CINEFORUM
ISSN: 0009-7039
Vol. 66. No. S6, 2026 (Special Issue 6)

A Hybrid Deep Learning Based Approach for Windows Malware Detection Using Static Feature Analysis

Mohd Hammad[1], Dr. Asif Irshad Khan[2]

[1,2] Department of Computer Science, Aligarh Muslim University, India, 202002
***Corresponding author:**
Mohd Hammad
Email: hammadmohammad667@gmail.com

Abstract
The modern computing infrastructures have continued to be greatly infected by malware, mostly owing to the increase in obfuscated code patterns and zero day exploits. Traditional signature based detection systems are ineffective when it comes to extending the challenge of such updated types of malwares. This paper suggests a hybrid malware detection system that would jointly utilize both the static features analysis and a deep learning classifier to enhance the detection rate and malware detection resilience. Through a combination of complementary methodological advantages, the methodology purports to overcome the weaknesses of a pure signature based or heuristic approach. The model takes high dimensional statistic attributes out of windows portable executives (PE) files and then passes such representations through a deep neural network that justifies benign and malignant samples. The feature space is designed as a fixed space, such that the structural, opcode, and metadata features that are useful to high quality classification are represented. The EMBER 2018 benchmark data is tested empirically. The hybrid model obtains the overall accuracy of 96.32 Percent and the area under the receiver operating characteristic curve (AUC) of 0.963, which proves a high level of discriminating power and shows the viability of deploying the model to the large-scale malware detections.....
Keywords : Malware Detection, Hybrid Model, Static Analysis, Deep Learning, EMBER Dataset, Windows PE Files...

INTRODUCTION

Malware is a consistent and ever becoming advanced threat to modern computing infrastructure and windows based platforms form the largest sphere of attack in the realms of enterprise as well as individual environments. The spread of malware strain types; including ransomware, trojan, spyware, and backdoor implants has caused significant security and economic impacts to people and organizations worldwide [1], [2]. In turn, the unceasing evolution of malicious codes by the authors, aimed at queue cutting the detection procedure, makes it clear that there is a strong necessity of effective and efficient detection systems [3]. In the traditional malware detection, the detection process is mostly carried out using signature-based matching where binaries of executable code are matched to the repository of known malware patterns [3]. The use of these methods provides computational efficiency, however, they demonstrate low effectiveness with zero-day exploits, polymorphic malware as well as highly obfuscated binaries, which do not have pre-defined signatures [4]. The failure of signature-based methods has in turn led to the development of research on more adaptive paradigms of detection. To address these issues, researchers have investigated a combination of techniques of the machine-learning with the use of the static analysis in the malware detection [5], [9]. The approach to executable artifact analysis Static analysis can be used to examine executable artifacts in an environment where dynamic execution is not available and then identifying a scalable intuition of discriminative patterns is issued by machine-learning models based on the results of the tensile analysis of the extracted features. Deep learning has also contributed to the performance of the detectors by automatically learning complex relationships within the high-dimensional spaces of features, thereby boosting the performance in generalizing to safeguard samples that were not encountered previously [6], [7], [12]. It is against this context that, the current paper proposes a hybrid architecture of malware detection that combines the use of both the static feature extraction and deep-learning classifier. The task of such a combine is to combine the performance of a static analysis with the representation learning that is robust and offered by deep neural networks. The effectiveness of the suggested architecture is evaluated with the help of the EMBER-2018 benchmark dataset which provides a compatible and reproducible basis of research in the field of finding malware in the statical Windows operating system [8].

A. Main Contributions

The key findings of this research will be listed as follows:

hybrid Malware Detector

Benchmark-Based Evaluation
In-depth Analysis of Performance

2. REVIEW OF LITERATURE

The earlier malware detection approaches were mainly based on signature methods, which matched the executable files with patterns of malware as stored in databases [3]. These methodologies can be considered to be insufficient in detecting new threats and other variations that employ code obfuscation or polymorphism though effective in detecting the already known malware [4]. In order to improve the effectiveness of detection, machine-learning methods based on statistic features of the Portable executable files were suggested. Conventional algorithms, such as the Support Vector Machines and Decision Trees, have been used in research to classify between the executionables that are normal and malicious, with better generalization as compared to the rule based applications [5], [9]. However, these approaches are extremely dependent on the performance of feature engineering carried out manually.The spread of massive malware datasets has launched the deep-learning methodologies to become the leading research in the malware detection area. Deep neural networks have demonstrated better performance through their independence in hierarchical feature representations of high-dimensional input data, thus reducing the need to select manually a feature set [6], [7], [12]. These models have shown strong abilities of detection particularly when they are trained on large and disparate datasets. More recently, hybrid detection paradigms that combine a hybrid of a static analysis with machine-learning or deep-learning classifier have been proposed to supplement their detection accuracy without compromising on scalability. These hybrid models combine domain specific, domain-independent static characteristics with powerful learning models, and they make synergistic improvements in performance compared to methods on their own [10], [11]. However, much of the current literature requires constrained or proprietary data, which makes the data publicly available and a hybrid between a static and deep-learning architecture especially crucial in the current research.

