doi:10.11959/j.issn.1000-436x.2016228

深度学习在僵尸云检测中的应用研究

寇广[1,2]，汤光明[1]，王硕[1]，宋海涛[1]，边媛[1]
(1.解放军信息工程大学，河南 郑州 450001；2. 信息保障技术重点实验室，北京 100072)

摘 要：僵尸云和正常云服务 2 种环境下的基本网络流特征差异不明显，导致传统的基于网络流特征分析法在检测僵尸云问题上失效。为此，研究利用深度学习技术解决僵尸云检测问题。首先，从网络流中提取基本特征；然后将其映射为灰度图像；最后利用卷积神经网络算法进行特征学习，提取出更加抽象的特征，用以表达网络流数据中隐藏的模式及结构关系，进而用于检测僵尸云。实验结果表明，该方法不仅能够提高检测的准确度，而且能减少检测所用时间。

关键词：僵尸云；云安全；深度学习；网络流；特征；卷积神经网络
中图分类号：TP309.5 **文献标识码**：A

Using deep learning for detecting BotCloud

KOU Guang[1,2], TANG Guang-ming[1], WANG Shuo[1], SONG Hai-tao[1], BIAN Yuan[1]
(1. PLA Information Engineering University, Zhengzhou 450001, China;
2. Science and Technology on Information Assurance Laboratory, Beijing 100072,China)

Abstract: The differences of the basic network flow characteristics between BotCloud and normal cloud services were not obvious, and this led to the inefficiency of the method in BotCloud detection based on network flow characteristics analysis. To solve this problem, a CNN(convolution neural network)-based method for detecting the BotCloud was proposed. First, it extracted the basic network flow characteristics from network flow data packets. Second, it mapped the basic network flow characteristics into gray image. Finally, in order to detect BotCloud, it utilized CNN algorithm to learn and extract characteristics that were more abstract to express the hidden model and structural relationship in the network data flow. The experimental results show that the proposed method can not only enhance the accuracy of detection, but also greatly reduce the time required for detecting.

Key words: BotCloud, cloud security, deep learning, network flow, characteristic, CNN

1 引言

云计算在改变 IT 世界的同时，也在催发新的安全威胁。云计算的多租户特性，使它可向任何人提供计算资源，而不管那些人的使用目的是好是坏。一方面，租户可以利用云服务进行高效计算；另一方面，租户同样可以利用云服务进行网络攻击、发送垃圾邮件和网络欺诈等恶意活动。

作为当今网络空间的最大威胁，僵尸网络[1]是指采用一种或多种传播手段，将大量主机感染 bot 程序，从而在控制者和被感染主机之间形成一个可一对多控制的网络。为了克服构建僵尸网络过程中要花费大量时间的问题，一些僵尸网络控制者正在伪装成云服务的合法租户，利用云服务提供商的虚拟机快速地构建僵尸网络并用于发动攻击，本文称这种用于构建僵尸网络的云服务为僵尸云。现实中，恶意攻击者利用僵尸云发动网络攻击的事件时有发生。2009 年，亚马逊的 EC2 云服务被用来发动 DDoS 攻击，就是广为人知的 Zeus botnet。2011 年 4 月，恶意攻击者利用亚马逊弹性云计算服务对

收稿日期：2015-10-19；**修回日期**：2016-10-28
基金项目：国家自然科学基金资助项目（No.61303074）；信息保障技术重点实验室开放基金资助项目（No.KJ-14-106）
Foundation Items: The National Natural Science Foundation of China (No.61303074), Foundation of Science and Technology on Information Assurance Laboratory (No.KJ-14-106)

索尼 PlayStation Network 和 Online Entertainment 服务发动攻击，造成了其重大损失。僵尸云是被租户滥用的云服务，它不仅会破坏云服务提供商良好的信誉，而且也极大地威胁着网络安全。2013 年，云安全联盟(CSA)发布名为“The Notorious Nine Cloud Computing Top Threats in 2013”的研究报告，将云服务滥用作为云计算安全的九大威胁之一。与传统僵尸网络相比，僵尸云威胁性更大，主要体现在 4 个方面：1) 僵尸云的构建是用户的合法行为，具有很强的隐蔽性，且僵尸云的构建极其迅速；2) 云计算扩展了其攻击能力与攻击范围；3) 能够快速向移动终端渗透；4) 攻击者从信誉良好的云服务上发动攻击，目标会变得更加难以定位。因此，检测僵尸云的存在、分析其行为和实施合适的防护手段是云计算安全要研究的重要问题。

目前，云计算安全的研究热点是保证租户免受来自外部的攻击，而对恶意租户滥用云服务的行为缺乏关注，特别是针对僵尸云检测问题的研究更是少见。针对这一问题，本文以如何高效检测僵尸云为问题牵引，通过分析进出云平台的网络流数据分组，提取合适的基本网络流特征，并将基本网络流特征映射为灰度图像，最终利用深度学习核心算法之一的卷积神经网络算法对特征进行深度学习，进而检测出僵尸云。

2　相关工作

随着云计算的发展，僵尸云以其构建快速、隐蔽性好、攻击力强等优点正成为网络攻击者的重要攻击手段，严重威胁着云计算安全。僵尸云检测的目的是尽早发现恶意租户利用云服务构建的僵尸网络，从而使云服务提供商能够通过关闭云服务等措施遏制恶意租户的行为，进而避免恶意租户发动网络攻击。

目前，针对传统僵尸网络的检测主要分为 3 类方法。1) 基于蜜网的检测方法。Artail 等[2]通过人工部署动态蜜网用于捕获攻击行为，但是自动化程序不高；北京大学诸葛建伟[3]带领的狩猎女神蜜网项目也是其中的典型代表。2) 基于主机行为的检测方法。此方法在主机端部署监测程序，检测主机是否感染了僵尸程序。文献[4~6]对此方法进行了深入的研究，比较有代表的是文献[4]，通过分析主机日志相关性检测僵尸程序，文献[6]则是通过分析程序执行数据来检测僵尸病毒。3) 基于网络流的检测方法。基于网络流的检测方法主要分为 2 类，一类是基于分组分析的方法，另一类是基于流特征分析的方法。基于分组分析的方法主要利用分析网络数据分组的恶意特征码来识别僵尸网络通信行为。文献[7~9]对此方法进行了深入的研究，此方法检测准确率高，但要求解析数据分组，一方面会造成计算量的增大，另一面会影响用户的隐私，并且对加密通信的僵尸网络失效，因此，并不适合僵尸云的检测。基于流特征分析的方法主要利用网络流的时空相关性，选取网络流的统计特征并构造特征向量，然后利用机器学习的方法进行有监督的学习，从而识别出异常网络流。文献[10]开启了这种方法的先河，它利用选取的流特征通过 C4.5 决策树分类、朴素贝叶斯分类和贝叶斯网络分类 3 种机器学习方法检测 IRC(internet relay chat)僵尸网络，其检测方法主要分为 2 步。第一步，将 IRC 聊天和非聊天流量分开，即识别出 IRC 聊天流；第二步，将僵尸网络中 IRC 聊天流量和真实流量分开，即识别恶意 IRC 聊天流量，但是检测的准确率并不高。文献[11]则利用 IRC 僵尸网络行为模式和流量特征的先验知识，结合聚类算法和图理论对 IRC 僵尸网络进行了检测，此方法适合在线检测，但容易被流干扰技术规避。其他基于网络流分析的僵尸网络检测方法见文献[12~14]。这种方法不要求解析数据分组，具有实时性好、准确率高的优点，逐渐成为僵尸网络检测的主流方法。基于网络流特征分析的方法正在引起众多学者的兴趣。此方法的核心是网络流统计特征的选取和机器学习算法的选择。文献[15]则从网络流特征选取的角度，提出了用于选取合适网络流统计特征的算法，很有借鉴意义。文献[16]应用了决策树算法对固定时间窗中网络流特征进行分类，进而检测僵尸网络，检测率达到了 99.5%，误报率只有 0.5%。文献[17]则利用改进的最大最小距离和 *k*-means 聚类算法，改善了数据聚类的效果，对检测 IRC 僵尸网络效果很好。文献[18]利用 SVM 实现了对僵尸主机的识别。

借鉴针对传统僵尸网络检测的方法，一些学者尝试对僵尸云检测问题进行了研究。文献[19]通过主成分分析方法识别云环境中虚拟机的异常行为，用来检测恶意租户用僵尸云发动 DDoS 攻击的行为。该方法本质上是基于主机端的检测方法，具有一定的局限性。文献[20]则认为针对云平台出口的网络流检测对检测僵尸云更加有效，并依据僵尸云

的工作特点提出了几个有效的网络流特征，实用性较强。但此方法忽略进入云平台的网络流，具有一定的主观性。文献[21]提出了一种针对僵尸云源端的协作式 DDoS 检测方法，但此方法主要从架构层次分析，并没有深入研究僵尸云的实际特征。文献[22]介绍了一种基于云的安卓终端僵尸网络检测系统，分析了云计算和移动互联网络发展过程中僵尸网络的变种特性，然而此系统着重于终端恶意代码检测，对僵尸云的检测并不深入。

