OMRDetector：一种基于深度学习的混淆恶意请求检测方法

杨秀璋 彭国军 罗 元 宋文纳 张 杰 操方涛

（武汉大学空天信息安全与可信计算教育部重点实验室 武汉 430072）

（武汉大学国家网络安全学院 武汉 430072）

摘 要 Web 恶意请求检测旨在快速精确地识别网络中的异常攻击. 当前 Web 恶意请求复杂多元、混淆明显，传统的检测技术存在过度依赖人工经验和规则库、易被绕过、误报率高，且忽略 Web 恶意请求特征语义关系等问题，无法第一时间感知未知网络攻击，无法对抗混淆和加密的恶意请求. 为此，本文设计并首次将三层 CNN-BiLSTM 融合注意力机制的模型应用于混淆恶意请求检测领域，并针对混淆的 Web 恶意请求特点进行模型优化，提出了一种基于深度学习的混淆恶意请求检测模型 OMRDetector. 该模型针对混淆恶意请求特征，提出抗混淆预处理方法，利用三层卷积神经网络（CNN）获取 Web 请求的局部特征，再引入双向长短时记忆（BiLSTM）网络捕获混淆恶意请求的长距离依赖关系以及上下文语义特征，通过注意力机制突出关键特征，最终由 Softmax 分类器计算恶意请求的检测结果，从而实现对抗混淆和检测未知恶意请求攻击. 实验结果表明，本文所提出的模型能有效检测出隐蔽性较高且带混淆加密的 Web 恶意请求，相比于现有模型在精确率、召回率、F_1 值和准确率上均有所提升，对应值分别为 97.734%、97.737%、97.735%和 97.754%. OMRDetector 模型恶意请求的漏报数量（1226 个）和误报数量（1244 个）均最少，其对混淆恶意请求的识别效果优于传统的机器学习模型（包括逻辑回归、决策树、朴素贝叶斯、支持向量机、随机森林和 AdaBoost）以及现有深度学习模型（包括 CNN、TextCNN、LSTM、BiLSTM、BiLSTM＋Attention）. 此外，本文结合 Web 请求恶意混淆的特点，归纳总结了 Web 恶意请求攻击常见的十二大类混淆方法，分别是大小写混淆、关键词复写混淆、含注释绕过混淆、特殊字符截断混淆、路径绕过混淆、URL 编码转换、特殊功能关键词混淆、组合规则绕过逻辑混淆、等价函数替换混淆、Base64 编码转换、DES 加密和流量魔改混淆，且最后两类混淆将用于评估 OMRDetector 模型对未知类型（无训练学习过程）的 Web 恶意请求的检测及对抗能力. 为进一步验证 OMRDetector 模型能更好地检测混淆恶意请求，本文进行详细的对比实验，最终证明 OMRDetector 能有效检测 Web 恶意请求的各种混淆类型并较好地感知未知类型的 Web 恶意请求攻击，具有一定的学术价值和应用前景.

关键词 混淆恶意请求检测；对抗混淆；卷积神经网络；双向长短时记忆；注意力机制；深度学习；网络安全

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2022.02167

OMRDetector: A Method for Detecting Obfuscated Malicious Requests Based on Deep Learning

YANG Xiu-Zhang PENG Guo-Jun LUO Yuan SONG Wen-Na ZHANG Jie CAO Fang-Tao

[1)] (*Key Laboratory of Aerospace Information Security and Trusted Computing of Ministry of Education, Wuhan University, Wuhan* 430072)

[2)] (*School of Cyber Science and Engineering, Wuhan University, Wuhan* 430072)

Abstract Web malicious request detection aims to identify attacks in the network quickly and accurately. At present, malicious Web requests are complex, heterogeneous, and obfuscated. Conventional detection methods suffer from multiple weaknesses, e.g., depending on experience

收稿日期：2021-09-14；在线发布日期：2022-05-18. 本课题得到 NSFC-通用技术基础研究联合基金（U1636107）、国家自然科学基金（62172308，61972297，62172144）资助. **杨秀璋**，博士研究生，中国计算机学会（CCF）学生会员，主要研究方向为网络与信息系统安全. E-mail：yangxiuzhang@whu.edu.cn. **彭国军**（通信作者），博士，教授，博士生导师，中国计算机学会（CCF）会员，主要研究领域为网络与信息系统安全. E-mail：guojpeng@whu.edu.cn. **罗 元**，博士研究生，主要研究方向为信息物理系统安全和移动系统安全. **宋文纳**，博士研究生，主要研究方向为信息安全和移动系统安全. **张 杰**，硕士研究生，主要研究方向为网络安全. **操方涛**，硕士研究生，主要研究方向为网络安全.

and rules, easy to be bypassed, high false-positive rates, fail to capture the semantic characteristics of malicious requests. Therefore, they cannot detect unknown network attacks in the first place and deal with obfuscated malicious requests. To this end, this paper designs and applies the three-layer CNN-BiLSTM fusion attention mechanism model to the field of obfuscated malicious request detection for the first time. Also, given the characteristics of obfuscated malicious requests, we optimize and propose an anti-obfuscation malicious request detection model named OMRDetector based on deep learning. This model adopts an anti-obfuscation preprocessing method for the characteristics of malicious requests. It uses a three-layer Convolutional Neural Network (CNN) to obtain the local characteristics of network requests. Then, a Bidirectional Long and Short-Term Memory (BiLSTM) network is designed to capture the long-distance dependence and contextual semantic features of the obfuscated malicious requests. Further, the attention mechanism extracts the key features. Finally, the softmax classifier calculates the detection results of malicious requests to combat obfuscation and detect unknown malicious request attacks. Experimental results show that our model can effectively detect highly stealthy and obfuscated malicious network requests. Compared with the conventional methods, the *Precision*, *Recall*, F_1-score, and *Accuracy* are improved, and the corresponding values are 97.734%, 97.737%, 97.735%, and 97.754%. In addition, the OMRDetector model has the lowest number of false negatives (1226) and false positives (1244). The performance is better than conventional machine learning models (including logistic regression, decision tree, naive Bayes, support vector machine, random forest, and AdaBoost) and existing deep learning models (including CNN, TextCNN, LSTM, BiLSTM, and BiLSTM + Attention) in recognition of obfuscated malicious requests. Moreover, this paper combines the characteristics of obfuscated malicious requests and summarizes twelve significant obfuscation types commonly used in malicious request attacks. Such types include case obfuscation, keyword copy obfuscation, comment bypass obfuscation, unique character truncation obfuscation, path bypass obfuscation, URL encoding, particular key word obfuscation, combination rules to bypass logic obfuscation, equivalent function substitution obfuscation, Base64 encoding, DES encryption, and traffic magic change obfuscation. Also, the last two obfuscation types will be used to evaluate the OMRDetector model's ability to detect and confront malicious Web requests of unknown types (no training and learning process). Meanwhile, this paper conducts comparative experiments to verify that the proposed OMRDetector model can detect obfuscated malicious requests better. Finally, the experimental results show that OMRDetector can effectively handle various obfuscation of malicious requests and better perceive unknown network attacks. Therefore, it has promising academic value and applications.

Keywords obfuscated malicious request detection; anti-obfuscation; convolutional neural network; bidirectional long and short-term memory; attention mechanism; deep learning; network security

1 引 言

Web 攻击常年危害网络空间安全,给社会造成巨大的安全隐患.根据国家互联网应急中心(CNCERT)[①]发布的报告,我国主要面临的 Web 攻击类型包括 SQL 注入、跨站脚本攻击(Cross Site Scripting,XSS)、服务器端请求伪造(Server-Side Request Forgery,SSRF)、远程代码执行、信息泄露、权限绕过、网站挂马等.面对日益复杂多样的 Web 攻击,如何构建一套高效精准的 Web 恶意请求检测系统至关重要.恶意请求检测(Malicious Request Detection)旨在快

① 2020 年中国互联网网络安全报告. http://www.cac.gov.cn/2021-07/21/c_1628454189500041.htm

速精确地识别网络中的异常攻击请求[1]. 传统方法主要通过规则库或黑白名单来识别 Web 恶意请求，但这些方法缺乏发现潜在未知攻击的能力，对混淆、加密和魔改的 Web 请求检测效果不理想，且过度依赖人工经验，需要定期更新规则，很容易被构造的混淆恶意请求绕过. 为了解决这些问题，基于机器学习的 Web 恶意请求检测成为了新的研究方向.

近年来，随着深度学习的兴起，利用深度神经网络(Deep Neural Network，DNN)识别恶意请求成为了新的热点，但这些方法对上下文语义的关联分析较弱，分类模型忽略了 Web 恶意请求特征的时序关系和语义强度，对混淆恶意请求检测效果较差，无法第一时间感知未知类型的 Web 恶意请求攻击. 而攻击者通常会利用不同方法和规则对 Web 恶意请求进行混淆、伪装、加密、魔改和欺骗处理，从而实现绕过防火墙和 Web 应用防护系统(Web Application Firewall，WAF)，实施可持续更强、隐蔽性更高、威胁性更大的攻击[2]. 混淆恶意请求(Obfuscated Malicious Request，OMR)是指通过某种变化将 Web 请求代码转换成一种功能等价的形式，从而躲避防火墙和 Web 应用防护系统的查杀，并能实施网络攻击的恶意请求. 因此，检测及对抗恶意混淆的 Web 请求方法已经成为网络安全研究领域的新型挑战和重点.

针对这些问题，本文设计并实现了基于 CNN-BiLSTM 和注意力机制的对抗混淆 Web 恶意请求的检测模型，称为 OMRDetector(Obfuscation Malicious Request Detector)模型. 本文贡献如下：

(1) 首次将三层 CNN-BiLSTM＋Attention 模型应用到混淆恶意请求检测领域，并针对混淆恶意请求特点进行模型改进，设计并实现了 OMRDetector 模型. 该模型能有效对抗混淆，检测未知类型的 Web 恶意请求攻击. OMRDetector 模型提出了抗混淆预处理方法，构建卷积神经网络(Convolutional Neural Network，CNN)获取 Web 请求的局部特征，引入双向长短时记忆(Bidirectional Long and Short-Term Memory，BiLSTM)网络捕获长距离依赖关系和语义信息，从而解决单一卷积神经网络无法获取长距离语义特征和单一循环神经网络梯度消失问题；最后通过注意力机制进行权重加成，从而突出字符级混淆、流量加密、编码转换和字符绕过等特征，再由 Softmax 分类器实现恶意请求检测.

(2) 针对 Web 恶意请求混淆和绕过 WAF 特点，总结归纳了十二类常见混淆方法，分别是大小写混淆、关键词复写混淆、含注释绕过混淆、特殊字符截断混淆、路径绕过混淆、URL 编码转换、特殊功能关键词混淆、组合规则绕过逻辑混淆、等价函数替换混淆、Base64 编码转换、DES 加密和流量魔改混淆. 首次对混淆、加密和魔改的 Web 恶意请求进行检测，开展多种方法的对比实验，并尝试对未知类型的混淆和加密恶意请求进行识别，旨在为科研工作者和安全技术人员检测混淆加密流量提供新思路.

(3) 本文方法对混淆的 Web 恶意请求检测具有良好的效果，相比于传统基于规则、机器学习和深度学习的模型，本文模型的精确率、召回率、F_1 值和准确率均有所提升，且误报数量和漏报数量均最低，能有效识别常见混淆和加密类型的 Web 恶意请求，较好地感知未知类型的 Web 恶意攻击.

需要注意，本文所述的“混淆”具有两层含义. 其中，狭义的混淆是指利用某种字符处理(含增加、删除、修改)将 Web 恶意请求代码转换成另一种等价的形式，从而有效躲避检测；广义的混淆是指通过字符级混淆、编码转换、加密和魔改等不同方式对恶意请求处理来躲避检测. 本文将在 2.3 小节和表 2 中给出 12 种混淆恶意请求的详细定义、分类和示例.

本文第 2 节介绍相关工作；第 3 节介绍模型设计与实现过程；第 4 节是实验过程和结果分析；第 5 节对本文方法进行总结和展望.

2　相关工作

本节将从三个方面介绍 Web 恶意请求检测的相关工作，包括基于规则、机器学习和深度学习的 Web 恶意请求检测方法.

2.1　基于规则的 Web 恶意请求检测

恶意请求检测作为一种有效的安全防护技术，在互联网中被广泛应用，如 Web 应用防护系统(WAF)、入侵检测系统(Intrusion Detection System，IDS)等[3]. 传统的 Web 恶意请求检测主要借助专家知识和人工经验实现，通过定义规则库和黑白名单来提取流量数据特征，再利用统计学相关的算法对 Web 请求进行分析，并设计检测模型来判断是否存在恶意 Web 攻击[4].

本文主要检测恶意 HTTP 请求，其核心数据为 URL 构造的 Payload. 恶意 URL 请求检测基本构造形式如下，由传输协议、用户名、密码、域名、端口

号、路径、文件名、查询和参数等组成.

```
scheme:[//[user[:password]@]host[:port]]
[/path][?query][#fragment]
```

其中,恶意 Web 请求的 URL 样本示例为

```
①SQL 注入
http://xxxx/main/show.asp?code=15' union all
select 1,2,3,null --
②XSS 攻击
http://xxxx/news/action.php?user=zs<script>
alert(123)</script>
③SSRF 攻击
http://xxxx/index.php?url=file:///var/www/html/
admin.php
```

正常 Web 请求的 URL 样本示例为

```
http://xxxx/main/show.asp?code=15
http://xxxx/url.php?name=admin&age=10
```

早期的 Web 恶意请求检测主要通过编写规则库或正则表达式来匹配特征,进而识别恶意请求. Denning[5]提出一种入侵检测模型,旨在通过检测系统的审计记录和度量规则来识别异常行为. Ma 等人[6]基于 URL 特征词、主机功能和在线学习算法来识别恶意站点. 潘峰等人[7]使用两种基于统计的异常检测技术识别网络入侵. Sperotto 等人[4]通过 IP 流量特征来划分攻击类别,分析了僵尸网络、漏洞扫描、蠕虫攻击和 DDoS 攻击的检测技术. 随后,在企业界出现了更多基于规则的入侵检测方法. Chimera[8]是一种用于流式网络流量分析的查询语言,它结合入侵检测系统和 SQL 语法及规则的优点. vNIDS[9]是一种新型网络入侵检测系统架构,它通过将虚拟 NIDS 配置为微服务并采用程序切片来检测程序的逻辑,从而保障网站的弹性安全. 然而,这类方法十分依赖规则库和专家经验,需要不断更新规则库,缺乏对混淆和加密的 Web 恶意请求进行检测,并且攻击者可以设计复杂的混淆及魔改请求绕过规则. 部分混淆恶意请求如下所示:

```
①大小写混淆
http://xxxx/main/show.asp?code=15' uNIon all
sELeCt 1,2,3,null --
②关键词复写混淆
http://xxxx/xss/action.php?user=zs
<sc<script>ript>alert(123)</sc<script>ript>
③URL 编码转换
http://xxxx/cmd/test.php?cmd=15%27+union+all+
select+1%2c2%2c3%2cnull+--
④路径绕过混淆
http://xxxx/xss/top.php?file=../../../../../../
../../../../var/log/apache/error.log
```

由于这些混淆操作会改变原始恶意请求的特征,从而导致规则库失效,不能第一时间感知恶意混淆的 Web 请求攻击. 此外,经过特定混淆构造的 Web 恶意请求很容易绕过入侵检测系统,造成严重的网络攻击事件. 随着机器学习技术的推广,网络安全领域也逐渐开启了基于机器学习的 Web 恶意请求检测研究,它具有更好的鲁棒性和准确率.

