网络中学习特征。

3) 实现 BotCatcher 原型系统，构造合理数据集进行性能评估实验。实验研究表明，BotCatcher 相比已有的深度学习检测模型，具有更高的准确性，能够有效检测大规模复杂的僵尸网络流量。

2　相关工作

僵尸网络检测已有工作总结如表 1 所示。从公开发表的文献看，僵尸网络检测领域目前有如下具有代表性的工作。

基于网络流量的僵尸网络检测技术主要包括误用检测和异常检测等。其中，误用检测基于通信特征码，使用事先配置的特征匹配规则对网络流量进行筛选，相关的入侵检测系统（IDS, intrusion detection system）包括 USTAT[13]、NetSTAT[14]等。文献[15]通过对 Snort（一款著名的开源 IDS）进行自定义规则配置，提出了一种以 IDS 为驱动的基于状态的僵尸网络检测系统 Bothunter。文献[16]通过在可控环境中观察僵尸主机行为，自动化提取特征，对不同协议的僵尸网络分别建立检测模型，并在实际检测过程中自动生成。误用检测技术虽然对已知的僵尸网络的准确率较高，但是对加密流量的识别能力较弱，而且无法检测未知攻击。

异常检测假设僵尸网络中 C&C 服务器与僵尸主机之间的通信模式与正常用户之间的通信模式有显著差异，因此可通过流量分析来对僵尸网络产生的异常流量进行检测，典型的异常特征包括高网络时延、非常规端口流量等。异常检测方法多使用机器学习技术，针对使用模型的不同，主要可以分为 2 种：无监督的聚类模型和有监督的分类模型。

聚类模型方面，文献[8]提出了一种与协议拓扑

表 1 僵尸网络检测已有工作总结

研究方向	研究进展	研究团队	存在问题
基于 IDS 的误用检测技术	提出一种以 IDS 为驱动基于状态的检测系统[15]	德克萨斯 A&M 大学	需要建立特征模板库、难以识别加密流量、无法检测未知攻击
	根据不同协议的僵尸网络分别建立检测模型[16]	维也纳技术大学	
基于聚类的异常检测技术	从主机行为和通信模式这 2 个方面对流量进行聚类[8]	德克萨斯 A&M 大学	需要人工提取特征、聚类关联分析繁杂
	采用时间窗模式对实时聚类结果进行关联[17]	沙希德贝赫什提大学	
	通过客户端重合度等特征聚类挖掘可疑服务器[10]	德克萨斯 A&M 大学	
基于分类的异常检测技术	针对 IRC 僵尸网络设计多种分类器[4]	雷神 BBN 技术中心	需要人工提取特征、多针对指定类型僵尸网络、多使用简单的分类算法
	针对 P2P 僵尸网络设计多种分类器[18]	维多利亚大学	
	采用随机森林模型动态选取特征构造检测系统[6]	赛门铁克研究实验室	
	采用算法过滤多余无关特征并剪枝缩小数据集[19]	哈立法大学	
基于深度学习的检测技术	利用带有反向传播机制的前馈神经网络进行分类[20]	PSG 技术学院	只使用一种深度学习算法、学习得到的特征类型单一
	利用 RNN 提取网络流状态序列特征[21]	门多萨大学	
	利用 CNN 在图形分类中的方法提取网络流特征[22]	中国科学技术大学	

无关的僵尸网络检测模型 Botminer，其基于僵尸网络具有时空相似性（space-time similarity）的假设，从主机行为与通信模式这 2 个层面对具有相似性的网络流量分别聚类，通过对聚类结果进行关联分析得出可疑的僵尸网络流量。文献[17]提出的检测模型与 Botminer 原理相似，在关联分析阶段采用了时间窗模式，并增加了实时检测功能。文献[10]提出了一种系统化的相关服务器挖掘模型，该模型没有对僵尸主机进行水平关联，而是通过流量聚类挖掘具有相似性的可疑服务器，所用的特征包括与服务器通信的客户端集合、服务器 IP 地址、whois 信息等。

分类模型方面，文献[4]针对 IRC 僵尸网络，利用 J48、朴素贝叶斯和贝叶斯网络算法设计了分类器，具有较低的漏报率。文献[18]针对 P2P 僵尸网络，比较了 SVM、KNN 等 5 种分类器在实时检测中的表现。文献[6]提出一种面向 NetFlow 数据的大规模高速检测系统 DISCLOSURE，通过采用随机森林模型动态选取特征，在不同的应用场景中可自适应地平衡漏报率和误报率。文献[19]提出了一种新型的随机化数据分割学习模型，采用改良的正向选择排序技术从特征集中过滤多余无关的特征，并通过基于泰森多边形的数据剪枝方法来减小庞大的训练数据集。

随着人工智能理念以及深度学习技术的发展，神经网络和深度学习算法被逐渐应用于僵尸网络检测领域。文献[20]提出利用带有反向传播机制的多层前馈神经网络建立分类器，并对算法进行改进使其在更新权值时可以动态调整模型的学习速率。文献[21]提出将僵尸网络流量转化成随时间变化的状态序列特征，采用循环神经网络（RNN, recurrent neural network）对这些特征进行学习来建立检测模型。文献[22-23]提出利用 CNN 学习网络流量的特征，利用图形分类的方法来实现流量分类。

不同于上述已有工作，本文结合多种深度学习算法来建立模型，通过多层神经网络逐步抽象，自动化地学习网络流量时间与空间这 2 个维度的特征，从而实现对大规模复杂僵尸网络的准确检测。

3 模型设计

3.1 BotCatcher 概述

BotCatcher 的目标为通过深度学习算法，从网络流量中自动化地提取时间与空间这 2 个维度的特征，并依此训练分类器。在特征学习模块中，空间维度特征提取采用 CNN 算法，主要方法为将数据流转换为二维灰度图像，然后利用 CNN 在图像识别领域应用的方法对流量特征进行学习；时间维度特征提取采用 RNN 算法，本系统具体选择 LSTM 神经网络对网络流中依时序排列的数据分组序列和字节序列进行特征学习。整体框架如图 1 所示。

