

Malware Detection by Machine Learning based on the LightGBM Model

AMJAD JUMAAH FRHAN

Department of Education and Islamic Studies,
Sunni Endowment Diwan,
IRAQ

National University of Science and Technology,
POLITEHNICA Bucharest,
Bucharest,
ROMANIA

Department of Cybersecurity Engineering, College of Engineering,
Al-Ameen University,
Baghdad 10001,
IRAQ

Abstract: - Wannacry has grown to be considered one of the biggest threats affecting computer security over the past ten years, which has prompted continuous research on detection and mitigation techniques. The EMBER 2018 dataset, which contains sample WPE files categorized as harmless and harmful, is used to synthesize research findings and data in order to build an efficient machine learning model for malware identification. The pre-processed data is sent into a deep learning system for training. The machine learning model was built using dense and conventional dropout layers to reduce constraints on resources and generate quicker, accurate results. By evaluating the underlying API functionality that a file utilizes, we can determine if its behavior could be considered benign or malicious. LightGBM has been built utilizing over 1.55 million tagged example sets, each represented by 1,000 API-call-based attributes. Every model, including the Random Forest model, which displayed outstanding results with 99.49% reliability and fine-grained characteristics like precision and recall to lower mistakes, is assessed using conventional criteria for performance. The study idea is based on the implementation of contemporary standard machine learning models (such LightGBM) to guaranty the highest-level using malware identification accuracy and the usage of reputable software programs to extract standardized features from PE data files. The research also explores classification thresholds and performance optimization methods, aiming to provide a reliable and effective initial solution to the problem of static malware detection.

Key-Words: - Malicious, benign, malware detection, Static PE files, classification, LightGBM, deep learning.

Received: March 27, 2026. Revised: June 2, 2026. Accepted: July 2, 2026. Published: October 5, 2026.

1 Introduction

Malware is created by the cybercriminals to breach infrastructure and take profit of flaws in systems., they have impacted many electronic devices, networks, and cloud computing tools in the digital age. According to Kaspersky Labs' 2017 definition, malware is computer software designed to disrupt and harm a legitimate user's computer in various ways [1].

Malware and malicious software are created for the illegitimate purposes of dangerous individuals who aim to attack networks. These attackers use

malicious software to uncover vulnerabilities in systems and programs, posing significant risks. Our everyday interactions have significantly shifted to electronic devices since The introduction of social networks, online learning, online banking, online shopping, and mobile technologies, these malware attacks have significantly risen. As a result, mobile gadgets have become increasingly important and integral to daily life [2].

In 2024, data and statistics from leading cybersecurity firms indicate that the average volume

of malware threats per minute will exceed 500,000 new attacks per day, equivalent to approximately 350 attacks per minute.

To protect computer and communication systems, malware variants must be detected efficiently. There are various techniques for detecting malware and malicious programs, such as signature, revealing the course of events, behavior, and model verification. The learning methods used in malware detection and the underlying systems are also different, including machine learning (ML) and deep learning (DL) based on the cloud or the Internet of Things [3]. Malware features must be extracted using static or dynamic analysis techniques prior to the start of the detection phase. The most crucial characteristics are then determined in order to categorize them into different categories. Lastly, malware and benign files can be distinguished using deep learning techniques and machine learning algorithms. Ten years ago, malware detection tools that relied on signatures were widely used [4].

However, due to the multi-variant nature of malware and new malware technologies, signature-based detectors are generally unable to identify this type of malware. The same rule applies to the failure of behavior-based detection techniques as well, (also known as event detection techniques), and although behavior detection techniques can detect pattern-based detectors and recognize some undiscovered malware varieties, they continue to miss new and more sophisticated malware variants that differ considerably from those that already exist [5]. Figure 1 depicts types of cyberattacks in the digital world or cyberspace.

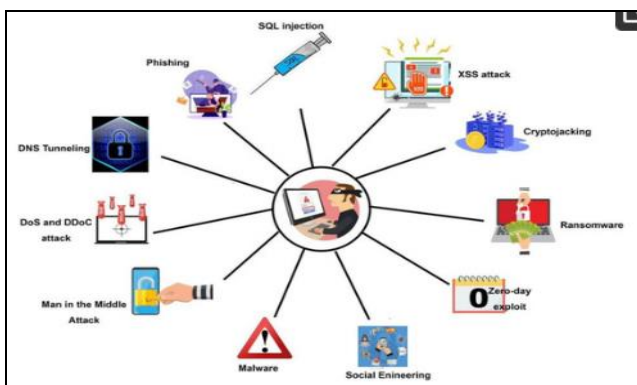


Fig. 1: Types of cyberattacks

Source: Created by the authors.