总结前人的研究成果，本文认为基于网络流特征分析的方法将是僵尸云检测的有效手段。然而，由于僵尸云网络中同时存在正常的云服务和恶意的僵尸活动，并且云计算与僵尸网络的工作机理相似，而网络流特征本质上反映了对真实网络行为的理解，使基本的网络流特征很难反映出二者的差异，从而导致用于检测传统僵尸网络的网络流特征分析方法在僵尸云检测问题上并不适用。鉴于 2006 年 Hinton[23]提出的深度学习方法对特征学习的有效性，以及相关应用已经在计算机视觉、图像识别、语音识别和语义分析等多个领域都取得了突破，将深度学习方法应用于网络安全研究逐渐引起了研究者的关注。文献[24]将网络流通过特征之间的距离转换为图像像素矩阵，探讨了利用计算机视觉技术处理入侵检测问题的可能性，但并没有利用深度学习算法进行实践。文献[25]利用深度学习算法提取视频序列中的时空域特征，用以检测视频序列中的异常事件，探索了深度学习在实时异常检测中的应用。文献[26, 27]则利用深度学习算法进行恶意代码检测，显著提高了检测率。文献[28]将恶意代码可视化 JPG 图片，通过搜索图片的纹理进行恶意代码的识别，但并没有利用深度学习算法进行图片特征的再学习。由于基本网络流特征很难反映正常云服务与恶意僵尸网络之间的差异，为了解决这一问题，本文在前人工作的基础上，将深度学习核心算法之一的卷积神经网络应用于对基本网络流特征的再学习过程，进而提取出更加抽象的特征，用以表达网络流数据中隐藏的模式以及结构关系，最终用于检测僵尸云，实验证明，本文提出的方法是可行的。

3 僵尸云特性分析

僵尸云(BotCloud)是指恶意租户从云服务提供商租用的用于构建僵尸网络的云服务。僵尸云作为云服务滥用的一种具体方式，是威胁云计算安全和网络空间安全的新型网络攻击手段。由于僵尸云中包含了 2 种服务方式，一种是提供正常的云服务，一种是用于构建恶意的僵尸网络，使僵尸云除了具有传统僵尸网络的一般特性外，还具有一些独有的特性，给僵尸云检测带来很大挑战。

3.1 传统僵尸网络的一般特性

传统的僵尸网络根据命令控制通信方式主要分为中心式、分布式和混合式。中心式的僵尸网络所有的僵尸主机均与一个僵尸服务器相连。利用的通信协议有 IRC、HTTP 和 HTTPS。IRC 方式需要构建一个 IRC 服务器，每个僵尸主机都与此 IRC 服务器相连用于被动接受命令(推模式“PUSH”，被动)。HTTP 和 HTTPS 方式则是通过僵尸主机与僵尸 Web 服务器之间的请求应答会话过程进行通信(拉模式“PULL”，主动)。中心式僵尸网络部署简单，但隐蔽性不好，容易被摧毁。分布式僵尸网络目前以 P2P 方式为主，稳定性很强，但是通信方式不能保证高可靠性和隐藏性。混合式僵尸网络则综合中心式和分布式的优点，利用多个僵尸服务器构建僵尸网络。

3.2 僵尸云的独有特性

僵尸云依托于云服务，它充分利用云计算安全可靠、方便快捷、动态可扩展等优点，在传统僵尸网络的基础上进一步发展，拥有自己独有的特性。本文总结了僵尸云相比传统僵尸网络具有的 3 个独有特性。

1) 从信誉良好的云服务提供商租用云服务用于构建僵尸网络的服务器，具有很强的隐蔽性，使其稳定性更强，更不易被检测与摧毁；

2) 云计算强大的计算能力使僵尸云的构建速度更快、发动攻击的速度更快、攻击能力更强；

3) 借助于云服务“资源共享”的特点，能够快速向移动终端渗透，对移动互联网络的威胁更大。

正是由于僵尸云这 3 个独有特性，一方面，僵尸云对整个网络空间的威胁相对传统僵尸网络更大；另一方面针对僵尸云的检测难度更大。因此，检测僵尸云的存在、分析其行为和实施合适的防护手段是云计算安全要研究的重要问题。

3.3 利用僵尸云发动网络攻击的方式

和传统僵尸网络一样，攻击者能够快速利用僵尸云发动诸如 DDoS 攻击、垃圾邮件、网络欺诈等多种网络攻击，一方面会严重影响到网络安全，另

一方面也会严重损坏云服务提供商的良好信誉。利用僵尸云发动网络攻击的模式如图 1 所示。

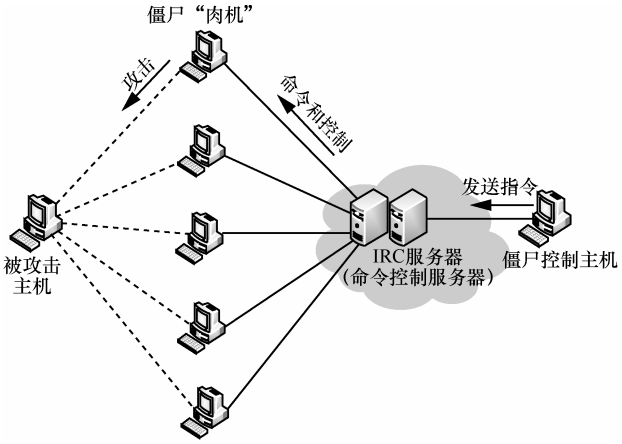


图 1　利用僵尸云发动网络攻击的模式

1) 僵尸网络的控制者通过向云服务提供商购买云服务搭建 IRC 服务器，构建一个聊天通道，等待僵尸主机加入聊天通道中。

2) 僵尸网络的控制者通过各种方法感染正常的计算机。

3) 被感染的计算机通过特定的用户名和密码加入到聊天信道中，等待命令的接收，同时维持和 IRC 服务器之间的连接。

4) 僵尸网络控制者通过 IRC 服务器向僵尸主机发送命令。

5) 僵尸主机接收到命令后，解析命令调用相应的模块对目标主机进行攻击。

3.4　僵尸云检测面临的挑战

鉴于僵尸云对网络空间和云计算的极大威胁，针对僵尸云的检测研究变得十分重要。通过对比分析传统僵尸网络与僵尸云的异同，针对僵尸云网络的检测主要存在 2 个方面的挑战。

1) 僵尸云利用云服务提供商提供的虚拟服务器作为 IRC 服务器，隐蔽性强、难以检测。云服务提供商一般都是信誉良好的大企业，很难对其产生怀疑。

2) 僵尸云中同时存在正常的云服务和恶意的僵尸活动，二者工作方式相似，难以分辨。僵尸网络和云计算的工作特点有 3 点相似之处：①都是基于大流量以及大数据的一种工作方式，对于网络流量都十分依赖，将分散分布的计算资源聚合起来完成某项大任务；②都是基于一对多的控制，对于云来说，它的集中端为云服务提供商，而对于僵尸网络，它的集中端为僵尸网络服务器；③无论是云计算还是僵尸网络，其使用模式和能力都超越了 CPU 的限制，令使用者能够以极低的代价高效地控制大量的资源为其服务。因此，正常的云服务为恶意的僵尸活动提供了掩护，使针对僵尸云检测更加困难。

4　基于卷积神经网络的僵尸云检测

4.1　概述

基于卷积神经网络的僵尸云检测方法的核心思想是通过选取合适的网络流特征并将特征编码，之后利用卷积神经网络对选取的特征进行学习，抽象出更有利于分类的特征，最终利用有监督的机器学习算法对特征数据进行分类，进而识别出异常网络流并检测出僵尸云。其总体框架如图 2 所示。