3.1.1 数据预处理

原始数据集文件为 pcap 格式，由多个数据分组（packet）构成，而 BotCatcher 进行特征学习的对象为数据流（flow），因此在进行特征提取建立模型之前需要对原始数据进行预处理，将其聚合为数据流形式。相关概念定义如下。

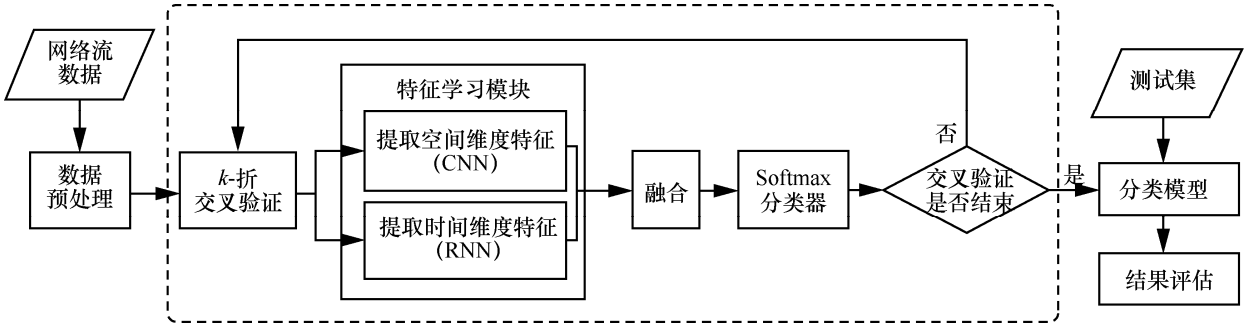


图 1　BotCatcher 整体框架

定义 1 数据分组。数据分组定义为 $p=(x_p, s_p, t_p)$，其中，x_p 表示五元组<*src_ip*, *src_port*, *dst_ip*, *dst_port*, *protocol*>，即源 IP、源端口、目的 IP、目的端口和传输协议，s_p 表示数据分组的大小，t_p 表示数据分组的起始时间。原始数据集可以表示为数据分组的集合 $P=\{p_1, p_2, \cdots, p_n\}$，$n$ 为数据集包含的数据分组数量。

定义 2 数据流。数据流由数据集中五元组相同的数据分组组合而成，且流中的数据分组按照时间顺序排列，即 $\{p_1=(x_1,s_1,t_1), p_2=(x_2,s_2,t_2), \cdots, p_i=(x_i,s_i,t_i)\}$，其中，$x_1=x_2=\cdots=x_i$，$t_1<t_2<\cdots<t_i$。单个数据流定义为 $f=(x_f, s_f, d_f, t_f)$，其中，x_f 表示流中所有数据分组相同的五元组，s_f 表示流中所有数据分组的大小之和，d_f 表示流的持续时间，t_f 表示流中第一个数据分组的起始时间。

因此，网络流量呈层次化结构，如图 2 所示。最底层为按照时间顺序排列的字节序列，这些字节序列根据不同的网络协议聚合形成不同的数据分组，数据分组序列根据五元组是否相同聚合形成数据流。

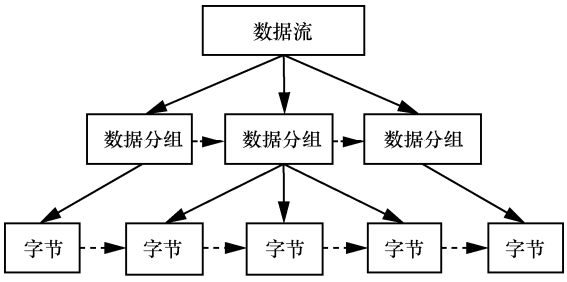


图 2　网络流量层次化结构

3.1.2　交叉验证

为了提高检测模型的泛化能力，同时更准确地评估模型的分类性能，BotCatcher 对数据流采用 *k*-折交叉验证（*k*-fold cross validation）方法构造训练集。*k*-折交叉验证指将样本数据集随机划分为 *k* 个相同大小的子集，在每次模型训练迭代过程中，按顺序选取其中的一个子集作为测试集，剩下的 *k*−1 个子集作为训练集。本文取 *k*=10，即将数据集分为 10 份，每次取其中 9 份进行训练，根据损失函数（用来评估预测值与实际值的差距）进行优化后执行下一次迭代。

3.2　空间特征学习

卷积神经网络是一种基于多层监督学习的人工神经网络，具有局部感知和权值共享等特点，能够自动化地学习目标的多尺度特征，相比传统的模式识别方法，具有更好的自适应性和容错能力，广泛应用于图像分类等领域，因此 BotCatcher 选择采用 CNN 对数据流的空间特征进行学习。

3.2.1　数据规范化

在学习前需要将每个数据流转化为一张二维灰度图像。为了提取出相同维度的特征，CNN 要求输入的图片大小相同，而经过数据预处理得到的训练集中，各个数据流大小不一，且方差可能很大。因此为了便于接下来的特征学习，BotCatcher 对所有的数据流进行截取，取每个数据流前 1 024 B（32×32）的数据（截取长度在第 4 节中进行评估），如果某条数据流长度不够 1 024 B，则在末尾用 0x00 进行填充。通常，一条数据流前面的数据主要包括连接信息（例如 TCP 连接中的三次握手、TLS 连接中的密钥交换）与少部分的内容交换，可以较好地反映整条数据流的主要特征。

3.2.2　图片转化

为了验证 CNN 对流量分类的科学性和可行性，本文从 NETRESEC 网站随机选取了几种恶意流量，同时捕获了几种常见的日常流量，并将其可视化。具体方法为：从某一类的流量中随机抽取若干条数据流，截取每个数据流中的前 1 024 B 的数据，并