As malware detection methods become more efficient due to machine learning and deep learning,

the necessity for manual labor is diminishing. Deep learning can be applied to a number of learning procedures, such as dimensionality reduction, feature extraction, and classification [6]. These tools can be combined with other models to improve performance, even when domain experts are available. Malware detection is a significant issue in this increasingly sophisticated digital world, and therefore malware detection is a critical task to prevent any financial loss [7].

Machine learning (ML) and deep learning (DL) have success stories in other disciplines such as computer vision, natural language processing, and robotics. Many scientific and academic organizations have collected numerous samples and models of malware in a database or combined all types of malwares to train and detect malware and made them available to researchers and academics specialized in this field, which saved a significant amount of computing resources [8].

Machine learning malware detection has gained significant attention due to the tremendous success of machine learning in addressing numerous applications such as image processing, natural language processing, and classification. Numerous features found in malware files may be utilized to develop standard machine learning models, including convolutional and standard convolutional networks (CNNs) [9].

By exploiting the advantages of algorithms based on deep learning, this study intends to improve the effectiveness and reliability the malware identification systems simultaneously addressing the challenges associated surrounding digital platforms and information protection. An effective machine learning-based malware detection system will not only improve cybersecurity, but it can also reduce the costs associated with data breaches and improve the accuracy of malware detection, making it more effective, flexible, and economical, thus enhancing overall system security.

2 Related Work

Many researchers have conducted studies on malware detection, some of which incorporate machine learning and deep learning techniques. For example, in 2018, [10] presented an approach for malware analysis and detection. They used a hybrid framework for malware classification. RBM was combined with MLP. Malicious files were taken

from VX Heaven, and benign files were collected from the internet, including Download.com and Softpedia.com. The model achieved an accuracy of 98.6% [10].

In this study, the researcher introduces the LightShield Intrusion Detection System (IDS), designed to enhance intrusion detection in IoT environments using high-performance computing. LightShield combines IoT data preprocessing, a ReliefF feature selection algorithm, and a novel detection model based on LightGBM, which is a gradient boosting framework. To facilitate quicker model validation, the system employs acceleration via graphics processing units (GPUs). The binary classification model exhibited remarkable performance, reaching an accuracy of 99.82% in recognizing potential attacks. The multi-class classification model, in the meantime, reached a commendable accuracy of 97.25% when identifying different kinds of attacks [11].

In this study, the researcher proposed a hybrid intrusion detection system using the LightGBM algorithm with the MobileNetV2 algorithm. LightGBM is used to filter anomalies at the traffic level, while MobileNetV2 is used for packet-level detection. The new Hybrid NNIDS model outperformed other intrusion detection models on the ACI-IoT-2023 dataset with an accuracy of 94%, an F1 score of 91%, and a precision rate of 93%. The results indicate that the constructed asymmetric algorithm can significantly reduce processing costs and may be suitable for use in IoT settings. The goal of the research is to improve response times and reduce false positives in intrusion detection [12].

In this study, the researcher proposed a fusion intrusion detection system (IDS) and denoising autoencoder to remove distortions from network traffic, using a LightGBM classifier to efficiently and quickly classify samples across nine benchmark datasets (e.g., CIDDs-001, CIC-IDS-2017, UNSW-NB15, etc.) for binary and multiclassification tasks, achieving detection rates exceeding 99% in many of them, with very short training times ranging from only 1.10 to 21.78 seconds. The effectiveness of the model confirms its potential for use as a real-time model in various industrial network traffic scenarios, such as smart IoT and fog cloud computing environments [13].

