DEF: DEEP ENSEMBLE NEURAL NETWORK CLASSIFIER FOR ANDROID MALWARE DETECTION

P Sumalatha, Dr. G.S. Mahalakshmi

Department of Computer Science and Engineering, Anna University, Chennai, Tamilnadu, India.

ABSTRACT

Malware is one of the threats to security of computer networks and information systems. Since malware instances are available sufficiently, there is increased interest among researchers on usage of Artificial Intelligence (AI). Of late AI-enabled methods such as machine learning (ML) and deep learning paved way for solving many real-world problems. As it is a learning-based approach, accumulated training samples help in improving thequality of training and thus leveraging malware detection accuracy. Existing deep learning methods are focusing on learning-based malware detection systems. However, there is need for improving the state of the art through ensemble approach. Towards this end, in this paper we proposed a framework known as Deep Ensemble Framework (DEF) for automatic malware detection. The framework obtains features from training samples. From given malware instance a grayscale image is generated. There is another process to extract the opcode sequences. Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) techniques are used to obtain grayscale image and opcode sequence respectively. Afterwards, a stacking ensemble is employed in order to achieve efficient malware detection and classification. Malware samples collected from the Internet sources and Microsoft are used for the empirical study. An algorithm known as Ensemble Learning for Automatic Malware Detection (EL-AML) is proposed to realize our framework. Another algorithm named Pre-Process is proposed to assist the EL-AML algorithm for obtaining intermediate features required by CNN and LSTM. Empirical study reveals that our framework outperforms many existing methods in terms of speed-up and accuracy.

KEYWORDS

Malware Detection, Artificial Intelligence, Machine Learning, Deep Learning, Ensemble Learning

1. INTRODUCTION

Malware detection is very important in case of mobile applications. Since people of all walks of life are using smart phones, the applications in such mobiles need to be protected from malware. As Android mobiles are widely used, it is indispensable to have mechanisms to prevent malware spreading in Android platform [1]. Of late Artificial Intelligence (AI) has changed the way solutions are made to the real-world problems. With the emergence of ML and deep learning models, it became easier to learn existing problems and provide solutions in a more scalable and optimized fashion. There are many ML approaches that are widely used for malware detection research. Since it supports supervised learning phenomena, learning based approach is found suitable to have a suitable solution. The rationale behind the wide usage of AI is that, the technology is capable of learning from historical data to gain knowledge and make appropriate decisions. There are many existing approaches that are based on AI for malware detection in the Android platform.

Different ML models are explored in [1] and [2] towards the malware detection process. There are some approaches that combine linear and non-linear models as explored in [3]. There are

other hybrid approaches that not only use ML approaches but also other methods in order to improve the prediction performance. Such methods are explored in [18], [19] and [20]. By combining models, there is evidence of the improved learning process thus leading to better performance. There are many deep learning based models [6], [10], [13], [14] and [15] found in the literature. Alzaylaeea *et al*. [6] proposed a malware detection method named DL-Droid which examines real devices in order to ensure secure mobile apps. It has dynamic analysis and also deep learning classifiers in order to identify malware. Zhang *et al*. [10] proposed a framework known as TC-Droid towards automatic detection of malware. It coverts apps into text sequences and feeds the same to CNN for feature extraction. It has automated feature engineering and detection of malware. Pektas and Acarman [13] explored malware detection process using API function calls represented in the form of graph embeddings. Singh *et al*. [14] used DREBIN dataset for malware detection research. They used an enhanced CNN model for extraction of features and classification of malware. Kalash *et al*. [15] exploited a CNN-based architecture along with tuning of parameters in order to detect malware samples more accurately. From the literature, it is ascertained that deep learning models are very efficient for malware detection. Existing models such as CNN and LSTM are widely used in the research of malware detection. However, there is need for ensemble approach towards more efficient detection of malware. Towards this end, in this research our contributions in this paper are as follows.