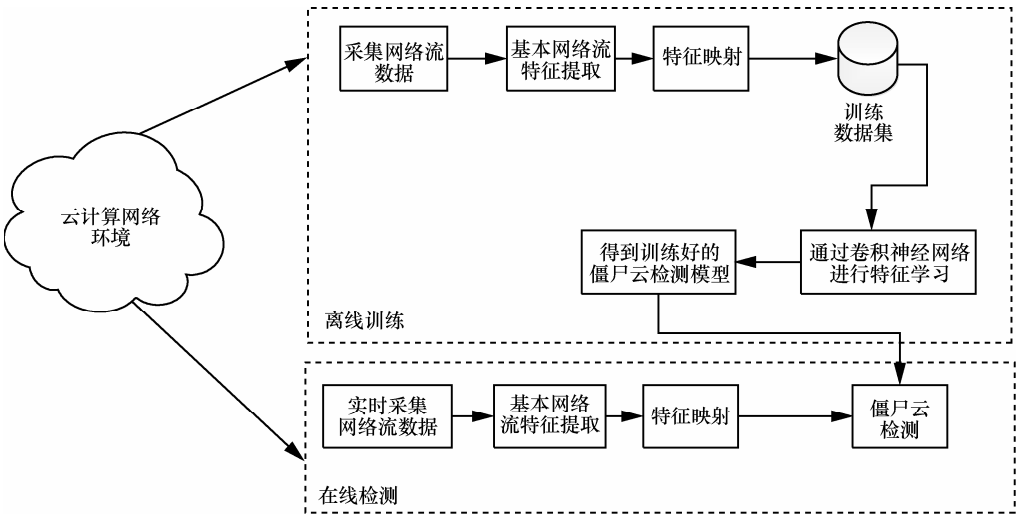


图 2　僵尸云检测总体框架

整个僵尸云检测过程分为离线训练和在线检测 2 个阶段。

1) 离线训练

① 采集网络流数据。通过利用 Wireshark 在云计算网络环境的出口和入口采集网络流数据。

② 基本网络流特征提取。依据僵尸云网络行为的时空相似性，选取用于检测僵尸云网络的网络流统计特征，并生成特征向量。

③ 特征映射。在时间窗口中提取基本网络流特征并生成流特征向量，然后将流特征向量标准化，进而利用特征之间的欧拉距离将流特征向量转换为欧拉矩阵并可视化为灰度图像。

④ 通过卷积神经网络进行有监督学习。将特征映射后的灰度图像作为卷积神经网络的输入，通过卷积神经网络的深度学习，将基本的网络流特征表达为更加精确和抽象的高层表示，从而更好地发现网络流数据中隐藏的模式以及结构关系，最终利用有监督学习对卷积神经网络进行训练。

2) 在线检测

① 实时采集一定时间窗的网络流数据；

② 提取基本网络流特征并映射为灰度图像；

③ 将灰度图像作为已训练好的卷积神经网络的输入、输出检测结果。

4.2 基本网络流特征提取

基于网络流分析的僵尸云检测方法，主要是利用僵尸云网络行为的时空相似性，如具有相似的通信周期性、分组大小、持续时间、数据分组时序、流量持续时间、报文到达时间间隔等统计特征的时间相似性和具有访问域名、端口、IP 等统计特征的空间相似性，然后采用数据挖掘、机器学习等技术提取特征向量，并构造分类检测器实施检测。通过分析僵尸云网络行为的特性，并参考文献[16]，本文选取了 20 个用来检测僵尸云网络的基本网络流特征，如表 1 所示。

4.3 特征映射

本文提出的基于卷积神经网络的僵尸云检测

表 1 选取的 20 个基本网络流特征

序号	特征	描述	类型
1	source IP	源 IP 地址	字符串
2	destination IP	目的 IP 地址	字符串
3	source port	源端口号	整型
4	destination port	目的端口号	整型
5	protocol	协议类型	字符串
6	PX (total number of packet exchanged)	数据分组的总数量	整型
7	NNP (number of null packets exchanged)	空数据分组的数量	整型
8	IOPR (ratio between the number of incoming packets over the number of outgoing packets)	进出数据分组数量的比率	浮点型
9	reconnect (number of reconnects)	重连接的数量	整型
10	duration (flow duration)	流持续的时间	浮点型
11	FPS (length of the first packet)	第一个数据分组的长度	整型
12	TBT (total number of bytes)	总共的字节数	整型
13	average bytes per packet	平均每个分组的字节数	浮点型
14	variance of bytes per packet	每个分组字节数的方差	浮点型
15	APL (average payload packet length)	平均分组长度	浮点型
16	DPL (total number of packets with the same length over the total number of packets)	相同长度的分组数量与总分组数量的比例	浮点型
17	PV (standard deviation of payload packet length)	数据分组长度的标准差	浮点型
18	BS (average bits-per-second)	平均每秒比特数	浮点型
19	AIT (average inter arrival time of packets)	数据分组到达的平均间隔	浮点型
20	PPS (average packets-per-second)	平均每秒的分组数	浮点型

方法本质上是利用特征分类的问题。对于一个分类问题，不同的特征表达对分类结果的准确率影响很大。网络流特征的选取本质上反映了对真实网络行为的理解，由于僵尸云中同时包含正常的云服务和异常的僵尸网络活动，且二者工作机理相似，使通过基本网络流特征很难将二者分开。为此，本文将初步提取的基本网络流特征作为输入，通过卷积神经网络的深度学习方法把低层次的特征通过组合转化成更加抽象的高层表示，用以表达网络流数据中隐藏的模式以及结构关系，进而提高分类和识别的准确率。特征映射的目的是将提取的基本网络流特征转化为适合卷积神经网络学习的灰度图像。整个特征映射过程如图 3 所示。

本文首先设定合适的时间窗口，在时间窗口中提取基本网络流特征并生成流特征向量，然后将流特征向量标准化，进而利用特征之间的欧拉距离将流特征向量转换为欧拉矩阵并可视化为灰度图像。特征映射结果发现正常网络流转换后的图片和僵尸云网络流转换后的图片具有很大的差异。

1) 特征标准化

为了消除网络流不同特征之间的量纲关系，从而使数据具有可比性，本文采用 Z-score 标准化方法对一个时间窗内的流特征向量进行标准化。

设定时间窗为 T，则 T 时间内假定提取的网络流有 n 个，记为 $\boldsymbol{X}=[\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\cdots,\boldsymbol{x}_n]$，其中，

$$\boldsymbol{x}_i=\begin{bmatrix} f_1^i \\ f_2^i \\ \cdots \\ f_m^i \end{bmatrix} \tag{1}$$

表示第 i 个 m 维（本文中 m 为 20）的网络流特征向量（下同），i=1,2,⋯,n,则利用式(2)进行数据的标准化。

$$f_j^i=\frac{f_j^i-\overline{f_j}}{\sigma_{f_j^i}} \tag{2}$$

其中，$\overline{f_j}$ 为 f_j^i 的平均值，$\sigma_{f_j^i}$ 为 f_j^i 的标准差，即

$$\overline{f_j}=\frac{1}{n}\sum_{i=1}^{n}f_j^i \tag{3}$$

$$\sigma_{f_j^i}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(f_j^i-\overline{f_j})^2} \tag{4}$$

2) 标准化后基本网络流特征向量转换为欧拉矩阵

对于每一个网络流特征向量，不同特征之间具有一定的相关性，利用标准化后的流特征之间的欧拉距离，将时间窗内的流特征向量集合转换为欧拉矩阵，从而实现对不同特征之间的多元关联。具体步骤如下。

步骤 1　将时间窗 T 内的 n 个流特征向量合并记为 $\boldsymbol{X}=[\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\cdots,\boldsymbol{x}_n]$，将 $\boldsymbol{x}_i$ 代入 $\boldsymbol{X}$ 中，$\boldsymbol{X}$ 能够被表示为

$$\boldsymbol{X}=\begin{bmatrix} f_1^1 & f_1^2 & \cdots & f_1^n \\ f_2^1 & f_2^2 & \cdots & f_2^n \\ \cdots & \cdots & \cdots & \cdots \\ f_m^1 & f_m^2 & \cdots & f_m^n \end{bmatrix} \tag{5}$$

其中，f_l^i 是第 i 个网络流特征向量中的第 l 个特征的数值，l=1,2,⋯,m，i=1,2,⋯,n。

步骤 2　为了充分利用第 i 个网络流特征向量中不同特征之间的相关性，将 $\boldsymbol{x}_i$ 转换成一个 m 行 m 列的矩阵，具体转换方法为

$$\boldsymbol{x}_i'=\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{I}=[f_1^i f_2^i\cdots f_m^i]\cdot \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{m\times m}=\begin{bmatrix} f_1^i & 0 & \cdots & 0 \\ 0 & f_2^i & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & f_m^i \end{bmatrix}_{m\times m} \tag{6}$$

步骤 3　矩阵 $\boldsymbol{x}_i'$ 的对角线元素为 m 维特征的值，矩阵 $\boldsymbol{x}_i'$ 的每一列用一个 m 维向量 $\boldsymbol{F}_j^i$ 表示，则

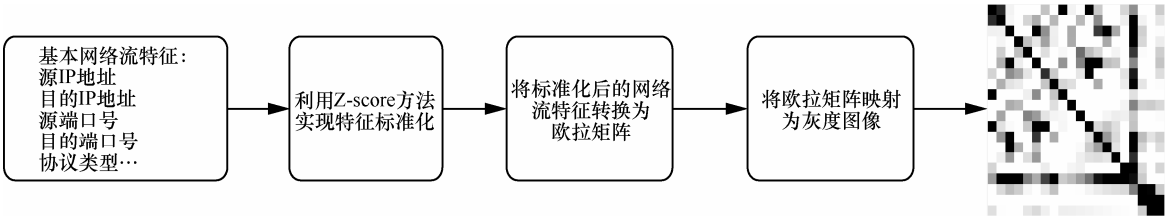


图 3　基本网络流特征映射为灰度图像过程

$$\boldsymbol{F}_j^i = \begin{bmatrix} \eta_{j,1}^i \\ \eta_{j,2}^i \\ \cdots \\ \eta_{j,m}^i \end{bmatrix} \tag{7}$$