2.2 基于机器学习的 Web 恶意请求检测

当前,国内外通过机器学习识别 Web 恶意请求已取得一定成果. 这类方法旨在构建机器学习模型,对所提取的 Web 恶意请求特征进行训练学习,再对未知 Web 请求的恶意性进行判定. 常见的机器学习模型包括逻辑回归、SVM、随机森林、K-Means 等. 基于机器学习的 Web 恶意请求检测模型框架如图 1 所示,训练样本和测试样本经过数据预处理和特征提取后,会被机器学习模型训练,训练好的模型再进行分类预测,最终输出检测结果.

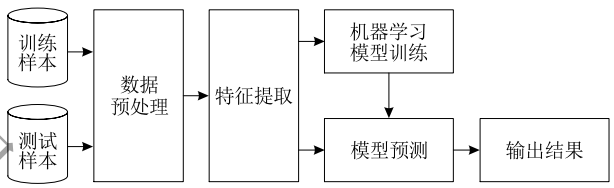

图 1 基于机器学习的 Web 恶意请求检测框架图

Abbes 等人[10]通过构建决策树分类器实现异常入侵检测. Khan 等人[11]提出一种融合 SVM 和层次聚类的入侵检测算法. Muda 等人[12]通过 K-Means 和朴素贝叶斯算法实现入侵检测,并在 KDD Cup 99 数据集上实验,有效识别出 DoS、R2L、U2R 和 Probe 四种类型的异常攻击. 陈春玲等人[13]通过 N-gram 和 TF-IDF 完成 URL 分词,再调用逻辑回归实现恶意请求识别. Fu 等人[14]通过聚类算法和频域特征构建 Whisper,从而实现对恶意流量的实时检测,提高准确度和吞吐量. 池亚平等人[15]通过 SVM 和 Adaboost 算法完成入侵检测,并在 NSL-KDD 数据集上进行训练与测试. Perdisci 等人[16]使用聚类算法对恶意样本的 HTTP 流量结构的相似性进行聚类分析. Zhang 等人[17]提出了一种基于 PU-Learning 的恶意 URL 检测算法,从而发现潜在的 URL 攻击. Li 等人[18]提出一种可持续学习的集成学习方法,通过重复学习历史知识来提升恶意请求检测的效果.

相对于基于规则的 Web 恶意请求检测,机器学习方法能提升一定程度的准确率,并降低误报率,但由于其属于浅层学习,缺乏深层语义知识和上下文

依赖分析，在检测复杂、混淆和加密的海量恶意请求时具有一定的局限性，且无法第一时间感知混淆后的 Web 恶意请求特征. 因此，传统的机器学习方法难以高效精准地对混淆 Web 恶意请求攻击进行检测及对抗.

2.3　基于深度学习的 Web 恶意请求检测

随着深度神经网络被应用于各安全领域，基于深度学习的 Web 恶意请求检测方法逐渐出现[19]. 它一方面能挖掘 Web 请求深层次的语义关系和关键特征，提升检测精准度，另一方面能避免因人工经验造成的误报和漏报，提升模型的鲁棒性. 2015 年，Atienza 等人[20]利用神经网络分析 HTTP 流量数据并进行可视化展现，对 HTTP dataset CSIC 2010 数据集实现 Web 攻击检测实验. Tang 等人[21]通过深度神经网络实现网络入侵检测，并取得良好的检测效果. Liang 等人[22]通过构造深度学习模型来区分异常请求和正常请求，并优于传统基于规则的 Web 攻击检测方法. 由于卷积神经网络在抽取局部特征方面效果良好，安全研究者也将其应用于 Web 恶意请求检测领域. 2017 年，Wang 等人[23]利用 CNN 实现入侵检测系统的流量分类，其将流量转换为图像后实施恶意请求检测. Saxe 等人[24]提出 eXpose，这是一种基于字符级 CNN 的恶意 URL 检测方法. Zhang 等人[25]提出一种基于单词级的 CNN 方法并对恶意 URL 进行分类，实验结果表明 CNN 具有较高的检测率和较低的误报率. Le 等人[26]优化了字符级卷积神经网络，提出 URLNet 方法来识别恶意请求. 崔艳鹏等人[27]整合了 HTTP 请求的 URL 及参数，再通过卷积神经网络进行 Web 恶意请求检测. Yu 等人[1]利用 TextCNN 深度学习方法进行恶意请求检测，并与其它模型进行对比. Sun 等人[28]利用 CNN 模型提取网络流量特征并构建 DL-IDS 入侵检测系统.

尽管基于 CNN 的 Web 恶意请求检测方法分类效果优于传统机器学习方法，但它忽略了特征序列和文本特征的先后顺序，无法解决长距离依赖、梯度消失和梯度爆炸等问题. 为了解决这些问题，长短时记忆(Long and Short-Term Memory，LSTM)网络被研究者提出，它可以联系上下文突出文本序列信息，LSTM 通过记忆状态单元保留重要信息，遗忘次要信息，从而有效阻止梯度消失或梯度爆炸问题[29]. Staudemeyer[30]应用 LSTM 检测网络攻击，并在 KDD Cup 99 数据集上验证了 LSTM 网络对入侵检测的有效性. Afzal 等人[31]提出一种名为 URLdeepDetect 的混合深度学习方法，利用 LSTM 网络检测恶意 Web 请求. 於帮兵等人[32]将 LSTM 网络应用在工业控制系统入侵检测领域. 赖训飞等人[33]提出一种基于实体嵌入和长短时记忆网络相结合的网络入侵检测方法. 刘月峰等人[29]融合 CNN 和 BiLSTM 网络完成入侵检测. 石乐义等人[34]基于信息熵和长短时记忆网络进行工业入侵检测，从而获取更高的准确率和较低的漏报率.

然而，上述方法主要是对非混淆恶意请求进行研究，难以实现对混淆、伪装、加密和欺骗处理后的 Web 恶意请求进行识别，并且无法第一时间感知未知类型的混淆或加密恶意请求攻击. 此外，混淆恶意请求具有极强的隐蔽性和威胁性，广泛存在于真实的网络攻击中，并带来了严重的威胁. 因此，针对混淆 Web 恶意请求检测的研究迫在眉睫. 本文将字符级混淆、编码转换、加密和魔改广义地归纳为 Web 恶意请求的混淆变化，从而更好地检测和感知不同类型的混淆恶意请求. 其中，字符级混淆是指通过增加、删减和修改字符内容，不改变原始字符含义的混淆，此时的脚本解释器仍然能理解代码的内容，常用的字符级混淆包括关键词复写、大小写替换、插入特殊字符等. 编码转换是指按预定义方法将信息从一种形式转换为另一种形式的过程，Web 请求需要执行解码操作才能理解代码的具体含义，常用的编码转换方式包括 URL 编码和 Base64 编码. 加密是指以某种特殊的算法改变原有的信息数据，信息在解密后才能理解其具体含义，Web 请求常用的加密算法如 DES 和 AES 加密算法. 魔改是网络安全领域中的技术术语，旨在通过组合各种规则和变化来实现恶意代码的混淆，从而躲避检测并完成攻击，本文将至少融合 2 种以上的混淆变化定义为魔改 Web 请求，包括组合规则绕过逻辑混淆和流量魔改混淆.

下面是经过流量加密或编码转换的 Web 恶意请求示例，它们可以通过各种 Web 渗透工具生成，其中 Base64 编码是最常用的流量编码方式之一，被攻击者所广泛使用，其威胁性巨大；而流量魔改混淆是指融合多种混淆规则，利用渗透工具发起 Web 恶意请求攻击的方法. 由于加密或魔改后的 Web 恶意请求具有极强的隐蔽性，因此工业界典型的 WAF 和在线防火墙对这类 Web 请求检测效果较差，当前基于深度学习的 Web 恶意请求检测算法也无法有效区分经过其处理后的正常及恶意请求.

①Base64 编码转换
http://xxxx/main/show.asp?code=PHNjcmlwdD5hbGVydCgndnVsbmVyYWJsZScpPC9zY3JpcHQ+
②DES 加密
http://xxxx/main/action.php?user=U2FsdGVkX1+bLrBm9UPqbQH4JKx9702ickLPbI9gZyKFdG9FH2mGGhwZPE48JcpOqq/u6jNh/Xg=
③URL 编码转换
http://xxxx/cmd/test.php?cmd=15%27+union+all+select+1%2c2%2c3%2cnull+--
④流量魔改混淆
http://xxxx/prefix/CNab')+UnIon+All+SelEct+1,2,3,4,5,Group_Concat(distinct+username,0x2b,password,0x2b,vozrast),7,8+From+users#

加密或魔改前的流量信息如下所示,可以发现,它们均能实现某种恶意攻击行为,经过加密或魔改处理后,其隐蔽性和威胁性更强.

①Base64 编码转换前
http://xxxx/main/show.asp?code=<script>alert('vulnerable')</script>
②DES 加密前
http://xxxx/main/action.php?user=-9 union select null,null,null,4,5
③URL 编码转换前
http://xxxx/cmd/test.php?cmd=15' union all select 1,2,3,null --
④流量魔改混淆前
http://xxxx/prefix/CNab')+union+all+select+1,2,3,4,5,username,7,8+from+users#

由于恶意 Web 请求上下文具有一定的关联性,即使是经过加密、编码或魔改混淆处理,前后文本时序也会影响检测的结果,CNN 能够较好提取 Web 请求的局部特征,BiLSTM 适合学习长期依赖和较长跨度的序列特征,综合 Web 恶意请求的多样性和上下文关联性,并且注意力机制通过计算关键特征的重要程度能有效识别字符信息,从而提升分类效果.因此,本文结合抗混淆预处理,提出一种基于 CNN-BiLSTM 和注意力机制的对抗混淆 Web 恶意请求的检测模型 OMRDetector,并与其它各模型进行详细的对比实验,对未知混淆类型的恶意请求进行检测,深入挖掘本文模型的优点.表 1 详细对比了现有 Web 恶意请求检测方法与本文工作的特点,本文模型在语义捕获和对抗混淆效果上表现良好.

表 1 现有 Web 恶意请求检测方法与本文工作对比

分类	相关工作	模型	数据集	捕获语义	对抗混淆
规则统计	Sperotto 等人[4]	IP 流量特征统计	自定义 URL 数据集	×	×
	Denning[5]	审计记录、规则度量	—	×	×
	Ma 等人[6]	URL 特征词统计	采集实时 URL 数据集	×	×
	潘峰等人[7]	两类统计算法	DARPA 1999	×	×
机器学习	Muda 等人[12]	*K*-Means、朴素贝叶斯	KDD Cup 1999	×	×
	陈春玲等人[13]	逻辑回归	Secrepo、Github 数据集	×	×
	池亚平等人[15]	SVM、Adaboost	NSL-KDD	×	×
	Perdisci 等人[16]	Behavioral Clustering	Malware Dataset HTTP	×	×
	Zhang 等人[17]	PU-Learning	自定义 URL 数据集	×	×
深度学习	Atienza 等人[20]	神经网络	HTTP dataset CSIC 2010	×	×
	Wang 等人[23]	CNN	USTC-TFC2016	×	×
	Saxe 等人[24]	字符级 CNN(eXpose)	VirusTotal URL	×	×
	Staudemeyer[30]	LSTM	KDD Cup 1999	✓	×
	石乐义等人[34]	CNN-BiLSTM+信息熵	工控系统入侵检测数据	✓	×
本文方法	OMRDetector (三层 CNN-BiLSTM+Attention+抗混淆预处理)		HTTP dataset CSIC 2010 Github 采集 Web 请求 安全工具生成 Web 请求	✓	✓

3 模型设计与实现

3.1 模型总体框架

本文构建基于 CNN-BiLSTM 和注意力机制的混淆 Web 恶意请求检测及对抗模型,称为 OMRDetector,该模型总体框架如图 2 所示.

OMRDetector 模型包括输入层、抗混淆预处理层、词嵌入层、卷积层、池化层、双向长短时记忆层、注意力机制、全连接层和输出层.首先,将包含正常 Web 请求和带恶意混淆的 Web 请求语料进行抗混淆预处理,接着利用 N-gram 和 Embedding 向量化处理原始数据并嵌入输入层.然后,设计三层卷积神经网络提取 Web 请求的代表性局部特征,再通过 BiLSTM 层捕获长距离依赖关系.最后,利用注意力机制计算各属性特征的重要性,突出混淆加密恶意请求的关键特征,赋予相关权重传入全连接层,并由 Softmax 分类器计算混淆 Web 恶意请求检测的最终结果.

