计算机科学 COMPUTER SCIENCE

http://www.jsjkx.com

DOI:10.11896/jsjkx.200100064

基于 TF-IDF 的 Webshell 文件检测

赵瑞杰[1] 施 勇[1,2] 张 涵[1] 龙 军[1] 薛 质[1,2]

1 上海交通大学网络空间安全学院 上海 200240

2 上海市信息安全综合管理技术实验室 上海 200240

(ruijiezhao@sjtu.edu.cn)

摘 要 随着互联网的飞速发展,网络攻击行为日益频繁。Webshell 是常见的网络攻击方式,而传统的检测手段已无法应对复杂灵活的变种 Webshell 攻击。为解决这一问题,提出了一种基于 TF-IDF 的 Webshell 文件检测方法。系统首先对不同类型的 Webshell 文件进行分类,并对不同文件进行相应的预处理转码,以降低混淆干扰技术对检测的影响;随后建立词袋模型,并采用 TF-IDF 算法加权提取相关特征;最后使用 XGBoost 算法训练得到检测模型。与传统机器学习算法进行的 10 折交叉验证对比测试表明,使用 TF-IDF 算法预处理后结合 XGBoost 算法的 Webshell 文件检测模型性能出色,检测效果相较于传统检测方法在准确率、精确率、召回率等方面均有所提高,同时具备更强的鲁棒性与泛化能力,其中对 PHP 类型文件检测的准确率达到了 98.09%,对 JSP 类型文件检测准确率达到了 97.09%。

关键词:Webshell 检测;特征提取;交叉验证;TF-IDF;多层神经网络;支持向量机;随机森林;XGBoost 算法

中图法分类号 TP393

Webshell File Detection Method Based on TF-IDF

ZHAO Rui-jie[1], SHI Yong[1,2], ZHANG Han[1], LONG Jun[1] and XUE Zhi[1,2]

1 School of Cyber Science and Engineering, Shanghai JiaoTong University, Shanghai 200240, China

2 Shanghai Information Security Integrated Management Technology Laboratory, Shanghai 200240, China

Abstract With the rapid development of Internet, cyber attacks are becoming more frequent. Webshell is a common cyber attack method, and traditional detection methods are unable to cope with complex and flexible variants of Webshell attacks. In order to solve this problem, webshell detection method based on TF-IDF is proposed. First of all, the system classifies Webshell files and transcodes different files accordingly to reduce the impact of confusion and interference technology on detection, then build a bag of words model and use TF-IDF algorithm to weight extract relevant features, and finally uses the XGBoost algorithm to train the detection model. Compared with the traditional machine learning algorithm, the Webshell detection model based on TF-IDF and XGBoost algorithm has higher accuracy than the traditional detection method, and has stronger robustness and generalization capabilities. The detection accuracy of XGBoost algorithm for PHP type files can reach 98.09%, and the accuracy for JSP type files can reach 97.09%.

Keywords Webshell detection, Feature extraction, Cross validation, TF-IDF, Multi-layer perception, Support vector machine, Random forest, XGBoost algorithm

1 概述

随着 Internet 的普及,网络上的共享资源成为了黑客们攻击的主要目标。随着网络攻击的急剧增多及其所带来的影响日益恶劣,信息安全问题成为了人们日益关注的焦点。在当今社会中,计算机网络在许多关键基础设施领域,如政府和军事组织以及企业,都扮演着极其重要的角色[1]。因此,对于网络管理者来说,研究如何成功阻止恶意网络黑客入侵,使网络系统和计算机处于安全的正常运行状态,无疑是一项迫切的任务。

Webshell 是以 PHP 或者 JSP 等网页文件形式存在的一种命令执行环境,也称为一种网页后门。一般说来,攻击者入侵一个网站后,会把这些 ASP,PHP 木马的后门文件放在该网站的 web 目录中,与正常的网页文件混杂,其命名可能与正常文件的命名很类似,让人无法第一眼通过文件名判断其为后门文件[2]。然后,入侵者就可以通过 Web 请求的方式,用 ASP 或者 PHP 木马后门对网站的服务器进行控制,具体包括上传或下载文件、调取数据库资料、执行恶意命令程序等一系列未经授权的入侵行为。

