Mixed-Models Method Based on Machine Learning in Detecting WebShell Attack

Lu Jinping
Nanjing Linkage Technologies
Nanjing, China
lujpnj@163.com

Tang Zhi
Computer Engineering College
JiMei University
Xiamen, China
2433038802@qq.com

Mao Jian*
Computer Engineering College
JiMei University
Xiamen, China
myjeans@sina.com

Gu Zhiling
Computer Engineering College
JiMei University
Xiamen, China
gzl18850075143@163.com

Zhang Jiemin
Computer Engineering College
JiMei University
Xiamen, China
zhangjm169@163.com

ABSTRACT

WebShell is a command execution environment in the form of web files and also a remote administration tool in web containers. However, it is also a web page backdoor for attackers. Malicious WebShell endangers safety of the Web services. Traditional detection methods, which suitable for general WebShell attack scripts, are based on rule matching. Effectively detecting mutant WebShell has become a great difficulty in computer security worldwide. Mutant scripts of PHP WebShell are the most numerous, complicated and difficult to detect in all kinds of mutant WebShell. This paper proposed a mixed model based on Machine Learning that is used to detect WebShell in different classifications. Using many feature engineering and sample balancing algorithms, the model mixes Machine Learning algorithms of Random Forest (RF) and Convolutional Neural Networks (CNN). It proposed a practicable intelligent solution for mutant WebShell attack detection with the optimized precision rate over 97%.

CCS CONCEPTS

CCS → Information systems → Information systems applications → Process control systems

KEYWORDS

Feature Extraction, Machine Learning, Opcode, Sample Balancing, WebShell

ACM Reference format:

Lu Jinping, Tang Zhi, Mao Jian, Gu Zhiling, Zhang Jiemin. 2020. Mixed-Models Method Based on Machine Learning in Detecting WebShell Attack. 2020 International Conference on Computers, Information Processing and Advanced Education (CIPAE 2020), October 16-18, 2020, Ottawa, Canada. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3419635.3419716

1 Introduction

Nowadays, Web System has become an essential part of modern society. It is widely used in areas like commerce, governmental affairs, military issues and social life and which plays an important role in economy, culture as well as politics. WebShell is a command execution environment in the form of web files such as ASP, PHP, JSP or CGI and also a remote administration tool running on the web server. In normal cases, Web System administrator can realize the Web Service remote management through WebShell.

However, WebShell is also a web page backdoor for attackers [1]. They use all kinds of vulnerabilities existing in Web application system to gain operation privileges and upload malicious scripts of WebShell (for example, PHP script) to Web service so that achieve the goal of long-term manipulating the Web service. WebShell attack makes the delivery of data and information real so that illegal activities like information theft, commercial blackmail and recreational opinion appear, which threaten security, stability and development of society.

Attack scripts of WebShell are much concealed, and they are mixed with normal dynamic web pages. It is difficult for administrators to notice the traces of attack intrusion in the system security log [2-5]. In order to detect WebShell attack, traditional method is based on rule matching and needs to establish rule base. After a long-term

WebShell attack and defense confrontation, general sorts of Webshell attack had been listed into blacklist mechanism by those security systems. Especially, their feature values and functions had been noted in the feature base of Web Application Firewall(WAF), which makes general WebShell attack be effectively defensed.

In recent years, in order to attack, attackers encrypt and disguise WebShell attack scripts to bypass the filtering of the firewall, which gives the birth of the mutant WebShell. Basing on rules is a good way to detect general WebShell attack with a better detection ratio. But for unknown or mutant WebShell scripts, this method faces difficulty in detecting.

In all kinds of mutant scripts formed by Web languages, due to PHP's simple, flexible and diverse feature, its WebShell mutations are numerous, concealed and most difficult to detect. Therefore, nowadays, it is most difficult to establish a WebShell defense system aimed for PHP [6-7]. Searching and studying methods to detect PHP WebShell attack scripts timely and effectively is a representative thing with applied value.

This paper studies a mixed model based on Machine Learning to realize classified detection of PHP WebShell attack sample. This model mixes Machine Learning algorithms of Random Forest(RF)and Convolutional Neural Networks(CNN) and applies many feature extraction technologies such as bag-of-words model, N-gram and TF－TIF. Besides, this model applies technologies of oversampling and undersampling to balance samples, which effectively solve the problem of large differences in identification accuracy between many kinds caused by imbalanced samples

Main contributions of this paper:

1) A variety of feature extraction techniques are studied to effectively extract the in-depth features in WebShell samples.

2) A variety of sampling algorithms are tried to solve the problem caused by unbalanced data set.

3) A variety of machine learning classification algorithms are experimented and the detection model is optimized by data analysis.