3.METHODOLOGY

This section details the design of the suggested hybrid malware detection system to combine conventional uses of the static analysis approach with a deep learning-based classifier. The hybrid architecture is set to leverage the efficiency and interpretability properties of the static feature analysis alongside leveraging the robust pattern-learning properties of deep neural networks that would increase the accuracy of malware detection.

A. General overview of the Hybrid Framework.

The suggested server acts like a background malware detector pipeline whereby an input file is subject to various analyses before it is categorized with a definite decision. The framework belongs to the hybrid category because it consists of the combination of rule-free static feature extraction combined with data-driven deep-learning classification. This combination allows the system to overcome the weaknesses found in signature based or heuristic only approaches to detecting threats. The proposed architecture is also in contrast to the traditional antivirus systems, which utilize signature matching as the only method of matching signatures, and instead focuses on learning of discriminative patterns based on executable characteristics. At the same time, it reduces the risk in overhead and security of dynamic analysis since it can work only within a strict environment of a static analysis.

B. Input File Processing

The system takes in Windows Portable Executable (PE) files. PE files are a popular victim of malware writers since they are being widely used in Windows operating systems. All input files go through a processing or processing stage without running hence making the analysis safe and appropriate to be deployed on a large scale. The input is first checked on simple file verification before feature extraction to ensure that the input is under PE format. The irrelevant files not within the required structural limitations are eliminated to ensure that invalid data do not spoil the detection process.

C. Static Feature Extraction Module

This module extracts feature vectors (in high dimensions) from each PE file encapsulating structural, statistical and metadata-based attributes. The features extracted include file header information, characteristics of the sections, imported libraries, entropy measures, and other indicator figures of potential malicious actions. The feature set follows the standardized representation that is used by the EMBER dataset and thus ensures

compatibility with benchmark-training neural networks as well as thirty reproducibility of experimental results. The process of analyzing executable files is machine-readable in a format suitable to learning features, which is performed in a simple and scalable process, called static feature extraction.

D. Preprocessing & Normalization of Features

Raw feature vectors will often contain missing values, scale differences as well as noise. In order to overcome these problems, the retrieved features are preprocessed by covering missing values and normalizing features. Normalization makes sure that the values of all features have a similar range which is crucial to the stable and effective training of deep-learning models.This initial processing step improves convergence training as well as the generalization capacity of the classifier on unseen examples.

E. Deep Learning Model Architecture

The deep-learning classifier is implemented as a feed-forward neural network which is fully connected and works with normalized, fixed input feature vectors of the data it is being applied to. The input layer has the size set to correspond to the size of features of Portable Executable (PE) binaries and then the network advances through several hidden layers, each using non-linear activation functions. The dropout regularization is also implemented in order to reduce the overfitting rate and, thus, increase the generalization. The terminal output layer in the end makes use of a sigmoid activation function to generate a probability value, thus showing whether the entered executable file is malicious or benign.

F. Decision Engine

The last element of the architecture is the decision engine that executes the probability output of the deep-learning model into a binary decision of classifying. An a priori standard is used to differentiate between benevolent and malevolent files. This modular design would allow the adjustment of the threshold to meet the deployment needs like focusing on a reduced false-positive rate in the enterprise settings.