本文主要对 Webshell 文件检测进行研究,具体工作包括 3 个方面。首先,通过预处理将 JSP 类型文件转为 Java 汇编码,将 PHP 类型文件转为 opcode 码,大幅降低混淆绕过技术

基金项目:国家重点研发计划项目(2017YFB0803200)

This work was supported by the National Key R&D Programe of China(2017YFB0803200).

通信作者:薛质(zxue@sjtu.edu.cn)

的干扰；随后，在此基础上建立词袋模型，采用 TF-IDF 算法在词频基础上使用加权处理技术降低高频无意义词汇的干扰，进一步提升数据质量；最后，引入具备极佳综合性能的 XGBoost 算法模型对数据集进行学习训练。10 折交叉验证测试结果表明，使用 XGBoost 算法的 Webshell 文件检测模型相较于其他机器学习检测算法，有着更佳的准确率、精准率与召回率，模型的综合性能处于领先水平。在 PHP 类型文件中使用 TF-IDF 为特征训练且选取 XGBoost 算法时准确率达到 98.09%，在 PHP 类型文件中使用 TF-IDF 为训练特征选取 XGBoost 算法时准确率达到 97.09%。

2 相关研究

早在 20 世纪 50 年代的后期，机器学习就以人工智能的概念被一些研究者提了出来。它从兴起至今，经历了五十余年。从学科角度来说，机器学习是一门与概率论、统计学、凸分析以及算法复杂度的理论等多门学科有关的交叉学科[3-5]。其核心是通过对先验知识的学习获取经验，自动分析数据并获取规律，然后推论出某种模型，据此对未来进行预测，其相关应用非常广泛。机器学习技术研究成为了一个热门话题，被广泛地应用到图像识别、语音识别、垃圾邮件过滤等多个领域，并取得了优异的成果，同时它也为解决入侵检测领域的问题提供了许多新兴的研究方向。

Stolfo 等[6]在 1999 年首次将机器学习方法应用到入侵检测模型中，其方案通过采集网络数据流量并对审计数据进行分析，得到一个异常检测模型。之后，陆续出现了有关注入侵检测方向的机器学习研究。其中研究最多的主要有基于聚类算法的入侵检测系统、基于支持向量机的入侵检测系统、基于人工神经网络的入侵检测系统等。2015 年，Ye 等[7]提出了一种基于支持向量机的 Webshell 黑盒检测方案，根据 HTML 文档的信息对 Webshell 进行检测，准确率达到 75%，具有一定的实际部署价值。2018 年，Fu[8]提出了一种基于 CNN 的 Webshell 检测方案，对 PHP 类型文件设计相应的机器学习检测方案，检测准确率较传统机器学习算法有了一定的提升，但仍存在着模型易过拟合和调参复杂等问题。

虽然将机器方法应用到入侵检测系统中已经取得了一定的效果，但是由于网络数据量的剧增和新攻击的层出不穷，入侵检测研究已经到达了一个瓶颈期，出现了众多较难解决的问题，如特征提取不合适、检测准确率较低，以及算法的复杂度较高等。同时，当前研究建立的 Webshell 检测模型大多仅适用于对 PHP 类型的 Webshell 文件进行检测，未对 JSP 类型的 Webshell 文件检测建立科学可行的方案。本研究将对上述问题进行讨论与解决。