1. We proposed a framework known as Deep Ensemble Framework (DEF) for automatic malware detection.
2. Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) techniques are used to obtain grayscale image and the opcode sequence respectively.
3. An algorithm known as Ensemble Learning for Automatic Malware Detection (EL-AML) is proposed to realize our framework.
4. Another algorithm named Pre-Process is proposed to assist the EL-AML algorithm for obtaining intermediate features required by CNN and LSTM.

The remainder of the paper is structured as follows. Section 2 reviews existing deep learning methods for automatic detection of malware. Section 3 presents our proposed ensemble-based framework along with an algorithm. Section 4 presents results of experiments while Section 5 concludes the paper and gives scope for the future research.

2. RELATED WORK

This section reviews the literature on recent research dynamics on malware detection. Liu *et al*. [1] review different methods of malware detection using machine learning techniques. It throws light on Android malware detection. Pan *et al*. [2] explored different static analysis methods for Android malware detection. This kind of analysis enables users to verify malware issues before installing an Android app. Static analysis can be done with neural network based analysis for better results. In the future, they intend to improve the static analysis method with certain guidelines. Damaševicius *et al*. [3] proposed combined ML models and neural networks to form an ensemble model for malware detection. In the process, they used linear and non-linear models in order to improve the detection procedure. In the future, they intend to improve their method with multiple malware datasets besides enhancing the learning architecture. Azeez *et al*. [4] also explored ensemble method for malware detection. Their ensemble model could exploit multiple ML models to improve performance. Kareem and Ali [5] proposed an approach based on CNN model with consideration on architectural optimization. It could verify permissions and API calls that are vulnerable. Alzaylaeea *et al*. [6] proposed malware detection method named DL-Droid which examines real devices in order to ensure secure mobile apps. It has dynamic analysis and also deep learning classifiers in order to identify malware. In the future, they intend to improve it with self-adaptation to meet runtime requirements without modifications.

Mahindru and Sangal [7] proposed a framework based on ML models. It is named as MLDroid which is designed to work with Android platform for malware detection. It has dynamic analysis models in order to detect malware samples. They evaluated it with 30 kinds of Android apps. Ma *et al*. [8] used ML algorithms and control flow graphs to realize an Android malware detection method. It is a hybrid method used based on supervised learning which includes training a classifier and detection of malware. In the future, they intend to improve it by locating the position of malware in source code and identification of malware families. Taheri *et al*. [9] proposed a method based on similarity-based phenomenon for malware detection. It combines two things like static binary features and hamming distance. Then it makes use of deep learning models for discrimination between malware and benign flows. Zhang *et al*. [10] proposed a framework known as TC-Droid towards automatic detection of malware. It coverts apps into text sequences and feeds the same to CNN for feature extraction. It has automated feature engineering and detection of malware. In future, they intend to improve it by combing static and dynamic features. Wang and Li [11] considered kernel task structures for machine learning towards malware detection. They used multiple dimensions of features in order to have a weight based approach for detection. In the future, they intend to improve it to explain parallel programming to achieve faster convergence.

Cai *et al*. [12] proposed a methodology where function calls are captured and represented in the form of enhanced graphs for deep learning based malware detection. Their methodology exploits CNN variant for learning features and discrimination. Pektas and Acarman [13] explored malware detection process using API function calls represented in the form of graph embeddings. Then the embeddings are transformed into feature sets that contain less number of dimensions for improving accuracy and speed in malware detection. Singh *et al*. [14] used DREBIN dataset for malware detection research. They used an enhanced CNN model for extraction of features and classification of malware. They also experimented the CNN along with traditional classifiers like SVM. They desired to improve their method with both static and dynamic analysis. Kalash *et al*. [15] exploited a CNN based architecture along with tuning of parameters in order to detect malware samples more accurately. Their modified CNN could perform better than its predecessors. The work of Frenklach *et al*. [16] for malware detection is based on app similarity graph. Since apps can have similarities, usage of similar apps and analysing could pave way for malware detection. Their method includes pre-processing and classification based on Random Forest based hybrid classifier. C ai *et al*. [17] built a framework known as JOWMDroid for the malware detection. It is based on weight-mapping and joint optimization procedures. It has feature extraction and joint optimization of feature learning process. In the future, they intend to improve it by considering feature correlations. A framework known as KronoDroid is proposed by Guerra-Manzanares *et al*. [18] for malware detection and characterization. It is based on hybrid features and temporal dimensions. They intend to introduce concept drift in their framework for to evaluate malware emergence over time. Surendrana *et al*. [19] explored the tree-augmented Naïve Bayes model in order to realize a hybrid model for malware detection. It has combination of static analysis and dynamic analysis. In future then tend to construct improved Bayesian models to leverage detection process. Zhang *et al*. [20] incorporated natural language processing (NLP) to realize a hybrid approach that is sequence-based. It combines both bi-LSTM and CNN models in order to have better understanding of features and improve classification accuracy. In the future, they intend to combine dynamic feature extraction and native compiled libraries. Sumalatha and Mahalakshmi [22] explored malware detection using stacking ensemble approach. Ahmed et al. [23] investigated on methods for detecting ransomware and proposed a methodology with strong pre-processing and dimensionality reduction. From the literature, it is observed that there are many ML approaches for Android malware detection. However, deep learning-based ensemble approach is still desired to improve prediction performance.