将每个字节转化为一个 8 位灰度像素（0x00 表示黑色，0xff 表示白色），最后形成一个 32×32 的灰度图像，如图 3 所示。

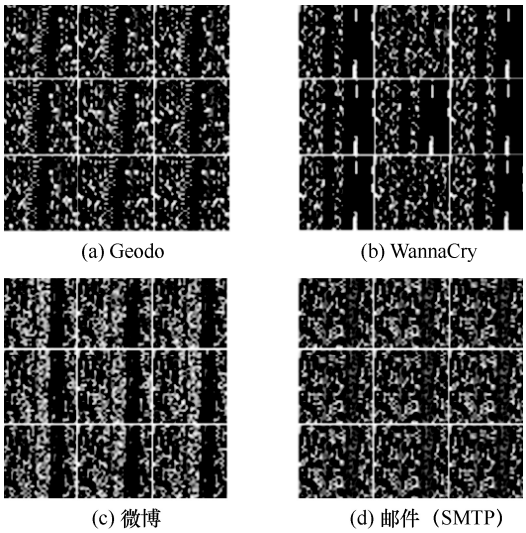


图 3　不同类别流量转化成的灰度图像

从图 3 可以看出，不同应用种类的 pcap 图像之间具有较为明显的差别，而每一类中的数据流则具有极高的相似度。因此可以推测僵尸网络数据流转化生成的图像与正常数据流的图像具有显著差异，即使用 CNN 对图片分类的方法进行流量识别是有效的。

3.2.3　CNN 结构设计

本文流量预处理得到的图片输入尺寸以及训练模型所使用的数据量与经典的 LeNet-5 结构十分相似。LeNet-5 共使用 7 层卷积神经网络结构，包括卷积层、池化层、卷积层、池化层、卷积层、全连接层和输出层，它避免了对图像进行复杂的预处理过程，在模式分类领域获得了广泛应用，尤其对于手写数字识别具有非常高的准确率。因此 BotCatcher 借鉴 LeNet-5 结构，同时由于数据流灰度图像相比于简单的手写数字具有更多复杂的细节，因此在使用 CNN 时加入了更多的过滤器来全面学习样本特征。BotCatcher 共采用 2 个卷积层，并在每个卷积层后的池化层进行最大池化（max-pooling）操作。

CNN 具体结构如图 4 所示，每层功能介绍如下。

1) 卷积层 C_1：卷积操作共使用 32 个过滤器，卷积核大小为 5×5，卷积步长为 1。该层由 32 个大小为 28×28 的特征图组成。

2) 池化层 S_1：C_1 层的每个特征图在该层进行一次大小为 2×2 的最大池化操作，即对每个特征图进行步长为 2 的 2×2 过滤，取 4 个输入中的最大值。该层由 32 个大小为 14×14 的特征图组成。

3) 卷积层 C_2：卷积操作共使用 64 个过滤器，卷积核大小为 5×5，卷积步长为 1。该层由 64 个大小为 10×10 的特征图组成，其中，每个特征图连接 S_1 层的所有 32 个或几个特征图。

4) 池化层 S_2：与 S_1 层进行的池化操作相同，该层由 64 个大小为 5×5 的特征图组成，特征图中的每个单元与 C_2 层中相对应特征图的 2×2 邻域相连接。

5) 全连接层 D_1：由 1 024 个神经元构成，与 S_2 层全相连，输出为 1 024 维向量。

6) 全连接层 D_2：由 10 个神经元构成，与 D_1

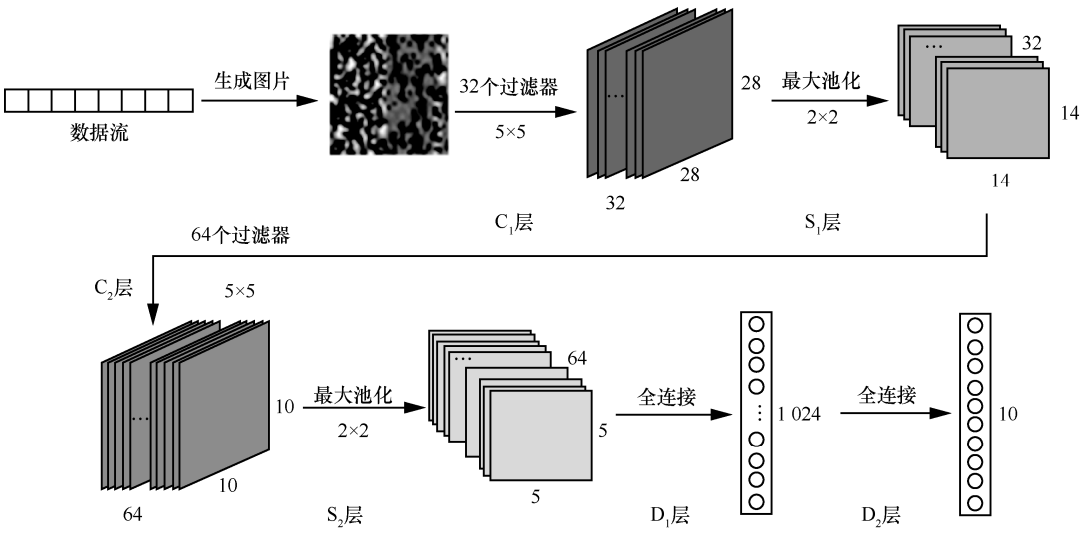


图 4　CNN 结构设计

层全相连，输出为 10 维向量。

3.3　时间特征学习

CNN 只能对数据流的空间特征进行学习，无法提取链式结构的输入集中各单元之间的依赖关系。为了更深层地挖掘数据流在时间序列上的特征，BotCatcher 采用 RNN 中的 LSTM 神经网络进行时间特征学习。相比于传统的前向反馈神经网络，RNN 引入了定向循环，能够处理输入之间前后关联的问题，目前，RNN 在自然语言处理（NLP, natural language processing）领域已经取得了巨大成功。如图 2 所示，网络流量中的每个数据流都由按照时间顺序排列的数据分组序列构成，因此可以使用 RNN 对其进行序列挖掘。LSTM 是一种特殊的 RNN，主要解决了序列数据中“长期依赖”的问题。通常，网络流量由于其协议的特殊性，数据流中的一些数据分组可能与在其之前的 N 个数据分组存在依赖关系（例如 TCP 握手阶段的超时重连），因此相比于简单循环神经网络，LSTM 更符合流量特征挖掘的场景需求。