2 General Design of the Mixed WebShell Detection Model

2.1 Technology Solutions of the Mixed WebShell Detection Model

The construction of mixed detection model mainly involves feature engineering technology, classification algorithm technology and sample balance processing technology.

Feature engineering: two types of WebShell samples are constructed. One is the text sequence of the original PHP sample. The other is the corresponding opcode sequence converted by the text information of the original PHP sample. For two different WebShell samples, text sequence feature engineering and opcode feature engineering are applied respectively.

Machine learning classification algorithms: the text feature engineering is respectively applied to NB, SVM, MLP, Xgboost and RF classification algorithms. The opcode feature engineering is applied to CNN classification algorithm.

Sample balance processing technology: through the analysis of experimental data, the problem of differences in the recognition rate caused by sample imbalance was found. Oversampling and undersampling techniques are respectively adopted to realize sample balance.

In order to implement each technology, multiple parameters' adjustment and optimizing are needed. By adjusting the corresponding parameters in different technical stages, the optimal detection performance of the model is approximated constantly. The technical scheme of constructing WebShell mixed detection model is shown in Fig. 1.

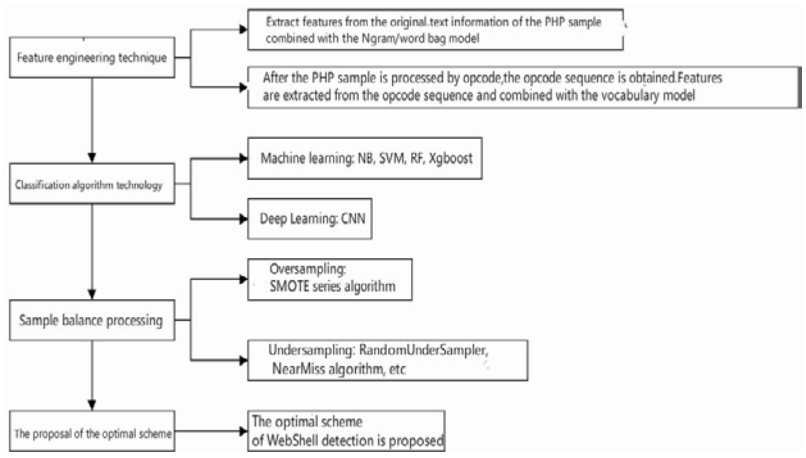
Feature engineering technique
Extract features from the original text information of the PHP sample combined with the Ngram/word bag model
After the PHP sample is processed by opcode,the opcode sequence is obtained.Features are extracted from the opcode sequence and combined with the vocabulary model
Classification algorithm technology
Machine learning: NB, SVM, RF, Xgboost
Deep Learning: CNN
Sample balance processing
Oversampling: SMOTE series algorithm
Undersampling: RandomUnderSampler, NearMiss algorithm, etc
The proposal of the optimal scheme
The optimal scheme of WebShell detection is proposed

Fig. 1. Technology scheme of mixed WebShell detection model

2.2 Experimental Design of WebShell Mixed Detection Model

WebShell attack mixed detection model experiment is shown in Fig. 2.

1) Firstly, MD5 dereprocessing was performed on the original

samples, and then the data set was divided into test set and training set by random sampling algorithm;
2) To conduct feature processing of samples, this experiment mainly uses two feature engineering schemes:

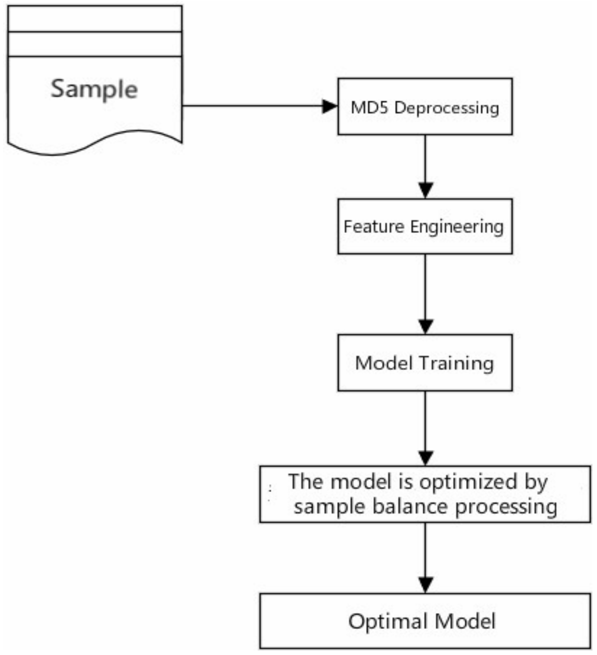

Fig. 2. Experimental design of WebShell mixed detection model

a) Feature words were extracted from the text information of PHP samples, and the text feature engineering was constructed by combining the N-gram word bag model.
b) Opcode processing was carried out for PHP samples, feature words were extracted from the sample opcode sequence, and the opcode feature engineeing was constructed by combining the vocabulary model.
3) A ccording to the characteristics of Machine Learning algorithm, five classification algorithms are selected: NB, SVM, MLP, Xgboost and RF classification algorithm, which are combined with text feature engineering for model training to seek a relatively optimized detection model. CNN classification algorithm was selected and the opcode feature engineering was used for experimental model training.
4) Judge the classification performance of the model by integrating multiple evaluation indexes. And at the same time, constantly adjust and optimize the corresponding parameters of the model, so as to give play to the greatest detection performance. Based on the experimental data, an optimal mixed scheme of Webshell detection is developed.
5) To further optimize the scheme, use oversampling and undersampling techniques such as SMOTE, Neighborhood Cleaning Rule to solve the problem of unbalanced sample distribution. In this experiment, the sample imbalance leads to the problem that the identification accuracy of the minority samples is not as good as that of the majority samples.
6) Optimal WebShell mixed detection model.

3 Sample Collection and Design of Evaluation Index

3.1 Data Sources

The data used in the experiment came from a security company in Xiamen. Because there were some duplicate samples in the original sample set, MD5 was firstly used to deprocess all the samples. A total of 15,567 samples remain after deprocessing, including 4,406 Webshell black samples and 11,161 Webshell white samples, as shown in Table 1.
The ratio of the white sample to the black sample is more than 2:1, so the black sample is unbalanced.

Table 1 Data details

Type	Number(unit:1)	Ratio (unit :%)
WebShell Black Sample	4406	28.30%
WebShell White Sample	11161	71.70%

3.2 Partition of Data Set

The data set is divided into training set and test set on a scale of 1:1. The details are shown in Table 2.
Among them, the training set is used as the training and validation data of the model, and the test set is used to test the classification performance of the model finally. Table 2 shows that in the training set and the test set， there is still a certain data imbalance in Webshell black and white samples. Therefore, this paper will adopt a variety of sampling methods to intervene the training sample set.

Table 2 Partition of data set

Data Type	Sample Type	Number(unit:1)
Training Set	WebShell Black Sample	About 2200
	WebShell White Sample	About 5580
Test Set	WebShell Black Sample	About 2200
	WebShell White Sample	About 5580

3.3 Sample Feature Engineering

1) A word bag model for text sequences
N-gram selects the first 10,000 fragments with the highest frequency of occurrence to generate word bags. In the detection method in this paper, n∈ [2, 4], and the eigenvector form obtained by the feature engineering is shown in Fig. 3.

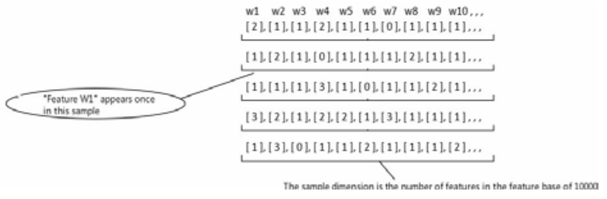

Fig. 3. The Wenxu sequence eigenvector form obtained by word bag

However, the text word bag generated by the N-gram model only considers the frequency of concurrency of the text sequence fragment, ignoring its importance to the sample. Therefore, in order to obtain higher quality sample vectors, TF_IDF algorithm is further used for optimizing.

TF_IDF algorithm is usually used to assess the importance of a corpus fragment to one of the corpus, that is, the distinguishing ability of the corpus fragment. The algorithm is composed of TF(word frequency) and IDF(reverse file frequency). TF refers to the number of occurrences of some corpus fragment X in a given text. IDF reflects the frequency of the corpus fragment in all texts.

2) Opcode Vocabulary Model

Depending on the lifetime of the PHP script, the PHP execution engine runs the opcode sequence[8].Therefore, by extracting the opcode of PHP scripts, a large amount of redundant information such as comment statement that may interfere with the performance of classification model detection can be eliminated on the premise of preserving the PHP operation semantics.

Opcode sequence uses vocabulary model to process features, that is, build vocabulary through the original set, and establish the only index value for each listed feature one after another, which means the earlier a feature is listed, the farther forward the feature will be. Then, using index to code each sample according to the sequence of features through vocabulary, feature vector is obtained through feature engineering, as shown in Fig. 4. Vocabulary model is usually used together with neural network-based distributed representation of words, that is, a word-embedding layer will be realized before the sample vector input classification model.