3. PROPOSED SYSTEM

We proposed an ensemble methodology based on deep learning techniques like LSTM and CNN. It is illustrated in Figure 1. The architecture is designed to support the binary classification of malware besides its detection. The proposed method is designed to exploit two deep learning models such as LSTM and CNN to realize a stacking ensemble for better performance in the detection of malware samples.

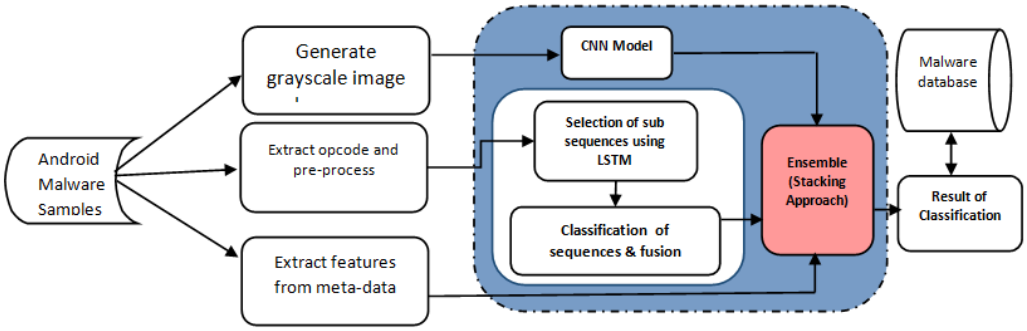

Figure 1: Proposed framework known as Deep Ensemble Framework (DEF)

The malware detection process has two important phases. In the first phase, the given Android malware sample is subjected to the generation of grayscale image from the given sample and pre-process of the data, extraction of the opcode and perform pre-processing and extracting the features from meta-data. The output of this phase is represented in such a way that it can be processed by CNN and LSTM models with ease. In the second phase, actual process pertaining to malware detection takes place. The deep learning models learn from given inputs and gain knowledge required forthe discrimination. Instead of using the knowledge of CNN and LSTM independently, we exploit a stacking ensemble of the two knowledge models to have an integrated approach for detection of malware. From given raw data, out framework has provision for learning from three types of features such as file structure features, code patterns and metadata features.