其中，

$$\eta_{j,p}^i = \begin{cases} 0, j \neq p \\ f_j^i, j = p \end{cases} \tag{8}$$

于是

$$\boldsymbol{x}_i' = [\boldsymbol{F}_1^i \boldsymbol{F}_2^i, \cdots, \boldsymbol{F}_m^i] \tag{9}$$

步骤 4 利用欧拉距离定义流特征向量不同特征之间的相关性如下

$$ED_{j,k}^i = \begin{cases} \sqrt{(\boldsymbol{F}_j^i - \boldsymbol{F}_k^i)^{\mathrm{T}} (\boldsymbol{F}_j^i - \boldsymbol{F}_k^i)}, j \neq k \\ 0, j = k \end{cases} \tag{10}$$

其中，i=1,2,…,n；j,k=1,2,…,m，则第 i 个流量记录可被表示成为一个 m 行 m 列对角线全为零的对称矩阵 ***EMD***。

$$\boldsymbol{EMD}_{x_i} = \begin{bmatrix} ED_{1,1}^i & ED_{1,2}^i & \cdots & ED_{1,m}^i \\ ED_{2,1}^i & ED_{2,2}^i & \cdots & ED_{2,m}^i \\ \cdots & \cdots & \cdots & \cdots \\ ED_{m,1}^i & ED_{m,2}^i & \cdots & ED_{m,m}^i \end{bmatrix} \tag{11}$$

3) 欧拉矩阵的可视化

将 ***EMD*** 矩阵中的每个元素看作像素点，矩阵中元素的值代表像素的灰度，元素值越大，灰度越大，越接近白色；元素值越小，灰度越小，越接近黑色。由此可将每一个时间窗口内的网络流特征向量集转换为一幅灰度图像。如图 4 和图 5 所示。

图 4 IRC 僵尸云网络流可视化

图 5 正常云服务的网络流可视化

由上述图像可知，对于正常和异常的网络流，利用欧拉距离将其转换成 ***EMD*** 后，存在显著的差异，可以用来进行异常检测，即上述映射方法是可行的。

4.4 利用卷积神经网络进行有监督的特征学习

随着云计算和大数据技术的发展，深度神经网络成为近些年特征学习方面的研究热点，并且在图像识别、语音识别、计算机视觉等多类应用中取得突破性的进展。作为深度学习的一种重要模型，卷积神经网络在图像特征学习上显示出了独特的优势。本文选取卷积神经网络作为特征学习方法，用于解决僵尸云检测问题。

1) 卷积神经网络的网络结构

卷积神经网络是从传统神经网络基础上发展而来的一个多层的神经网络，它本质是通过学习一种深层非线性网络结构，实现复杂函数逼近，表征输入到输出的映射关系，具有强大的从少数样本集中学习数据集本质特征的能力。卷积神经网络是一个多层网络结构，由卷积(convolution)层、次抽样(subsampling)层和全连接(full connection)层这 3 种操作组合而成，其中，卷积层和次抽样层在中间层里交替出现，全连接层常用于连接最后 2 层节点，用于输出最终结果。

① 卷积层

在卷积层，上一层的特征映射图与训练好的卷积核进行卷积运算，运算结果经过激活函数后，输出本层的特征映射图。本层的特征映射图可能是前一层的几个特征映射图组合的结果。一般地，卷积层的形式为

$$x_j^l = f\left(\sum_{i \in M_j} x_j^{l-1} k_{ij}^l + b_j^l\right) \tag{12}$$

其中，l 代表层数，k 是卷积核，M_j 表示输入选择的特征映射图，b 为偏置。激活函数则通常采用 sigmoid、tanh 或 softsign 等饱和非线性函数，实现输出范围的映射。

② 次抽样层

一个次抽样层对输入进行抽样操作。次抽样层不改变特征映射层的数量，但会缩小特征映射图的尺寸。次抽样层的一般形式为

$$x_j^l = f(\beta_j^l \mathrm{down}(x_j^{l-1}) + b_j^l) \tag{13}$$

其中，down(·)表示次抽样函数。次抽样函数通常是对输入图像中每一个 $n \times n$ 大小的区域加权求和，故输出图像的大小变为输入图像大小的 $\frac{1}{n}$。每一个输

出的特征映射图有自己的β和偏置 b。

③ 全连接层

全连接层通常位于 CNN 的最后位置，作用是计算网络的输出结果。分类任务中通常在这一层训练一个分类器，将学习到的高层特征作为分类器的输入，输出结果是分类的结果。

作为一种典型的卷积神经网络应用模型，LeNet-5 最初用于手写数字识别并取得了巨大的成功[29]。LeNet-5 使用了 5 层卷积神经网络模型，包括卷积层、次抽样层、卷积层、次抽样层、全连接层。它避免了对数字图像进行复杂的预处理过程，在模式分类领域得到了广泛的应用。

针对僵尸云检测与手写数字识别的不同，对传统的 LeNet-5 进行了改进。不同之处主要体现在以下 2 点。

① 根据特征映射得到的灰度图像尺寸，将网络的输入层设计为 20×20 像素。

② 僵尸云检测的目的是为了区分正常的云服务与恶意的僵尸网络，其本质是二分类问题。因此，将传统的 LeNet-5 的输出层由 10 个神经元改为 2 个神经元。

本文设计的用于检测僵尸云网络的卷积神经网络结构如图 6 所示。

输入为设定时间窗内网络流信息转换成的 20×20 的图片，下面介绍各个层的结构。

C1 层为卷积层。使用感知阶段预训练得到的 6 个 3×3 大小的卷积核对输入图像进行卷积，得到 6 张 18×18 的特征映射图。

S2 层为次抽样层。使用 2×2 大小的窗口对 C1 层的 6 张图像池化得到 6 张 9×9 的特征映射图。

C3 为卷积层。使用预训练的 16 个 4×4 的卷积核对 S2 的 6 张特征映射图进行卷积，得到 16 张 6×6 的特征映射图。本文采取的卷积组合规则和 LeNet-5 给出的规则一样，如表 2 所示。

S4 为次抽样层。使用 3×3 大小的窗口抽样，得到 12 张 3×3 的特征映射图。

C5 层为全连接层。利用 80 个 3×3 大小的卷积核将 16 张 3×3 的特征映射图全连接为一个 80×1 的

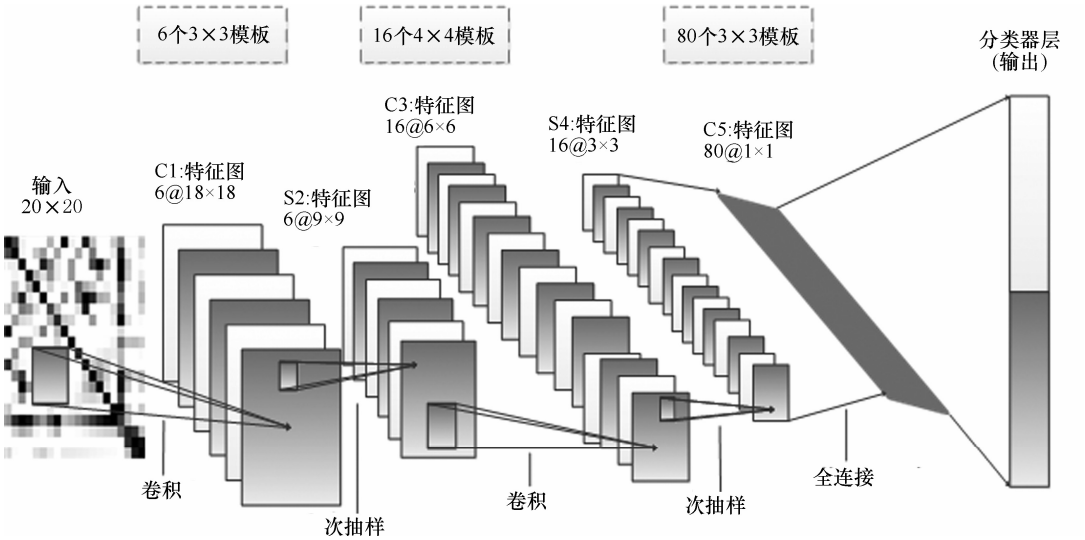


图 6　本文设计的 5 层 CNN 网络结构

表 2　　C3 卷积层卷积组合规则

编号	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	√				√	√	√			√	√	√	√		√	√
1	√	√				√	√	√			√	√	√	√		√
2	√	√	√				√	√	√			√		√	√	√
3		√	√	√			√	√	√				√		√	√
4			√	√	√			√	√				√	√		√
5				√	√	√			√					√	√	√

向量。

输出层利用 Soft-max 分类器，输出结果为 2 类，正常（编码为 0）和异常（编码为 1）。

2) 卷积神经网络的训练过程

卷积神经网络的训练主要是利用误差反向传播算法，即 BP(back propagation)算法[30]。BP 算法由数据流的正向传播和误差信号的反向传播 2 个过程构成。利用输出后的误差来估计输出层的直接前导层的误差，再用这个误差估计更前一层的误差，如此一层一层地反传下去，就获得了所有其他各层的误差估计。权值调整的过程也就是网络的学习训练过程。

