Executable Program Feature Cleaning and Forgery Technology Based on Machine Learning

Jiahong Wang, Yibo Chang*
College of Electronic Engineering, National University of Defense Technology, Hefei, Anhui, China
**Corresponding Author.*

Abstract: As we delve into the digital age, the rapid evolution of information technology has brought about a surge in network security threats. Among the various tools used by cybercriminals, executable programs stand out as particularly potent vectors for malicious activities. The intricate nature of these programs means that their feature cleaning and forgery technologies are under intense scrutiny. Traditional methods of feature cleaning and forgery, while once effective, now face significant limitations in the face of the ever-evolving complexity of network security landscapes. In the last few years, machine learning has emerged as a groundbreaking force in cybersecurity, offering innovative solutions to the challenges posed by executable program threats. Our study introduces a novel approach to feature cleaning and forgery, grounded in the principles of machine learning. Through rigorous experimental verification, we have demonstrated that this method not only achieves high accuracy but also proves to be highly practical in the real-world application of feature cleaning and forgery. The aim of this research is to delve deeper into the feature cleaning and forgery technologies of executable programs, leveraging machine learning to pave the way for more robust and effective means of network security protection. The purpose of this study is to explore the cleaning and counterfeiting technology to provide effective means for network security protection.

Keywords: Machine Learning; Executable Program; Feature Cleaning; Forgery Technology; Network Security

1. Feature Cleaning Technology of Executable Programs

1.1 Feature Extraction Method

At present, there are mainly two methods for feature extraction of executable programs: one is based on the features during the program's runtime, including static analysis methods and dynamic analysis methods. Static analysis refers to obtaining the features of the program during runtime by analyzing the execution behavior of the program [1]. Dynamic analysis means that during the dynamic execution process, according to the changes in the code within the program, the features of the program during runtime are dynamically extracted [2]. The feature extraction methods based on static analysis and dynamic analysis are mainly aimed at removing or filtering out the unnecessary information and retaining the most useful information. The feature extraction method based on machine learning algorithms can classify unknown samples through a learning model, and retain more valuable information from the known samples, thus better meeting the application requirements [3].

1.2 Feature Selection and Dimensionality Reduction

In the analysis of executable programs, feature selection is an important task. When performing feature selection, the issue of dimensionality reduction is the first consideration. In the field of machine learning, dimensionality reduction refers to reducing the dimensionality of a dataset from a high-dimensional space to a low-dimensional space, with the aim of improving the efficiency and accuracy of algorithms [2]. The choice of dimensionality reduction method has a great impact on the effect of executable program analysis. Common selection methods include the minimum distance method, correlation coefficient method, mutual information method, etc. The purpose of feature selection is to select features that have little impact on the analysis results and can effectively reduce the algorithm complexity while maintaining the effectiveness of the algorithm.

Therefore, in the analysis of executable programs, it is often necessary to select and reduce the dimensionality of existing features. Commonly used feature selection methods include Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Multi-class Support Vector Machine (SVM). Principal Component Analysis (PCA) is a classic unsupervised dimensionality reduction method. Its core idea is to project high-dimensional data onto a low-dimensional space through a linear transformation while retaining as much of the main information of the data as possible. Specifically, PCA calculates the covariance matrix of the data, finds the directions (principal components) with the largest variance in the data, and projects the data onto these directions to achieve dimensionality reduction. The advantages of this method include small computational complexity, comprehensive feature selection, and it is suitable for handling large-scale data. However, it is sensitive to outliers, which may affect the selection of principal components [2]. The formula of the PCA algorithm is as follows:

$$W = \arg\max_{W} \frac{1}{N} \sum_{i=1}^{N} (W^T x_i)(W^T x_i)^T \quad (1)$$

Among them, W is the projection matrix, x i is the sample data, and N is the number of samples. The Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction method. Its core idea is to find a projection direction by maximizing the between-class distance and minimizing the within-class distance, so that the projected data can distinguish different classes to the greatest extent [1]. The LDA algorithm realizes variable selection by calculating the distances between the variables in the original dataset and the variables after projection. Its advantages include relatively low computational complexity, and the projection direction has a clear discriminative ability, making it suitable for classification tasks and small to medium-scale data. The disadvantages are that it has relatively strict assumptions about the data distribution (assuming that the data follows a Gaussian distribution), and when the number of classes is large, the computational complexity will increase significantly. LDA is usually used for feature extraction and dimensionality reduction in classification tasks, for example, in fields such as biometric identification, text classification, and medical diagnosis. The formula of the LDA algorithm is as follows:

$$w = \arg\max_{w} \frac{w^T S_b w}{w^T S_w w} \quad (2)$$

Among them, S_b is the between-class scatter matrix, S_w is the within-class scatter matrix.