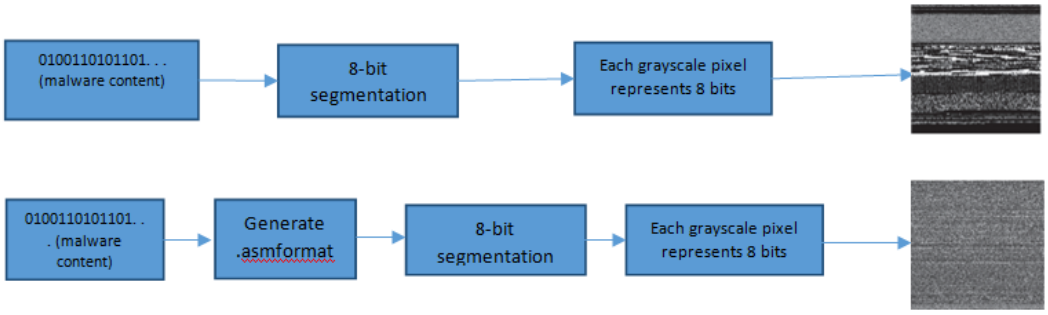

Figure 2: Process of grayscale image generation

As presented in Figure 3, there are two approaches in which a malware sample is transformed into the corresponding grayscale image. In both approaches the malware raw data is subjected to transformation. In the former case, it is subjected to 8-bit segmentation and each grayscale pixel represents 8 bits prior to final generation of grayscale image. In the latter approach, before generation of 8-bit segments there is transformation of mal ware sample into .asm format.

Table 1: Notations used in this paper

Notation	Description
$x_j.y$	corresponding labels
δ	Threshold
$y = 1 \mid x_j)$	LSTM's output
M	Hyperparameter
w_i	the weight of subsequence discrimination
α_{ij}	Each sample length
l_i	the total length of the sequence of category
d_i	the step length
γ	upper limit on the number of samples
θ	parameter

In either case, the final outcome is a grayscale image that is subjected to further analysis. After the generation of intermediate results in first phase, LSTM and CNN are applied on those results. CNN performs its feature extraction on the grayscale image. In the same fashion, LSTM operates on the opcode sequences for selection of sub sequences and classification of the same. In order to perform the selection of sub sequences, maximum likelihood probability is computed as in Eq. 1.

$$\delta \geq \max\{P(y = 1 \mid x_j),\ 1 - P(y = 1 \mid x_j)\} \qquad (1)$$

After making selection of sub sequences, LSTM classifies them and then performs fusion of the same. For the purpose of the fusion, majority voting approach is used as expressed in Eq. 2.

$$H(x) = C_{argmax_j} \sum_{i=1}^{T} w_i h_j^i(x) \qquad (2)$$

For further processing in LSTM there is need for computing length of sequence as expressed in Eq. 3.

$$len(l_i) = \sum_{j=0}^{\beta_i} \alpha_{ij} \qquad (3)$$

There is also need for computing sliding window as in Eq. 4.

$$d_i = \frac{\sum_{j=0}^{\beta_i} \alpha_{ij}\tau}{\gamma},\ 1 \leq d_i \leq \tau \qquad (4)$$

With respect to sequence length there is need for expanding γ value which is done as expressed in Eq. 5.

$$\gamma \leq \sum_{j=0}^{\beta_i} \alpha_{ij} - \tau \qquad (5)$$

The proposed architecture makes use of CNN and LSTM as first level learners while logistic regression is used for next level of learning and whose objective function is computed as in Eq. 6.

$$P_\theta(x) = \frac{1}{1 + exp(-\theta^T x)} \qquad (6)$$

In order to train the regression model SGD is employed in the proposed framework. Two algorithms are proposed to realize the proposed framework for stacking ensemble. All notations used in this paper are provided in Table 1.

Algorithm 1: Pre-Processing

```
Algorithm: Pre-Processing
Input: Malware samples S
Output: Intermediary result map M
Begin
For each sample s in S
img←GenerateGrayScale(s)
opcode←GenerateOpcodeSequence(s)
  features←ExtractMetaDataFeatures(s)
  Add s and (img, opcode, features) to M
End For
Return M
End
```

As presented in Algorithm 1, it performs phase 1 processing of the proposed framework. It takes given malware samples and performs an iterative process to acquire a grayscale image, opcode sequence and metadata features. These intermediary outcomes are populated in a map object. This map object contains all inputs required by CNN and LSTM. The second algorithm is known as Ensemble Learning for Automatic Malware Detection (EL-AML) which invokes Pre-Process algorithm to complete phase 1 of the framework and reuses its outcome for further processing in phase 2.