为了更准确地提取特征，BotCatcher 采用双向 LSTM，即对每个数据流进行正向、反向 2 个方向的序列扫描。同时采用双层 LSTM 架构，先以数据分组中的字节序列作为 LSTM 的输入，将每个数据分组转化为一个向量，再将得到的数据分组向量序列作为 LSTM 的输入，生成最终的时间维度特征。

3.3.1　输入规范化

与 3.2 节相同，为了进行模型训练，需要对所有输入的数据流进行结构规范化，统一格式。BotCatcher 对每个数据流截取前 8 个数据分组，每个数据分组取前 100 个字节（截取长度在第 4 节进行评估），若长度不够，则在末尾用 0x00 填充。

3.3.2　LSTM 结构设计

LSTM 具体结构如图 5 所示，各层功能介绍如下。

1) 独热编码层 One-Hot：首先将数据流分为 8 个数据分组，每个数据分组为 100 维向量 $\boldsymbol{a}=(a_1,a_2,\cdots,a_i)$, $i=1,2,\cdots,n$，$a_i\in N^*$ 且 $0\leqslant a_i\leqslant 255$。这样，数字化的向量在进行训练时，模型会将字节这种离散值误认为连续值，从而影响权重学习，降低准确率。因此在进行第一层 LSTM 学习之前，需要对数据分组向量进行编码。BotCatcher 采用 One-Hot 编码，将每个字节编码为 256 维的向量，其中只有一位为 1，其他位均为 0，数据分组即变为 100×256 的稀疏矩阵。

2) LSTM 层 L_1：由 100 个 LSTM 单元构成，输入为 One-Hot 编码后的字节序列，输出为 100 个 256 维向量。

3) 全连接层 D_1：由 256 个神经元构成，输出

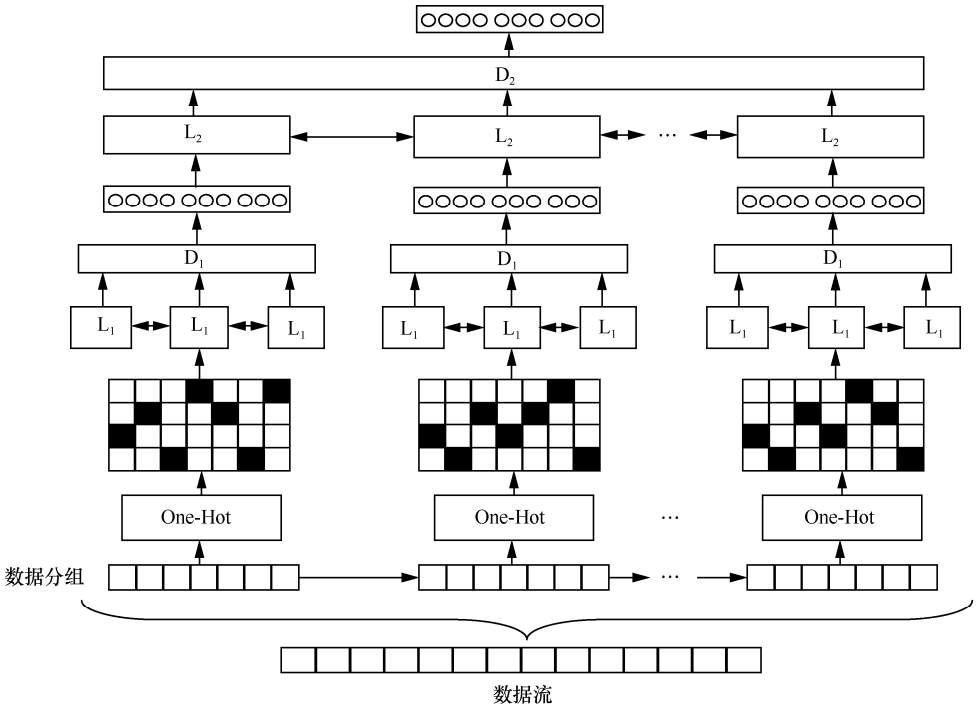


图 5　LSTM 结构设计

为 256 维向量。

4) LSTM 层 L_2：由 8 个 LSTM 单元构成，输入为 D_1 层生成的数据分组序列，输出为 8 个 256 维向量。

5) 全连接层 D_2：由 10 个神经元构成，输出为 10 维向量。

3.4 分类学习

在进行分类之前，需要对数据流时间和空间这 2 个维度的特征进行聚合。聚合过程有累加、累乘、取最大值等多种方式，为了尽可能地保留网络流量的时空特性，BotCatcher 采用串联的方式进行聚合，即将 2 个 10 维的特征向量相接，构成一个 20 维的特征向量。

1) 分类器选择：基于数据流特征，使用 Softmax 分类器来判断输入的数据流为正常流量还是僵尸网络流量，如式(1)所示。其中，V_i 为向量的第 i 个元素，该元素的 Softmax 值 S_i 为

$$S_i = \frac{\mathrm{e}^{V_i}}{\sum_j \mathrm{e}^{V_j}} \tag{1}$$

Softmax 会将多个神经元的输出映射到(0,1)内，各个输出之和为 1，符合概率形式，简单易用。本文的 Softmax 作为二分类器，输出结果为 2 类。

2) 损失函数：模型训练时，在分类器后需要根据损失函数计算损失，继而进行反向传播来进行参数调整，即 BP（back propagation）过程。BotCatcher 选用多分类交叉熵损失（categorical cross-entropy loss）函数作为模型的损失函数。此外，为了防止训练结果过拟合，使权重的分配更加均匀，在损失函数中加入了正则项，如式(2)所示。

$$J(\boldsymbol{W},b) = -\sum_i y_i \log(f_{\boldsymbol{W},b}(x_i)) + \frac{1}{2}\lambda \sum_i \boldsymbol{W}_i^2 \tag{2}$$