The Support Vector Machine (SVM) is a supervised learning algorithm based on statistical learning theory, which is mainly used for classification and regression tasks. In feature selection, SVM can select the features that have a greater impact on the classification results by maximizing the classification margin, demonstrating excellent performance. The advantages of this algorithm are as follows: it has good generalization ability for high-dimensional data, can effectively handle non-linear data, has high classification performance, strong robustness, and is suitable for dealing with small sample data. However, this algorithm has a relatively high computational complexity, especially when dealing with large-scale datasets. The selection of the kernel function and the adjustment of parameters have a significant impact on the performance of the model. It is also relatively sensitive to noisy data. SVM is widely applied in fields such as image recognition, text classification, and bioinformatics. The formula of the SVM algorithm is as follows:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \quad (3)$$

Satisfy the constraint conditions:

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0$$

Among them, w is the weight vector, b Is the bias term, C is the penalty parameter and ξ_i is the relaxation variable.

1.3 Design and Implementation of Feature Cleaning Algorithm

To clean the data, this paper employs machine learning methods. In the feature space of the program, two classifiers are used: decision trees and random forests. Both decision trees and random forests classify data according to certain specific rules. A decision tree is a linear classifier. Its principle is to take each element in the data as a feature, calculate the distance between each feature and all other elements, and select the feature with the minimum distance as the classification label. A random forest is a non - linear classifier. Its principle is to automatically select the optimal classifier to classify data by learning a large number of training samples. Due to the presence of a large amount of noise in the program, this paper

adopts the Support Vector Machine (SVM) as the feature cleaning algorithm. SVM is a single - objective linear model that performs well when the data is imbalanced [4].

2. Forgery Technology Based on Machine Learning

2.1 Principle of Forgery Technology

Machine learning is an important branch of artificial intelligence. It uses a large amount of data (i.e., a dataset), extracts useful information from the dataset through certain algorithms, and ultimately achieves the automatic learning of certain rules or patterns [5]. The main purpose of machine learning is to train a system to recognize patterns and identify new patterns based on these recognized patterns. For example, machine learning automatically learns the patterns in the data by minimizing the loss function. Given a training set

$$D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\} \quad (4)$$

Among them, x_i is the sample characteristic, y_i Is the label, The goal of the model is to find the parameter that minimizes the prediction error. The commonly used Mean Squared Error (MSE) loss function is:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f_\theta(x_i))^2 \quad (5)$$

Among them $f_\theta(x_i)$ is the prediction output of the model. Update the parameters iteratively through the gradient descent algorithm:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L\,\theta_t) \quad (6)$$

Here, η is the learning rate, ∇_θ Represents the gradient. This optimization process enables the model to extract features from the data and generate the key patterns of forged programs [5,6].

The forgery technology based on machine learning employs machine learning algorithms to conduct feature analysis on the target executable programs, extract and classify the features of the target executable programs, and then design new executable programs according to these features. Leveraging machine learning algorithms to automatically learn from a large number of sample data is currently the most commonly used forgery technique [6]. In machine learning, each sample can serve as part of the training set, and each new sample can be used as a test - set. By training the system, it can automatically recognize patterns or rules through learning from the sample data. This learning process is automatic and does not require human intervention. Through this approach, machine learning algorithms can extract useful information from the data. For example, in the field of image recognition, classification algorithms can be used to categorize images into different classes. These classification algorithms mainly rely on a large amount of training data for training to identify patterns or rules.