In this study, the researcher proposed combining the LightGBM algorithm with a multi-layer perception network to select important features,

enhancing the new model's ability to detect security threats more accurately and quickly and distinguish attacks from normal traffic. The data was split 80% for training and 20% for testing, with performance accurately evaluated through metrics such as precision, recall, and F1-score. Based on the application of this new combined model to detect industrial Internet of Things (IoT) security, its accuracy reached 96.2%, its recall rate reached 95.6%, and its F1-score reached 96.5%, offering significant advantages over other models. This research aims to support comprehensive analysis of intrusion detection of attacks in the IoT using an effective technique that combines feature selection and deep learning [14].

In this study, the researcher proposed developing and evaluating an automated parameter tuning framework for machine learning models to improve malware classification performance. AutoGluon-Tabular and Microsoft NNI techniques were used to tweak the LightGBM model's parameters to fine-tune its ability to detect malicious features with high accuracy. A couple of datasets have been utilized for extensive experimental purposes: a private dataset that the investigator acquired from an insurance firm which contained fresh samples of malicious software, and the public EMBER the data set, a standard that is well-known for its prevalence in classification of malware studies. Experimental results demonstrate that the AutoML frameworks are able to identify a hyperparameter set that significantly outperforms the model with the best-known hyperparameter setup, improving the LightGBM classifier's true positive rate from 86.8% to 90%, with a false positive rate of 0.1% on the EMBER dataset [8].

The researcher proposed creating and using the EMBER dataset, which includes features extracted from 1.1 million binary files: 900,000 training samples (300,000 malicious, 300,000 benign, and 300,000 unlabeled samples) and 200,000 test samples (100,000 malicious, 100,000 benign samples) to train machine learning models using the LightGBM classifier, comparable in performance to deep learning models such as MalConv. Performance was compared based on specificity metrics such as accuracy, sensitivity, and ROC-AUC, and the results showed that the LightGBM-based model outperforms the MalConv model even with default settings in deep learning malware detection [15].

Researcher [16] proposed the use of several machine learning and deep learning models for

malware detection. They utilized of the Ember malware dataset, which has 69,869 malware records and 70,140 benign records. The machine learning (ML) and deep learning (DL) models that were used were KNN, SVM, Random Forest (RF), AdaBoost, Logistic Regression (LR), Naive Bayes (NB), and Deep Neural Network (DNN). The standard Adam optimization approach was used to train the phenomenon over 200 epochs. Maximum accuracy of 98.9% had been achieved by the LSTM model [17].

For improvement in the performance of detection of malware, investigator suggested employing the LightGBM algorithm with a modified (logistic) loss function. This was based on two types of data: a private, secret (FFRI) dataset together with a public (EMBER) dataset consisting of pre-formatted features from PE files. LightGBM models were programmed on the static physical features that were recovered from PE files by conducting a file surface analysis technique. Their performance was analyzed using benchmarks such as the AUC, which reached 97.9% on the EMBER dataset at $\alpha=430$ and $\beta=339$, demonstrating a high ability to correctly classify malicious files. The aim of the research is to address the challenges associated with cipher wrapping and obfuscation techniques used by malware developers by adjusting the weights to improve the model's sensitivity in detecting threats. The research's weaknesses include its reliance on limited datasets, especially non-public data, which limits the generalizability of the results. Furthermore, the use of static features from PE files may not be sufficient to counter attacks that increasingly use obfuscation techniques [18].

The following are the research's contributions:

1. A method for analyzing persistent malware by extracting features from PE files was presented. This technique solves the problems associated with constructing a feature set based on analysis of PE files.
2. Data preprocessing techniques were used on the extracted features to address issues related to data or feature set imbalances when constructing them.
3. A deep learning model that solved the issues of the high computational resources and analysis time was presented to categorize PE files utilizing dense and low-level layers.

2.1 Machine-Learning-Based Detection

Since 2015, methods for detecting malware that are based on machine learning have gained popularity and continue to be employed in numerous scientific investigations. Markel proposed a method for detecting malware that utilized machine learning. The metadata, particularly the header data from each Windows Portable Executable (PE32), were gathered. Subsequently, learning methods were applied to the gathered metadata in order to identify malware. The proposed approach was evaluated using realistic datasets. The findings of the test indicated that classifiers did not perform well on test data with a low malware frequency, even when ensemble learning was used. It was identified that for optimal performance, the training and testing data should be equivalent to each other [19].