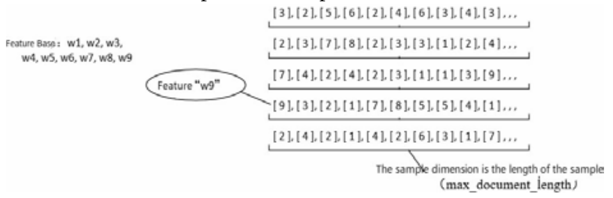

Fig. 4. The form of an eigenvector obtained from a vocabulary

3.4 Evaluation Index System

Webshell detection is a dichotomization problem, and the samples are tagged with positive or negative labels. In this paper, the Webshell black samples are taken as the positive category and the classification performance of the training model is compared and determined by four indicators which are extended from confusion matrix: Accuracy, Precision, Recall and F1.

4 Experimental Data and Analysis

4.1 experimental results of classification algorithm

This experiment adapts to different selected classification algorithms with corresponding feature engineering training sample sets. After adjusting and optimizing parameters, the best detection performance of each algorithm is obtained. The specific step is that when the feature engineering is "text sequence + word bag", classification algorithms of NB[8] MLP[9], SVM[10-11], RF[12], and Xgboost[13] are correspondingly used. When "opcode sequence + vocabulary" is taken as the feature engineering, it is cooperated with CNN classification detection algorithm. The experimental results are shown in Table 3.

It can be seen that these six models have achieved good recognition results, all of which can achieve Accuracy closing to or over 95%. Among which RF has the best recognition result and Accuracy value reaches 96.83%. The

second was the Xgboost model, with Accuracy of 96.54%. The third is the CNN model, and Accuracy is close to 96%. The detection performance of different algorithms are shown in Fig. 5.

Table 3 Experimental results of different algorithms

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1_score**
NB	0.944502	0.914451	0.88123	0.897533
MLP	0.951439	0.916627	0.906381	0.911475
SVM	0.954779	0.983306	0.850489	0.912088
RF	0.968268	0.985685	0.897997	0.9398
Xgboost	0.965442	0.979571	0.89334	0.93447
CNN	0.958247	0.965957	0.876025	0.918796

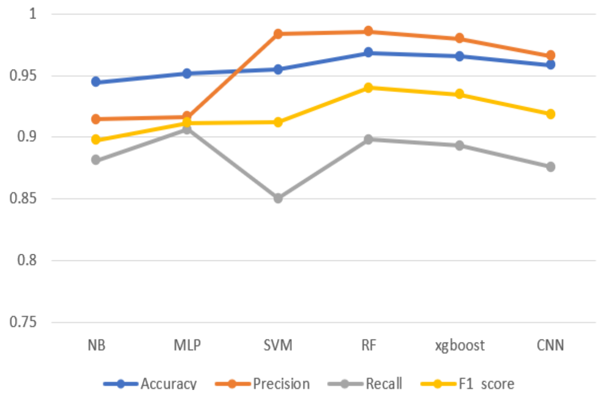

Fig. 5. Comparison of detection performance of different algorithms

4.2 Experimental data analysis of classification algorithm

By analyzing the error detection samples of the CNN model, it is found that a large part of the error detection samples have relatively short opcode sequence length, such as E O E ECHO RETURN, and its distribution is shown in Fig. 6. In fact, the short opcode sequence means that it contains less information, and it is likely that the original text samples of positive and negative WebShell will get the same sequence after opcode. In addition, there is a sample imbalance in the training data itself, which has a negative impact on the detection performance of the model.

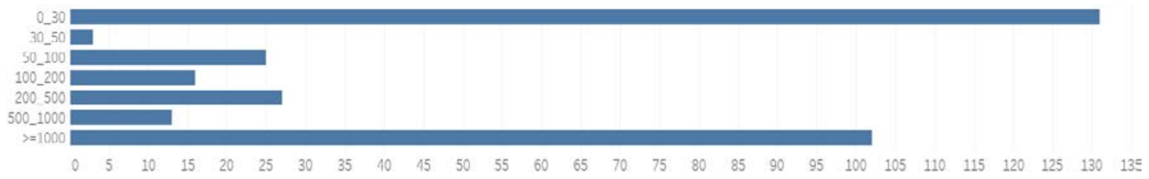

Fig. 6. The length distribution of error detection sample opcode sequence

4.3 CNN Plus Model Optimization

The samples that affect the detection performance in the opcode sample set are eliminated, that is, the samples whose opcode sequence length is less than or equal to 30 do not participate in the training and prediction of the CNN model, and all this above is named as the CNN Plus model. A total of 1049 samples were removed from the experiment and Machine Learning training was carried out again. The experimental results are shown in Table 4.

Table 4 Experimental results of the CNN Plus model

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1_score**
CNN	0.97316	0.959055	0.94224	0.95057
Sample : The opcode sequence length>30				

Experimental results show that the opcode sequence length is less than or equal to 30 samples after excluding and compared to before, CNN's classification performance is greatly improved, with Accuracy improved by 1.5%, Recall value improved closing to 7%, and F1 value increased by 3%. Among them, 78 Webshell white samples and 112 Webshell black samples were missed in 5140 Webshell white samples and 1938 Webshell black samples.
The performance comparison between the CNN detection model of WebShell attack and CNN Plus detection model is shown in Fig. 7.