2.2 Design and Implementation of the Forgery Method

Machine learning methods are employed to construct models of false executable programs for fitting the input dataset. In machine learning, the input dataset is a collection of data that can depict the characteristics of the input data. For example, in the field of image recognition, the features in the input data can be represented by the pixel values of the images, and then a mapping relationship between the input features and the output features can be constructed. In machine learning, the output result is a classification label, which can be used to classify the categories in the input data set. An executable program contains a large number of operation sequences, including startup, suspension, invocation, etc. These operation sequences may be controlled by different users or not controlled by users during execution. The existence of these operation sequences in the executable program is a kind of feature. For example, in a classification task, feature mapping can be achieved through logistic regression. Let the input feature vector be x, and the output label be a binary classification result.

$$y \in \{0,1\} \quad (7)$$

Then the hypothesis function is:

$$h_\theta(x) = \sigma(\theta^\top x) = \frac{1}{1+e^{-\theta^\top x}} \quad (8)$$

Where σ is the sigmoid function, θ for the model parameters. By maximizing the log-likelihood function:

$$\max_\theta \sum_{i=1}^{n}\left[y_i \log h_\theta(x_i) + (1-y_i)\log\left(1-h_\theta(x_i)\right)\right] \quad (9)$$

The model learns the mapping relationship between the input features (such as the operation sequences of the executable program) and the malicious behavior labels, thus generating forged programs with similar features [7]. Therefore, the features in the executable program can be treated as a special kind of data sample.

2.3 Application Scenarios of the Forgery Technology

In the current fields of software development and network security, forgery technology has become an important research direction. The technical solution proposed in this paper mainly focuses on the detection and analysis of malicious behaviors in computer programs, and it has practical application values in multiple aspects. Specifically, this technology is mainly reflected in the following key application levels. Firstly, this technology can effectively identify malicious code. As is well known, malicious code often contains a large number of functions, and these functions may be carefully designed by malicious hackers to hide their true intentions. By applying machine learning algorithms, we can quickly identify and locate these suspicious functions, and then conduct in-depth analysis of them to find out the vulnerabilities or abnormal behaviors in the code, providing a basis for subsequent security reinforcement. Secondly, it can also be used to tamper with the program logic [1]. The program will process the user input data accordingly during operation. Through machine learning technology, we can discover some potential defects or vulnerabilities in these data processing logics. By taking advantage of these logical loopholes, attackers can construct deceptive means to achieve control or interference with the program, so as to achieve their malicious purposes. Of course, equally importantly, this technology can also be used to verify the reliability of existing programs. By inputting test samples into existing programs and observing whether the programs can run normally, machine learning algorithms can evaluate the stability, security and other related characteristics of existing programs. This method can be used not only to discover and fix known problems, but also to explore new functions or improve old functions to ensure the long-term stable operation of the program.

3. Case Analysis

3.1 Case Selection and Background Introduction

An Internet financial platform, due to the needs of business development, urgently needs to deploy the security policies required by the financial platform in a certain key business system. To this end, the financial platform has designed a set of security policies for the business system and deployed the relevant security policies. However, after the deployment of these security policies, some malicious software emerged in the network of this business system. These malicious software appear in the network in various forms, including but not limited to phishing websites, Trojan viruses, etc. These malicious software usually launch attacks on some sensitive resources within the system. For example, a certain business system requires logging in with a specific account or accessing other sensitive resources. If this account or other resources are stolen by attackers, the account or other resources will be used for other purposes. Therefore, this business system is facing the risk of network attacks.

3.2 The Implementation Process of the Case

First, through static analysis, we found that there was a mismatch between the input parameters and output results of the program, indicating a problem of feature redundancy. Then, we combined machine - learning algorithms to extract and clean the corresponding features from the input parameters and output results of the program. The extracted features mainly include three categories: program - called functions, input parameters, and output results. Next, we selected features by analyzing the differences between the feature vectors corresponding to each feature and the actual output result vectors of the program. Finally, we classified and identified the cleaned features using machine - learning algorithms and performed data cleaning based on the classification and identification results. We used multiple classification algorithms, including the K - Nearest Neighbors (KNN) algorithm, the Naive Bayes algorithm, and the Support Vector Machine (SVM) algorithm, to classify and identify the cleaned data [8]. Among them, the Naive Bayes algorithm is a classification algorithm based on probability theory. When classifying a dataset, it first selects features from the dataset, then builds a classifier, and finally obtains a good classification result through learning from the training samples. The Support Vector Machine algorithm is a classification algorithm based on statistical learning theory. It has strong generalization ability, high accuracy, and good robustness.