A detection and classification schema based on machine learning was proposed by Sethi et al. System activities were collected using the Cuckoo sandbox when program samples were executed. Subsequently, a property extraction and a selection method were suggested and carried out on the gathered activities to distinguish malware from benign samples. Ultimately, a range of machine learning algorithms were executed for malware detection and fine-grained classification [20]. The obtained detection and classification results were better than those of the state-of-the-art techniques, according to the paper. The performance results that were achieved for specific machine learning classifiers, though, were low [21].

2.2 Dataset and Method

1. Dataset

Previously, the majority of researchers collected both benign and malicious samples at their own expense and developed feature extraction techniques and machine learning datasets on their own. Comparing different approaches with those of other security researchers was difficult due to the lack of a consistent methodology for feature extraction and dataset segmentation. Malware-related datasets are less publicly available due to the specificity of the security field. In the past few years, certain malware detection datasets, including the EMBER dataset [5], have been accepted and are widely used in academic research.

The EMBER dataset represents the first large, public dataset for malware detection using machine learning. The authors expect the dataset to spur

innovation in malware detection using machine learning [15]. Using this dataset, we also released a benchmark, unoptimized LightGBM model. Model performance can be improved through simple techniques, such as feature selection to remove unwanted features and hyperparameter optimization using grid search. The pre-trained LightGBM model, trained on these features, demonstrated superior performance compared to recently published studies on end-to-end malware detection using deep learning [22]. Therefore, in addition to serving as a benchmark dataset, we hope that EMBER will provide a simple way to benchmark the performance of models based on new architectures, such as end-to-end deep learning. The source code and dataset are available at [15].

A. Data Preparation:

- Data is collected from the EMBER resource, either as sample files or as features extracted via open-source tools.
- Data is cleaned and labeled in a balanced manner, with the possibility of using unlabeled samples to support semi-supervised learning methods.

B. Feature Extraction

This process requires selecting a specific set of features from the provided dataset. By extracting features, the feature space of the dataset will be reduced. Reducing the feature space can help reduce computational cost and avoid needless processing of the full dataset during training [23]. The EMBER dataset consists of PE files in JSON format. To train a deep learning model, features must be extracted from these JSON files. To extract features from PE (Portable Executable) files, Endgame Inc. suggests utilizing the Library to Instrument Executable Formats (LIEF), a tool that is well-known for its efficacy in this regard [24].

C. Data cleaning

This is sometimes referred to as data cleansing or scrubbing. By identifying and eliminating the flaws and inconsistencies found in the datasets, this enhances their quality. Researchers [24] stated that a number of resources might be employed and demonstrated how data gathered from one or more sources is necessary to carry out data cleansing. There are training and testing sets in the EMBER datasets.

The EMBER dataset, a well-known open-source dataset designed to train and evaluate machine learning models for malware detection, specifically using PE files (portable executable files) on Windows, was used. The dataset was split into two sets, 80% training and 20% testing. EMBER does not provide raw binary files (to avoid malware propagation), but rather provides pre-extracted features from PE files. These features include a byte histogram, PE header information, import/export operations, partition metadata, string statistics, and general file metadata.

Each sample is classified as benign (0), malicious (1), or unclassified (-1). The unclassified set can be used for semi-supervised learning. The class imbalance problem helps detect when the number of samples from one class is significantly greater than the number of others. Figure 2 shows the histogram to provide a quick visual understanding of the composition of the data set.

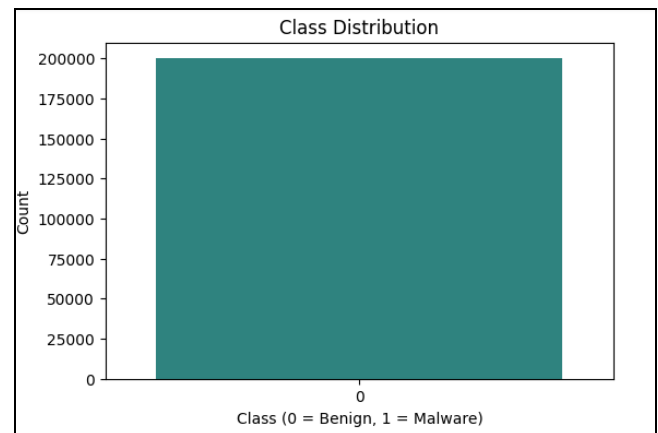


Fig. 2: Confusion Matrix

Source: Created by the authors.