OMRDetector 整个算法的流程如下所示,该模

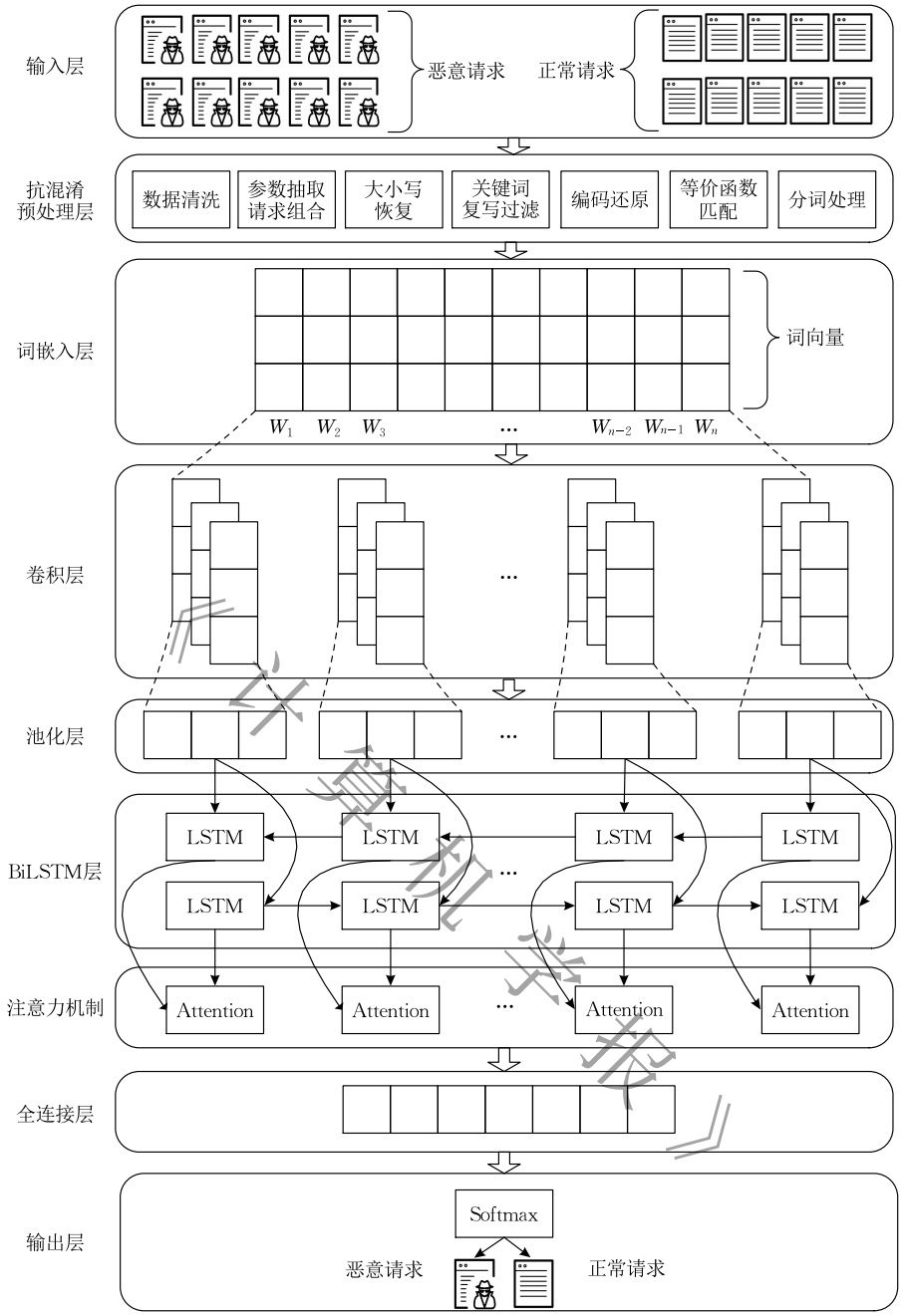

图 2　OMRDetector 模型总体框架图

型的优点在于能够较好地捕获混淆、加密和魔改恶意请求的上下文语义信息和长距离依赖特征，有效识别正常请求和恶意请求.

算法 1.　OMRDetector混淆恶意请求检测模型.

输入：Web 请求数据集 $\boldsymbol{D}=\{(X^{(k)},y^{(k)})\}_{k=1}^{n}$

输出：Web 请求检测预测类别 $\boldsymbol{C}(c^{(1)},\cdots,c^{(n)})$

1. $\boldsymbol{V}(v^{(1)},\cdots,v^{(n)})=Anti\_obfuscation\_process(\boldsymbol{D})$
2. $Initialize\_model(\boldsymbol{V})$
3. FOR each *epoch* DO
4. 　FOR each *batch* DO
5. 　　① 三层 CNN 模型提取文本局部特征
6. 　　(a) $\boldsymbol{H}=Convolution\_processing(\boldsymbol{V})$
7. 　　(b) $\boldsymbol{S}=Pooling\_processing(\boldsymbol{H})$
8. 　　② BiLSTM 模型学习时序特征
9. 　　(a) $h_1=Forward\_LSTM\_processing(\boldsymbol{S})$
10. 　　(b) $h_2=Backward\_LSTM\_processing(\boldsymbol{S})$
11. 　　(c) $\boldsymbol{Y}=BiLSTM(h_1,h_2)$
12. 　　③ Attention 机制突出关键特征
13. 　　(a) $\boldsymbol{U}=Attention\_processing(\boldsymbol{Y})$
14. 　$loss^{*}=Loss\_function(\boldsymbol{U})$
15. 　IF $loss^{*}>loss$ THEN
16. 　　$loss\leftarrow(loss^{*})$
17. 　END IF

18. Softmax 计算分类结果并预测输出 C
19. END FOR
20. END FOR
21. RETURN C

3.2 抗混淆预处理

本文通过分析混淆、编码、加密和魔改后的恶意 Web 请求的特点，将其归纳为常见的十二大类混淆恶意请求，其混淆类型、定义和示例如表 2 所示.

表 2 常见混淆恶意请求

编号	混淆类型	定义	分类	示例
①	大小写混淆	将关键词转换成大小写混合的形式，从而绕过检测规则	字符级混淆	SeLEct
②	关键词复写混淆	通过重复填充关键词来躲避过滤规则，关键词即使被入侵检测系统过滤仍能构造恶意请求并实施攻击	字符级混淆	〈sc〈script〉ript〉
③	含注释绕过混淆	利用注释字符来构造绕过规则的恶意请求	字符级混淆	/*!SELECT*/
④	特殊字符截断混淆	利用截断字符来躲避防火墙检测. 在文件上传漏洞攻击中，某些函数会将“00”截断当作结束符处理，从而绕过文件类型检测	字符级混淆	php. php%001. jpg
⑤	路径绕过混淆	通过构造特定的路径匹配来实施网络攻击，常用于获取特殊的日志文件或关键信息	字符级混淆	../../../../var/log/error. log
⑥	URL 编码转换	将字符串转换为 URL 编码，从而躲避检测. 比如将“加号”转换为“%2B”后能躲避防火墙或在线 WAF 的查杀	编码转换	select%201%2Cnull
⑦	特殊功能关键词混淆	利用某些关键字来替换易被检测的功能. 比如关键字“between”可以固定范围来替换 SQL 注入的数字判断	字符级混淆	between 1 and 1
⑧	组合规则绕过逻辑混淆	融合多种混淆规则来绕过网站的业务逻辑，从而实施网络攻击	魔改	ascii(substr((sEleCt FroM),1,1))>1
⑨	等价函数替换混淆	将某些容易被检测的函数替换为冷门的函数或正常网站必须要使用的函数，从而躲避防火墙的查杀	字符级混淆	left('password',1)
⑩	Base64 编码转换	将字符串转换为 Base64 编码形式，提升数据在传输中的兼容性	编码转换	code=PHNjcmlwdD5hbGVydCgndnVsbmVyYWJsZScpPC9zY3JpcHQ+
⑪	DES 加密	利用 DES 对称加密算法对 Web 恶意请求实施加密，从而躲避检测，接收到请求后再通过解密并构造 Payload，完成网络攻击	加密	user=U2FsdGVkX1+bLrBm9UPqbQH4JKx9702ickLPbI9gZyKFdG9FH2mGGhwZPE48JcpOqq/u6jNh/Xg=
⑫	流量魔改混淆	Web 攻击常用术语，旨在通过组合各种规则、编码和变化来实现恶意代码的混淆，从而躲避检测并完成攻击，该类混淆通常为网络渗透工具生成	魔改	poc=1) and if((ascii(substr((SeleCt flag FroM flag),"+str(pos)+",1))="+str(i)+"),sleep(3),0) --

这十二大类混淆恶意请求分别是大小写混淆、关键词复写混淆、含注释绕过混淆、特殊字符截断混淆、路径绕过混淆、URL 编码转换、特殊功能关键词混淆、组合规则绕过逻辑混淆、等价函数替换混淆、Base64 编码转换、DES 加密和流量魔改混淆. 它们通过不同的方法和规则对恶意请求进行混淆加密处理，从而实现绕过防火墙或 Web 应用防护系统，并实施网络攻击.

针对混淆和加密 Web 恶意请求的特点，本文提出一种抗混淆预处理方法，整个处理流程如图 3 所示，共包括七个核心步骤，每个步骤有对应预期解决的混淆类型，其类型编号对应于表 2. 比如大小写恢复阶段主要解决“①”类混淆，而相对复杂的“⑤路径绕过混淆”、“⑧组合规则绕过逻辑混淆”、“⑪DES 加密”、“⑫流量魔改混淆”将通过本文提出的 OMR-Detector 模型综合解决. 抗混淆预处理七个步骤的具体处理过程如下：

(1) 数据清洗. 将所采集的 Web 请求数据集进行数据清洗，结合不同类型的混淆生成对应规则，并对数据集的 Web 请求进行数据标注，筛选出不同类型的恶意请求. 接着过滤掉重复、错误、长度过短或过长的 Web 请求.

(2) 参数抽取及请求组合. 从 Web 请求中抽取 GET 和 POST 请求的 URL 及 Payload 的请求参数，

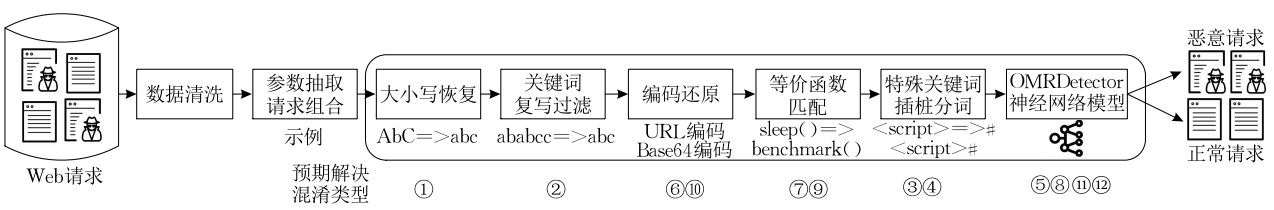

图 3 抗混淆预处理流程

接着组合成新的 Web 请求，并过滤掉 Web 请求的域名，因为它们对 Web 请求恶意性检测影响较小.

(3) 大小写恢复. 所有 Web 请求统一转换成小写形式，比如将“SeLEct”转换成“select”.

(4) 关键词复写过滤. 对所有 Web 请求的复写关键词进行过滤清洗，比如将“〈sc〈script〉ript〉”转换成“〈script〉”.

(5) 编码还原. 判断 Web 请求是否包含编码转换，再实施 URL 编码和 Base64 编码还原. 即将所有包含 URL 编码转换的 Web 请求还原成初始形态，比如“%20”对应“空格”，“%2B”对应“加号”，“%2C”对应“逗号”，“%3D”对应“等号”.

(6) 等价函数匹配. 自定义对照表将 Web 恶意请求中常见的等价函数进行匹配还原，比如“hex()”等价于“ascii()”，“mid()”等价于“substring()”，“sleep()”等价于“benchmark()”. 表 3 展示了常用等价函数对照表，其第 1 列表示易被识别的功能函数，第 2 列表示常用混淆的等价函数. 在混淆 Web 恶意请求中，通常会将某些易被识别的功能函数转换为隐蔽性相对更高的等价函数，比如将“sleep()”转换为“benchmark()”来躲避 WAF 的查杀. 在该阶段中，本文模型会结合上下文语义信息和共现概率按照对照表将等价函数进行还原.

表 3　常用等价函数对照表

易被识别的功能函数	常用混淆的等价函数
hex()	ascii()
bin()	ascii()
mid()	substring()
substr()	substring()
substr()	left()
substr()	right()
sleep()	benchmark()
concat_ws()	group_concat()
concat()	group_concat()
updatexml()	extractvalue()

(7) 特殊关键词插桩分词处理. 由于混淆恶意请求通常包含特殊关键词或特殊字符，本文利用插桩方式在这些关键词前后补充“#”，进一步凸显其恶意性. 接着通过 N-gram 进行分词处理，利用 Embedding 层将 Web 请求数据转换为词向量，随后传入 OMRDetector 中的神经网络模型进行恶意请求检测.

需要注意，为了确保抗混淆预处理不会对恶意请求的关键特征提取产生影响，并尽可能还原真实场景的 Web 恶意请求，本文不对加密算法执行解密操作，而是利用深度学习去挖掘加密流量上下文之间的语义特征，更好地检测未知混淆及加密类型的恶意请求. 因此，抗混淆预处理主要是通过多种方法对提取的特征进行优化，更好地服务后续的检测模型. 此外，由于字级嵌入一定程度会损失 Web 恶意请求的语义信息，并且恶意请求构造复杂，“/”、“&”、“=”、“.”等字符均是恶意 URL 的常见特殊字符，采用英文分词并转换成词向量会丢失上下文语义关系，影响检测效果，本文最终选择 N-gram 和 Embedding 进行词嵌入表示.

3.3　卷积神经网络模型

卷积神经网络(CNN)旨在利用卷积核滚动来提取局部特征，常被应用于处理分类问题，典型模型由输入层、卷积层、池化层和全连接层四部分组成. 在网络攻击检测领域，CNN 模型在抽取局部特征方面具有良好的性能，本文拟构造三层 CNN 模型来提取 Web 请求具有代表性的局部特征，所提取混淆恶意请求常见的特征包括“admin”、“union”、“〈script〉”、“alert”、“select”、“null”等. 假设存在如下所示的 XSS 攻击恶意请求示例.

```
http://xxxx/news/action.php?
code=〈script〉alert('xss')〈/script〉
```

在抗混淆预处理中，通过基于特殊关键词插桩的分词处理，将有效提取特征词，比如“〈script〉”. 此外，由于特殊字符是 Web 恶意请求的重要组成，该插桩分词方法也将保留关键字符，比如“?”. 为更好地理解卷积神经网络提取 Web 恶意请求关键局部特征的过程，本文将该示例去除无关域名后的核心特征归纳为 9 个，即“news”、“action.php”、“?”、“code”、“=”、“〈script〉”、“alert”、“('xss')”、“〈/script〉”，其卷积神经网络提取关键特征的过程如图 4 所示.

