2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA) | 978-1-6654-9991-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICAICA54878.2022.9844642

An Android Malware Detection Method Using Deep Learning based on Multi-features

Menglong Chen
School of Computer Science and Cyber Engineering
Guangzhou University
Guangzhou, China
1654639685@qq.com

Quan Zhou*
School of Mathematics and Information Science
Guangzhou University
Guangzhou, China
zhouqq@gzhu.edu.cn
*Corresponding author

Kemeng Wang
School of Computer Science and Cyber Engineering
Guangzhou University
Guangzhou, China
956250234@qq.com

Zhikang Zeng
School of mathematics and Information Science
Guangzhou University
Guangzhou, China
13414799390@163.com

***Abstract*—As we all know, android is an operating system used by all walks of life, and many developers have developed a large number of applications. Because of its openness, it is becoming the target of malware, which may pose a serious security threat to all aspects of life. In order to deal with these security threats, we propose an excellent malware detection method based on text classification. This method uses androguard tool to extract permission, service, receiver and intent from android application package, and combines them into a text report representing the android application. We apply BiLSTM network to mine important information in the text. The accuracy of our experiment is 97.47%, so our system can effectively detect Android malware.**

Keywords—android malware detection, deep learning, text classification

I. Introduction

Due to the deep development of internet, mobile phone has entered all aspects of life. Android operating system is widely used in various industries. Given the openness and popularity of android, it is often attacked by malware, which will have an extremely serious impact, leading to file destruction, information theft, data leakage, national security and terrorist events [1], [2]. The security of the android operating system cannot be underestimated. Given a report released by Kaspersky [3], although the number of malware attacking personal terminals and industrial operating systems decreased in 2021 compared with 2020, its number also reached the level of one million.

Due to the popularity of android operating system, most industries are using Android operating system, so Android malware came into being. The Android operating system allows users to install software downloaded from third-party platforms. Attackers distribute the malware to the third-party application store. If users download the software package from the third-party application store, the mobile phone is likely to be infected.

In our paper, The method we use is text processing technology combined with deep learning. This method is based on APK's static analysis report. Due to the powerful ability of neural network structure to extract features, there is no need to extract features manually. Each report consists of many sentences with specific content, which are vectorized by word2vec. We use the BiLSTM algorithm to analysis the report contentt, and expect that this BiLSTM algorithm can deeply excavate the features of the sentence. Our experimental result proves the effectiveness of the method.

II. Related Work

In this part, We will introduce the development of Android malware in recent years. The feature in these works can be classified into three categories: static feature, dynamic feature and hybrid feature. The three types of features correspond to three analysis methods

A. Static Analysis

Static analysis method refers to using the static information in the document to identify malware when the application is not executed.

Shudong Li et al. [4] detect malware from the perspective of binary files and extracts string features of malware.Their experiments prove that string features detection of malware is excellent. Enrico Mariconti et al. [5] used Markov chains to represent API call sequences and extracted features, and then classified Malware with different algorithms.

B. Dynamic Analysis

Dynamic analysis method refers to constructing a safe and controlled environment, monitoring and recording its behavior mode, and realizing the dynamic analysis of files.

P. Feng, J. Ma, C. Sun, X. Xu and Y. Ma [6] extracted various behavior features, then used the artificial intelligence algorithm to judge its maliciousness. When applying integration learning stack, it performs well in malware detection. Meihua fan et al.[7] used malicious behavior to create a heterogeneous graph and used graph algorithm to detect malware.

C. Hybird Analysis

Hybrid analysis method refers to the use of static and dynamic characteristics, and make full use of their advantages.

M. Spreitzenbarth, T. Schreck, F. Echtler, D. Arp and J. Hoffmann[8] combine static and dynamic features to detect Android malware from a more comprehensive perspective.

III. Proposed Methodology

In this part, We will introduce our proposed method, which deals with Android malware from the perspective of text classification. As shown in Fig.1 below. The new method is composed of three main components: feature extraction, feature vectorization and text classification, that will also be introduced below.