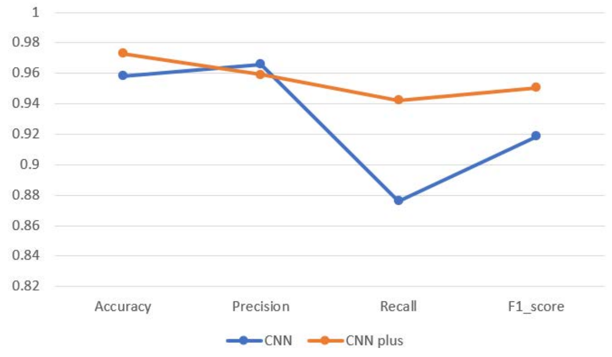

Fig. 7. Comparison of detection performance between CNN model and CNN Plus model

5 Webshell Mixed Detection Model

5.1 Description of Problem

In the forward experiment, "text sequence + word bag" was used as a feature engineering, and classification algorithms of NB, MLP, SVM, RF and Xgboost were used together. In which, the RF model showed the best detection performance. Although the detection performance of CNN model which was obtained by "Opcode sequence + vocabulary" as a feature engineering ranks the third, its performance index is not much different from that of RF model. The key point is that, the CNN Plus model, obtained from WebShell samples with eliminated opcode sequence of length less than or equal to 30, its detection performance is greatly improved. Table 5 shows performance comparisons of CNN, CNN Plus and RF detection models.

Table 5 Performance comparison of CNN, CNN Plus and RF detection model

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1_score**
CNN	0.95825	0.96596	0.87603	0.9188
CNN plus	0.97316	0.95906	0.94224	0.95057
RF	0.96827	0.98569	0.898	0.9398

The experimental results in Table 5 show that CNN Plus has the optimal WebShell detection performance. But CNN Plus has the following problems:
1) WebShell samples that fail in opcode cannot be detected;
2) WebShell samples with opcode sequence length less than or equal to 30 cannot be detected.

5.2 Mixed detection model solution

In general consideration, in order to obtain the best WebShell detection performance, this paper proposes a mixed detection model scheme, that is, the CNN Plus model is used to identify samples whose opcode sequence length is more than 30, the RF model identifies samples which fail in opcode and the samples whose opcode sequence length is less than 30, as shown in Fig. 8.

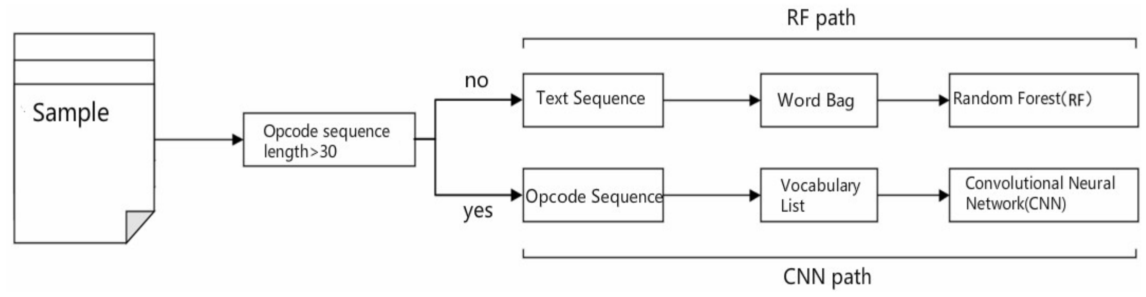

Fig. 8. WebShell mixed detection model

5.3 Experimental results of mixed detection model

Experimental design of the mixed model: the data set was divided with the opcode sequence length of 30 as the boundary (the sample length of the failed opcode was 0). The data set of the Webshell white sample and the Webshell black sample was divided as shown in Table 6. For sample sets with opcode sequence length less than or equal to 30, the corresponding WebShell original samples are taken for training and testing the RF model. The training sets and test sets are divided by 1:1.WebShell sample sets with opcode sequence length more than 30 are used for training and testing of the CNN Plus model. The training sets and test sets are divided by 1:1.

The experimental results are shown in Table 7. The CNN plus detection model detected 5140 Webshell white samples with 78 of which were missed and 1938 Webshell black samples with 112 of which were missed. The RF detection model detected 497 Webshell white samples with 5 of which were missed and 208 Webshell black samples with 36 of which were missed.