3 Methodology

In 2017, Ke and associates proposed LightGBM, which is an enhancement of the GBDT algorithm [25]. By excluding samples with smaller gradients during training, gradient-based one-sided sampling lowers the processing requirements.

Model complexity is also decreased by using an exclusive feature pooling approach when it deeply on smart cloud systems, which lowers the quantity of features used during model training. This model has drawn a lot of interest and has shown effective in a variety of tasks, including ranking, regression, and classification [18]. The LightGBM model has the

advantages of fast training and high efficiency [26], this could prevent risk in cloud environment to save our resources [27].

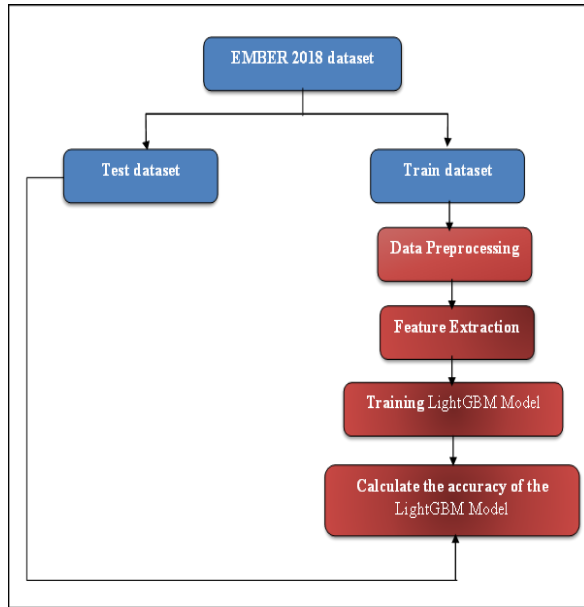


Fig. 3: The proposed methodology
Source: Created by the authors.

In the field of machine learning, a widely used model is the Gradient Boosting Decision Tree (GBDT). LGBM serves as a framework for the implementation of the GBDT model. In order to address the issue of fault diagnosis in belt conveyors, this paper utilized LGBM an efficient method for data classification that finds application across various domains such as industry, medicine, and economics, [25], [26]. The GBDT learning model serves as the foundation of the LGBM model, which combines the Exclusive Feature Bundling (EFB), Gradient-based One-Side Sampling (GOSS), and Histogram algorithms, [28]. While ensuring the model's accuracy [29], it can enhance the model's learning efficiency. Figure 3 proposed the methodology. Recent developments took a look at phishing detection methods utilizing both machine learning and NLP procedures [30]. The finding of rolling bearing problems has been additionally done using conventional deep learning mechanisms [31].

Predictive forecasting and categorization of severe to severe the convective weather has been accomplished by applying the LightGBM based algorithm [32].

Hierarchical agglomerative clustering has been used for real-time event detection from online social media networks [33].

The LGBM algorithm is as follows [14]:

Firstly, assuming that there is a training set $X = \{x_1, x_2 \dots x_n\}$, the LGBM model will initialize a tree as a constant:

$$y_i^0 = f_0 = 0 \quad (1)$$

where y_i^t is the prediction of the i -th example at iteration t .

Then, the followed tree is trained using the lowest loss function:

$$f_t(x_i) = \arg \min L_t = \arg \min L(y_i, y_i^{(t-1)} + f_t(x_i)) \quad (2)$$

where $f_t(x_i)$ represents the learning model of the t -th decision tree

The following model is then expected as:

$$y_i^t = y_i^{(t-1)} + f_t(x_i) \quad (3)$$

Equations (2) and (3) are repeated until the model reaches the termination condition. The final model formula is:

$$y_i = \sum_{t=0}^{M-1} f_t(x_i) \quad (4)$$

where M is the number of iterations.