A. Preliminaries

It is essential for us to introduce the internal structure of Android application for understanding Android malware detection. Android software installation package may be equivalent to compressed (zip) or archive files, which are composed of some significant files. The five main composition of APK file are: METF-INF, androidmanifest.xml, resources.arsc, Classes.dex and res.

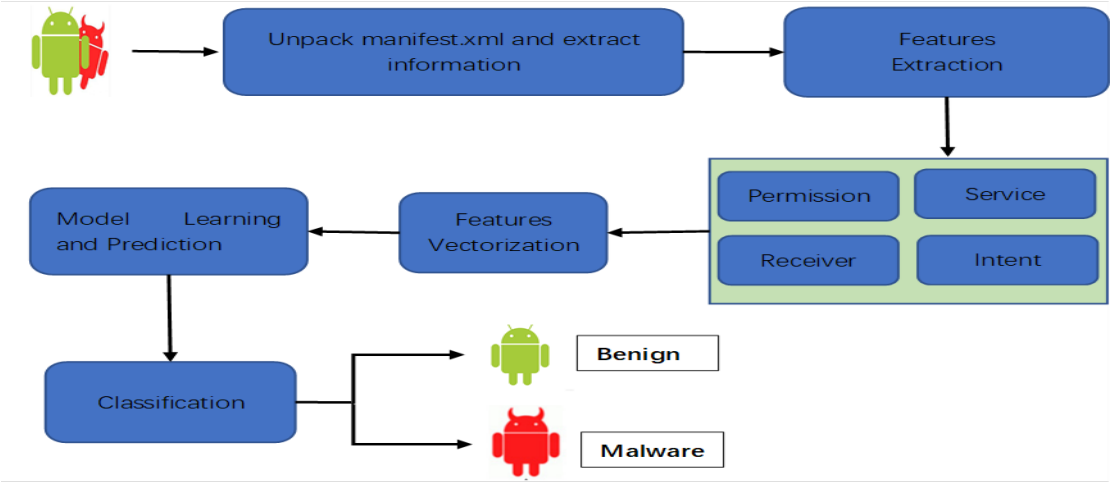

Fig. 1. Framework android malware detection

AndroidManifest.xml: For Android applications, the permission information that needs to be applied and the registration information of activities, services and other components need to be in androidmainfest.xml. Androidmanifest.xml contains important information that android system must master before running programs, including service, intent, receiver, application name, icon, package name, module composition and authorization

B. Feature Extraction

In order to detect Android malware, We need very valuable features to describe Android software. In order to get the features we need from the Android Software installation package, we use the androdguard tool to extract features. We selected four components as our features: permission, service, receiver and intent. These four features will be introduced below.

1)Permission: Permission is a security mechanism. Android permissions are mainly used to restrict the use of some restrictive functions within applications and component access between applications.

2)Service: It is one of the four major components. It cannot run by itself, can only run in the background, and can interact with other components.

3)Receiver: It is one of the four components and a mechanism widely used to transmit information between applications. It transmits our data by sending intent.

4)Intent: Intent is a kind of runtime binding mechanism, which can connect two different components during the process of program running.

C. Feature Vectorization

We use word2vec [9] algorithm to process sentence into vector, that can achieve very good results for text processing. In 2013, Google created the Word2vec algorithm that converts text into word vectors. Word2vec is a lightweight network. It is similar to the structure of deep learning neural network, which is also composed of several network layers. The composition of the network is relatively simple, which is composed of three common layers: input layer, hidden layer and output layer. The network mainly includes cbow and skip gram model according to different input and output.Through this tool, we convert permission, service, receiver and intent into word vectors.