其中，$\boldsymbol{W}$ 为权值矩阵，b 为偏移量，λ 为权重衰减系数，y 为真实分类结果，$f_{\boldsymbol{W},b}(x)$ 为分类器输出结果。

3) 优化算法：BotCatcher 采取的优化算法为基于 mini-batch 的随机梯度下降法（SGD, stochastic gradient descent），并加入动量参数 ρ 来提高稳定性，加快学习速度。参数更新过程如式(3)所示。

$$\begin{aligned} \Delta\theta_i &= \rho\theta_{i-1} - \alpha\nabla_{\theta_i} J(\theta) \\ \theta_{i+1} &= \theta_i + \Delta\theta_i \end{aligned} \tag{3}$$

其中，α 为学习率，$J(\theta)$ 为损失函数。

模型训练过程如算法 1 所示。

算法 1 模型训练过程

输入 训练集$\{(x_1,y_1),\cdots,(x_i,y_i)\}$，神经网络层数 N，网络矩阵 $\boldsymbol{W}$，偏移量 b，学习率 α，动量 ρ，准确率变化量阈值 a，batch_size 大小为 m。其中，$J(\boldsymbol{W},b)$为带有正则项的损失函数，$z^{(l)}$为第 l 层的输出，$\boldsymbol{W}_f$、b_f分别为上一轮迭代时的权重矩阵和偏移量

输出 更新后的网络权重矩阵 $\boldsymbol{W}'$，偏移量 b'

1) while $\Delta accuracy > a$
2) 从训练集中随机选取 m 对样本
3) $\delta^{(N)} \leftarrow \frac{1}{m} J(\boldsymbol{W},b)$
4) for $l \leftarrow N-1$ to 1 do
5) $\delta^{(l)} \leftarrow (\boldsymbol{W}^{(l)})^{\mathrm{T}} \delta^{(l+1)} f'(z^{(l)})$
6) $\boldsymbol{W}^{(l)} = \boldsymbol{W}^{(l)} - (\rho \boldsymbol{W}_f^{(l)} - \alpha \nabla_{\boldsymbol{W}^{(l)}} \delta^{(l)})$
7) $b^{(l)} = b^{(l)} - (\rho b_f^{(l)} - \alpha \nabla_{b^{(l)}} \delta^{(l)})$
8) end for
9) 使用验证集进行验证
10) end while

4 效果评估

4.1 数据集

本文实验采用的数据集由僵尸网络流量、正常网络流量和背景流量 3 个部分组成。其中，僵尸网络流量取自 CTU 大学（布拉格捷克理工大学）组织建立的 Stratosphere IPS 项目，该项目的一个姊妹项目 Malware Capture Facility 专门负责收集和捕获各种类型的恶意流量和正常流量，著名的 CTU-13 数据集[24]就由该项目建立。本文从中选择了若干个具有代表性的僵尸网络作为实验数据集，并对其中过大（超过 1 GB）的 pcap 分组进行了一定裁剪，如表 2 所示。正常网络流量取自 ISOT 2010 数据集[25]，该数据集混合了 French chapter of Honeynet[26]、Ericsson Research in Hungray[27]等多个项目的公开数据集，包括 Storm 和 Zeus 这 2 种 P2P 僵尸网络以及 HTTP、P2P 应用（例如 bittorrent）、游戏等多种非恶意流量，本文选取其中未感染主机的流量作为本文实验的正常流量。

此外，为了使实验数据更接近用户日常上网环境，对本地 10 台日常使用的电脑进行了流量采集，生成流量作为本文实验的背景流量。

表 2　　数据集

僵尸网络	CTU 序号	处理
Neris	42	原始（56 MB）
Rbot	44	原始（123 MB）
Virut	46	原始（30 MB）
Zeus	78-2	裁剪（309 MB）
Conficker	90	原始（13 MB）
Geodo	125-1	裁剪（327 MB）
Miuref	127-1	原始（16 MB）
Bunitu	141-1	原始（323 MB）

4.2　评估指标

为了对检测模型的表现进行评估，本文共选取 3 个实验指标，分别为准确率（*ACC*）、误报率（*FPR*）和 F_β 值，如式(4)所示。

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

$$FPR = \frac{FP}{FP + TN}$$

$$F_\beta = \frac{(1+\beta^2)TP^2}{\beta^2 TP(TP+FN) + TP(TP+FP)} \tag{4}$$

其中，F_β 值为查准率和查全率的加权调和平均值，由于在流量检测中查准率比查全率更加重要，因此 β 应介于 0~1 之间，本文将 β 定为 0.5。*TP* 是将正类预测为正类的数量，*TN* 是将负类预测为负类的数量，*FP* 是将负类预测为正类的数量，*FN* 为将正类预测为负类的数量。

4.3　参数选择

第 3 节中，在对流量进行特征学习时，需要对数据集进行预处理，通过截取相同大小的长度将流量数据处理成深度学习框架需要的输入形式，相关参数包括以下 3 个部分。

1) *bpf*：空间特征学习模块中，每条数据流的字节数，即输入图片的大小。

2) *ppf*：时间特征学习模块中，每条数据流的数据分组数量。

3)*bpp*：时间特征学习模块中，每个数据分组的字节数。

合理的模型输入规格可以更好地保留流量数据自身特性，同时有助于构造出学习能力更强的神经网络结构，加快模型学习的效率。因此，为了方便选择合适的参数，本文取 Conficker、Neris、Zeus 这 3 个较为典型的僵尸网络数据集，对它们的流量特征进行了 3 个方面的统计，即每条数据流的字节数、每条数据流的数据分组数和每个数据分组的字节数，统计数据分组包括总数、最大值、最小值、平均值和众数，如表 3 所示。