由图可知，CNN 模型可以利用卷积层完成 Web 恶意请求的特征提取任务，通过构建卷积核对输入的词向量矩阵进行卷积操作，其滤波器将提取不同区域的关键特征，比如“action.php”、“〈script〉”、“alert”、“('xss')”、“〈/script〉”等. 卷积层的计算公式如式(1)所示.

$$h_i^d = f(\omega_d \times \boldsymbol{V}_i + b_d) \tag{1}$$

其中，Web 请求 URL 特征词对应的词向量为 $\boldsymbol{V}_i$，且 $\boldsymbol{V}_i \in R^{n\times k}$，$n$ 表示特征词数量，k 表示词向量维度，ω_d 表示尺寸为 d 的卷积核，b_d 表示偏置项，采用的激活函数 f 通常为 $ReLU$，最终得到新的特征 h_i^d. 通过卷积处理后，映射得到局部特征集合 $\boldsymbol{H}_d$，其计算公式

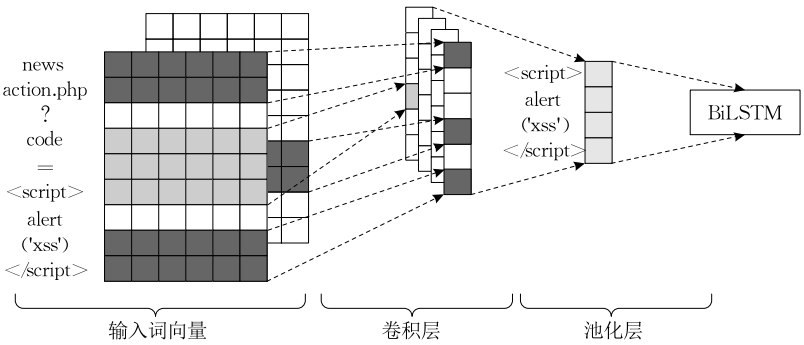

图 4 CNN 模型网络结构及 Web 恶意请求特征提取过程

如式(2)所示.

$$\boldsymbol{H}_d=\{h_1^d,h_2^d,\cdots,h_{n-d+1}^d\} \tag{2}$$

接着执行池化层操作,将卷积层输出的恶意请求特征集进行采样处理,并计算局部特征最优解,从而降低数据维度并保持核心特征属性.该示例计算出的最优解代表着 XSS 恶意请求攻击的局部最关键特征,对应图 3 的池化层输出,即"〈script〉"、"alert"、"('xss')"、"〈/script〉",它们经常同时出现并象征着 XSS 攻击请求.同时,本文采用 MaxPooling 技术对局部特征 $\boldsymbol{H}_d$ 进行池化处理,计算公式如式(3)所示.

$$M_i=\max\{\boldsymbol{H}_d\} \tag{3}$$

池化层提取局部重要特征后,会将所得到的特征添加至后续网络模型,组合形成输出向量 $\boldsymbol{S}$,如式(4)所示.

$$\boldsymbol{S}=\{M_1,M_2,\cdots,M_n\} \tag{4}$$

最终,经过三层 CNN 模型处理后,其输出向量 $\boldsymbol{S}$ 会作为下一阶段 BiLSTM 模型的输入,它们将融合共同完成针对混淆 Web 恶意请求的检测及对抗任务.需要注意的是,本文实验的真实处理过程采用 N-gram 分词,它不同于经过英文分词如图 4 所示的 9 个特征.采用 N-gram 方法分词,一方面是为了更好地保存上下文语义信息,提取 Web 恶意请求中常见的特殊字符;另一方面 CNN 模型同样能捕获 N-gram 分词后的局部关键特征,它们也能有效表征不同类型的混淆和加密恶意请求.

3.4 BiLSTM 模型

长短时记忆网络(LSTM)是循环神经网络的一种变体,通过记忆状态单元保留重要信息,遗忘次要信息,从而有效阻止梯度消失或梯度爆炸问题.双向长短时记忆网络(BiLSTM)模型由前向 LSTM 和后向 LSTM 组合而成,旨在从前向和后向对句子进行编码,提取上下文语义特征并捕获长距离依赖关系,其网络结构如图 5 所示.

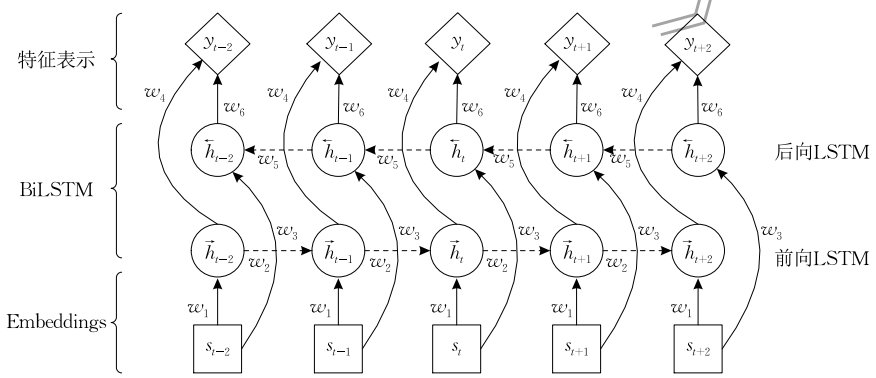

图 5 BiLSTM 模型网络结构图

本文通过 BiLSTM 模型能够更好地捕捉恶意混淆的 Web 请求的长距离依赖关系和上下文语义信息.例如,它既能从常规恶意请求中发现"union"和"select"同时用于 SQL 注入攻击,"〈script〉"和"alert"同时用于 XSS 攻击,它们存在较强的语义关联;也能从混淆恶意请求中发现"file"、"../"和"log"这种长距离依赖关系,它们常用于路径绕过混淆的远程代码执行攻击;还能挖掘加密恶意请求中的上下文语义特征和共现特征.

BiLSTM 模型从前后两个方向对混淆 Web 恶意请求的特征词深度训练,并进行细粒度的恶意请求分类.该模型的计算公式如下所示:

$$\begin{cases}\overrightarrow{h}_t = f(w_1 \cdot \boldsymbol{s}_t + w_2 \cdot \overrightarrow{h}_{t-1}) \\ \overleftarrow{h}_t = f(w_3 \cdot \boldsymbol{s}_t + w_5 \cdot \overleftarrow{h}_{t+1}) \\ \boldsymbol{y}_t = g(w_4 \cdot \overrightarrow{h}_t + w_6 \cdot \overleftarrow{h}_t)\end{cases} \tag{5}$$

式中，$\overrightarrow{h}_t$表示 t 时刻前向 LSTM 层的状态，$\overleftarrow{h}_t$表示 t 时刻后向 LSTM 层的状态，对应 Web 恶意请求文本中的上下文特征信息；$\boldsymbol{s}_t$表示 t 时刻输入的词向量；w_1至 w_6表示权重参数；f 和 g 表示激活函数，通常包括 sigmoid、tanh 等；$\boldsymbol{y}_t$是双向长短时记忆网络的最终输出结果.

本文将卷积神经网络的池化层输出序列转化为 BiLSTM 模型的输入，从而构建 CNN-BiLSTM 模型. 该模型结合上下文语义知识来提取混淆、加密及魔改后的 Web 恶意请求文本局部特征，一方面解决局部特征丢失问题，另一方面保障对长距离依赖和前后语义关联的特征，有效提升后续对抗和检测混淆恶意请求的实验效果. 同时，为防止出现过拟合现象，本文引入 Dropout 机制随机抽取数据进行训练，后续还会进一步融合注意力机制完成最终的混淆恶意请求检测任务.

需要注意，由于混淆、加密及魔改后的 Web 恶意请求命名规则复杂，缺乏海量的预训练语料，而部分特征又存在语义关联，关键特征的距离间隔较远，并且企业对 Web 恶意请求检测的时间性能要求较高. 因此，本文综合多个因素考虑，最终选择 BiLSTM 结合注意力机制进行长距离依赖和上下文语义提取，而未选择需要预训练且消耗较多计算资源的 Bert 模型[35-36]. 此外，RNN 模型在处理富含长距离依赖和上下文时序数据时，其效果较好，甚至优于 Transformer 和 CNN 模型[37]，并能解决梯度爆炸和梯度消失的问题. 综上，本文选择更适用于该 Web 请求数据集的 BiLSTM 模型.

3.5　注意力机制

注意力(Attention)机制旨在将有限的注意力资源分配到关键信息点上，从而增强文本特征词与输出结果的关联性，被广泛应用于机器翻译、自然语言处理领域[37]. 本文通过引入注意力机制有效增强神经网络模型对混淆 Web 恶意请求重要特征的关注，提高关键特征词对于整个 Web 恶意请求文本分类的贡献度，并赋予混淆、加密及魔改特征重要权重，从而区分 Web 请求文本中的重要性和恶意性.

本文在 CNN-BiLSTM 模型输出后融合注意力机制，如果特征词对 Web 请求语义的贡献较大，则赋予更大的权重，反之赋予更小的权重，通过这种方式有效地提高关键特征词对恶意请求分类效果的影响，从而更全面地把握输入 Web 请求的核心内容. 比如上述示例中的“union”、“select”和“null”是 SQL 注入攻击的关键特征，“〈script〉”、“alert”和“〈/script〉”是 XSS 攻击的关键特征；而“code”、“name”、“page”和“index.php”通常出现在正常 Web 请求中，它们属于正常 Web 请求的核心特征，注意力机制同样会增强其权重，从而更好地区分正常 Web 请求和恶意 Web 请求. 同理经过加密、混淆和魔改后的恶意 Web 请求关键特征，也将会被注意力机制赋予更高的权重.

注意力机制的计算过程如式(6)至式(8)所示.

$$v_t = \tanh(\boldsymbol{w}_c \cdot \boldsymbol{y}_t + b_c) \tag{6}$$

$$\boldsymbol{a}_t = \mathrm{Softmax}(\boldsymbol{w}^{\mathrm{T}}, v_t) \tag{7}$$

$$\boldsymbol{u} = \sum_{t=1}^{n} \boldsymbol{a}_t \cdot \boldsymbol{y}_t \tag{8}$$

其中，式(6)生成目标注意力权重，值为 v_t，它是激活函数 tanh 对 $\boldsymbol{y}_t$非线性变换的结果，$\boldsymbol{y}_t$是 CNN-BiLSTM 网络模型输出的向量，$\boldsymbol{w}_c$是训练的参数向量，b_c是对应的偏置项. 式(7)采用 Softmax 函数计算各分量 v_t的权重，得到权重的概率向量 $\boldsymbol{a}_t$，$\boldsymbol{w}^{\mathrm{T}}$是转置矩阵. 最终通过式(8)将生成的注意力权重匹配给对应的网络模型输出向量 $\boldsymbol{y}_t$，$\boldsymbol{u}$ 表示对 $\boldsymbol{y}_t$重要性加权和的句子向量.

最后，经过注意力机制得到向量表示 $\boldsymbol{u}$，将其输入到 Softmax 分类器中得到最终分类结果，从而完成混淆 Web 恶意请求检测任务.

4　实验及评估

本文采用 TensorFlow 和 Keras 深度学习框架设计神经网络模型，编程语言为 Python3.7. 该部分对所提出的混淆 Web 恶意请求检测及对抗模型进行有效性评估和详细的对比实验，进一步突出检测模型的优势.

在实验环境方面，本文采用 Windows10 64 位操作系统，处理器为 Intel(R) Core i7-8700K，内存为 64 GB，GPU 为 GTX 1080Ti，本文通过 GPU 加速器来提高实验的效率. 在模型参数配置方面，最大特征词数设为 10 000，文本序列长度设为 600，三层卷积神经网络的卷积核数量设为 256，卷积层和全连接层统一采用非线性激活函数 *ReLU*. 同时，BiLSTM 模型正反两个方向的神经元数均设置为

256,优化算法选择 *Adam* 优化器,*Epoch* 设为 12,初始学习率为 0.001,误差评价选择交叉熵损失函数,并且增加 Dropout 方法防止出现过拟合现象,其 *keep_prob* 参数值设为 0.4.

4.1 实验数据及预处理

实验数据为经典数据集 HTTP data set CSIC 2010[①] 和 Github[②③④⑤] 所采集公开 Web 请求的集合,同时通过常用的网络安全攻击工具模拟生成混淆、加密和流量魔改的 Web 恶意请求.接着利用不同混淆类型所生成的规则对数据集进行恶意性及混淆类型标注,将正常 Web 请求标注为"0",恶意 Web 请求标注为"1",并为混淆加密类型添加额外的标注属性.通过对数据标注后的 Web 请求数据集实施预处理,从中抽取 GET 和 POST 请求的 URL 及 Payload 参数进行组合,最终形成 38.5 万条 Web 请求.其中,正常 Web 请求 20 万条,恶意 Web 请求 18.5 万条,包括各种混淆加密的 SQL 注入、XSS 攻击、SSRF 攻击、命令注入等恶意请求.

本文将实验数据集的常见十大类混淆 Web 恶意请求按照一定比例随机划分为训练集、验证集和测试集,其数据分布情况如表 4 所示.为验证本文提出的 OMRDetector 模型能有效检测未知类型的混淆及加密恶意请求,本文增加了 DES 加密和恶意流量魔改混淆两大类型,它们分别存在 5000 条测试数据,并且不参与训练过程,将在 4.3.5 小节的未知 Web 恶意请求感知分析部分详细介绍.此外,为有效评估模型对混淆请求的检测效果,正常请求中包含常见的混淆方式,约占 20%,如大小写混淆、URL 编码转换和 Base64 编码转换等,也包含部分常用字符,如路径连接符、参数拼接符等.