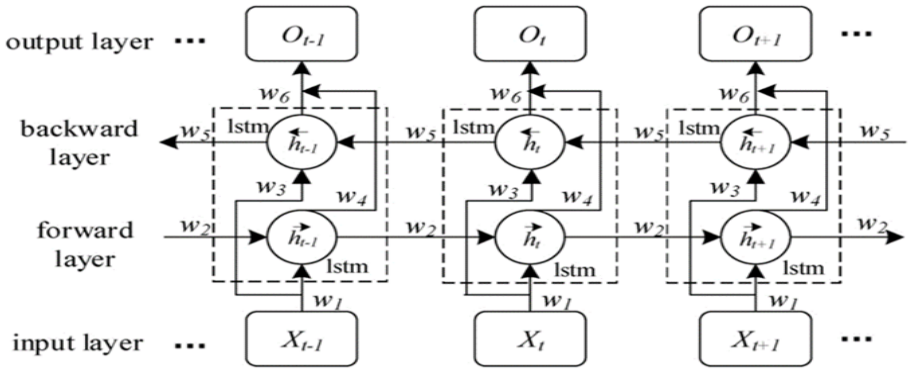

Fig. 2. The BiLSTM classification model

D. Classification Model

In view of the advanced natural pretreatment technology, we use natural language processing technology to deal with malware detection. We will use the BiLSTM introduced in [10] to complete the text classification problem. It is mature

enough for BiLSTM algorithm to deal with text classification. This algorithm is capable of fully understanding text content. The classification model of BiLSTM is shown above in Fig.2.

$(\overrightarrow{h_{t-1}}, \overrightarrow{h_t}, \overrightarrow{h_{t+1}})$ is a forware hidden state sequence. Similarly,$(\overleftarrow{h_{t-1}}, \overleftarrow{h_t}, \overleftarrow{h_{t+1}})$is the reverse hidden state sequence. The hidden layer of the BiLSTM algorithm needs to obtain two values. $\overrightarrow{h_t}$ is the result of the forward layer. $\overleftarrow{h_t}$ is the result of the backward layer. The combination of the outputs of the forward layer and the backward layer is the output result O_t , which can be shown as follows:

$$\overrightarrow{h_t} = f(w_1 x_t + w_2 \overrightarrow{h_{t-1}}) \quad (1)$$

$$\overleftarrow{h_t} = f(w_3 x_t + w_5 \overleftarrow{h_{t+1}}) \quad (2)$$

$$O_t = g(w_4 \overrightarrow{h_t} + w_6 \overleftarrow{h_t}) \quad (3)$$

After splicing $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ horizontaly, we get a vector. Then it passes through a full connection layer, and then normalizes the features with softmax layer to obtain the probability value.

IV. Experiments

A. Experimental Preparation

In our experiment, the Android malware samples we used came from the Drebin data set [11], which has 5560 Android malware samples from 179 different malware families. We downloaded nearly 3997 Android software installation packages from this data set for our experiment.

We use MalDroid-2020 and CICInvesAndMal2019 Dataset from [12], which saved many android benign samples. They inspected all samples with VirusTotal to ensure that thes APK files are not malware. In total, we collected about 4000 benign APK files.

B. Metrics

There are four performance indicators of the detection method: Accuracy(ACC), Precision, Recall, and F1-score(F1). Each indicator is calculated with the following formula.

TABLE I. The Result of Experiments

Approach	ACC	Precision	Recall	F1-Score
FastText	0.9734	**0.9791**	0.9677	0.9734
DPCNN	0.9651	0.9615	0.9677	0.9646
MY	**0.9747**	0.9681	**0.9806**	**0.9743**

Accuracy measures the proportion of correct classification. Its calculation formula is simple and direct, and the expression method is as follows:

$$ACC = \frac{TP+TN}{(TP+FP+TN+FN)} \quad (4)$$

The precision represents the accuracy of the prediction in the results of the positive samples, the expression method is as follows:

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

Recall refers to the probability of being predicted as a positive sample in the actual positive sample,Recall is computed as , the expression method is as follows:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

Precision and recall sometimes have contradictions. In order to comprehensively consider them, our commonly used indicator is F1-score. The higher the F1-socre value, the more effective the model is. the expression method is as follows:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (7)$$

C. Experimental Results

In our experiment, for the sake of evaluating the detection effect of our framework using text classification technology. We used 3997 malicious samples and 4000 benign samples to conduct the experiment. In the process of allocating data sets, we use three quarters of the data sets as training sets and the rest as test sets. We compare our method with two common deep learning models DPCNN and Fasttext for text classification. The experiment results are shown in Table I.