Table 6 Data set partitioning for mixed models

Type / Number	<=30	>30
Webshell White Sample	977	10284
Webshell Black Sample	432	3974

Table 7 Experimental results of mixed detection model

Model	Accuracy	Precision	Recall	F1_score
Mixed Detection Model	0.97032	0.96013	0.93107	0.94537

6 Sample Balance and Optimization

6.1 Description of problem

The mixed model proposed in this paper achieves the best detection performance, but the identification accuracy of minority class samples is lower than that of majority class samples, whether it is CNN path or RF path. For example, there were 5140 Webshell white samples in CNN test path, 78 of which were missed, and the identification accuracy was 0.98482. There were 1939 Webshell black samples were tested in CNN path with 112 missed samples, and the identification accuracy was 0.94223. Black and white sample identification accuracy difference is 4 percentage points. There were 497 Webshell white samples in RF test path, 5 of which were missed, and the identification accuracy was 0.99003. There were 208 Webshell black samples were tested in RF path with 36 missed samples, and the identification accuracy was 0.82692. Black and white sample identification accuracy difference is 17 percentage points. This paper believes that one of the main reasons for this situation is the imbalance of training data. In particular, RF path more obviously reflects the difference in detection accuracy caused by sample imbalance. In this paper, a variety of oversampling and undersampling methods are tried to make the samples as balanced as possible.

6.2 RF Model

Oversampling is used to solve the problem of sample imbalance, and the experimental results are shown in Table 8. From the perspective of various indicators, the best effect can be obtained by combining SVMSMOTE[14-16]: Accuracy, F1 value increased by 1%, while Recall value increased by 5.6%, the identification Accuracy of Webshell white sample of 0.98579, and Webshell black sample identification Accuracy of 0.88. The experimental data shows that after balancing the sample by SVMSMOTE oversampling algorithm, the problem of minority sample identification accuracy lower than that of the majority sample is effectively alleviative.

When using the undersampling way to solve the problem of unbalanced samples, compared with the original way, the optimization effect is not obvious. Either is no improvement of the differences in identification accuracy between categories, or a big fall in the overall recognition rate, which proves the undersampling method combined with RF path can't solve the problems due to unbalanced samples very well, the results as shown in Table 9. This was predictable because the RF path had very few training samples and was not suitable for undersampling.

Table 8 The experimental results of RF path with oversampling

Fit Sample Method	Accuracy	Precision	Recall	F1_score
SMOTE	0.95036	0.95288	0.875	0.91228
Borderline SMOTE	0.89929	0.76245	0.95673	0.84861
ADASYN	0.89645	0.75281	0.96635	0.84632
SMOTE+ENN	0.93475	0.92188	0.85096	0.885
SMOTE Tomek	0.9461	0.94271	0.87019	0.905
SVMSMOTE	0.95036	0.94819	0.87981	0.91272
K-Means SMOTE	0.94184	0.97175	0.82692	0.89351

Table 9 The experimental results of RF path with undersampling

Fit Sample Method	Accuracy	Precision	Recall	F1_score
Cluster Centroids	0.84681	0.66779	0.95673	0.78656
Random Under Sampler	0.94326	0.92424	0.87981	0.90148
Near Miss	0.94043	0.94624	0.84615	0.8934
Neighborhood Cleaning Rule	0.90638	0.78175	0.94712	0.85652

6.3 CNN Plus Model

Oversampling is used to solve the problem of sample imbalance, and the experimental results are shown in Table 10. From the perspective of various indicators, the best effect can be obtained by combining the Borderline SMOTE [14] [17-18] : Accuracy and F1 value have a small increase, while Recall value increased by nearly 1%. The recognition Accuracy of normal samples is 0.98151, and that of the Webshell sample is 0.95152, which effectively alleviate the problem that the recognition Accuracy of a few samples is lower than that of the majority.

Table 10 The experimental results of cnn path with oversampling

Fit Sample Method	Accuracy	Precision	Recall	F1_score
SMOTE	0.97175	0.94982	0.94688	0.94835
Borderline SMOTE	0.9733	0.95103	0.95152	0.95128
ADASYN	0.97358	0.96203	0.94069	0.95124
SMOTE+ENN	0.9726	0.96042	0.93863	0.9494
SMOTE Tomek	0.97118	0.93791	0.95823	0.94796
SVMSMOTE	0.97231	0.96136	0.93657	0.9488
K-Means SMOTE	0.97358	0.96695	0.93553	0.95098

However, when the undersampling method is used to intervene the sample, the sample recognition accuracy between categories does not improve, that is, the Recall value does not change much, which proves that the undersampling method combined with the CNN path is not a good solution to the problem caused by the imbalance sample. The experimental results are shown in Table 11.