Algorithm 2: Ensemble Learning for Automatic Malware Detection (EL-AML)

```
Algorithm: Ensemble Learning for Automatic Malware Detection (EL-AML)
Inputs: Malware dataset D
Output: Detection results R
Begin
Initialize intermediary results map M
M←Pre-Process(D)
For each entry m in M
featuresByCNN←ApplyCNN(m.img)
featuresByLSTM←ApplyLSTM(m.opcode)
knowledge←StackingEnsemble(featuresByCNN, featuresByLSTM, m.features)
  result←Classification(knowledge)
Add m.s and result to R
End For
Display R
End
```

As presented in Algorithm 2, it takes malware dataset D as input and invokes the Pre-Process algorithm to achieve extraction of different features required by CNN, LSTM and the stacking ensemble. Here M is the map which holds the intermediate results for each sample. Then there is an iterative process to apply CNN on grayscale image, LSTM on opcode and make an ensemble of these two outcomes long with metadata features of each sample. Thus for each sample classification is carried out through the knowledge gained by stacking ensemble. Eventually results of the classification of each sample is added to R and displayed.

4. EXPERIMENTAL RESULTS

Experiments are made with malware dataset collected from [21]. Different existing models are compared with the results of the proposed ensemble model.

Table 2: Performance comparison of different malware detection models

Models	Accuracy(%)	AUC	TPR(%)	FPR(%)
N-gram	88.5495	93.708	84.759	0.1
LSTM	94.1735	94.9905	93.7555	0.08
CNN	93.233	94.8955	92.074	0.07
Proposed	98.8812	98.9901	98.1486	0.05

As presented in Table 2, the performance of the proposed ensemble model is compared against three existing models.

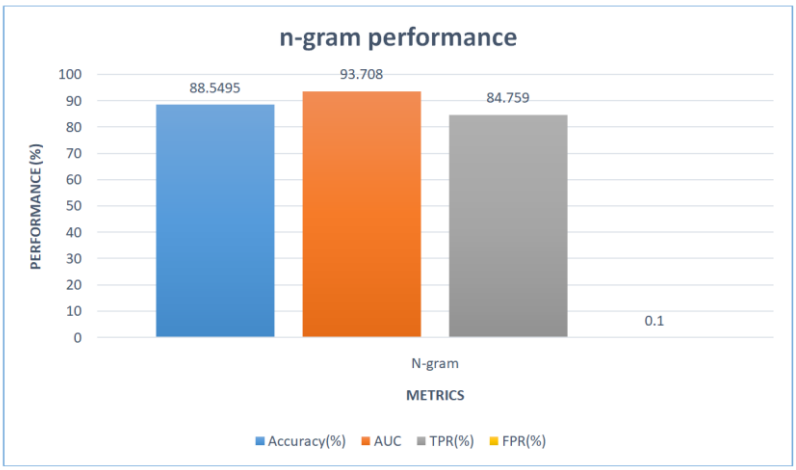

Figure 3: Performance of the N-gram model for malware detection

As presented in Figure 3, N-gram model performance for malware detection is provided. It has 88.54% accuracy, 93.70% AUC, 84.75% TPR and 0.1% FPR.

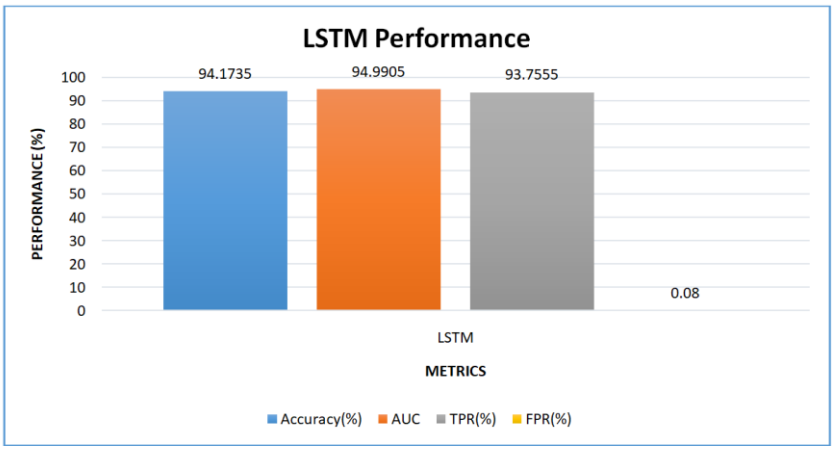

Figure 4: Performance of LSTM model for malware detection

As presented in Figure 4, LSTM model performance for malware detection is provided. It has 94.17%% accuracy, 94.99% AUC, 93.75% TPR and 0.08% FPR.