Table I shows that the deep learning method of text classification we use is better than the compared deep learning model in some indicators. Our experimental results are very good, with an accuracy ratio of 97.47% and a precision of 96.81%. As we all know, the higher the ACC, the better the detection performance. It can be seen that In practice, BiLSTM algorithm exceeds the Fasttext and DPCNN deep learning model.

For the sake of measuring the detection performance of this method for unfamiliar data, we test it with a new test set. We take 500 malware APK files from MalDroid-2020 and 500 benign APK files from the official website of android(http://www.anzhi.com/). The final test result is 91.77%, which verifies its feasibility again. Perhaps because this test set is relatively new and the previous training set contains limited types of malware, the detection performance of new malware is reduced. However, the accuracy of 91.77% also verifies the feasibility in practice.

V. Conclusion and Future Work

In our paper, our method uses text classification technology combined with deep learning algorithm to detect Android Software. We utilize Androguard,which is the packag in python, to extract the permission, service, receiver and intent sequences of Android malware, and then use word2vec to vectorize the sequences. We use BiLSTM as a classification algorithm to detect android malware.Our evaluation results depict the potential of this method. Among them, my model is superior to other methods. It can identify malware with high accuracy(97.47%). And we also use the new data set to measure the practical robustness of this method. In the future, we may try to consider static features

and dynamic features together as detection features, so that we can more fully mine the characteristics of malware.

ACKNOWLEDGMENT

In the process of generating this paper, all the expenses are from the National Natural Science Foundation of China (NO.12171114), National Key R&D Program Project (NO.2021YFA1000600).

REFERENCES

[1] W. Yaokumah, M. Rajarajan, J.-D. Abdulai, I. Wiafe, F. Katsriku, Modern Theories and Practices for Cyber Ethics and Security Compliance, 2020,http://dx.doi.org/10.4018/978-1-7998-3149-5.

[2] F. Ullah et al., "Cyber Security Threats Detection in Internet of Things Using Deep Learning Approach," in IEEE Access, vol. 7, pp. 124379-124389, 2019, doi: 10.1109/ACCESS.2019.2937347.

[3] Kaspersky:'Mobile malware evolution 2021'. Available,at,https://securelist.com/mobile-malware-evolution-2021/105876/

[4] Shudong Li, Yuan Li, Weihong Han, Xiaojiang Du, Mohsen Guizani, Zhihong Tian, Malicious mining code detection based on ensemble learning in cloud computing environment, Simulation Modelling Practice and Theory, Volume 113, 2021,102391,ISSN 1569-190X,https://doi.org/10.1016/j.simpat.2021.102391.

[5] Enrico Mariconti, Lucky Onwuzurike, Panagiotis Andriotis, Emiliano De Cristofaro, Gordon Ross, Gianluca Stringhini, MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models,https://doi.org/10.48550/arXiv.1612.04433.

[6] P. Feng, J. Ma, C. Sun, X. Xu and Y. Ma, "A Novel Dynamic Android Malware Detection System With Ensemble Learning," in IEEE Access, vol. 6, pp. 30996-31011, 2018, doi: 10.1109/ACCESS.2018.2844349.

[7] Meihua Fan, Shudong Li, Weihong Han, Xiaobo Wu, Zhaoquan Gu, and Zhihong Tian. 2020. A Novel Malware Detection Framework Based on Weighted Heterograph. DOI:https://doi.org/10.1145/3444370.3444545

[8] Spreitzenbarth, M., Schreck, T., Echtler, F. et al. Mobile-Sandbox: combining static and dynamic analysis with machine-learning techniques. Int. J. Inf. Secur. 14, 141–153 (2015). https://doi.org/10.1007/s10207-014-0250-0.

[9] Yoav Goldberg, Omer Levy. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method.https://doi.org/10.48550/arXiv.1402.3722

[10] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," in IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, Nov. 1997, doi: 10.1109/78.650093.

[11] Institute for System Security-Technische Universitat Braunschweig¨ :' The Drebin Dataset' Available at https://www.sec.tu-bs.de/danarp/drebin/download.html

[12] Canadian Institute for Cybersecurity, Available: https://www.unb.ca/cic/datasets/maldroid-2020.html.