Table 11 The experimental results of cnn path with UNDERsampling

Fit Sample Method	Accuracy	Precision	Recall	F1_score
Cluster Centroids	0.97415	0.96653	0.93811	0.95211
Random Under Sampler	0.97274	0.96685	0.93244	0.94933
Near Miss	0.9709	0.96362	0.92883	0.9459
Neighborhood Cleaning Rule	0.97189	0.95886	0.9376	0.94811

6.4 Optimized hybrid detection model

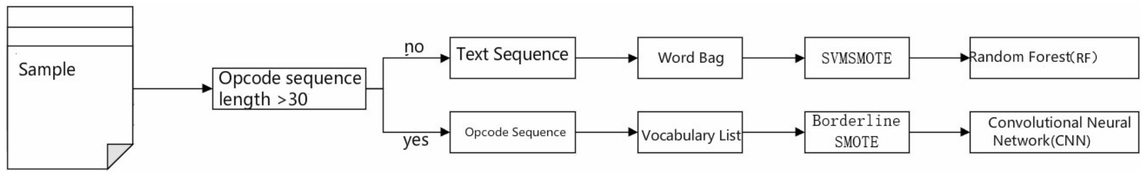

Fig. 9. Webshell detection model

Table 12 The experimental results of the optimal model

Detection Model	Accuracy	Precision	Recall	F1_score
Optimal Model	0.97122	0.95077	0.94457	0.9477

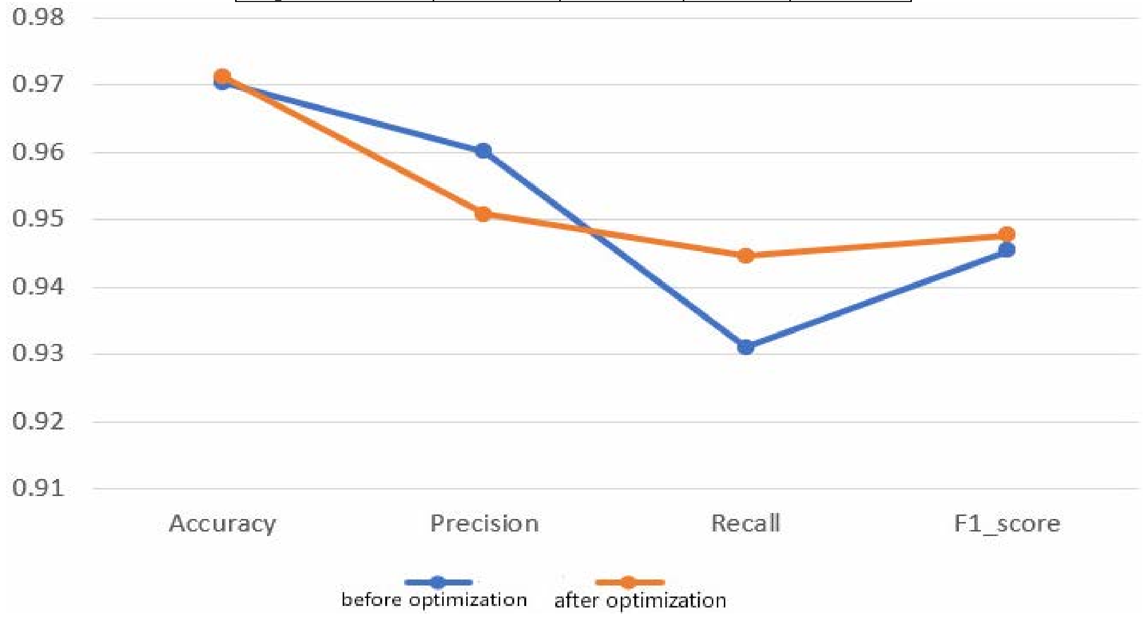

Fig. 10. Comparison of results before and after balancing and optimization

Through theory and a lot of experimental support, the optimal Webshell detection model was obtained in this paper, as shown in Fig. 9, that is, for the samples of opcode failure and the samples with the length of opcode sequence less than or equal to 30, the way is to use the word bag model as the characteristic engineering in combination with oversampling algorithm of SVMSMOTE, and using the random forest (RF) to detect. For the sample with opcode sequence length more than 30, the way is to use the vocabulary as the characteristic engineering in combination with the Borderline SMOTE oversampling algorithm, and using convolutional neural network (CNN) to detect. The detection performance of this detection scheme is shown in Table 10. It can be seen from Fig. 10. that after balancing and optimizing sample, the detection performance of the model is better.

7 Conclusion

This paper proposes a model to detect Webshell effectively. According to the different situation, this model is used to detect the difference, that is, all samples will be opcode processed. And for samples with the opcode failure or opcode sequence length less than and equal to 30 samples, the word bag model is used as the feature engineering, in combination with SVMSMOTE sampling algorithm and random forest (RF) to detect. For the sample with opcode sequence length more than 30, use the vocabulary as the feature engineering in combination with the Borderline SMOTE algorithm, using convolutional neural network (CNN) to detect. The feasibility of the model has been verified by a large number of experiments. When WebShell detection is carried out, the recognition accuracy of the model can be greater than 97%, and the problem of low identification accuracy of a small number of samples caused by sample imbalance can be effectively solved.