表 3　　流量参数统计数据

统计数据	Conficker	Neris	Zeus
总数据流	18 855	9 480	23 146
总数据分组	154 838	195 971	488 858
总字节	9.36×10^6	3.48×10^7	3.17×10^8
每条数据流字节数最大值	19 942	864 526	81 095
每条数据流字节数最小值	232	60	306
每条数据流字节数平均值	496	3 675	13 678
每条数据流字节数众数	480	372	3 496
每条数据流数据分组数最大值	268	6 662	92
每条数据流数据分组数最小值	4	1	5
每条数据流数据分组数平均值	8	20	21
每条数据流数据分组数众数	8	6	10
每个数据分组字节数最大值	1 466	10 274	1 474
每个数据分组字节数最小值	54	60	54
每个数据分组字节数平均值	60	177	647
每个数据分组字节数众数	62	60	54

综合表 3 提供的信息，本文将 2 个特征学习模块作为单独的模型分别进行了模拟实验（即只用空间特征训练或只用时间特征训练），并根据结果对模型参数进行了如下选择。

1) 对于空间特征学习模块，本文对 *bpf* 分别取 25×25、26×26、…、37×37，构造 13 组训练集对模型进行训练，训练结果如图 6 所示。从图 6 可以看出，

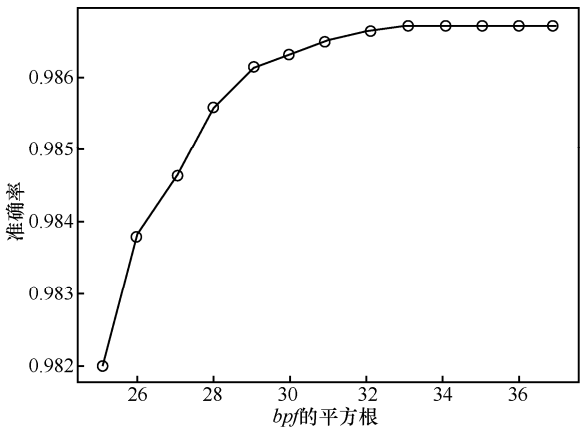


图 6　*bpf* 参数选择

当 *bpf* 大小超过 1 089（33×33）时，模型准确率不再有显著上升。结合 CNN 模型的学习经验，本文将 *bpf* 确定为与 LeNet-5 的输入集大小相同，即 1 024（32×32）。

2) 对于时间特征学习模块，虽然在一定范围内取更大的 *ppf* 和 *bpp* 可以更好地反映流量内部的时间关联，但是由于该模块输入的训练集格式为四维矩阵，随着 2 个参数取值的增加，得到的训练集会成倍增大，并最终对模型训练速率造成很大的影响，因此该模块的参数选择在保证一定准确率的前提下，训练时间成本与空间占用成本成为主要的考虑因素。此外，从表 3 可以看出，数据流所含的数据分组多数为 6~10 个，数据分组中所含的字节数多数都不超过 100 B，由此可以判断较小的 *ppf* 和 *bpp* 即可反映大部分流量的时间特性。本文对 *ppf* 分别取 6、8、10、12，对 *bpp* 分别取 80、100、120，构造 12 组训练集对模型进行训练，在训练集大小为 1 000 个样本的情况下，10 轮 epoch 所需的训练时间（包括读取训练数据的时间）与准确率如图 7 所示。从图 7 可以看出，当 *ppf* 和 *bpp* 分别为 8 和 100 时，模型有较高的准确度，且训练时间成本较小，随着参数取值变大，准确率不再有显著增长，但是时间成本大幅度提高。因此本文将 *ppf* 确定为 8，*bpp* 确定为 100。

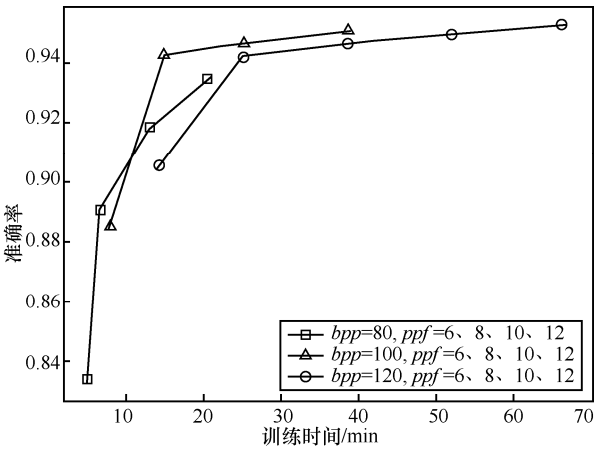


图 7　*ppf* 和 *bpp* 参数选择

4.4　评估结果

4.4.1　实验环境

数据预处理阶段，采用 pkt2flow 工具将原始 pcap 分组转化为数据流。模型搭建阶段，采用 keras 作为神经网络框架，计算机配置为 8 核 16 GB 内存，搭载 64 位的 Ubuntu 16.04，显卡为 AMD R7 350。参数方面，mini-batch 大小为 128，训练时间为 30 轮 epoch。此外，为了提升训练速度、防止过拟合，模型采用 dropout 方法，丢弃率为 0.25。

4.4.2　实验结果

正如本文第 2 节所述，僵尸网络检测方面已经有利用 CNN、RNN 等深度学习技术构建检测模型的例子。本文为了更全面、更精确地刻画网络流量特征，没有采用单一的方法，而是从数据流和数据分组等多种数据结构出发，结合 CNN 和 LSTM 技术提取流量中的时间和空间这 2 个维度的特征。因此，为了对 BotCatcher 的性能进行对比评估，本文利用数据集对以下 3 种模型分别进行了多轮训练，并观察检测结果指标。