具体来讲，对于一个 N 层结构的神经网络模型，将第 1 层记为输入层，第 N 层记为输出层，第 l 层的节点数记为 S_l，则网络输出层的值可以从输入层开始逐层计算出来。假设有 m 对训练样本$\{(x^{(1)}, y^{(1)}),\cdots,(x^{(m)},y^{(m)})\}$，本文使用梯度下降算法来训练神经网络学习一个输入到输出的映射。

$$f_{\boldsymbol{W},b}(x^{(i)}) = \hat{y}^{(i)} \cong y^{(i)} \tag{14}$$

模型在训练样本上总体代价函数定义为

$$\Gamma(\boldsymbol{W},b) = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\| f_{\boldsymbol{W},b}(x^{(i)}-y^{(i)})\|^2\right)+\frac{\lambda}{2}\sum_{l=1}^{N}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}(W_{ji}^{(l)})^2 \tag{15}$$

其中，前一项是训练样本实际值与网络输出值之间误差的平方和的均值，后一项是权值衰减项，即网络所有权值的平方和与权值衰减因子 λ 的乘积。对于一个输入样本$(x^{(i)},y^{(i)})$，它的 2 个传播过程如下。

正向传播过程。首先计算第 2 层所有节点的激活值，以此为输入计算第 3 层的输出值，一层一层不断向前推进，最终计算出输出层的输出值$f_{\boldsymbol{W},b}(x^{(i)})$。

反向传播过程。利用网络输出层的输出值$f_{\boldsymbol{W},b}(x^{(i)})$和目标值$y^{(i)}$之间的误差$\delta_i^{(N)}$作为输出层的误差，然后将输出误差反传，即将误差分摊给各层所有单元，从而获得各层单元的误差$\delta_i^{(l)}$，进而修正各单元的权值。

利用 BP 算法训练卷积神经网络的过程如算法 1 所示。

算法 1 BP 算法

输入：m 对训练样本$\{(x^{(1)},y^{(1)}),\cdots,(x^{(m)},y^{(m)})\}$,网络层数为 N，网络权重矩阵 $\boldsymbol{W}$,偏置 b,学习速率 a,权值衰减因子 λ。其中，$z^{(i)}$表示第 i 层的输出值，$\boldsymbol{a}^{(i)}$表示经激活函数激活后的第 i 层输出值。

输出：更新后的网络权重矩阵 $\boldsymbol{W}'$,偏置 b'

1) for $l\leftarrow 1$ to N do $\Delta W^{(l)}\leftarrow 0$, $\Delta b^{(l)}\leftarrow 0$

2) for $t\leftarrow 1$ to m do

3) for $i\leftarrow 1$ to N do $z^{(i)}\leftarrow f_{\boldsymbol{W},b}(x^{(t)})$, $a^{(i)}\leftarrow \text{sigmoid}(z^{(t)})$

4) $\delta_i^{(N)}\leftarrow -(y_t-a^{(N)})f'(z^{(N)})$

5) End for

6) for $l\leftarrow N-1$ to 2 do $\delta^{(l)}\leftarrow((\boldsymbol{W}^{(l)})^{\mathrm{T}}\delta^{(l+1)}f'(z^{(l)})$

7) for $l\leftarrow N-1$ to 1 do {

8) $\nabla_{\boldsymbol{W}^{(l)}}\Gamma(\boldsymbol{W},b)\leftarrow\delta^{(l+1)}(\boldsymbol{a}^{(l)})^{\mathrm{T}}$, $\nabla_{b^{(l)}}\Gamma(\boldsymbol{W},b)\leftarrow\delta^{(l+1)}$

9) $\Delta\boldsymbol{W}^{(l)}\leftarrow\Delta\boldsymbol{W}^{(l)}+\nabla_{\boldsymbol{W}^{(l)}}\Gamma(\boldsymbol{W},b)$, $\Delta b^{(l)}\leftarrow\Delta b^{(l)}+\nabla_{b^{(l)}}\Gamma(\boldsymbol{W},b)$

10) End for

11) End for

12) End for

13) for $l\leftarrow N-1$ to 1 do

14) $\boldsymbol{W}^{(l)}\leftarrow\boldsymbol{W}^{(l)}-\alpha[(\frac{\Delta\boldsymbol{W}^{(l)}}{m})+\lambda\boldsymbol{W}^{(l)}]$

15) $b^{(l)}\leftarrow b^{(l)}-\alpha(\frac{\Delta b^{(l)}}{m})$

16) End for

17) End for

5 实验与结果分析

5.1 实验环境与数据来源

实验采用的实验数据是UNB ISCX 2012 intrusion detection evaluation dataset，该数据集是全世界公开的入侵检测研究的最佳数据集之一。根据原数据集的网络拓扑环境，构建了本文的实验环境，如图 7 所示。

整个实验环境包含 3 个主要部分：私有云服务、5 个用户子网和 IDS。私有云服务主要由 IRC 服务器、Web 服务器和主机构成，其 IP 为 192.168.5.0/24；5 个用户子网的 IP 分别为 192.168.1.0/24、192.168.2.0/24、192.168.3.0/24、192.168.4.0/24 和 192.168.6.0/24。IDS 用于获取私有云服务器和用户子网之间的网络流数据。最后，结合本文的研究内容，选取了数据集中 2011 年 6 月 15 日的数据（这一天的数据记录了利用 IRC 僵尸网络发动 DDoS 攻击的网络活动）在搭建的实验平台上进行重放，获取了用于检测 IRC 僵尸云的数据集，如表 3 所示。此外，保留了原始流数据分组，用于实时检测实验。

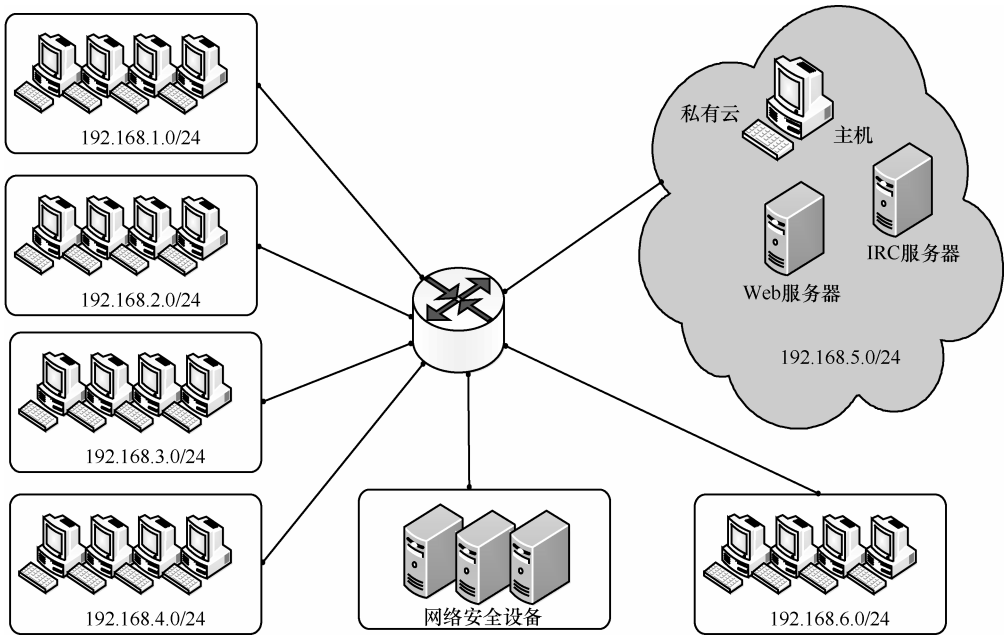


图 7　IRC 僵尸云检测实验环境

表 3　　**实验所用数据集**

数据集类型	标签	流数量(比例)	总数
训练	正常	106 725(78.6%)	135 697
	攻击	28 972(21.4%)	
测试	正常	12 716(69.7%)	18 194
	攻击	5 478(30.3%)	

对于采集到的实验数据，本文实验的硬件环境是 Dell 服务器(内存 8 GB)并配置 GPU Tesla K20c(显存 4.8 GB)，利用 java 实现对网络流特征的提取、特征映射，最后借助贾扬清博士提供的 caffe[31]深度学习架构实现基于卷积神经网络的异常网络流检测。

5.2　实验算法与流程

主要有卷积神经网络初始化、网络训练和网络测试 3 步，具体流程如图 8 所示。

实验流程如下。

Step1　原始数据收集与处理。选择从表 3 所示的数据集中导入原始数据，分为训练样本和测试样本数据，并形成统一的数据格式存放到矩阵中。

Step2　卷积神经网络初始化。借鉴 LetNet-5 网络的层次结构和卷积核，结合本文算法的特性，设计用于检测僵尸云的卷积神经网络。

Step3　卷积神经网络训练。将训练数据输入识别模型，训练确定卷积神经网络的各层权值系数。

Step4　卷积神经网络测试。将测试数据输入卷积神经网络进行僵尸云流量分类，判断检测率是否高于设定下限，如果高于下限，则保存参数后微调重复 Step3；否则直接调整参数跳转到 Step3。