表 4 数据集描述

类别	请求类别	混淆类型	训练集	验证集	测试集
0	正常请求	混淆请求约占 20%	100000	40000	60000
1	恶意请求	常规恶意请求	20000	5000	10000
		大小写混淆	8000	2000	4000
		关键词复写混淆	8000	2000	4000
		含注释绕过混淆	8000	2000	4000
		特殊字符截断混淆	8000	2000	4000
		路径绕过混淆	8000	2000	4000
		URL 编码转换	8000	2000	4000
		特殊功能关键词混淆	8000	2000	4000
		组合规则绕过逻辑混淆	8000	2000	4000
		等价函数替换混淆	8000	2000	4000
		Base64 编码转换	8000	2000	4000
		DES 加密	—	—	5000
		流量魔改混淆	—	—	5000
总计			**200000**	**65000**	**120000**

图 6 展示了未经预处理的恶意 Web 请求和正常 Web 请求的词云分布情况.其中恶意请求以"script"、"select"、"sleep"、"passwd"、"echo"、"union"等关键词为主,并且包含字符编码、特殊字符、大小写等混淆特征,以及部分加密特征;正常请求主要包含常规的 URL 路径、网页文件和请求参数,如"index"、"php"、"jsp"、"path"等.

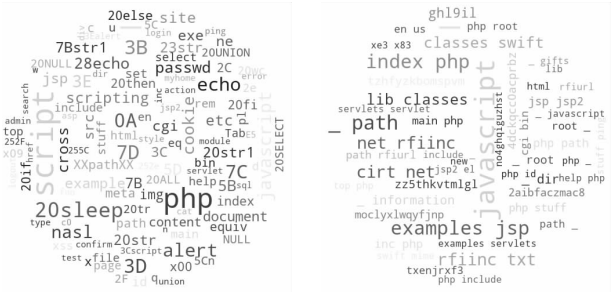

图 6 Web 请求词云分析图

4.2 评估指标

本文将数据集中的正常 Web 请求和恶意 Web 请求分别使用正样本(类别为 0)和负样本(类别为 1)标记,分类情况如表 5 所示.其中,真阳性(True Positive,*TP*)表示 Web 请求的检测结果和真实结果都是恶意请求的数量;真阴性(True Negative,*TN*)表示 Web 请求的检测结果和真实结果都是正常请求的数量;假阳性(False Postive,*FP*)表示将正常 Web 请求的样本归类为恶意请求的数量;假阴性(False Negative,*FN*)表示将 Web 恶意请求的样本

① HTTP DATASET CSIC 2010. https://www.isi.csic.es/dataset/
② Github. https://github.com/exp-db/AI-Driven-WAF
③ Github. https://github.com/Echo-Ws/UrlDetect
④ Github. https://github.com/duoergun0729/1book
⑤ Github. https://github.com/cwellszhang/DetectMalicious-URL

表 5　恶意请求检测模型的分类情况

预测类别	真实类别	
	恶意请求	正常请求
恶意请求	True Positive(TP)	False Positive(FP)
正常请求	False Negative(FN)	True Negative(TN)

归类为正常请求的数量.

为了评估检测模型对混淆、加密及魔改恶意请求识别的性能，选取精确率($Precision$)、召回率($Recall$)、F_1值(F_1-score)和准确率($Accuracy$)作为恶意请求检测的评估指标. 其计算公式如下所示：

$$Precision=\frac{TP}{TP+FP} \tag{9}$$

$$Recall=\frac{TP}{TP+FN} \tag{10}$$

$$F_1\text{-score}=\frac{2\times Precision\times Recall}{Precision+Recall} \tag{11}$$

$$Accuracy=\frac{TP+TN}{TP+TN+FP+FN} \tag{12}$$

精确率旨在评估模型预测正确的类别数量占预测该类别总数量的比值，其值越大说明该模型检测正常 Web 请求样本的能力越强；召回率旨在评估模型预测正确的类别数量占该类别实际数量的比值，其值越大说明该模型对恶意 Web 请求样本的检测能力越强；F_1值是综合考虑精确率和召回率的结果，通常用于评估分类模型的效果；准确率旨在评估分类预测正确的样本数量占该类别 Web 请求总数量的比值，可以直观地反映分类效果的好坏，本文主要用它来对比各深度学习模型随训练迭代次数的学习效果.

为验证本文提出模型的有效性和实用性，最终实验精确率、召回率、F_1值和准确率为 10 次 Web 请求分类结果的平均值，通过多次试验来减小某次异常实验的噪声影响.

4.3　实验结果与分析

4.3.1　实验总体评价

本文首次将改进的三层 CNN-BiLSTM 和注意力机制模型应用于 Web 恶意请求检测，并进一步融合抗混淆预处理设计 OMRDetector 模型. 为验证本文提出混淆恶意请求检测及对抗模型的有效性，分别与传统机器学习模型和深度学习模型进行对比实验，并详细还原参考文献的部分检测模型. 其中，传统机器学习模型包括逻辑回归(LR)、决策树(DT)、朴素贝叶斯(NB)、支持向量机(SVM)、随机森林(RF)和 AdaBoost 算法，现有深度学习模型包括CNN模型、TextCNN模型、LSTM 模型、BiLSTM 模型、BiLSTM 融合注意力机制模型，这些对比模型的参数均为多次实验的最优解，部分模型是参考文献所提出算法的重现. 实验结果如表 6 所示.

表 6　各模型实验结果对比

分类	模型	*Precision*	*Recall*	F_1-score	*Accuracy*
机器学习模型	LR	0.95384	0.95370	0.95377	0.95416
	DT	0.95901	0.95942	0.95921	0.95952
	NB	0.89975	0.89798	0.89886	0.89981
	SVM	0.96004	0.96023	0.96013	0.96045
	RF	0.96121	0.96140	0.96130	0.96162
	AdaBoost	0.91570	0.91618	0.91594	0.91658
深度学习模型	CNN	0.97171	0.97169	0.97170	0.97194
	TextCNN	0.97520	0.97537	0.97528	0.97549
	LSTM	0.97186	0.97224	0.97205	0.97226
	BiLSTM	0.97513	0.97524	0.97518	0.97539
	BiLSTM+Attention	0.97552	0.97560	0.97556	0.97576
本文模型	三层 CNN-BiLSTM+Attention	0.97593	0.97609	0.97601	0.97621
	OMRDetector(混淆恶意请求检测器)	**0.97734**	**0.97737**	**0.97735**	**0.97754**

由表 6 对比可知，本文设计的 OMRDetector 模型能够较好地检测混淆的 Web 恶意请求，其精确率、召回率、F_1值和准确率均有一定程度的提升. 在传统机器学习模型中，随机森林(RF)表现效果更好，本文 OMRDetector 模型相较于随机森林模型，其精确率、召回率、F_1值和准确率分别提升1.613%、1.597%、1.605%和 1.592%. 本文OMRDetector 模型与现有的 CNN、TextCNN、LSTM、BiLSTM、BiLSTM 融合 Attention 模型对比，其 F_1值分别提升 0.565%、0.207%、0.530%、0.217%和 0.179%；准确率分别提升 0.560%、0.205%、0.528%、0.215%和 0.177%. 此外，本文设计并首次应用于混淆 Web 恶意请求检测的三层 CNN-BiLSTM 融合 Attention 模型的实验效果也相对较好，但 OMRDetector 模型表现最优. 通过深入分析发现，OMRDetector 模型检测效果更佳的原因来自两个方面. 一是通过抗混淆预处理层

能有效提升 OMRDetector 对字符级混淆、编码转换和流量魔改的 Web 恶意请求检测效果;二是融合三层 CNN 和 BiLSTM 网络的模型能更好地捕获加密和混淆请求的长距离依赖关系和上下文语义特征,突出混淆关键特征和语义关联.综上,OMRDetector 模型的优势具体表现为:三层 CNN 网络结构能够较好地学习经过抗混淆预处理后的 Web 请求深层次的局部特征;BiLSTM 网络能捕获前后时序特征和长距离依赖关系,有效学习混淆恶意请求特征和加密共现特征,比如“file”和“log”、“union”和“--”、“img”和“alert”等,并且融合注意力机制更加重视关键信息对分类结果的影响,进而提升整个模型的检测效果.最终该模型取得最好的混淆恶意请求检测效果,且优于本文提出的三层 CNN-BiLSTM+Attention 模型.

需要注意,尽管 OMRDetector 模型的性能相对于其它深度学习模型的提升并不是特别明显,其准确率提升最多的是 0.560%(超出 CNN 模型),提升最低的是 0.177%(超出 BiLSTM+Attention 模型),但该模型的性能提升在 12 万条 Web 请求测试集上映射,会发现 OMRDetector 模型比 CNN 模型多正确识别 672 条,比 BiLSTM 及 Attention 模型多正确识别 212 条,比传统机器学习朴素贝叶斯模型多正确识别 9324 条.面对当前企业每日甚至每时千万级别的流量请求,OMRDetector 方法将发挥着重要作用,能有效提升工业界 Web 恶意请求检测的性能.同时,本文更加注重对混淆、加密及魔改后的 Web 恶意请求检测及感知,OMRDetector 模型提供了一种检测混淆恶意请求的思路,并通过对十二大类混淆 Web 恶意请求进行系统实验,进一步体现本文所提出 OMRDetector 模型的优点,为学术界和工业界检测混淆及加密的 Web 请求提供借鉴.综上所述,该程度的性能提升显然是有价值的,后续实验将进一步验证本文所提出模型对混淆恶意请求的检测效果更好,具有一定对抗混淆、加密及魔改的作用,从而突出本文方法的有效性和必要性.

4.3.2 恶意请求检测对比分析

为进一步分析各模型识别 Web 恶意请求特征的效果,本文对 Web 恶意请求的漏报数量和误报数量进行统计,得出结果如表 7 所示.由表可知,OMRDetector 模型未检测出的恶意请求数量和恶意请求检测错误数量均最少.其中,恶意请求漏报数量为 1226 个,比六类机器学习模型(LR、DT、NB、SVM、RF、AdaBoost)的平均值少 1995 个,比五类深度学习对比模型(CNN、TextCNN、LSTM、BiLSTM、BiLSTM+Attention)的平均值少 148 个,比本文改进并首次应用在混淆加密恶意请求检测的三层 CNN-BiLSTM+Attention 模型少 31 个,该模型缺少抗混淆预处理过程.Web 恶意请求误报数量为 1244 个,比六类机器学习模型的平均值少 1912 个,比五类深度学习模型的平均值少 223 个,比三层 CNN-BiLSTM+Attention 模型少 116 个.本文模型的误报数量提升更高,对混淆 Web 恶意请求具有较强的鲁棒性.

表 7 各模型恶意请求检测效果对比

模型	恶意请求漏报数量	恶意请求误报数量
LR	2567	2475
DT	2085	2367
NB	6107	4914
SVM	2114	2236
RF	2048	2174
AdaBoost	4407	4769
CNN	1551	1536
TextCNN	1292	1404
LSTM	1402	1649
BiLSTM	1319	1388
BiLSTM+Attention	1307	1359
三层 CNN-BiLSTM+Attention	1257	1360
OMRDetector	**1226**	**1244**

通过对比各模型 Web 恶意请求检测的漏报数量和误报数量,进一步验证本文提出的 OMRDetector 模型能有效检测混淆加密的恶意请求.经过深入分析其原因,发现 OMRDetector 模型能够通过抗混淆预处理层的七个核心步骤对 Web 请求进行清洗和过滤,再利用三层 CNN 模型提取局部特征,利用 BiLSTM 网络捕获长距离混淆特征,最终通过注意力机制增强关键特征,从而突出混淆特征的权重,这些方法结合后能有效降低漏报数量和误报数量,提升混淆和加密 Web 恶意请求的识别效果.此外,由于网站的正常 Web 请求数量巨大,其误报率也会影响检测模型的性能,因此本文对正常 Web 请求的误报率和漏报率进行统计分析,发现 OMRDetector 模型正常请求的误报率为 2.07%,漏报率为 2.04%,均优于现有机器学习模型和深度学习模型.其中,误报率比现有六类机器学习模型的平均误报率低 3.19%,比现有五类深度学习模型的平均误报率低 0.35%.

本文不仅与现有的机器学习模型、深度学习模型进行综合性对比实验,还分别针对 Web 恶意请求数据集和正常 Web 请求数据集进行实验评估,得出如图 7 和图 8 所示的结果.

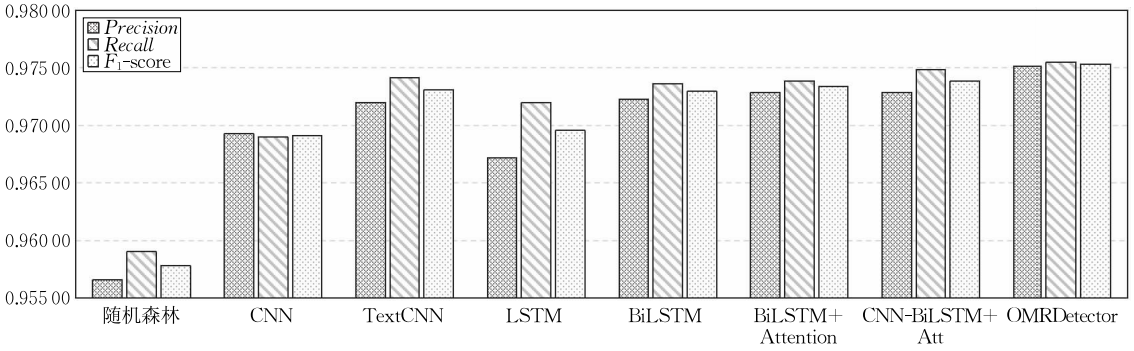

图 7　各模型 Web 恶意请求性能对比

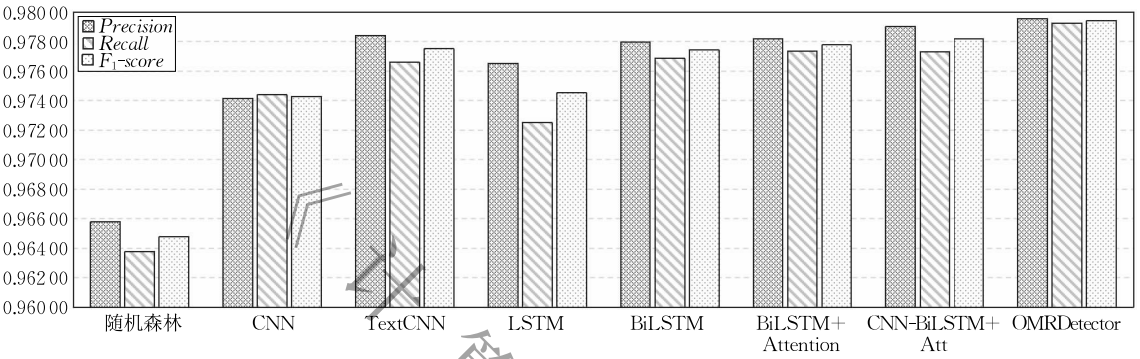

图 8　各模型正常 Web 请求性能对比

从图 7 展示的实验结果来看，OMRDetector 模型在 Web 恶意请求测试集上的精确率、召回率和 F_1 值分别为 0.975 13、0.975 48 和 0.975 30，它们均达到上述模型中最好效果. 其中，本文模型恶意 URL 请求检测的 F_1 值比机器学习中表现较好的随机森林模型提高 1.75 个百分点，比 CNN 模型提高 0.62 个百分点，比 TextCNN 模型提高 0.22 个百分点，比 LSTM 模型提高 0.57 个百分点，比 BiLSTM 模型提高 0.24 个百分点，比 BiLSTM 融合 Attention 模型提高 0.19 个百分点，比本文提出的三层 CNN-BiLSTM 融合 Attention 模型提高 0.14 个百分点. 该实验充分说明本文模型能够学习到混淆加密 Web 恶意请求的深层次特征，从而获取更好效果. 同时，结合抗混淆预处理、三层 CNN、BiLSTM 和注意力机制多种算法的优点让其效果更佳.