3 基于 TF-IDF 的 Webshell 检测模型

Webshell 的编写十分灵活，变体众多，根据脚本程序的功能不同，Webshell 通常可以分为“大马”“小马”“一句话木马”3 种[9]。越来越多的 Webshell 通过绕过、混淆技术干扰着现有 Webshell 检测工具的检测性能，传统的基于特征的静态检测手段已无法适应现在复杂多变的攻击环境，需要设计出一套更加有效的算法模型对 Webshell 进行特征提取和检测。本研究中对不同类型的 Webshell 预处理方式不同，因此系统首先对 Webshell 进行分类，随后对 PHP 代码和 JSP 代码进行不同方式的转码处理，最后通过 TF-IDF 提取其相关特征后，分别使用 SVM、随机森林、XGBoost 3 种机器学习算法进行模型搭建。系统整体框架如图 1 所示。

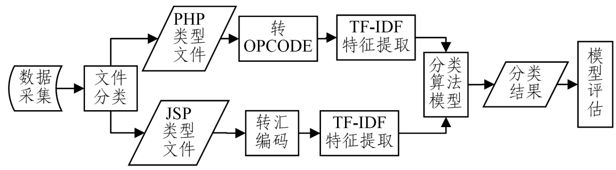

图 1 整体框架图

Fig. 1 System framework

3.1 Webshell 文件预处理

为了获得最佳的训练效果，我们先对不同类型的 Webshell 文件分类，并使用相应的方法进行预处理。预处理工作主要分为文件分类、去重、转码、特征提取 4 个部分。其中，文件分类通过判别文件类型将数据样本分为 JSP 和 PHP 类型文件；去重通过 MD5 校验去除重复样本，降低了重复样本对训练效果的影响；转码根据文件的不同类型将 JSP 文件内容转为 Java 汇编码，PHP 文件内容转为 opcode 码，以大幅降低绕过混淆等技术对检测性能的影响；特征提取均使用 TF-IDF 算法进行，较传统词频提取方法具备更加有效的性能表现。

3.1.1 TF-IDF 算法模型

目前对 Webshell 的特征提取大多使用词频的方法，然而词频特征提取方法只考虑到了相关词汇在整表中出现的频率，无法消减高频无意义的词汇对整体性能带来的影响。为了进一步提升模型性能，本研究采用 TF-IDF 词频特征提取方法对 JSP 与 PHP 文件特征进行提取。

TF-IDF 是一种在词频基础上进行信息检索与数据挖掘的加权技术。TF 是指 Term Frequency，即文本中某关键词出现的频率；IDF 是指 Inverse Document Frequency，表示逆文本频率指数。该方法较传统的词频提取，更加能够筛选出有用的重要信息。通过计算 TF-IDF 的值对数据集特征进行分类训练，TF-IDF 的计算值越大，说明该词对文本的相关性越大。

以 JSP 文件为例，首先统计文件中不同汇编码出现的次数，例如 iconst_1 出现过 X 次，文本共有 N 个词，易得 iconst_1 出现的词频为 X/N；随后计算逆文本频率指数 IDF，计算方法为 $\log(D_A/D)$，其中 D 为出现 iconst_1 的 JSP 样本的具体数量，D_A 为全部 JSP 样本数量；最后进行 TF-IDF 具体数值的计算，计算公式如下所示。

$$TFIDF = TF \times IDF = X/N \times \log(D_A/D)$$

PHP 文件 TF-IDF 数值的计算过程与 JSP 文件相似，通过 TF-IDF 的模型对所有 JSP 与 PHP 文件进行具体的数值计算，得到 JSP 文件汇编码如 iconst_1 和 iconst_2 等相应的数值；同理得到 PHP 文件 OPCODE 码如 ADD 和 ASSIGN 等具体的相应数值。最后，将这些数值作为数据集特征进行训练。

3.1.2 JSP 类型 Webshell 的特征提取

JSP 类型的 Webshell 中的关键代码是 Java 代码，但是本身的 Java 代码可以利用各种绕过混淆技术来改变本身 Webshell 的静态特征。为了解决这一问题，设计方案通过将 JSP