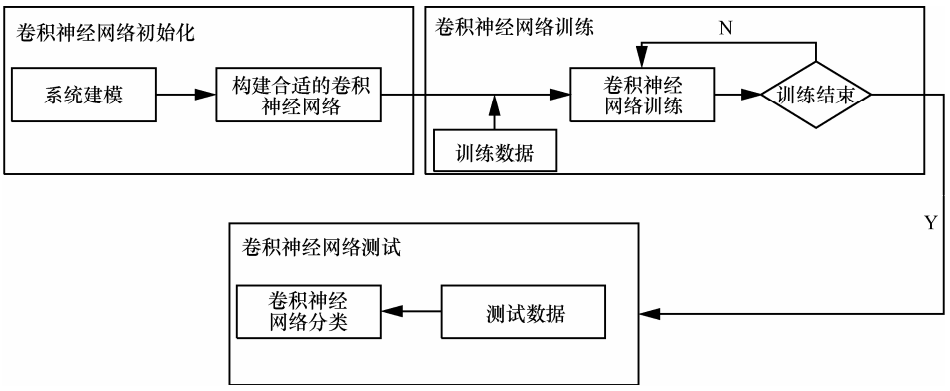


图 8　利用卷积神经网络检测僵尸云全流程

5.3 结果分析

5.3.1 评价指标

基于网络流分析的IRC僵尸云检测方法的性能评价指标主要分为 2 个方面。一是检测的准确度，包括检测率、误报率和漏报率；二是检测的速度，包括离线训练时间和测试时间。具体而言，在本实验中

$$\text{检测率}=\frac{\text{正确检测出的攻击样本数量}}{\text{总攻击样本数量}}\times 100\% \quad (16)$$

$$\text{误报率}=\frac{\text{被误判为攻击的正常样本数量}}{\text{总的正常样本数量}}\times 100\% \quad (17)$$

$$\text{漏报率}=\frac{\text{被误判为正常的攻击样本数量}}{\text{总的攻击样本数量}}\times 100\% \quad (18)$$

离线训练时间和测试时间均是在同一实验条件下针对同一训练和测试样本测试得来。为达到最佳效果，实验中的所有评价指标均是多次测量取平均值。

5.3.2 训练最优卷积神经网络

1) 卷积神经网络层次结构设计

由于卷积神经网络的层次结构直接影响了网络训练和测试的结果，在卷积神经网络初始化阶段，根据 LeNet-5 网络的基本层次结构，设计了 6 种具有不同结构的卷积神经网络，丰富了网络的可选择性，为得到适用于本文算法的卷积神经网络打下基础，如表 4 所示。

针对设计过程中存在的问题，有以下 3 点说明。

① 由于输入层的数据样本大小为 20×20，计算难度并不太大，故为避免损失信息，和 LeNet-5 网络保持一致，设定卷积层的步长为 1，次抽样层的采样窗口为 2×2。

② 由于特征图的边长不一定是 2 的倍数，为了避免损失信息，本文采用保留边缘的方法进行次抽样的预处理，即将特征图的边长填充为 2 的倍数，然后再次抽样。

③ 当卷积核尺寸大于 6×6 时，5 层网络显得过“深”，容易出现过拟合的现象，不利于后续的训练和测试，故设计的网络结构中的卷积核尺寸均不大于 6×6。

2) 不同层次结构卷积神经网络的检测结果分析

卷积神经网络的次抽样层主要是利用图像局部相关性的原理，对卷积层的输出进行抽样，从而减少数据处理量，同时保留有用信息。常见的次抽样方法主要有最大池化和均值池化 2 种。其中，最大池化是取采样窗口中所有元素的最大值；均值池化是取采样窗口中所有元素的均值。采用不同的次抽样方法能够影响卷积神经网络的检测结果。因此，本文利用表 3 所示的数据集，分别研究了不同抽样方法下 6 种网络的检测结果，具体如表 5 和表 6 所示。

表 4 实验设计的 6 种不同网络结构的卷积神经网络

编号	C1 卷积层		S2 次抽样层		C3 卷积层		S4 次抽样层		C5 全联接层	
	卷积核	输出	采样窗口	输出	卷积核	输出	采样窗口	输出	卷积核	输出
1	6×(3×3)	6×(18×18)	2×2	6×(9×9)	16×(3×3)	16×(7×7)	2×2	16×(4×4)	80×(4×4)	80×1
2	6×(3×3)	6×(18×18)	2×2	6×(9×9)	16×(3×3)	16×(7×7)	2×2	16×(3×3)	80×(3×3)	80×1
3	6×(3×3)	6×(18×18)	2×2	6×(9×9)	16×(4×4)	16×(6×6)	2×2	16×(3×3)	80×(3×3)	80×1
4	6×(4×4)	6×(17×17)	2×2	6×(9×9)	16×(4×4)	16×(6×6)	2×2	16×(3×3)	80×(3×3)	80×1
5	6×(5×5)	6×(16×16)	2×2	6×(8×8)	16×(5×5)	16×(4×4)	2×2	16×(2×2)	80×(2×2)	80×1
6	6×(6×6)	6×(15×15)	2×2	6×(8×8)	16×(6×6)	16×(3×3)	2×2	16×(2×2)	80×(2×2)	80×1

表 5 采用最大池化方法时的检测结果

编号	检测率	误报率	漏报率	训练时间/s	测试时间/s
1	0.925 1	0.073 9	0.072 8	1 023	63
2	0.921 6	0.077 2	0.061 4	1 064	61
3	0.936 7	0.089 1	0.070 1	1 013	68
4	0.882 3	0.121 4	0.091 4	988	64
5	0.869 2	0.118 4	0.099 5	956	67
6	0.827 1	0.136 9	0.112 8	899	72

表 6　采用均值池化方法时的检测结果

编号	检测率	误报率	漏报率	训练时间/s	测试时间/s
1	0.931 5	0.072 4	0.061 1	1 054	68
2	0.915 6	0.089 6	0.052 3	1 018	63
3	0.946 3	0.065 7	0.057 1	1 003	65
4	0.894 7	0.102 3	0.086 1	957	71
5	0.875 4	0.091 1	0.116 4	976	69
6	0.849 8	0.131 1	0.105 2	913	69

实验结果表明，对于本文的算法，卷积核越大，检测准确度相对较差，而检测速度较快。总体而言，6 种不同层次结构卷积神经网络的检测速度相差不大且在可接受范围内；而检测准确度是衡量僵尸云检测问题更为重要的评价指标。鉴于此，本文选取采用均值池化方法编号为 3 的卷积神经网络为最优卷积神经网络，其检测率、误报率和漏报率分别达到 94.63%、6.57%和 5.71%。

5.3.3　实验结果对比与分析

1) 离线状态下不同样本数目下检测结果对比

目前，检测僵尸网络公认效果较好的是 SVM 和决策树方法。因此，本文利用上文得出的最优卷积神经网络，在给定不同样本数量条件下将三者的检测结果进行统计并对比，结果如表 7 所示。

图 9 给出了不同样本数量下卷积神经网络、SVM 和决策树用于检测僵尸云的检测率、误报率、漏报率和测试时间。从图 9 中可以看出，随着样本数量的增加 3 种方法的检测率和测试时间都呈上升趋势，而误报率和漏报率呈下降趋势。在同等样本数量下，卷积神经网络的检测准确度和检测速度均优于 SVM 和决策树。

2) 实时状态下不同时间窗下检测结果对比

离线状态下的测试数据均是有标签的网络流特征向量，而实时的僵尸云检测问题则需要从设定时间窗中网络流数据分组信息中提取特征。因此，时间窗大小的设定对实时检测方法的准确度有着重要影响。为了设定合适的时间窗，设定 10 s、30 s、60 s、120 s、180 s、240 s、300 s 为 7 个不同的时间窗，取 10%原始流数据分组（约 2.3 GB）并分别按不同时间窗各形成 12 000 条测试样本，通过实验比较了不同时间窗条件下 3 种方法的检测率、误报率、漏报率和测试时间，结果如表 8 所示。

图 10 给出了不同时间窗下卷积神经网络、SVM 和决策树用于检测僵尸云的检测率、误报率、漏报率和测试时间。从图 10 中可以看出，随着时间窗的增大 3 种方法的检测率都呈上升趋势，而误报率和漏报率呈下降趋势，测试时间则呈相对稳定趋势。在相同时间窗下，卷积神经网络的检测准确度和检测速度均优于 SVM 和决策树。当时间窗大于 180 s 时，检测率、误报率和漏报率趋于稳定。因此，本文认为在实时检测僵尸云时设定时间窗大小为 200 s 是合适的。