从图 8 展示的实验结果来看，OMRDetector 模型在正常 Web 请求测试集上的精确率、召回率和 F_1 值分别为 0.979 56、0.979 27 和 0.979 41，它们均达到上述模型中最好效果. 其中，与传统机器学习随机森林模型对比，本文模型的精确率提高 1.38 个百分点，召回率提高 1.55 个百分点，F_1 值提高 1.46 个百分点. 与现有的深度学习 CNN 模型对比，本文模型的精确率提高 0.54 个百分点，召回率提高 0.49 个百分点，F_1 值提高 0.51 个百分点. 与经典的深度学习 LSTM 模型对比，本文模型的精确率提高 0.30 个百分点，召回率提高 0.68 个百分点，F_1 值提高 0.49 个百分点. 该实验有效验证了本文模型在正常 Web 请求检测中也有较好的效果.

4.3.3　抗混淆效果分析

为深入挖掘各模型的抗混淆效果，通过对大小写、关键词复写、含注释绕过等十大类混淆恶意请求进行分析，得出如表 8 所示的检测结果. 其中，本文提出 OMRDetector 模型融合抗混淆预处理、三层 CNN、BiLSTM 和注意力机制后，在各类混淆恶意请求中表现均较好，关键词复写混淆和等价函数替换混淆识别率为 100%. 传统机器学习模型在大小写混淆、含注释混淆、路径绕过混淆、特殊功能关键词混淆、组合规则绕过逻辑混淆和 Base64 编码转换中表现较差，该类混淆恶意请求未识别数量约占 4%到 8%之间；常用深度学习对比模型在大小写混淆、含注释绕过混淆、路径绕过混淆、特殊功能关键词混淆和 Base64 编码转换中表现较差，而本文提出的模型在这些混淆加密恶意请求检测中均有一定程度的提升. 此外，为了提高网络请求传输过程中的安全性，现实中的正常请求也会采取编码转换、混淆、加密等处理，本文数据集中的正常请求包含 20%的混淆处理，它们一定程度也会造成各入侵检测模型的误报和漏报.

表 8 各类混淆 Web 恶意请求检测结果对比

模型	恶意请求混淆类型									
	大小写混淆	关键词复写混淆	含注释绕过混淆	特殊字符截断混淆	路径绕过混淆	URL编码转换	特殊功能关键词混淆	组合规则绕过逻辑混淆	等价函数替换混淆	Base64编码转换
LR	○	◐	○	○	○	◐	○	○	◐	◐
DT	○	◐	◐	○	◐	◐	○	○	◐	○
NB	○	○	○	○	○	○	○	○	○	○
SVM	○	◐	○	◐	○	◐	○	◐	◐	◐
RF	○	●	◐	◐	◐	◐	○	◐	◐	◐
AdaBoost	○	○	○	○	○	◐	○	○	○	○
CNN	○	●	◐	◐	◐	◐	○	◐	◐	◐
TextCNN	○	●	◐	◐	◐	●	◐	◐	●	●
LSTM	○	●	◐	◐	○	◐	◐	◐	◐	◐
BiLSTM	○	●	◐	◐	○	●	◐	◐	◐	◐
BiLSTM+Attention	◐	●	◐	◐	○	●	◐	◐	●	◐
三层 CNN-BiLSTM+Attention	◐	●	●	◐	◐	◐	◐	●	●	●
OMRDetector	●	●	●	●	●	●	◐	●	●	●

注:"●"表示能够基本识别该类带混淆的恶意请求,未识别数量占总数的比例小于 1‰(0～4 个);"◉"表示能够较好地识别该类带混淆的恶意请求,未识别数量占总数的比例为 1‰至 10‰区间(5～40 个);"◐"表示识别该类带混淆的恶意请求时,未识别数量占总数的比例为 10‰至 40‰区间(41～160 个);"◐"表示识别该类带混淆的恶意请求时,未识别数量占总数的比例为 40‰至 80‰区间(161～320 个);"○"表示识别该类带混淆的恶意请求效果较差,未识别数量占总数的比例为大于 80‰(320 个以上).

通过进一步分析发现:(1) URL 编码转换后的特征与正常 Web 请求、常规恶意请求的特征区别较大,其检测效果相对较好,但经 Base64 编码转换的正常请求一定程度会影响实验结果.此外,转码或加密流量的内容恶意性检测仍是未来亟需突破的关键;(2) 特殊字符截断混淆、组合规则绕过逻辑混淆由于其本质是通过增加或组合字符,利用网站漏洞绕过防火墙实现攻击,而恶意请求特征仍然存在,其恶意行为会被有效识别;(3) 特殊功能关键词混淆、等价函数替换混淆通过调用函数实现恶意攻击,这些函数规则特征能被机器学习和深度学习识别,本文模型的双向长短时记忆网络能较好地捕获上下文函数特征;(4) 大小写混淆、关键词复写混淆、含注释绕过混淆、路径绕过混淆和 Base64 编码转换对传统检测模型的干扰性较大,本文一方面通过注意力机制关注这些混淆特征,BiLSTM 能捕获长距离依赖,另一方面抗混淆预处理能过滤部分干扰项,最终取得更好的检测结果.

恶意请求测试集各混淆类型的分类结果如图 9 所示,它有效地将各类别 Web 恶意请求数据集进行区分.各 Web 恶意请求按 N-gram 分词后形成多维特征,经过 PCA 算法降维处理形成两维具有代表性的特征,能有效表征多个维度的特点,最终将这两个维度特征分别作为横坐标和纵坐标,并绘制对应的各混淆类型的分类结果.其中,位于左下方的实心三角形表示 URL 编码转换,右中部实五边形表示等价函数替换混淆,右中部实心圆形表示特殊功能关键词混淆,右下方三叉形表示组合规则绕过逻辑混淆,其它混淆类别也都在各区域交叉分布着,主要是因为混淆恶意请求特征存在各种规则融合的情况,但这并不会影响实验的整体效果.该实验充分说明本文 OMRDetector 模型能较好区分十类混淆 Web 恶意请求,该结果与表 6 的各类混淆检测结果相辅相成.

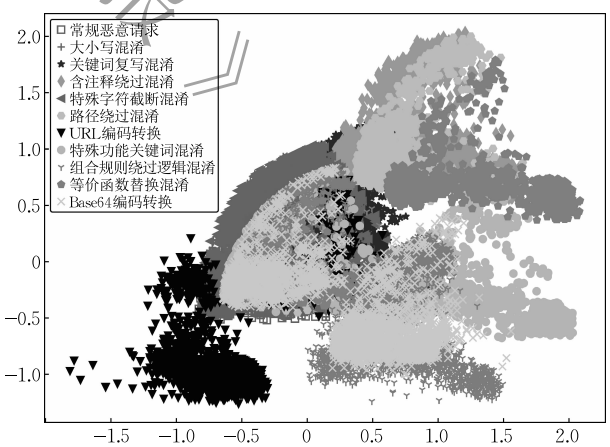

图 9 Web 恶意请求各混淆类型的分类结果

由于正常 Web 请求中也存在混淆、加密和编码转换的情况,为更好地评估 OMRDetector 对混淆正常请求的检测能力,本文进行了详细的对比实验,并绘制如图 10 所示的效果图.其中,OMRDetector 模型检测混淆的正常 Web 请求的准确率为 0.972 39,比随机森林、CNN、TextCNN、LSTM、BiLSTM、

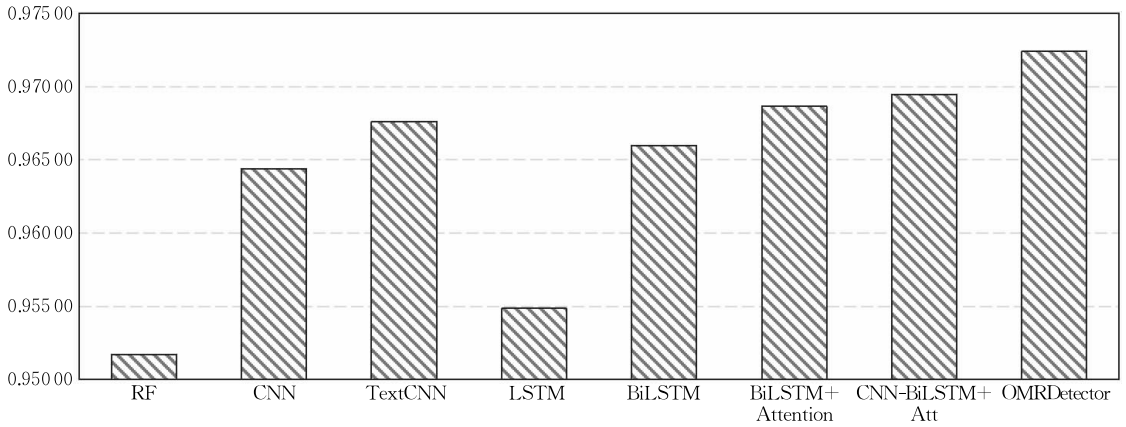

图 10　混淆的正常 Web 请求各模型检测结果对比

BiLSTM＋Attention、CNN-BiLSTM＋Att 模型分别提升 2.07%、0.80%、0.48%、1.75%、0.64%、0.37% 和 0.29%. 这充分说明 OMRDetector 模型能更好地区分混淆及加密的 Web 请求(包括恶意请求和正常请求).

4.3.4　性能对比分析

为进一步衡量分类模型的检测效果，评估其对恶意 Web 请求和正常 Web 请求的识别结果，本文进行了混淆矩阵对比实验. 分别选取了机器学习模型效果更好的随机森林模型、深度学习对比模型效果更好的 BiLSTM＋Attention 模型和本文提出的 OMRDetector 模型进行比较，实验结果如表 9 所示. 由表可知，本文提出的模型能正确识别测试集中正常 Web 请求 58 756 条，比随机森林(RF)模型多识别 930 条，比深度学习 BiLSTM＋Attention 对比模型多识别 115 条；本文提出的模型能正确识别 Web 恶意请求 48 774 条，比随机森林模型多识别 822 条，比现有深度学习对比模型多识别 81 条. 综上，本文模型能更好地感知 Web 恶意请求攻击并检测正常的 Web 请求.

表 9　Web 请求检测的混淆矩阵对比结果

模型	混淆矩阵		真实类别	
			恶意请求	正常请求
RF	预测类别	恶意请求	47 952	2174
		正常请求	2048	57 826
BiLSTM＋Attention	预测类别	恶意请求	48 693	1359
		正常请求	1307	58 641
OMRDetector	预测类别	恶意请求	48 774	1244
		正常请求	1226	58 756

同时，为了更形象地表现本文 OMRDetector 模型的良好性能，使用训练数据和验证数据进一步挖掘模型的学习过程，得出如图 11 所示的准确率变化曲线和如图 12 所示的误差变化曲线. 在图 11 中，横坐标为训练周期(*Epoch*)，纵坐标为训练集的准确率(*Accuracy*). 由图 11 可知，与其它各模型相比，本文所提出的 OMRDetector 模型训练过程更稳定，整个曲线收敛速度更快，并且在训练数据量较少时取得较高的准确率，最终稳定在 0.979 12 的位置，其对混淆加密恶意 Web 请求的检测效果优于已有的 CNN、LSTM 及其改进模型.

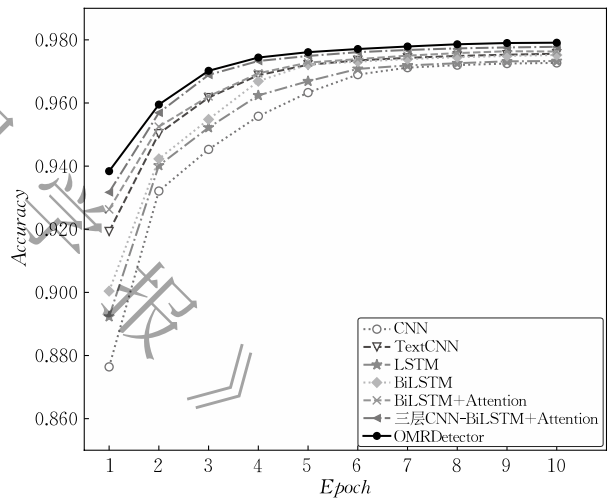

图 11　各深度学习模型的准确率变化曲线

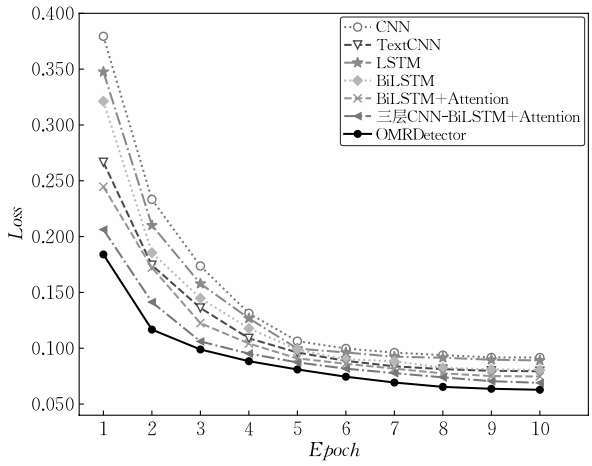

图 12　各深度学习模型的误差变化曲线

图 12 展现各深度学习模型的误差随训练周期变化的曲线，本文模型的误差(*Loss*)随训练周期收敛速度更快，曲线更平缓，取得较好的收敛效果，这进一步验证了将该模型应用到含混淆恶意请求检测是可行的.