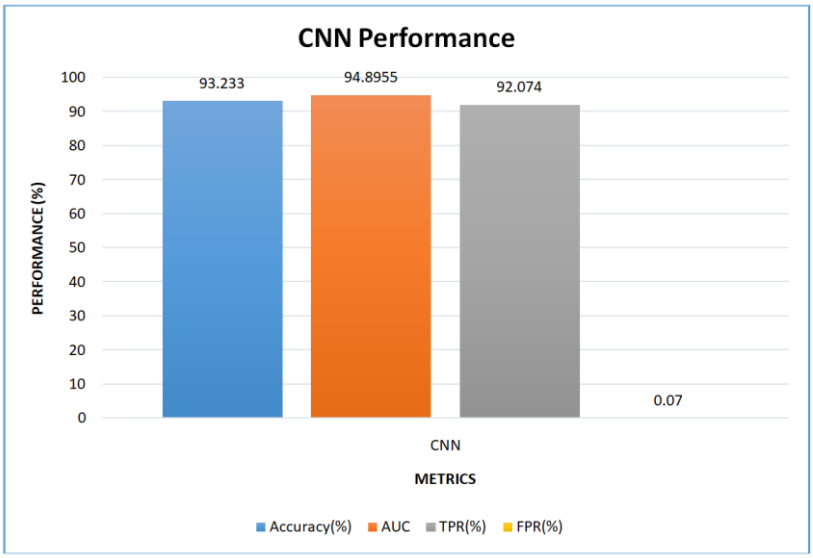

Figure 5: Performance of CNN model for malware detection

As presented in Figure 5, CNN model performance for malware detection is provided. It has 93.23%% accuracy, 94.89% AUC, 92.07% TPR and 0.07% FPR.

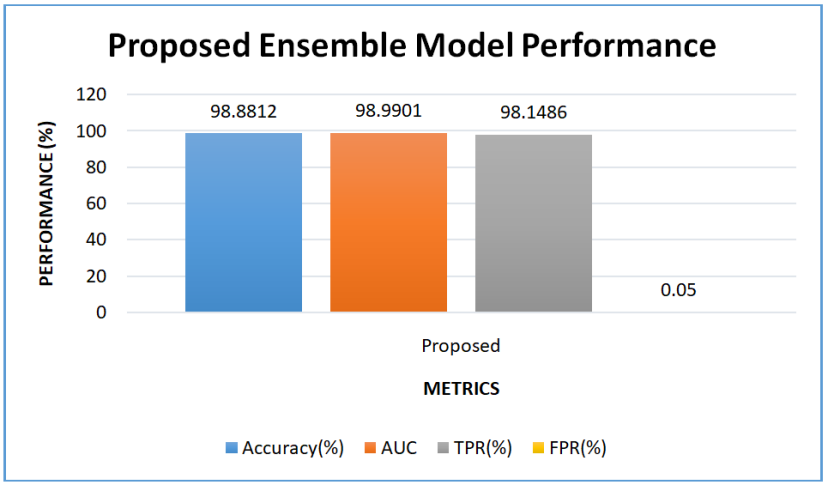

Figure 6: Performance of CNN model for malware detection

As presented in Figure 6, the proposed model performance for malware detection is provided. It has 98.88%% accuracy, 98.99% AUC, 98.14% TPR and 0.05% FPR.