1) 模型 A：只使用 CNN 提取空间特征。

2) 模型 B：只使用 LSTM 提取时间特征。

3) 模型 C：同时使用 CNN 和 LSTM 提取时空特征，即 BotCathcer 模型。

图 8 和图 9 显示了 3 种模型在不同训练时间下测试得到的 F_β 值和 *FPR*，共进行 10 次实验，取 10 次结果的平均值作为最后结果。

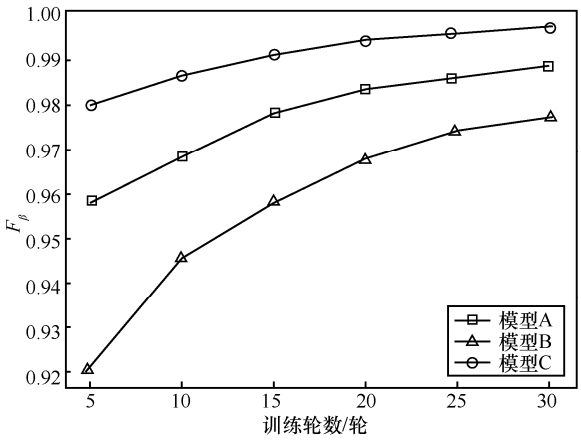


图 8　检测结果 F_β 值

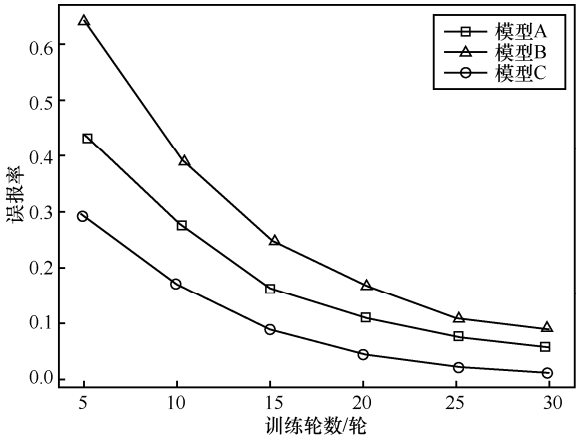


图 9　检测结果 *FPR*

从图 8 可以看出，在只使用一种特征的情况下，模型 B 的性能相对较差，30 轮训练后的 F_β 值为 0.977 8，模型 A 的性能相对较好，30 轮训练后的 F_β 值为 0.988 4，这说明本文所用到的空间特征比时间特征可以更好地反映网络流量特性。

BotCatcher 使用时间和空间这 2 种特征，从图 9 可以看出，在同样训练时间下，模型 C 的性能比模型 A 和模型 B 均有明显提升，30 轮训练后 F_β 值达到 0.997 6，*FPR* 仅为 0.012。综上，在建立僵尸网络检测模型时，利用 CNN 和 LSTM 提取出来的多维特征相比单一特征能够取得更高的准确率和更低的误报率，得到的模型准确率可以满足实际使用需求。

5　讨论

BotCatcher 由 2 种深度神经网络构成，模型结构相比传统机器学习模型要复杂得多，主要体现在以下 2 个方面。

1) CNN 和 LSTM 这 2 种深层模型并行工作，模型层数较多，共有近 300 万个参数可以调节。

2)本文所提模型需要的 2 种输入格式分别为三维和四维，多维输入导致训练集数据庞大。

以上 2 个原因导致 BotCatcher 虽然检测准确性较好，但是实际运行速度较为缓慢，且对计算环境的配置要求较高。BotCatcher 运行的时间主要体现在神经网络特征的提取过程，尤其是输入高维数据的 RNN。由 4.3 节的结果可知，训练 1 000 个样本所需时间大约在 10~20 min，相比朴素贝叶斯、SVM 等人工选取特征的常见机器学习算法具有较高的时间成本。但是计算机硬件对于深度学习的效率影响非常大，高性能的设备可以显著提高深度学习的处理速度。本文的实验过程采用 CPU 进行模型准确性的评估，如果使用具有大容量显存和高吞吐量的 GPU 将会大幅度降低时间成本。总之，考虑到效率因素，BotCatcher 可能暂时不适合实时检测场景。接下来的工作中需要对其进行优化，在保证性能的同时提高检测效率。

此外，本文特征提取部分所用到的结构还可以选择其他深度神经网络进行进一步尝试，例如，可以使用 LSTM 网络的变体门控循环单元（GRU, gated recurrent unit）对时间特征进行学习，通过对比检测效果进一步优化模型结构。

6　结束语

由于僵尸网络形态和机理的发展，基于机器学习的僵尸网络检测系统开始面临人工提取特征困难的问题。本文提出一种基于深度学习的检测模型并实现了其原型系统 BotCatcher。该模型使用 CNN 和 RNN 这 2 种深层神经网络架构，对原始流量自动化地提取时间与空间这 2 个维度的特征。其中，提取空间特征时，先将每条数据流转化为一张灰度图像，然后利用 CNN 在图像识别领域应用的方法从中学习特征；提取时间特征时，分别将每个数据分组中的字节序列以及每条数据流中的数据分组序列作为输入建立双层双向 LSTM 神经网络，并从中学习特征。

BotCatcher 通过结合时空特征对网络流量进行全面刻画，将特征提取与模型训练过程串联起来。系统不依赖于任何有关协议和拓扑的先验知识，也不需要人工参与特征选择。实验证明，该检测系统性能良好，相比只使用单一神经网络的检测模型具有更高的准确率与更低的误报率，能够满足实际使用需求。

参考文献：

[1] CUI X, FANG B, SHI J, et al. Botnet triple-channel model: towards resilient and efficient bidirectional communication botnets[C]// International Conference on Security and Privacy in Communication Systems. 2013: 53-68.

[2] KOLIAS C, KAMBOURAKIS G, STAVROU A, et al. DDoS in the IoT: mirai and other botnets[J]. Computer, 2017, 50(7): 80-84.