本文对 OMRDetector 模型的卷积神经网络层数进行了对比实验，在相同的抗混淆预处理和模型参数情况下，对比卷积神经网络不同层数的效果，得出如表 10 所示的实验结果，该表仅显示 1 至 6 层 CNN 的实验结果. 由表可知，本文方法的 F_1 值为 0.97735，准确率为 0.97754，均优于其他层数. 这可能与本文构建的混淆 Web 恶意请求数据集的长度有关，三层 CNN 能够更好地提取这些 Web 请求的局部特征，并与 BiLSTM 和注意力机制相互融合. 因此，考虑到模型的准确率和时间效率，本文构建的 OMRDetector 模型最终选择三层卷积神经网络.

表 10 卷积神经网络层数对比实验

CNN 层数	F_1-score	Accuracy	训练时间
1	0.97597	0.97626	219.1134
2	0.97615	0.97638	238.1435
3	**0.97735**	**0.97754**	**339.4455**
4	0.97658	0.97681	392.1290
5	0.97613	0.97632	632.1037
6	0.97639	0.97679	1005.5542

分类算法的精准率、召回率和 F_1 值是评价模型的重要指标，同时对 Web 请求的推理时间也是一个重要指标，它能有效评估一个模型的时间成本和算法复杂度，以及该模型是否能有效地实现对恶意请求攻击的实时检测，从而更好地为模型在工业界应用提供基础. 本文详细对比了各深度学习模型的推理时间，其是对测试集所有 Web 请求检测的总用时，最终结果如表 11 所示，时间单位为秒钟.

表 11 各模型推理时间成本对比

模型	推理时间	每秒检测请求数量	*Accuracy*
CNN	62.7990	1911	0.97194
TextCNN	86.1353	1393	0.97549
LSTM	111.2271	1079	0.97226
BiLSTM	150.7662	796	0.97539
BiLSTM+Attention	151.7832	791	0.97576
三层 CNN-BiLSTM+Attention	115.3786	1040	0.97621
OMRDetector	**105.8507**	**1134**	**0.97754**

本文主要分析含混淆、加密和魔改的 Web 请求数据集. 由表可知，七种深度学习模型对测试集的推理时间范围是 60 至 160 s 之间. 其中，CNN 模型的推理时间最短，其值为 62.7990 s，每秒检测出 1911 条 Web 请求；本文所提出 OMRDetector 模型的推理时间为 105.8507 s，每秒检测出 1134 条 Web 请求，其推理时间效率处于中等偏上，其推理时间成本是可接受范围，未呈现指数级增长. 此外，通过该实验评估，验证了 OMRDetector 模型能有效应用于 Web 恶意请求的实时检测.

4.3.5 未知 Web 恶意请求感知分析

为更好地评估各模型对未知混淆、加密及魔改类型的 Web 恶意请求检测效果，检验 OMRDetector 模型对未知 Web 攻击的感知能力. 本文选择四种混淆类型进行对比实验，其中 DES 加密和流量魔改混淆是新增加的混淆类型，未在前文的实验中出现，它们是工业界或 Web 攻击工具经常使用的混淆加密类型，测试数据各为 5000 条；另外两种混淆类型选择 Base64 编码转换和 URL 编码转换，这两种编码转换混淆类型的恶意流量在真实网络攻击中经常出现. 区别于前文的实验，该部分将训练集和验证集中 Base64 编码转换和 URL 编码转换两种类型的 Web 恶意请求清除，仅设置各 5000 条作为测试集. 最终通过对四种类型的 20000 条 Web 恶意请求和 5000 条正常 Web 请求实验，来实现对未知 Web 恶意请求的检测和对抗能力评估. 此外，正常 Web 请求包含经混淆、编码和加密处理后的网络请求. 实验结果如表 12 所示.

表 12 未知 Web 恶意请求检测结果对比

模型	*Precision*	*Recall*	F_1-score
LR	0.83376	0.88723	0.85966
DT	0.82898	0.88196	0.85465
NB	0.80195	0.86512	0.83234
SVM	0.81590	0.85498	0.83498
RF	0.83751	0.88929	0.86262
AdaBoost	0.80017	0.85611	0.82720
CNN	0.89299	0.93319	0.91265
TextCNN	0.92587	0.94419	0.93494
LSTM	0.90900	0.92993	0.91935
BiLSTM	0.91407	0.93409	0.92397
BiLSTM+Attention	0.91603	0.93636	0.92608
三层 CNN-BiLSTM+Attention	0.92888	0.94736	0.93803
OMRDetector	**0.93968**	**0.95552**	**0.94753**

由表可知，本文 OMRDetector 模型能较好地识别未知类型的 Web 恶意请求，其精确率、召回率和 F_1 值分别为 0.93968、0.95552 和 0.94753，它们均为各种模型的最优结果. 同时，六种机器学习模型对该数据集的未知 Web 恶意请求的检测效果较差，对抗性较弱，它们的平均精确率、召回率和 F_1 值分别为 0.81971、0.87245 和 0.84524，比本文提出的 OMRDetector 模型分别降低 11.996%、8.308% 和

10.229%.其它五种常用于 Web 恶意请求检测的深度学习模型，其检测性能比机器学习稍好，但也低于本文提出的 OMRDetector 模型，它们的平均精确率、召回率和 F_1 值分别降低 2.808%、1.997% 和 2.414%.通过对未知类型的 Web 恶意请求检测实验，进一步突出本文提出 OMRDetector 模型能有效对抗混淆加密请求，并发现未知恶意 Web 攻击.

此外，本文分别对比了不同模型对未知类型 Web 请求和已知类型 Web 请求的检测结果，形成如图 13 和图 14 所示的结果，其横坐标是各种算法，纵坐标是不同算法对应检测结果的 F_1 值.其中，图 13 详细对比了机器学习模型与 OMRDetector 模型的实验结果，它们分别对未知混淆加密类型（未在训练集出现）和已知混淆加密类型（训练集存在部分）的 Web 恶意请求进行检测.图 14 详细对比各种深度学习模型的实验结果.由图可知，本文提出的 OMRDetector 模型不仅能更好地检测出已知混淆类型的恶意请求，还能更好地检测出未知混淆类型的恶意请求，其 F_1 值分别为 0.97735 和 0.94753，均优于机器学习模型和深度学习模型，并且各模型对已知类型的恶意请求检测结果优于未知类型.

最后，为有效评估加密恶意请求的检测效果.本文对未知类型的 DES 加密恶意请求进行实验，得出各模型的 F_1 值，如图 15 所示.由图可知，各模型对

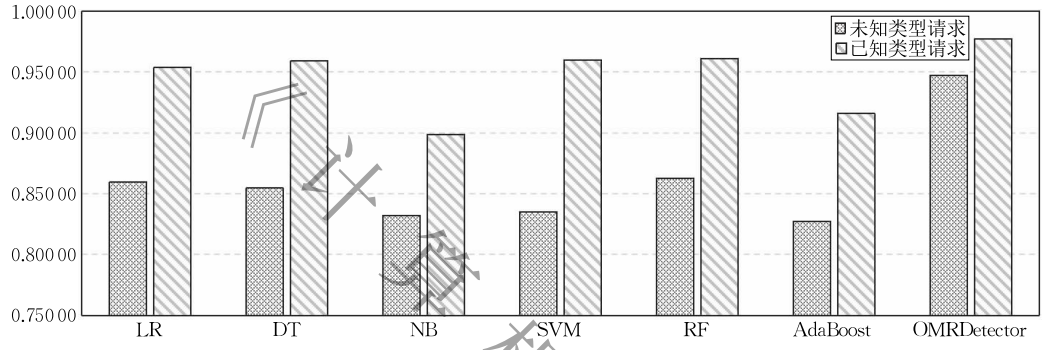

图 13　未知及已知类型 Web 请求的机器学习模型检测结果对比

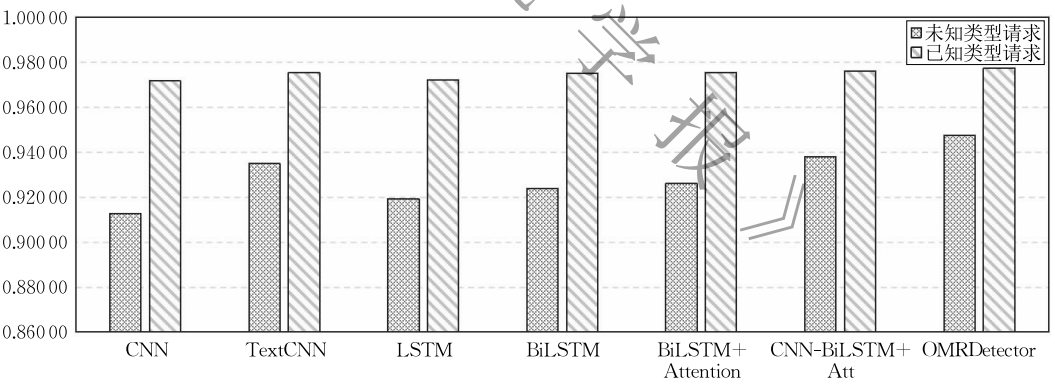

图 14　未知及已知类型 Web 请求的深度学习模型检测结果对比

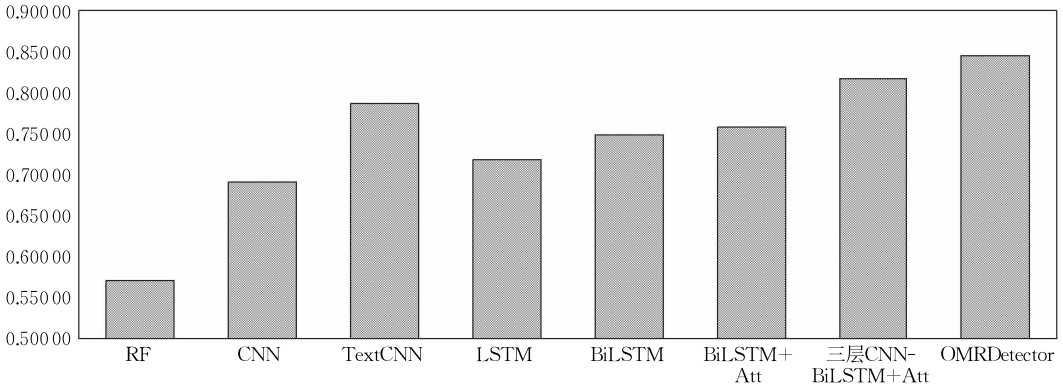

图 15　各模型检测 DES 加密恶意请求的结果对比

DES 加密的 Web 恶意请求检测效果不佳，机器学习模型有大幅度的降低，其表现最好的随机森林模型的 F_1 值仅为 0.570 91. 同时，各深度学习对比模型的 F_1 值均未超过 0.8，而本文 OMRDetector 模型的检测结果为 0.845 14，远低于表 6 和表 12 的效果. 通过进一步分析，本文认为加密恶意流量的语义关联性较弱，未呈现足够的特征规律或共现关系，从而导致传统机器学习和深度学习方法仍较难识别加密流量的恶意性. OMRDetector 模型虽然能较好地识别 DES 加密请求，其 F_1 值优于各对比模型，但仍有较大的提升空间. 未来，将继续开展加密恶意流量检测的研究.

4.3.6 讨论

本文提出了一种基于深度学习的混淆恶意请求检测方法，称为 OMRDetector. 该模型融合了抗混淆预处理、三层卷积神经网络、BiLSTM 和注意力机制. 首先，与现有 Web 恶意请求检测相关研究不同，本文主要针对混淆恶意请求进行分析，通过详细的对比实验验证 OMRDetector 模型的有效性，并定义了十二大类混淆方法，涉及字符级混淆、编码转换、加密和流量魔改类型，涵盖了真实场景中容易出现的混淆恶意请求. 其次，本文的方法优于传统机器学习[12-14,16-17]和深度学习[20,23-24,27,29]模型，通过抗混淆预处理能提升模型的泛化能力，更好地识别字符级混淆、编码转换和流量魔改的 Web 恶意请求，而之前的方法缺乏对混淆和加密请求的考虑，更偏向于模型的组合及准确率比较，未系统分析模型对不同混淆类型的检测能力，并且这些高隐蔽、高威胁的混淆恶意请求已对网络空间安全造成严重的危害. 最后，本文结合混淆恶意请求的特点，利用抗混淆预处理来对抗混淆，构建三层卷积神经网络来提取局部特征，融合 BiLSTM 网络捕获加密和混淆请求的长距离依赖关系和上下文语义特征，能较准确地识别出不同混淆及加密类型的 Web 恶意请求和正常请求，可以广泛应用于入侵检测领域.

综上所述，本文针对混淆 Web 恶意请求特点及 Web 攻击的现状，提出 OMRDetector 模型，并进行了详细的论证和实验分析，以期为科研工作者和安全技术人员检测混淆加密请求提供借鉴. 同时，本文从多个维度丰富了现有混淆恶意请求检测工作.