3.3 Case Achievements and Enlightenments

In this case, we have successfully constructed an intelligent system that can automatically achieve feature cleaning and forgery, which effectively

improves the detection efficiency of executable programs. Compared with the traditional manual feature extraction and analysis methods, the method based on machine learning is adopted in this case [9]. Machine learning is a new machine learning theory, which predicts unknown data by learning from a large amount of data. The traditional feature extraction method mainly relies on manual operations, and this approach requires a large amount of manual processing procedures [10]. In this case, we have realized the feature cleaning and forgery of executable programs through the machine learning-based feature extraction technology. This system can not only automatically identify the malicious code features existing in executable programs, but also effectively and automatically identify the features of unknown malicious codes [11]. By using the machine learning-based method, this system can not only extract the features of a large number of executable programs, but also extract the features of unknown malicious codes to automatically identify executable programs. The machine learning technology in this system is capable of automatically identifying the features of unknown malicious codes.

4. Conclusion

In this study, an executable program feature cleaning and forgery technology based on machine learning has been proposed, and the effectiveness of the proposed method has been verified through experiments. The research results show that machine learning technology has great application potential in the feature cleaning and forgery of executable programs. However, this study still has certain limitations, such as the optimization of feature extraction and selection, and the further research on forgery technology. The future research directions include: expanding the application of machine learning algorithms in the field of network security, improving the practicality of feature cleaning and forgery technology, and exploring more effective network security protection strategies. It is hoped that this study can provide useful references for the technological development in the field of network security.

References

[1] Xiaoyu Liu, Haichen Zhu. Research on New PE File Packer and Shelling Methods//International Conference on Information Sciences, Machinery, Materials and Energy (ICISMME 2015). 2015-04-11.

[2] Pillay, K.D.A. Relocating loader for MS-DOS. EXE executable files. Microprocessors and Microsystems, 1990, Vol.14(7): 427-434.

[3] Christian Janiesch, Patrick Zschech, Kai Heinrich. Machine learning and deep learning. Electronic Markets, 2021, Vol. 31(3): 685-695.

[4] T. Sangeetha, T. Meyappan, M. Phil. New Technique of Hidden Data in PE-File with in Unused Area Two. International Journal of Engineering Trends and Technology, 2012, Vol. 3(3): 304-310.

[5] A. A. Zaidan, B. B. Zaidan, Shihab A. Hameed. Novel Approach for Secure Cover File of Hidden Data in the Unused Area within EXE File Using Computation between Cryptography and Steganography. International Journal of Computer Science & Network Security, 2009, Vol. 9(5): 294-300.

[6] Madani, Houria, Ouerdi, et al. Ransomware: Analysis of Encrypted Files. International Journal of Advanced Computer Science and Applications, 2023, Vol. 14(1): 213-217.

[7] Stanislav Igorevich Shterenberg, Andrey Vladimirovich Krasov, Igor Aleksandrovich Ushakov. Analysis of Using Equivalent Instructions at The Hidden Embedding of Information into the Executable Files. Journal of Theoretical and Applied Information Technology, 2015, Vol. 80(1): 28-34.

[8] De Prado, Paula Martin. Machine learning. Biologist, 2024, Vol. 71(1): 8-9.

[9] Lenover, Michael, Bedi, et al. Improvements to a Machine Learning Machining Feature Recognition System. Computer-Aided Design & Applications, 2025, Vol. 22(1): 119-135.

[10] A. A. Zaidan, B. B. Zaidan, Hamid. A. Jalab. A New System for Hiding Data within (Unused Area Two + Image Page) of Portable Executable File using Statistical Technique and Advance Encryption Standared. International Journal of Computer Theory & Engineering, 2010, Vol. 2(2).

[11] Nisha P. Shetty, Ikhil Ranjan. Using Steganography & Cryptography to Hide Data in EXE Files. International Journal of Engineering & Technology, 2018, Vol. 7.