[3] EHRENFELD J M. Wannacry, cybersecurity and health information technology: a time to act[J]. Journal of Medical Systems, 2017, 41(7): 104.

[4] LIVADAS C, WALSH R, LAPSLEY D, et al. Usilng machine learning technliques to identify botnet traffic[C]//31st IEEE Conference on Local Computer Networks. 2006: 967-974.

[5] KONDO S, SATO N. Botnet traffic detection techniques by C&C session classification using SVM[C]//International Workshop on Security. 2007: 91-104.

[6] BILGE L, BALZAROTTI D, ROBERTSON W, et al. Disclosure: detecting botnet command and control servers through large-scale netflow analysis[C]//The 28th Annual Computer Security Applications Conference. 2012: 129-138.

[7] FRANÇOIS J, WANG S, ENGEL T. BotTrack: tracking botnets using NetFlow and PageRank[C]//International Conference on Research in Networking. 2011: 1-14.

[8] GU G, PERDISCI R, ZHANG J, et al. BotMiner: clustering analysis of network traffic for protocol-and structure-independent botnet detection[C]//USENIX Security Symposium. 2008: 139-154.

[9] CUI X, FANG B X, YIN L H, et al. Andbot: towards advanced mobile botnets[C]//The 4th Usenix Workshop on Large-scale Exploits and Emergent Threats. 2011: 11.

[10] ZHANG J, SAHA S, GU G, et al. Systematic mining of associated server herds for malware campaign discovery[C]//2015 IEEE 35th In-

ternational Conference on Distributed Computing Systems (ICDCS). 2015: 630-641.

[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(9): 154-157.
CUI P F, QIU Y, SUN R. Research on image recognition technology for the network content security[J]. Netinfo Security, 2015(9): 154-157.

[12] GUL K S Q, 尹继泽, 潘丽敏, 等. 基于深度神经网络的命名实体识别方法研究[J]. 信息网络安全, 2017(10): 29-35.
GUL K S Q, YIN J Z, PAN L M, et.al. Research on the algorithm of named entity recognition based on deep neural network[J]. Netinfo Security, 2017(10): 29-35.

[13] ILGUN K. USTAT: a real-time intrusion detection system for UNIX[C]// 1993 IEEE Computer Society Symposium on Research in Security and Privacy. 1993: 16-28.

[14] VIGNA G, KEMMERER R A. NetSTAT: a network-based intrusion detection approach[C]//14th Annual Computer Security Applications Conference. 1998: 25-34.

[15] GU G, PORRAS P A, YEGNESWARAN V, et al. BotHunter: detecting malware infection through IDS-driven dialog correlation[C]//USENIX Security Symposium. 2007: 1-16.

[16] WURZINGER P, BILGE L, HOLZ T, et al. Automatically generating models for botnet detection[C]//European Symposium on Research in Computer Security. 2009: 232-249.

[17] ARSHAD S, ABBASPOUR M, KHARRAZI M, et al. An anomaly-based botnet detection approach for identifying stealthy botnets[C]//2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE). 2011: 564-569.

[18] SAAD S, TRAORE I, GHORBANI A, et al. Detecting P2P botnets through network behavior analysis and machine learning[C]//2011 Ninth Annual International Conference on Privacy, Security and Trust (PST). 2011: 174-180.

[19] AL-JARRAH O Y, ALHUSSEIN O, YOO P D, et al. Data randomization and cluster-based partitioning for botnet intrusion detection[J]. IEEE Transactions on Cybernetics, 2016, 46(8): 1796-1806.

[20] VENKATESH G K, NADARAJAN R A. HTTP botnet detection using adaptive learning rate multilayer feed-forward neural network[C]//WISTP. 2012: 38-48.

[21] TORRES P, CATANIA C, GARCIA S, et al. An analysis of recurrent neural networks for botnet detection behavior[C]//2016 IEEE Biennial Congress of Argentina (ARGENCON). 2016: 1-6.

[22] WANG W, ZHU M, ZENG X, et al. Malware traffic classification using convolutional neural network for representation learning[C]//2017 International Conference on Information Networking (ICOIN). 2017: 712-717.

[23] 王勇, 周惠怡, 俸皓, 等. 基于深度卷积神经网络的网络流量分类方法[J]. 通信学报, 2018, 39(1): 14-23.
WANG Y, ZHOU H Y, FENG H, et al. Network traffic classification method basing on CNN[J]. Journal on Communications, 2018, 39(1): 14-23.

[24] HADDADI F, PHAN D T, ZINCIR-HEYWOOD A N. How to choose from different botnet detection systems?[C]//Network Operations and Management Symposium (NOMS). 2016: 1079-1084.

[25] ZHAO D, TRAORE I, SAYED B, et al. Botnet detection based on traffic behavior analysis and flow intervals[J]. Computers & Security, 2013, 39: 2-16.

[26] WATSON D, RIDEN J. The honeynet project: data collection tools, infrastructure, archives and analysis[C]//WOMBAT Workshop on Information Security Threats Data Collection and Sharing. 2008: 24-30.

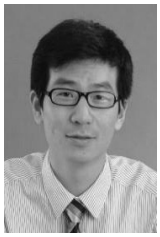
[27] SZABÓ G, ORINCSAY D, MALOMSOKY S, et al. On the validation of traffic classification algorithms[C]//International Conference on Passive and Active Network Measurement. 2008: 72-81.

[作者简介]

吴迪（1991−），男，辽宁抚顺人，中国科学院大学博士生，主要研究方向为网络攻防技术。

方滨兴（1960−），男，江西万年人，中国工程院院士，北京邮电大学教授、博士生导师，主要研究方向为计算机体系结构、计算机网络与信息安全。

崔翔（1978−），男，黑龙江讷河人，博士，广州大学研究员，主要研究方向为网络攻防技术。



刘奇旭（1984−），男，江苏徐州人，博士，中国科学院副研究员、中国科学院大学副教授，主要研究方向为网络攻防技术、网络安全评测。