代码转换为最后的 Java 汇编码得到相应的训练特征,从而提高训练准确率。实现过程为:首先将 JSP 文件使用 Tomcat 作为容器,先将 JSP 转换为 Servlet 类源文件(.java),再将该源文件编译相应为 Servlet 类的.class 文件,最后直接使用 Java 命令将.class 文件转化为 Java 汇编码进行特征提取。特征提取流程如图 2 所示。

图 2　JSP 类型特征提取流程图

Fig. 2　JSP type feature extraction flowchart

对 JSP 代码进行 TF-IDF 特征提取,并以此建立词汇表模型。Java 汇编码经加权处理后的词频特征如图 3 所示。

图 3　JSP 代码的 TF-IDF 特征值

Fig. 3　TF-IDF characteristics of JSP type

为了更加清晰地展示图示汇编码加权词频特征,表 1 列出部分汇编码加权特征值。

表 1　汇编码 TF-IDF 特征值

Table 1　TF-IDF feature values ofassembly code

汇编码	TF-IDF 值
iconst_1	0.3879
iconst_2	0.0577
dconst_1	0.0083
sipush	0.0041
ldc	0.1651
l2d	0.0041
swap	0.1238
…	…

3.1.3　PHP 类型 Webshell 的特征提取

PHP 类型的 Webshell 是 PHP 代码,同样可以通过绕过、混淆等技术改变本身 Webshell 的特征。因此,可以通过将 PHP 代码转换为中间层次的 OPCODE 码,来提取相应的训练特征。OPCODE 码是由编译器对高级语言进行相应的词法分析转换后得到,是一种在 PHP 脚本编译之后的中间语言。本文使用 PHP 插件 VLD 进行 PHP 文件的 OPCODE 转换。特征提取流程图如图 4 所示。

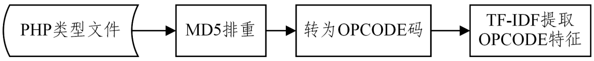

图 4　PHP 类型特征提取流程图

Fig. 4　PHP type feature extraction flowchart

将 PHP 代码转为 OPCODE 码后建立的词汇表模型与进行加权词频特征提取后对应的特征值如图 5 所示。

图 5　PHP 代码的 TF-IDF 特征值

Fig. 5　TF-IDF feature ralues of PHP type

以数据形式整理图中部分 OPCODE 加权特征值,如表 2 所列。

表 2　OPCODE 码的 TF-IDF 特征值

Table 2　TF-IDF feature values of OPCODE

OPCODE	TF-IDF 值
ADD	0.0051
ADD_ARRAY_ELEMENT	0.0112
ASSIGN	0.2103
ASSIGN_ADD	0.0020
ASSIGN_CONCAT	0.0345
ASSIGN_OBJ	0.0081
NOP	0.0224
…	…

3.2　分类器模型

本文使用两种监督学习算法测试对比使用 XGBoost 算法的 Webshell 检测模型的有效性与性能表现,其分别为支持向量机(Support Vector Machine,SVM)和随机森林算法。

3.2.1　SVM、随机森林、XGBoost 算法模型

SVM 提高学习机泛化能力是通过寻求结构化风险最小来实现的,目的是达到目标风险的最小化,同时对置信范围也进行最小化,即便是在统计样本量不是非常充足的条件下,也一样可以取得良好的统计规律[10-11]。

决策树算法是一种常用的机器学习算法,在分类问题中,决策树算法通过样本中某一维度属性的数值,将样本划分到不同的类别中。随机森林是指利用多棵树对样本进行训练并预测的一种分类器,即随机森林是在决策树的基础上通过投票进行优化的分类器。如果有了新的数据需要通过随机森林得到分类结果,就可以通过对子决策树的判断结果投票,来获得随机森林的输出结果[12-13]。它的优点是避免了决策树可能出现的过拟合情况,并且由于投票机制的存在,往往能获得令人满意的分类效果。