5 结 语

当前 Web 恶意请求复杂多元化、混淆伪装和加密魔改明显，防火墙和 Web 应用防护系统很容易被绕过，传统基于规则和机器学习的方法忽略了 Web 恶意请求特征语义关系，无法第一时间感知未知类型的网络攻击，且对混淆、加密和魔改的 Web 恶意请求效果较差. 本文提出一种基于三层 CNN-BiLSTM 和注意力机制的混淆 Web 恶意请求检测及感知模型 OMRDetector. 该模型针对混淆加密的恶意请求特征，提出了抗混淆预处理方法，利用 CNN 获取 Web 请求的局部特征，再引入 BiLSTM 网络捕获混淆恶意请求的上下文语义特征和长距离依赖关系，通过注意力机制进行特征融合，最终由 Softmax 分类器计算恶意 Web 请求的检测结果.

为验证模型性能，本文进行了详细的对比实验和未知恶意请求检测实验. 实验结果表明，本文所提出的 OMRDetector 模型能有效检测出隐蔽性较高且带混淆、加密和魔改的 Web 恶意请求，相比于传统模型在精确率、召回率、F_1 值和准确率上均有所提升，并且能够较好地识别大小写混淆、关键词复写混淆、含注释绕过混淆、特殊字符截断混淆、路径绕过混淆、URL 编码转换、特殊功能关键词混淆、组合规则绕过逻辑混淆、等价函数替换混淆和 Base64 编码转换十大类常见的混淆 Web 恶意请求攻击，能有效识别包括 DES 加密和流量魔改混淆等 Web 恶意请求攻击，以及混淆的正常 Web 请求. 同时，本文详细验证了 OMRDetector 模型能够充分利用语义特征和前后时序，注意力机制突出混淆特征，从而提升恶意请求的检测结果，有效对抗恶意请求的各种混淆，检测未知混淆加密的攻击. 这些实验进一步证明了本文的模型具有一定的学术价值和应用前景.

在下一步研究工作中，本文计划从以下两个方面进行探索.

(1) 由于真实场景的 Web 请求混淆、加密及魔改操作更复杂，Web 请求数据量更大，本文将针对工业界安全产品及 WAF 所生成的 Web 请求或流量数据进行实时检测及对抗，发现隐蔽性更高、对抗性更强的混淆加密恶意请求. 同时，降低变种恶意请求逃逸和绕 WAF 的风险，弥补本文 OMRDetector 模型仅针对十二大类混淆恶意请求的缺点，以及缺乏对真实安全产品的恶意行为分析.

(2) 对现有神经网络模型进行改进，结合迁移学习模型和语义关系进行真实复杂场景的扩充，尝试利用图神经网络和知识图谱构建混淆恶意请求的关联，实现多分类混淆 Web 恶意请求检测模型，力争实现恶意 Web 请求的溯源.

致　谢　感谢《计算机学报》编辑部老师们的辛勤工作，感谢审稿专家提出的宝贵意见！

参　考　文　献

[1] Yu L, Chen L, Dong J, et al. Detecting malicious Web requests using an enhanced TextCNN//Proceedings of the 2020 IEEE Annual International Computer Software and Applications Conference(COMPSAC). Madrid, Spain, 2020: 768-777

[2] Ji Tian-Tian, Fang Bin-Xing, Cui Xiang, et al. Research on deep learning-powered malware attack and defense techniques. Chinese Journal of Computers, 2021, 44(4): 669-695(in Chinese)
(冀甜甜，方滨兴，崔翔等. 深度学习赋能的恶意代码攻防研究进展. 计算机学报，2021，44(4)：669-695)

[3] Luo B, Xia J. A novel intrusion detection system based on feature generation with visualization strategy. Expert Systems with Applications, 2014, 41(9): 4139-4147

[4] Sperotto A, Schaffrath G, Sadre R, et al. An overview of IP flow-based intrusion detection. IEEE Communications Surveys and Tutorials, 2010, 12(3): 343-356

[5] Denning D E. An intrusion-detection model. IEEE Transactions on Software Engineering, 1987, 13(2): 222-232

[6] Ma J, Saul L K, Savage S, et al. Identifying suspicious URLs: An application of large-scale online learning//Proceedings of the 26th International Conference on Machine Learning(ICML). Montreal, Canada, 2009: 681-688

[7] Pan Feng, Ding Yun-Fei, Wang Wei-Nong. Anomaly detection techniques based on statistics. Journal of Shanghai Jiaotong University, 2004, 38(s1): 204-207(in Chinese)
(潘峰，丁云飞，汪为农. 两种基于统计的入侵检测技术. 上海交通大学学报，2004，38(s1)：204-207)

[8] Borders K, Springer J, Burnside M. Chimera: A declarative language for streaming network traffic analysis//Proceedings of the 21st USENIX Security Symposium. Bellevue, USA, 2012: 365-379

[9] Li H, Hu H, Gu G, et al. vNIDS: Towards elastic security with safe and efficient virtualization of network intrusion detection systems //Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security(CCS). Toronto, Canada, 2018: 17-34

[10] Abbes T, Bouhoula A, Rusinowitch M. Efficient decision tree for protocol analysis in intrusion detection. International Journal of Security and Networks, 2010, 5(4): 220-235

[11] Khan L, Awad M, Thuraisingham B. A new intrusion detection system using support vector machines and hierarchical clustering. The VLDB Journal, 2007, 16(4): 507-521

[12] Muda Z, Yassin W, Sulaiman M N, et al. A *K*-means and naive Bayes learning approach for better intrusion detection. Information Technology Journal, 2011, 10(3): 648-655

[13] Chen Chun-Ling, Wu Fan, Yu Han. A classification and recognition model of malicious requests based on logistic regression. Computer Technology and Development, 2019, 29(2): 124-128(in Chinese)
(陈春玲，吴凡，余瀚. 基于逻辑斯蒂回归的恶意请求分类识别模型. 计算机技术与发展，2019，29(2)：124-128)

[14] Fu C, Li Q, Shen M, et al. Realtime robust malicious traffic detection via frequency domain analysis//Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS). Virtual Event, Korea, 2021: 3431-3446

[15] Chi Ya-Ping, Ling Zhi-Ting, Wang Zhi-Qiang, et al. Intrusion detection system based on support vector machine and Adaboost. Computer Engineering, 2019, 45(10): 183-188(in Chinese)
(池亚平，凌志婷，王志强等. 基于支持向量机与 Adaboost 的入侵检测系统. 计算机工程，2019，45(10)：183-188)

[16] Perdisci R, Lee W, Feamster N. Behavioral clustering of HTTP-based malware and signature generation using malicious network traces//Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI). San Jose, USA, 2010: 391-404

[17] Zhang Y, Li L, Zhou J, et al. POSTER: A PU learning based system for potential malicious URL detection//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security(CCS). Dallas, USA, 2017: 2599-2601

[18] Li X, Zhu M, Yang L T, et al. Sustainable ensemble learning driving intrusion detection model. IEEE Transactions on Dependable and Secure Computing. 2021, 18(4): 1591-1604

[19] Luo Y, Xiao Y, Cheng L, et al. Deep learning-based anomaly detection in cyber-physical systems: Progress and Opportunities. ACM Computing Surveys, 2021, 54(5): 1-36

[20] Atienza D, Herrero Á, Corchado E. Neural analysis of HTTP traffic for Web attack detection//Proceedings of the 8th Computational Intelligence in Security for Information Systems Conference (CISIS). Burgos, Spain, 2015: 201-212

[21] Tang T A, Mhamdi L, McLernon D, et al. Deep learning approach for network intrusion detection in software defined networking//Proceedings of the 2016 International Conference on Wireless Networks and Mobile Communications(WINCOM). Fez, Morocco, 2016: 258-263

[22] Liang J, Zhao W, Ye W. Anomaly-based Web attack detection: A deep learning approach//Proceedings of the 2017 VI International Conference on Network, Communication and Computing (ICNCC). Kunming, China, 2017: 80-85

[23] Wang W, Zhu M, Zeng X, et al. Malware traffic classification using convolutional neural network for representation learning//Proceedings of the 2017 International Conference on Information Networking (ICOIN). Da Nang, Vietnam, 2017: 712-717

[24] Saxe J, Berlin K. eXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs, file paths and registry keys. arXiv preprint arXiv: 1702.08568, 2017

[25] Zhang M, Xu B, Bai S, et al. A deep learning method to detect Web attacks using a specially designed CNN//Proceedings of the 24th International Conference on Neural Information Processing(ICONIP). Guangzhou, China, 2017: 828-836

[26] Le H, Pham Q, Sahoo D, et al. URLNet: Learning a URL representation with deep learning for malicious URL detection. arXiv preprint arXiv: 1802.03162, 2018

[27] Cui Yan-Peng, Liu Mi, Hu Jian-Wei. Malicious Web request detection technology based on CNN. Computer Science, 2020, 47(2): 281-286(in Chinese)
(崔艳鹏，刘咪，胡建伟. 基于 CNN 的恶意 Web 请求检测技术. 计算机科学，2020，47(2)：281-286)

[28] Sun P, Liu P, Li Q, et al. DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system. Security and Communication Networks, 2020, Article ID: 8890306

[29] Liu Yue-Feng, Cai Shuai, Yang Han-Xi, et al. Network intrusion detection method integrating CNN and BiLSTM. Computer Engineering, 2019, 45(12): 127-133(in Chinese)
(刘月峰，蔡爽，杨涵晰等. 融合 CNN 与 BiLSTM 的网络入侵检测方法. 计算机工程，2019，45(12)：127-133)

[30] Staudemeyer R C. Applying long short-term memory recurrent neural networks to intrusion detection. South African Computer Journal, 2015, 56(1): 136-145

[31] Afzal S, Asim M, Javed A R, et al. URLdeepDetect: A deep learning approach for detecting malicious URLs using semantic vector models. Journal of Network and System Management, 2021, 29(21): 1-27

[32] Yu Bang-Bing, Wang Hua-Zhong, Yan Bing-Yong. Intrusion detection of industrial control system based on long short term memory. Information and Control, 2018, 47(1): 54-59
(in Chinese)
(於帮兵，王华忠，颜秉勇. 基于长短时记忆网络的工业控制系统入侵检测. 信息与控制，2018，47(1)：54-59)

[33] Lai Xun-Fei, Liang Xu-Wen, Xie Zhuo-Chen, et al. Intrusion detection method based on entity embedding and long short-term memory networks. Journal of University of Chinese Academy of Sciences, 2020, 37(4): 553-561(in Chinese)
(赖训飞，梁旭文，谢卓辰等. 基于实体嵌入和长短时记忆网络的入侵检测方法. 中国科学院大学学报，2020，37(4)：553-561)

[34] Shi Le-Yi, Zhu Hong-Qiang, Liu Yi-Hao, et al. Intrusion detection of industrial control system based on correlation information entropy and CNN-BiLSTM. Journal of Computer Research and Development, 2019, 56(11): 2330-2338(in Chinese)
(石乐义，朱红强，刘祎豪等. 基于相关信息熵和 CNN-BiLSTM 的工业控制系统入侵检测. 计算机研究与发展，2019，56(11)：2330-2338)

[35] Li B, Zhou H, He J, et al. On the sentence embeddings from pre-trained language models//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing(EMNLP). Online, 2020: 9119-9130

[36] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting//Proceedings of the 35th Conference on Artificial Intelligence (AAAI). Vancouver, Canada, 2021: 11106-11115

[37] Tang G, Müller M, Rios A, et al. Why self-attention? A targeted evaluation of neural machine translation architectures//Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP). Brussels, Belgium, 2018, 4263-4272

YANG Xiu-Zhang, Ph. D. candidate. His research interests include network security and information system security.

PENG Guo-Jun, Ph. D., professor, Ph. D. supervisor. His research interests include network security and information system security.

LUO Yuan, Ph. D. candidate. His research interests include information physical system security and mobile system security.

SONG Wen-Na, Ph. D. candidate. Her research interests include information security and mobile system security.

ZHANG Jie, M. S. candidate. His research interest is network security.

CAO Fang-Tao, M. S. candidate. His research interest is network security.

Background

Network Security is a hot topic that has been discussed in the electronic and computer domains for a long time. Nowadays, we are witnessing a rapid growth of sophisticated cyber attacks, and Web attacks have been endangering cyberspace security for years, causing substantial security risks to our society and enterprises. Such Web attacks include SQL injection, XSS attack, server-side request forgery, and remote code execution vulnerability. Unfortunately, the prior

works suffer from multiple weaknesses, e. g. , depending on experience and rules, easy to be bypassed, high false-positive rates, fail to capture the semantic characteristics of malicious requests. Therefore, they cannot detect unknown network attacks in the first place and deal with obfuscated malicious requests. Moreover, attackers usually use different rules to obfuscate and encrypt malicious requests to bypass the traditional defenses such as firewalls and intrusion detection systems, thereby breaching critical infrastructures and causing devastating catastrophes.

To address these problems, we design and apply the three-layer CNN-BiLSTM fusion attention mechanism model to the field of obfuscated malicious request detection for the first time. Also, given the characteristics of obfuscated malicious requests, we optimize and propose an anti-obfuscation malicious request detection model named OMRDetector based on deep learning, which can combat obfuscation and detect unknown malicious request attacks effectively. OMRDetector includes an anti-obfuscation preprocessing method, a three-layer CNN network, a BiLSTM network, and the attention mechanism, which can capture long-distance dependencies and contextual semantic information. Meanwhile, this paper combines the characteristics of obfuscated malicious requests and summarizes twelve significant obfuscation types commonly used in malicious request attacks. Such types include case obfuscation, keyword copy obfuscation, comment bypass obfuscation, unique character truncation obfuscation, path bypass obfuscation, URL encoding, particular key word obfuscation, combination rules to bypass logic obfuscation, equivalent function substitution obfuscation, Base64 encoding, DES encryption, and traffic magic change obfuscation. In addition, we design and implement an extensive evaluation framework to compare OMRDetector and the baseline models. Compared with the conventional methods, the *Precision*, *Recall*, F_1-score, and accuracy are improved, and the corresponding values are 97.734%, 97.737%, 97.735%, and 97.754%. In short, Our model can effectively detect highly stealthy and obfuscated malicious network requests.

This work is supported in part by the National Natural Science Foundation of China (Nos. U1636107, 62172308, 61972297, 62172144) and completed under the guidance of Professor Guojun Peng. The authors have proposed a deep learning model to combat various obfuscation of malicious requests and detect unknown network attacks. This work is a significant extension and improvement over anti-obfuscation malicious request detection. Guojun Peng is the corresponding author.