In Webshell attack detection, traditional detection methods rely on rule matching of feature base and generally have problems such as easy to avoid, high error rate, hysteresis and high maintenance cost. In particular, the traditional methods are difficult to detect the mutant WebShell attack. Attackers even use intelligent algorithms for WebShell mutation. The mixed detection model based on Machine Learning has the characteristics of fast running speed, low maintenance cost and high detection rate of mutant WebShell attack samples. The detection accuracy of mixed model for Webshell attack is up to 97%, which provides a new way to solve the problem of difficult detection of mutant WebShell.

ACKNOWLEDGMENTS

Upon the completion of the thesis, I would like to take this opportunity to express my sincere gratitude to my workmates. Without our efforts, this thesis would have been impossible. This research was funded in part by NATURAL SCIENCE FOUNDATION OF FUJIAN PROVINCE OF CHINA, grant number 2018H0025, in part by XIAMEN SCIENCE AND TECHNOLOGY PROJECT, grant number 3502Z20183035, 3502Z20183037, 3502Z20183038. Whether in the capital or equipment, the project teams has given us great support. Their help have accompanied us though all the courses of the thesis.

REFERENCES

[1] Long Xiao, Fang Yong, Huang Cheng, et al. A survey on webshell: game of detection and escape[J]. Cyberspace Security, 2018, 9(01): 62-68.)4
[2] Martin M, Lam M S. Automatic generation of XSS and SQL injection attacks with goal-directed model checking[C]. San Jose, CA, United states: USENIX Association, 2008.

[3] Kiezun A, Guo P J, Jayaraman K, et al. Automatic creation of SQL injection and cross-site scripting attacks[C]. Vancouver, BC, Canada: IEEE Computer Society, 2009.

[4] Balzarotti D, Cova M, Felmetsger V, et al. Saner: Composing static and dynamic analysis to validate sanitization in web applications[C]. Oakland, CA, United states: Institute of Electrical and Electronics Engineers Inc., 2008.R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[5] Hong JunBin. Research and Implementation of Web Application Vulnerabilities Detecting Technology[D]. Guangdong: Guangdong University of Technology, 2016.

[6] Dinh Tu T, Guang C, Xiaojun G, et al. Webshell detection techniques in web applications[C]. Hefei, China: Institute of Electrical and Electronics Engineers Inc., 2014.

[7] Zhang Hewei, Liu Xiaojie, Php-Webshell detection method based on the text vector[J].Data Communication, 2019(04):16-21.

[8] Hu Biwei. Research on Webshell Detection Method Based on Bayesian Theory[J]. Science Mosaic, 2016(06): 66-70.

[9] XU Xiaobo NIE Xiaoming. A Method of Detecting WebShell Based on Multi-layer Perception[J]. Communications Technology, 2018, 51(04): 895-900.

[10] MENG Zheng, MEI Rui, ZHANG Tao, et al. Research of Linux WebShell Detection based on SVM Classifier[J]. Netinfo Security, 2014(05): 5-9.

[11] Ye Fei, Gong Jian, Yang Wang. Black Box Detection of Webshell Based on Support Vector Machine[J]. Journal of Nanjing University of Aeronautics & Astronautics, 2015, 47(06): 924-930.

[12] Jia Wenchao, Qi Lanlan, Shi Fan, et al. WebShell detection method based on random forest improved algorithm[J]. Application Research of Computers, 2018, 35(05): 1558-1561.

[13] CUI Yanpeng, SHI Kexing, HU Jianwei. Research of Webshell Detection Method Based on XGBoost Algorithm[J]. Computer Science, 2018, 45(S1): 375-379.

[14] Li Chunxue, Xie Linsen, Lu Chengbo. An undersampling method based on clustering for unbalanced data sets[J]. Maths Practice and knowledge, 2019,49(01):203-209.

[15] Shi Hongbo. Chen Yuwen, Chen Xin. Review on SMOTE oversampling and its improved algorithm[J]. Journal of Intelligent Systems, 2019,14(06):1073-1083.[doi:10.11992/tis.201906052]

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[17] H. Han, W.-Y. Wang, B.-H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," In Proceedings of the 1st International Conference on Intelligent Computing, pp. 878-887, 2005.

[18] Hongyun Qin, Houpan Zhou, Jiuwen Cao. Imbalanced learning algorithm based intelligent abnormal electricity consumption detection [J]. Neurocomputing, 2020. DOI: 10.1016/j.neucom.2020.03.085.