XGBoost 算法通过弱分类器的迭代过程来进行并行计算并实现准确的分类效果,是基于梯度 Boosting 的集成学习算法[14]。Boosting 是一种使用多个更简单的模型来拟合数据的学习算法,它所用的这些更加简单、基础的模型也被称为基本学习器或弱学习器。其学习的方法是使用参数设置一样或稍有不同的基本学习器来自适应地拟合数据。实验表明 XGBoost 的性能在随机森林算法的基础上又有进一步提升,XGBoost 算法的优点非常的显而易见,能对各个类型的数据进

行灵活的处理。XGBoost 的训练速度大约是传统的 GBDT 的 10 倍量级[15-16]，其因速度快、准确率高而在众多机器学习算法中表现卓越。同时，即使在调参时间相对较少的情况下，采用该算法模型进行预测的准确度也非常高。

3.2.2 XGBoost 训练模型的参数设置

为了得到适合本系统的最佳算法性能，我们对 XGBoost 算法进行了一定的参数调试。测试中，我们选择数据集中的 70%数据样本进行训练，30%数据样本进行测试，对不同树深度进行测试。首先对 JSP 样本进行调参训练，根据训练结果，由于欠拟合的情况，在训练深度为 1～5 时的准确率不如深度为 6 时表现理想，当训练深度为 6 时，准确率达到最高的 96.90%。随后我们对 PHP 样本进行相应的调参测试，与之前结果类似，在训练深度为 6 时，算法准确率达到 98.41%。测试结果如图 6 所示，为了尽可能避免过拟合，不使用过大深度对算法模型进行训练，因此我们将树深度 6 设置为本系统采用的 XGBoost 深度参数。

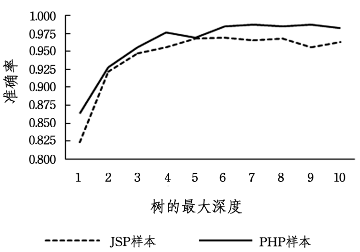

图 6 不同训练深度的准确率

Fig. 6 Accuracy of samples in different training depth

4 模型测试

系统测试方法有很多，在模型建立后，选择不一样的参数必然导致分类器产生性能差异。为了增强方案验证的可靠性，本文使用 sklearn 的 cross_val_score 10 折交叉验证方法进行实验测试。10 折交叉验证是将原始数据均分成为 10 组，数据的验证集是其中的每一个子集，训练集则是剩下的 9 组子集数据，这样会得到 10 个验证集生成的分类准确率，此分类器的性能由这 10 个分类准确率的平均数体现[18-19]。通过比较不同 Webshell 检测算法在 10 折交叉验证下的性能表现，对结果进行相应分析。

4.1 实验数据

PHP 训练中使用的样本集数据为 4404 个，其中 Webshell 样本为 1903 个，普通样本为 2501 个。JSP 训练中使用的样本集数据为 1533 个，其中 Webshell 样本为 352 个，普通样本为 1181 个。Webshell 样本主要来源于 github 中公开的 Webshell 收集项目，对其进行相应的 MD5 校验去重处理；PHP 正常页面样本主要来源于 wordpress，phpcms，yii 等开源的 PHP 内容管理系统(CMS)。CMS 系统在某些功能表现上类似于 Webshell，所以选取这类 PHP 文件作为白样本，可以使检测更加准确、有效。

4.2 实验环境

本文实验所使用的主机为 Ubuntu18.04 操作系统，处理器为 Intel Core i5-8259U@2.3GHz，内存为 16GB，实验使用 Python 语言实现。实验环境的具体配置与虚拟机实现环境如表 3 所列。

表 3 虚拟机实现环境

Table 3 Virtual machine environment

配置	描述
Ubuntu18.04	操作系统
Python 2.7	编译器环境
JDK1.8	编译器环境
Tomcat8.0	编译器环境
Numpy 1.16.3	扩展程序库
Scikit-learn	机器学习库