综上所述，由于僵尸云和正常云服务 2 种环境

表 7　不同样本数量下的 3 种算法检测结果

样本数量	本文算法				SVM				决策树			
	检测率	误报率	漏报率	测试时间/s	检测率	误报率	漏报率	测试时间/s	检测率	误报率	漏报率	测试时间/s
10 000	0.861 1	0.114 9	0.107 1	9.3	0.753 5	0.154 6	0.127 3	21.1	0.781 3	0.149 7	0.129 1	25.4
20 000	0.870 5	0.101 0	0.104 2	17.5	0.740 4	0.149 8	0.124 5	39.2	0.797 9	0.142 5	0.126 2	48.3
30 000	0.873 0	0.103 7	0.099 6	24.7	0.762 8	0.146 9	0.118 7	57.6	0.807 7	0.147 7	0.117 3	71.1
40 000	0.882 9	0.097 0	0.094 2	31.4	0.778 4	0.140 3	0.115 4	73.4	0.793 4	0.136 5	0.113 0	93.6
50 000	0.878 5	0.093 4	0.089 0	37.6	0.781 8	0.145 2	0.109 4	88.1	0.822 5	0.134 6	0.103 7	114.7
60 000	0.908 4	0.087 6	0.086 4	43.1	0.804 8	0.131 6	0.110 2	104.3	0.838 0	0.132 3	0.100 4	134.8
70 000	0.912 3	0.081 1	0.078 7	48.3	0.815 6	0.128 7	0.102 8	118.0	0.854 0	0.124 0	0.096 1	150.7
80 000	0.927 3	0.076 4	0.072 5	52.6	0.829 4	0.115 4	0.099 5	131.4	0.846 6	0.110 7	0.099 8	167.2
90 000	0.939 8	0.070 3	0.067 2	58.1	0.822 9	0.116 4	0.102 0	143.5	0.851 7	0.111 4	0.095 2	182.5
100 000	0.942 7	0.068 2	0.064 3	63.2	0.841 8	0.111 8	0.098 7	154.6	0.855 0	0.110 2	0.093 2	195.3

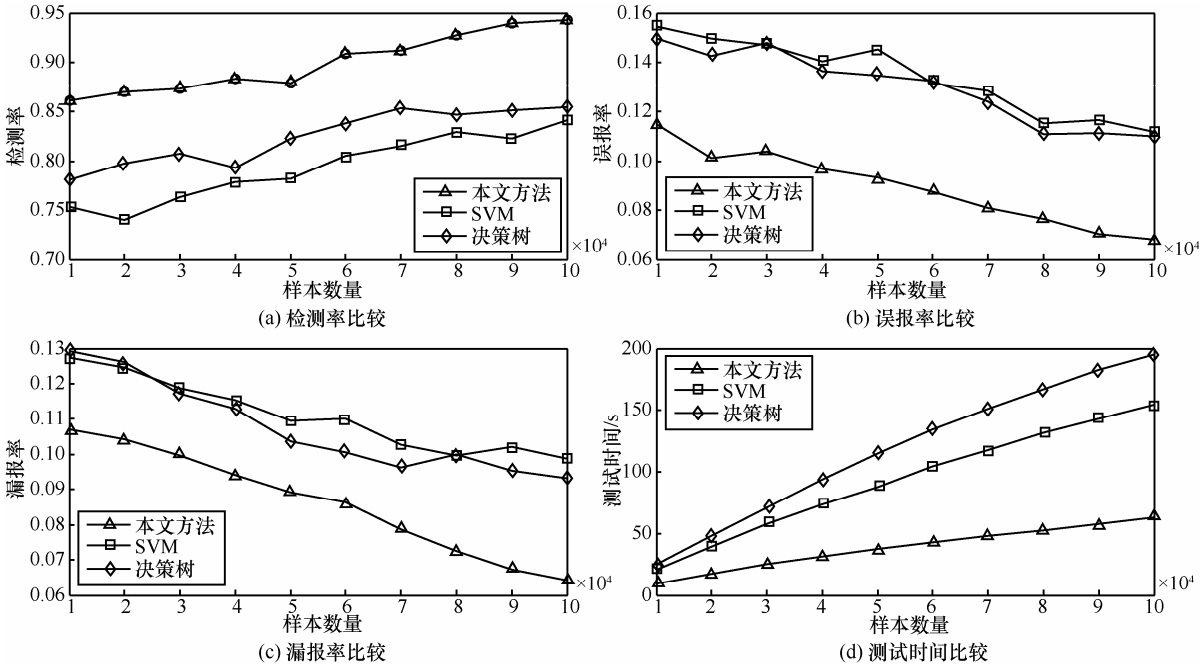


图 9　不同样本数量下 3 种方法检测结果

表 8　　**不同时间窗条件下 3 种算法实时检测结果**

时间窗/s	本文算法				SVM				决策树			
	检测率	误报率	漏报率	测试时间/s	检测率	误报率	漏报率	测试时间/s	检测率	误报率	漏报率	测试时间/s
10	0.835 2	0.452 7	0.381 6	11.3	0.625 2	0.490 3	0.461 6	25.5	0.635 3	0.484 9	0.452 3	30.2
30	0.841 1	0.397 5	0.312 1	11.9	0.630 7	0.458 7	0.425 1	24.9	0.641 5	0.451 8	0.415 6	30.3
60	0.855 6	0.364 6	0.234 5	10.9	0.634 6	0.427 6	0.375 4	25.3	0.645 9	0.402 6	0.365 8	29.1
120	0.876 5	0.245 2	0.193 4	11.6	0.658 3	0.356 1	0.285 5	23.6	0.672 4	0.275 8	0.254 7	27.6
180	0.892 3	0.136 8	0.154 2	11.4	0.678 4	0.254 4	0.199 7	23.9	0.689 5	0.152 4	0.171 2	28.9
240	0.895 1	0.128 7	0.132 4	12.3	0.685 9	0.205 3	0.163 3	24.6	0.693 7	0.132 6	0.160 3	29.5
300	0.896 3	0.111 6	0.091 3	9.7	0.687 5	0.175 7	0.152 4	25.4	0.697 7	0.121 9	0.142 0	27.9

下的基本网络流特征差异不明显，从而导致用于检测传统僵尸网络的网络流特征分析方法在僵尸云检测问题上并不适用。在基于网络流的僵尸网络检测中 Zhao[16]的方法具有代表性，他通过决策树的方法使检测率达到 99.7%，而误报率只有 0.4%。然而，样本数量为 20 000 时，本文利用他构造的决策树对僵尸云网络进行检测，发现检测率迅速下降为 79.8%，而误报率则迅速提高到 14.2%。除此之外，本文利用 SVM 进行了实验，作为经典的机器学习算法，SVM 针对僵尸云的检测率为 74.0%，误报率为 15.0%。这一实验结果充分说明了僵尸云检测的难度大于传统僵尸网络的检测，传统用于检测僵尸网络的方法并不适用于僵尸云检测问题。而利用卷积神经网络对基本网络流特征进行再学习的方法，将检测的准确率提高了将近 10%，达到 87.1%，将误报率降低到 10.1%，说明本文提出的方法是有效的。此外，从训练时间和测试时间上看，卷积神经网络也有着很大的优势，这就为实时检测僵尸云提供了可能。

6　结束语

本文认为基于网络流特征分析的方法将是僵尸云检测的有效手段。然而，由于僵尸云网络中同时存在正常的云服务和恶意的僵尸活动，并且云计算与僵尸网络的工作机理相似，使僵尸云和正常云服务 2 种环境下的基本网络流特征差异不明显，从而导致用于检测传统僵尸网络的网络流特征分析方法在僵尸云检测问题上并不适用。为了解决这一

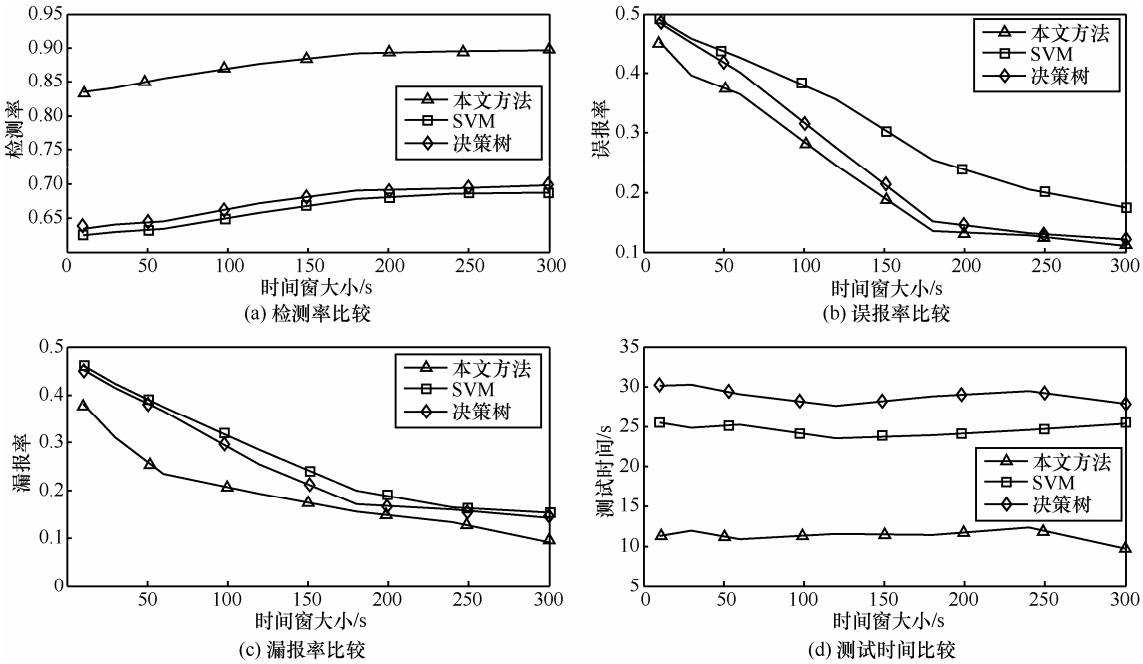


图 10　不同时间窗下 3 种方法检测结果

问题，本文在前人工作基础上，将深度学习重要算法之一的卷积神经网络应用于对基本网络流特征的再学习过程，进而提取出更加抽象的特征，用以表达网络流数据中隐藏的模式以及结构关系，最终用于检测僵尸云，实验结果表明，本文提出的方法不仅能够提高检测的准确度，而且能够减少检测所用的时间。