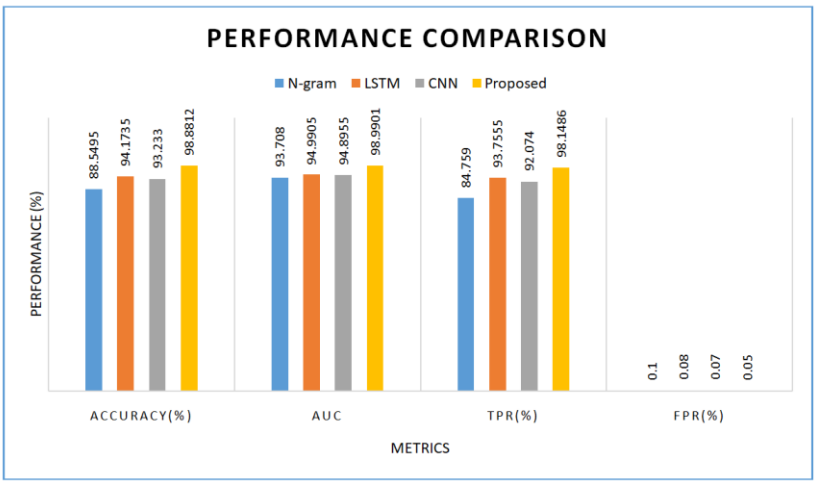

Figure 7: Performance comparison of malware detection models

As presented in Figure 7, our proposed ensemble model is compared against CNN, LSTM and N-gram models. It is observed from the empirical study that the proposed model outperforms existing ones. Each model has shown different level of performance due to their internal architecture and functionality. The least accuracy is exhibited by the N-gram model with 88.54%. The accuracy of LSTM is 94.17% and CNN is 93.23% while the proposed model exhibited higher accuracy with 98.88%.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a framework known as Deep Ensemble Framework (DEF) for automatic malware detection. The framework obtains features from training samples. From the given malware instance a grayscale image is generated. There is another process to extract opcode sequences. Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) techniques are used to obtain grayscale image and opcode sequence respectively. Afterwards, a stacking ensemble is employed in order to achieve efficient malware detection and classification. Malware samples collected from Internet sources and Microsoft are used for empirical study. An algorithm known as Ensemble Learning for Automatic Malware Detection (EL-AML) is proposed to realize our framework. Another algorithm named Pre-Process is proposed to assist the EL-AML algorithm for obtaining intermediate features required by CNN and LSTM. Empirical study reveals that our framework outperforms many existing methods in terms of speed-up and accuracy. The proposed method achieved 98.88% accuracy besides considerable speed-up in the malware detection process. In the future, we intend to explore Generative Adversarial Network (GAN) architecture for further improvement in malware detection performance.

CONFLICTS OF INTEREST STATEMENT

Manuscript Title : DEF: Deep Ensemble Neural Network Classifier for Android Malware Detection

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements),

or non-financial interest (such as personal or professional relationships, affiliations, knowledge and beliefs) in the subject matter or materials discussed in this manuscript.

Authors Name:

[1.]*P Sumalatha,
[2]Dr. G.S. Mahalakshmi

REFERENCES

[1] Liu, K., Xu, S., Xu, G., Zhang, M., Sun, D. and Liu, H., 2020. A review of android malware detection approaches based on machine learning. IEEE Access, 8, pp.124579-124607.
[2] Pan, Y., Ge, X., Fang, C. and Fan, Y., 2020. A systematic literature review of android malware detection using static analysis. IEEE Access, 8, pp.116363-116379.
[3] RobertasDamaševičius;AlgimantasVenčkauskas;JevgenijusToldinas;ŠarūnasGrigaliūnas; (2021). Ensemble-Based Classification Using Neural Networks and Machine Learning Models for Windows PE Malware Detection . Electronics, p1-23.
[4] NureniAyofeAzeez;OluwanifiseEbunoluwaOdufuwa;SanjayMisra;JonathanOluranti;RobertasDamaševičius; (2021). Windows PE Malware Detection Using Ensemble Learning . Informatics, p1-20.
[5] Karrar Ahmed Kareem, Ethar Sabah Mohammad Ali. (2022). MALWARE RECOGNITION SCHEME FOR ANDROID PLATFORM USING AN OPTIMIZED CNN-BASED APPROACH. Journal of Hunan University（Natural Sciences）. 49(1), pp.1-21.
[6] Alzaylaee, Mohammed K.; Yerima, Suleiman Y.; Sezer, Sakir (2019). DL-Droid: Deep Learning Based Android Malware Detection Using Real Devices. Computers & Security, p1-28.
[7] Mahindru, Arvind; Sangal, A. L. (2020). MLDroidâ€ ” framework for Android malware detection using machine learning techniques. Neural Computing and Applications, p1-28.
[8] Ma, Zhuo; Ge, Haoran; Liu, Yang; Zhao, Meng; Ma, Jianfeng (2019). A Combination Method for Android Malware Detection Based on Control Flow Graphs and Machine Learning Algorithms. IEEE Access, p1–11.
[9] Taheri, Rahim; Ghahramani, Meysam; Javidan, Reza; Shojafar, Mohammad; Pooranian, Zahra; Conti, Mauro (2020). Similarity-based Android malware detection using Hamming distance of static binary features. Future Generation Computer Systems, 105, p230–247.
[10] Nan Zhang;Yu-anTan;ChenYang;Yuanzhang Li; (2021). Deep learning feature exploration for Android malware detection . Applied Soft Computing, p1-7.
[11] XinningWang;Chong Li; (2021). Android malware detection through machine learning on kernel task structures .Neurocomputing, p1-25.
[12] Cai, Minghui; Jiang, Yuan; Gao, Cuiying; Li, Heng; Yuan, Wei (2021). Learning features from enhanced function call graphs for Android malware detection. Neurocomputing, 423, p301–307.
[13] Pektaş, Abdurrahman; Acarman, Tankut (2019). Deep learning for effective Android malware detection using API call graph embeddings. Soft Computing, p1-17.
[14] Singh, Jaiteg; Thakur, Deepak; Ali, Farman; Gera, Tanya; Kwak, Kyung Sup (2020). Deep Feature Extraction and Classification of Android Malware Images. Sensors, 20(24), p1-29.
[15] Mahmoud Kalash, MrigankRochan, Noman Mohammed, Neil D. B. Bruce, Yang Wan. (2018). Malware Classification with Deep Convolutional Neural Networks. IEEE, pp.1-5.
[16] Frenklach, T., Cohen, D., Shabtai, A., &Puzis, R. (2021). Android malware detection via an app similarity graph. Computers & Security, 109, 102386., p1-16.
[17] Cai, Lingru; Li, Yao; Xiong, Zhi (2020). JOWMDroid: Android Malware Detection Based on Feature Weighting with Joint Optimization of Weight-Mapping and Classifier Parameters. Computers & Security, p1-21.
[18] Guerra-Manzanares, A., Bahsi, H., &Nõmm, S. (2021). KronoDroid: Time-based Hybrid-featured Dataset for Effective Android Malware Detection and Characterization. Computers & Security, 110, 102399., p1-32.
[19] Surendran, R., Thomas, T., & Emmanuel, S. (2020). A TAN based hybrid model for android malware detection. Journal of Information Security and Applications, 54, 102483, p1-11.

[20] Nan Zhang;JingfengXue;YuxiMa;RuyunZhang;TiancaiLiang;Yu-an Tan; (2021). Hybrid sequence-based Android malware detection using natural language processing . International Journal of Intelligent Systems, p1-15.

[21] Microsoft Malware Dataset, https://www.kaggle.com/c/malware-classification

[22] P Sumalatha1 and G.S. Mahalakshmi. (2023). MACHINE LEARNING BASED ENSEMBLE CLASSIFIER FOR ANDROID MALWARE DETECTION. *International Journal of Computer Networks & Communications (IJCNC)*. 15(4), pp.1-18.

[23] Ahmed Dib1 ,Sabri Ghazi2 and Mendjel Mohamed Said Mehdi. (2023). RANSOMWARE ATTACK DETECTION BASED ON PERTINENT SYSTEM CALLS USING MACHINE LEARNING TECHNIQUES. *International Journal of Computer Networks & Communications (IJCNC)*. 15(4), pp.1-23.