4.3 性能测试

4.3.1 性能评价指标

为了进一步对 3 种机器学习算法在 Webshell 检测方面进行各项性能比较，使用准确率、精确率、召回率 3 个指标进行相应性能评估。评估使用的参数及相关计算方法如下所示，其中 T_p 为真正例，T_n 为真负例，F_p 为假正例，F_n 为假负例。

准确率是最常用也最直观的性能指标，模型表示判断正确的数据 T_p+T_n 占总数据的比例，是对整体分类结果正确程度的体现，计算公式如下：

$$Acc=\frac{T_p+T_n}{T_p+T_n+F_p+F_n}$$

精确率是判断检测中真正例在所有正例中的占比情况，体现为检测的所有正例中检测正确的情况，计算公式如下：

$$Precision=\frac{T_p}{T_p+F_p}$$

召回率是正确判断出的正例占数据集中所有正例的比例，也叫查全率，相应的计算公式如下：

$$Recall=\frac{T_p}{T_p+F_n}$$

4.3.2 PHP 类型 Webshell 文件的检测

本节从准确率、精确率、召回率 3 个方面对使用 SVM、随机森林、XGBoost 算法的模型进行比较，算法训练过程中内部参数均调至理想水平。PHP 类型文件使用 TF-IDF 进行特征提取得到的各模型性能指标情况如表 4 所列。

表 4 PHP 类型 Webshell 文件检测算法的性能比较

Table 4 Performance comparison of PHP type Webshell file detection algorithms

(单位：%)

PHP 类型	准确率	精准率	召回率
SVM	92.54	98.02	89.54
随机森林	97.58	98.68	97.21
XGBoost	98.09	99.67	98.16

为了更加直观地体现不同算法的性能差异，对 PHP 类型 Webshell 文件检测性能情况绘制性能指标图，如图 7 所示。

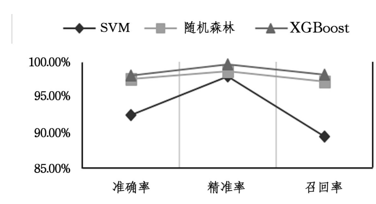

图 7 PHP 类型 Webshell 文件检测算法的性能指标图

Fig. 7 Performance of PHP type Webshell detection

4.3.3 JSP 类型 Webshell 文件的检测

JSP 类型文件使用 TF-IDF 进行特征提取后，对 SVM、随机森林、XGBoost 算法模型的测试性能指标情况如表 5 所列。

表 5 JSP 类型 Webshell 文件检测算法的性能比较

Table 5 Performance comparison of JSP type Webshell file detection algorithms

（单位：%）

JSP 类型	准确率	精准率	召回率
SVM	86.22	95.43	76.27
随机森林	96.08	98.53	94.41
XGBoost	97.09	99.38	96.76

对 JSP 类型 Webshell 文件检测性能情况绘制性能指标图，如图 8 所示。

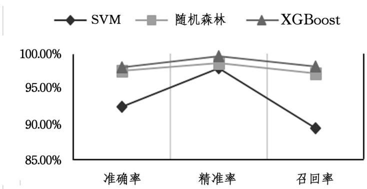

图 8 JSP 类型 Webshell 文件检测算法的性能指标图

Fig. 8 Performance of JSP type Webshell detection

结束语 进行 TF-IDF 预处理后的算法模型均能够较好地学习 Webshell 文件特征进行检测，同时 XGBoost 算法较支持向量机与随机森林算法在 Webshell 文件检测中均有更优的性能表现，其中在 PHP 类型 Webshell 文件检测中的准确率高达 98.09%，在 JSP 类型 Webshell 文件检测中的准确率高达 97.09%。值得一提的是，XGBoost 算法在设计时借鉴了随机森林的做法，支持列抽样，不仅可以防止过拟合，还能进一步减少计算，在数据集的适应性方面无疑有着更加优秀的能力。综合来看，XGBoost 是这 3 个算法中最适合用于 Webshell 检测的算法。

本方案也具有一定的局限性：由于公开数据集中样本数量有限，导致算法模型容易出现过拟合现象，整理收集更加丰富全面的数据集是后续在 Webshell 检测方面非常重要的工作；同时，干扰、加密情况下的 Webshell 检测效果受到一定的影响，下一步将结合异常样本进行深入研究，尝试使用语法树分析等方法来进一步提升检测性能。

参 考 文 献

[1] SHI L Y,FANG Y. Research on Webshell Detection Method Based on Web Log [J]. Information Security Research, 2016, 2(1):66-73.