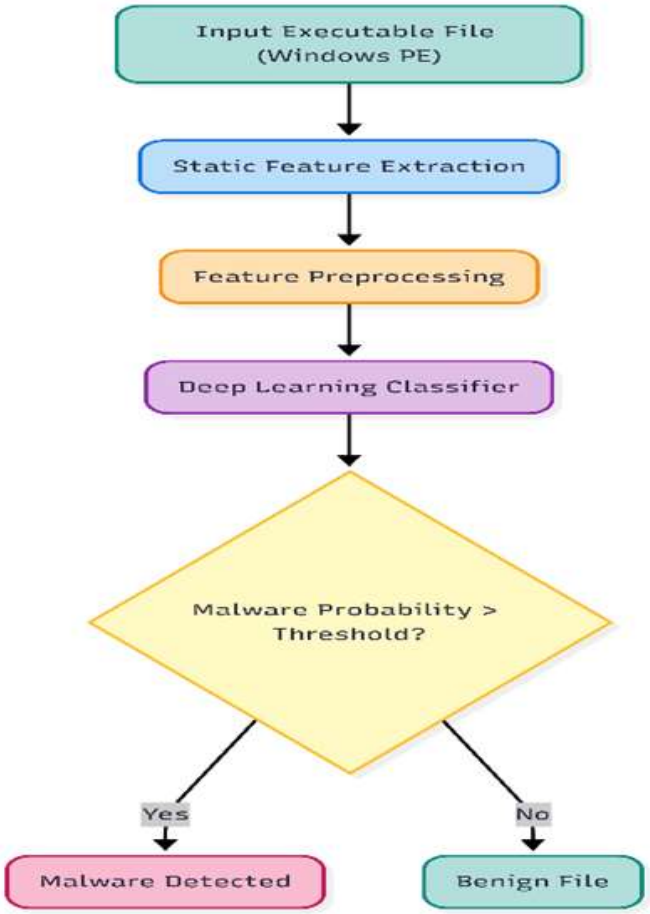

Fig. 1. Architecture of the proposed hybrid malware detection framework

4. DATASET AND EXPERIMENTAL SETUP

The empirical evaluation of the hybrid malware detecting framework is the use of EMBER-2018 which is a freely downloaded benchmark developed to aid research on the subject of detecting malware by static code. The data set is a set of labelled benign and malicious Windows Portable Executable (PE) files, in the form of fixed-length numerical feature vectors generated by use of a static analysis.

A. Dataset Description

EMBER-2018 data includes a vast amount of about 1.1million PE samples comprising benign and malicious binaries, which are obtained in the real world. Every of the instances is represented in feature space as a 2381 dimensional feature vector derived based on executable metadata and structural properties. The dataset also provides labeled training data and unlabeled test data, which makes it appropriate in testing machine learning and neural networks models.

B. Feature Analysis

The dataset includes multiple categories of static features that capture structural and behavioral indicators of executable files. The major feature groups include:

General file information
File size
Virtual size
Number of sections
PE header features
Compilation timestamp
Characteristics flags
Machine type
Section statistics
Section names
Section entropy
Raw size and virtual size of sections
Import table features
Imported libraries (DLLs)
API calls used by the executable
Byte histogram and entropy features
Statistical distribution of byte values
Entropy measurements indicating possible code obfuscation

These feature categories provide meaningful indicators of malicious behavior without requiring code execution, making them highly suitable for large-scale static malware detection.

C. Generality of Data

EMBER dataset is very popular in the malware study due to its variety and true reflection of the malware families. It refers to samples that are not only collected in various sources but also in various time frames and in this manner, the trained model can be capable of learning patterns that cut across a wide collection of malware behaviours. This heterogeneity achieves a better generalisation power of the deep learning model, as it is able to notice samples that it has not previously encountered, and reduces that of the deep learning model to one specific family.

D. Experimental Configuration

The dataset is divided into training, validation, and testing subsets using stratified sampling to maintain class balance between benign and malicious samples. The deep learning classifier is trained using the following experimental configuration:

Optimizer: Adam
Loss Function: Binary Cross-Entropy
Batch Size: 128
Learning Rate: 0.001
Epochs: 20–30
Regularization: Dropout layers to prevent overfitting

E. Evaluation Metrics

The determination of standard classification measures is done to measure model performance:
- Accuracy
- Precision
- Recall
- F1-Score
- Area Under Receiver Operating Characteristic Curve (AUC)

These measurements give a broad evaluation on how the model will correctly classify malicious and benign executable files.

5. RESULTS AND ANALYSIS

In this section, the experimental analysis of the proposed hybrid malware detection model based on the EMBER-2018 is given. The empirical research studies were aimed to test how effective the combination of the analysis of the static features and a deep learning classifier is.

A. Experimental Set up and Behavior of Training.

Throughout the training period, the deep neural network learned discriminative patterns of high-dimensional feature space which were generated by Portable Executable (PE) files. After multiple epochs, convergence was obtained, which was manifested by the gradually leveled improvements in classification. The architecture integrated dropout layers that helped it avoid overfitting and increase the generalization ability of the model. Through the training process, a steady decrease in loss and associated and corresponding increment in the level of classification accuracy across validation samples were observed.

B. Evaluation of Performance Quantitatively.

Table I provides the statistics of quantitative performance of the proposed model.