网络攻击技术愈发隐蔽高超，深度学习应用于僵尸云检测问题的研究还处于起步阶段。在将来的工作中，本文将针对如下 3 个方面进行更进一步的研究。

1) 利用深度学习的其他算法横向比较检测的效果。

2) 增加卷积神经网络的层数，比较检测的效果。

3) 利用云计算环境实现分布式的僵尸云检测，提高检测的效率。

参考文献：

[1] 江健, 诸葛建伟, 段海新, 等. 僵尸网络机理与防御技术[J]. 软件学报, 2012, 23(1): 82-96.
JIANG J, ZHUGE J W, DUAN H X, et al. Research on botnet mechanisms and defenses[J]. Journal of Software, 2012, 23(1): 82-96.

[2] ARTAIL H, MASTRI Z A, SRAJ M, et al. A dynamic honeypot design for intrusion detection[C]//IEEE/ACS International Conference on Pervasive Services. 2004. 95-104.

[3] 诸葛建伟, 韩心慧, 周勇林, 等. HoneyBow:一个基于高交互式蜜罐技术的恶意代码自动捕获器[J]. 通信学报, 2007, 28(12): 8-13.
ZHUGE J W, HANG X H, ZHOU Y L, et al. HoneyBow: an automated malware collection tool based on the high-interaction honeypot principle[J]. Journal on Communications , 2007,28(12):8-13.

[4] ALHAMMADI Y, AICKELIN U. Detecting botnets through log correlation[C]//The Workshop on Monitoring, Attack Detection and Mitigation. 2010.

[5] STINSON E, MITCHELL J C. Characterizing bots' remote control behavior[C]//The 4th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment. 2007: 89-108.

[6] LIU L, CHEN S, YAN G, et al. Bottracer: Execution-based bot-like malware detection[C]//The 11th International Conference on Information Security. 2008: 97-113.

[7] KOLBITSCH C, COMPARETTI P M, KRUEGEL C, et al. Effective and efficient malware detection at the end host[C]//The 18th Conference on USENIX Security Symposium. 2009: 351-366.

[8] ROESCH M. Snort: lightweight intrusion detection for networks[C]// The 13th USENIX Conference on System Administration. 1999: 229-238.

[9] GOEBEL J, HOLZ T. Rishi: identify bot contaminated hosts by IRC nickname evaluation[C]//The first conference on First Workshop on Hot Topics in Understanding Botnets. 2007.

[10] LIVADS C, WALSH R, LAPSLEY D, et al. Using machine learning techniques to identify botnet traffic[C]//31th IEEE Conference on Local Computer Networks. 2006: 967-974.

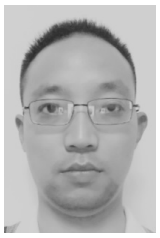
[11] STRAYER W T, LAPSELY D, WALSH R, et al. Botnet detection based on network behavior[C]//2006 ARO Workshop on Botnets. 2007: 1-24.

[12] ZENG Y, HU X, SHIN K. Detection of botnets using combined host and network-level information[C]//International Conference on Dependable Systems and Networks (DSN). 2010: 291-300.

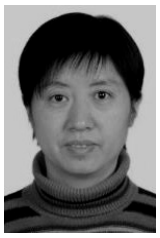
[13] WANG H, HOU J, GONG Z. Botnet detection architecture based on heterogeneous multi-sensor information fusion[J]. Journal of Networks, 2011, 6 (12): 1655-1661.

[14] GU G, ZHANG J, LEE W. BotSniffer: detecting botnet command and control channels in network traffic[C]//The 15th Annual Network and Distributed System Security Symposium. 2008: 269-286.
[15] BEIGI E B, JAZI H H STAKHANOVA N, et al. Towards effective feature selection in machine learning-based botnet detection approaches[C]//International Conference on Communications and Network Security. 2014: 247-255.
[16] ZHAO D, TRAORE I, SAYED B, et al. Botnet detection based on traffic behavior analysis and flow intervals[J]. Computers & Security, 2013, 4(7): 2-16.
[17] 闫健恩, 袁春阳, 许海燕, 等. 基于多维流量特征的 IRC 僵尸网络频道检测[J]. 通信学报, 2013, 34(10): 49-64.
YAN J E, YUAN C Y, XU H Y, et al. Method of detecting IRC botnet based on the multi-features of traffic flow[J]. Journal on Communications, 2013, 34(10): 49-64.
[18] YAMAUCHI K, HORI Y, SAKURAI K. Detecting HTTP-based botnet based on characteristic of the C&C session using by SVM[C]//8th Asia Joint Conference on Information Security. 2013: 63-68.
[19] BADIS H, DOYEN G, KHATOUN R. Toward a source detection of botclouds: a PCA-based approach[C]//International Conference on Autonomous Infrastructure, Management, and Security. 2014: 105-117.
[20] TULASIRAM N, ANUSHUA K, BHANU SMS, et al. An extrusion detection system against botclouds[C]//Seventh International Conference on Communication Networks (ICCN-2013). 2013: 207-215.
[21] BADIS H, DOYEN G, KHATOUN R. A collaborative approach for a source based detection of botclouds[C]//International Symposium on Integrated Network Management. 2015: 906-909.
[22] JADHAV S, DUTIA S, CALANGUTKAR K, et al. Cloud-based android botnet malware detection system[C]//17th International Conference on Advanced Communication Technology. 2015: 347-352.
[23] HINTION G E, SALAKHUTDINOV R R. Reducing the dimensionality of data with neural networks[J]. Science, 2006, 313(28): 504-507.
[24] TAN Z Y. Detection of denial-of-service attacks based on computer vision techniques[D]. Sydney: University of Technology, 2013.
[25] FANG Z J,　FEI F C, FANG Y M, et al. Abnormal event detection in crowded scenes based on deep learning[J]. Multimedia Tools & Applications, 2016: 1-23.
[26] YUAN Z L, LU Y Q, XUE Y B. Droid detector: Android malware characterization and detection using deep learning[J]. Tsinghua Science & Technology, 2016, 21(1): 114-123.
[27] WANG Y, CAI W D, WEI P C. A deep learning approach for detecting malicious javascript code[J]. Security & Communication Networks, 2016, 51(8): 28656-28667.
[28] 韩晓光, 曲武, 姚宣霞, 等. 基于纹理指纹的恶意代码变种检测方法研究[J]. 通信学报, 2014, 35(8): 125-136.
HAN X G, QU W, YAO X X, et al. Research on malicious code variants detection based on texture fingerprint[J]. Journal on Communications, 2014, 35(8): 125-136.
[29] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[C]//The IEEE. 1998: 1-46.
[30] 敖道敢. 无监督特征学习结合神经网络应用于图像识别[D]. 广州: 华南理工大学, 2014.
AO D G. Integration of unsupervised feature learning and neural networks applied to image recognition[D]. Guangzhou: South China University of Technology, 2014.
[31] JIA Y Q, SHELHAMER E, DONAHUE J, et al. Caffe: convolutional architecture for fast feature embedding[C]//The 22nd ACM international conference on Multimedia. 2014: 675-678.

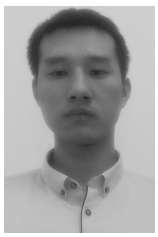
作者简介：



寇广（1983-），男，河南许昌人，解放军信息工程大学博士生、讲师，主要研究方向为网络安全态势感知、云安全等。



汤光明（1963-），女，湖南常德人，解放军信息工程大学教授、博士生导师，主要研究方向为信息安全、数据挖掘和体系对抗。



王硕（1991-），男，河南南阳人，解放军信息工程大学硕士生，主要研究方向为网络安全。

宋海涛（1990-），男，山东烟台人，解放军信息工程大学博士生，主要研究方向为网络安全。

边媛（1992-），女，陕西渭南人，解放军信息工程大学硕士生，主要研究方向为网络安全、信息隐藏等。