[2] DAI H,LI J,LU X Y,et al. Machine learning algorithm for intelligent detection of WebShell [J]. Journal of Network and Information Security,2017,3(4):51-57.

[3] GOLDBERG D E. Genetic algorithms in search,optimization and machine learning [M]. Addison-wesley Longman Publishing Co. ,1989.

[4] BUCZAK A L,GUVEN E. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection [J]. IEEE Communications Surveys & Tutorials, 2017, 18(2): 1153-1176.

[5] XIAO H,RASUL K,VOLLGRAF R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms [J]. arXiv:1708.07747,2017.

[6] STOLFO S J,LEE W. A data mining framework for constructing features and models for intrusion detection systems (computer security, network security) [M]. Columbia University 2960 Broadway,1999:227-261.

[7] YE F,GONG J,YANG W. Webshell black box detection based on support vector machine [J]. Journal of Nanjing University of Aeronautics and Astronautics,2015(6):924-930.

[8] FU J M,LI L,WANG Y J. Webshell File Detection Based on CNN [J]. Journal of Zhengzhou University (Science Edition), 2019,51(2):4-11.

[9] QI J J. Stealing WebShell Detection Method [J]. Computer and Network,2015(13):38-39.

[10] MEI R,ZHANG T. Research on WebShell detection method based on SVM classifier under Linux [J]. Information Network Security,2014(5):5-9.

[11] CHI Y P,LING Z T,WANG Z Q,et al. Intrusion Detection System Based on Support Vector Machine and Adaboost [J/OL]. Computer Engineering,2019,45(10):183-188.

[12] WANG Y. Design and implementation of pedestrian detection algorithm based on random gradient boosting decision tree [D]. Hangzhou:Zhejiang University,2017.

[13] CHEN J,LI K,TANG Z,et al. A Parallel Random Forest Algorithm for Big Data in a Spark Cloud Computing Environment [J]. IEEE Transactions on Parallel & Distributed Systems, 2017,PP(99):1-1.

[14] TU X Y,YU L,GENG Z C,et al. A Method for Early Warning of Leakage Accidents Based on Large-scale Time Series [J]. Information Technology,2018,42(12):1-4.

[15] CHEN T,GUESTRIN C. XGBoost: A Scalable Tree Boosting System [C] // ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2016:785-794.

[16] ZHENG H,YUAN J,CHEN L. Short-Term Load Forecasting Using EMD-LSTM Neural Networks with a Xgboost Algorithm for Feature Importance Evaluation [J]. Energies, 2017, 10(8): 1168.

[17] LI Y F,WANG Y,LI J H. Repeatability of several cross-validation tests [J]. Journal of Taiyuan Normal University (Natural Science Edition),2013(4):46-49.

[18] WANG K,HOU Z R,WANG C L. Network Intrusion Detection Based on Cross-Validation SVM [J]. Journal of Test and Measurement Technology,2010,24(5):419-423.

[19] GUTLEIN M,HELMA C,KARWATH A,et al. A Large-Scale Empirical Evaluation of Cross-Validation and External Test Set Validation in (Q)SAR [J]. Molecular Informatics,2013,32(5/6):516-528.

ZHAO Rui-jie, born in 1995, postgraduate. His main research interests include network security, machine learning and data mining.

XUE Zhi, born in 1971, Ph.D, professor, Ph.D supervisor. His main research interests include cyber security, communication technology and machine learning.