Metric	Value
Accuracy	96.32%
Precision	96.20%
Recall	96.43%
F1-score	96.32%
AUC	0.963

Table I Performance Metrics of the Proposed Hybrid Malware Detection Model

The results verify that the hybrid model achieves high detection and keeps equal precisions and recall, which is an important aspect of the mitigation of false positives and false negatives related to malware detection systems.

C. ROC Curve Analysis

Figure 2 is a Receiver Operating Characteristic (ROC) curve of the model being considered. The AUC of 0.963 shows that a significant amount of discrimination exists between the benign and malicious samples over a series of decision levels hence confirming the strength of the classifier.

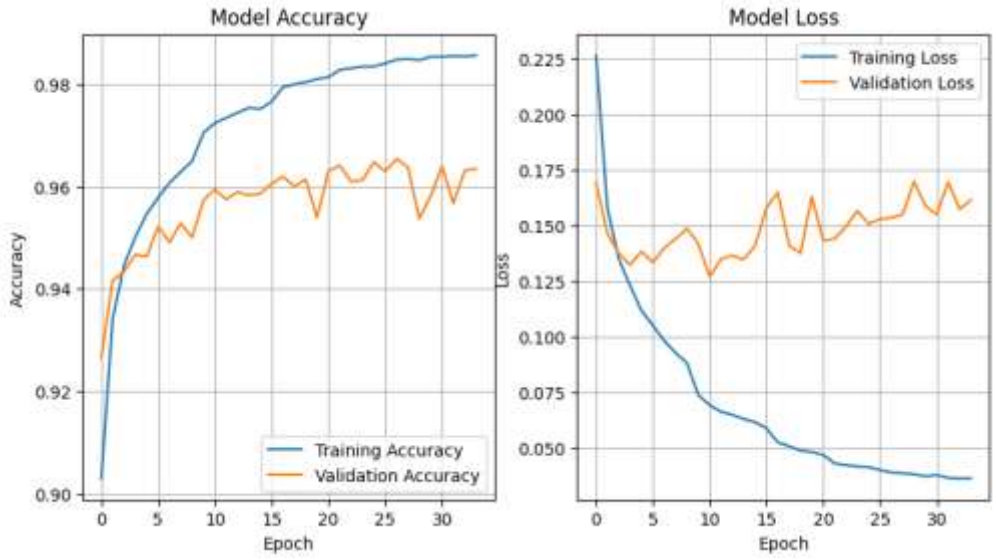

Fig. 2. ROC curve of the proposed hybrid malware detection framework.

D. Comparative Analysis.

The effectiveness of the suggested approach has been compared to the studies on the malware detection that can be found in the literature. Table II provides a comparison in terms of the methodology of detection, type of dataset used, and accuracy as reported.

Ref.	Study	Approach	Feature Type	Dataset	Reported Accuracy
[5]	Shabtai et al.	Classical Machine Learning	Static PE features	Private dataset	91.0
[9]	Kolter & Maloof	Classical Machine Learning	Static executable features	Wild malware samples	93.0
[6]	Saxe & Berlin	Deep Learning	Binary program features	Benchmark dataset	94.0
[7]	Vinayakumar et al.	Deep Learning	Statistical / static features	Benchmark dataset	94.5
[10]	Garg et al.	Hybrid(Static+ Dynamic)	Mixed features	Mixed dataset	95.1
[11]	Baldangombo et al.	Hybrid Machine Learning	Static + ML features	Public dataset	95.0
Proposed	This work	Hybrid Static + Deep Learning	Static PE features	EMBER-2018 [8]	96.32

TABLE II. Comparison with Existing Malware Detection Studies

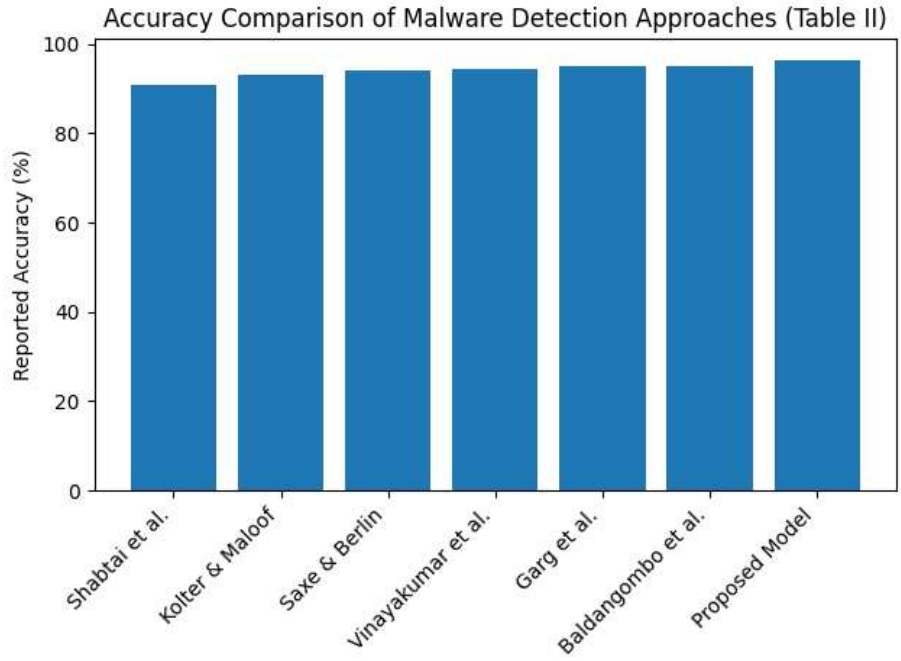

Fig. 3. Accuracy comparison of malware detection approaches based on Table II

The comparison suggests that the suggested hybrid model performs better than classical machine learning and standalone deep learning methods based solely on the use of a static analysis approach and thus avoids the expenses of having a dynamic methodology.

CONCLUSION AND FUTURE WORK

This paper would present a hybrid malware dictionary framework, which combines analysis of the malware features using the two methods of deep learning classification and analysis of static features to detect malicious Windows Portable Executable files. The proposed system reaches a high detection result on the EMBER-2018 benchmark data set when using structural and statistical features of PE binaries and a deep neural network classifier. According to results of experimental assessment, the hybrid system can use the hybrid method to achieve high accuracy (96.32) and AUC (0.963), which demonstrates the presence of a strong classification performance and the ability to effectively distinguish benign and malicious executable files. It is also further proved in the paper that the given model outperforms most models that are still in existence which use machine-learning and deep-learning to detect malware.

Limitations of the Study

Although the current study has positive outcomes, it has a number of interesting limitations:

The system relies solely on the feature of a static analysis and such dependency can make it insufficient when faced with highly obfuscated or packed malware variants.

The experimental analysis capitalized on one benchmark dataset thus, having the possibility of limiting the applicability of the results to different malware distribution.

By definition, the static analysis cannot accommodate the runtime behavioral characteristics displayed by malware

Future Research Directions

Several directions could be taken by the further research of this work:

To begin with, the combination of dynamic behavioral analysis and fixed sets of features and features might increase the detection of advanced malware.

The application of the incremental learning methods would enable the model to be updated with new samples of malware continuously.

The ability to add support to cross-platform malware detection such as Linux and Android systems would make the framework more practical.

It might be beneficial to consider the transformer-based architecture and graph neural networks, which may provide feature representations..

References

[1] AV-TEST Institute, "Malware statistics and trends," 2023. Available: https://www.av-test.org
[2] M. Christodorescu and S. Jha, "Static analysis of executables to detect malicious patterns," Proc. 12th USENIX Security Symposium, Washington, DC, USA, 2003, pp. 169–186.
[3] E. Gandotra, D. Bansal, and S. Sofat, "Malware analysis and classification: A survey," Journal of Information Security, vol. 5, no. 2, pp. 56–64, 2014.
[4] Y. Ye, T. Li, Q. Jiang, and Y. Wang, "CIMDS: Adapting postprocessing techniques of associative classification for malware detection," IEEE Trans. Systems, Man, and Cybernetics, vol. 40, no. 2, pp. 298–307, 2010.
[5] S. Shabtai, R. Moskovitch, Y. Elovici, and C. Glezer, "Detection of malicious code by applying machine learning classifiers on static features," Security Informatics, vol. 1, no. 1, pp. 1–18, 2012.
[6] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," Proc. IEEE Malware, 2015, pp. 11–20.
[7] R. Vinayakumar et al., "Deep learning approach for intelligent intrusion detection system," IEEE Access, vol. 7, pp. 41525–41550, 2019.
[8] S. Garg, R. S. Bali, and S. K. Khatri, "A hybrid malware detection framework using static and dynamic analysis," Proc. IEEE Int. Conf. on Computing, Communication and Automation, 2020.
[9] A. Baldangombo, J. R. Tapamo, and S. O. Oladele, "Hybrid malware detection using machine learning techniques," Computers & Security, vol. 105, 2021.
[10] H. S. Anderson and P. Roth, "EMBER: An open dataset for training static PE malware machine learning models," arXiv preprint arXiv:1804.04637, 2018.
[11] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, pp. 2721–2744, 2006.
[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436–444, 2015..

..