3.1 Model Evaluation and Optimal Threshold Determination

Following the training phase, the model's performance was rigorously evaluated, and an optimal classification threshold was determined. The Receiver Operating Characteristic (ROC) curve points, specifically the False Positive Rate (FPR) and True Positive Rate (TPR), were calculated for various thresholds using the roc_curve function from sklearn.metrics on the training dataset (X_{train} , y_{train}) and the predictions generated by the trained gbm model (utilizing its best_iteration). To identify the most effective classification threshold, Youden's J statistic ($J = TPR - FPR$) was computed, and the threshold corresponding to the maximum J-statistic value (best_thresh) was selected, signifying the best balance between maximizing true positives and minimizing false positives. The Area Under the ROC Curve (AUC-ROC) was subsequently calculated

using auc to quantify the model's overall discriminative power. Finally,

The ROC curve was visually represented using Roc Curve Display, Figure 4 represents a graphical illustration of the model's ability to distinguish between harmful and benign samples.

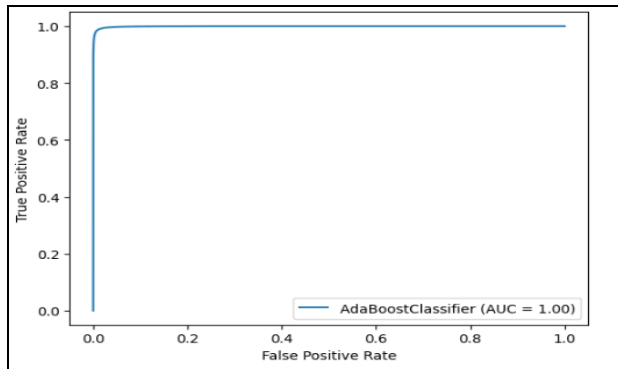


Fig. 4: Receiver Operating Characteristic (ROC) Curve

Source: Created by the authors.

After that step these probabilities transformed to binary diagnoses using a predetermined decision threshold (in this example, 0.5, though this could have been changed based on earlier ROC curve study). Number of performance measures was computed in order to comprehensively assess the correctness of the framework. These consists of a thorough classification report (classification report) and confusion matrix (confusion matrix) created using the simulated binary classifications as well as the space under the receiver operating characteristic curve (ROC-AUC) using the predicted chances. These measurements offer an in-depth assessment of the model's performance in identifying legitimate malware on heretofore unknown data along with its potential to distinguish a malware across various classifications.

3.4 Model Evaluation Results

The framework performed well considering reviewed on the test set, displaying exhibiting a ROC-AUC of 99%49. The model's accuracy for both categories (0.0 for benign files and 1.0 for malicious files) is fully described in the classification test report and standard misunderstanding matrix. As shown in the Table 1, the model achieved a precision, recall, and F1-score of 97% for both categories, indicating balanced and high performance in identifying both benign and malicious files. The overall accuracy of

the model was 97. Matrix of errors shows how many samples that were categorized were correctly and incorrectly categorized. 96,946 benign normal and abnormal samples and 97,045 samples that are hazardous have been accurately categorized, contrary to there were 2,926 false negative (malicious files classified benign) and 3,039 false-positive results (benign files classified exactly as malicious). Table 1 Illustrate the final LightGBM results:

Table 1. The final LightGBM results

Metrics	The results %
AUC	99.49
Precision	96.85
Recall	97.12
F1	96.98

Source: Created by the authors.

3.5 Analyzing Model Performance at Different Decision Thresholds

The aims are analyzing the impact of the choice of decision threshold. The things are scrutinized at 0.4, 0.5, then and 0.6 specifications. The structure of the model predictive potential outcomes on the test set is transformed into binary classifications for each chosen threshold; snapshots fall under the category as positive if their prediction probable is higher greater than or equivalent to the required threshold. The threshold used is then printed, followed by a comprehensive classification report for each threshold, which includes key performance metrics such as precision, recall, and F1-score for each class (benign and malicious).

Threshold: 0.4				
	precision	recall	f1-score	support
0.0	0.98	0.96	0.97	99985
1.0	0.96	0.98	0.97	99971
accuracy			0.97	199956
macro avg	0.97	0.97	0.97	199956
weighted avg	0.97	0.97	0.97	199956
Threshold: 0.5				
	precision	recall	f1-score	support
0.0	0.97	0.97	0.97	99985
1.0	0.97	0.97	0.97	99971
accuracy			0.97	199956
macro avg	0.97	0.97	0.97	199956
weighted avg	0.97	0.97	0.97	199956

Fig. 5: Model classification report at different decision thresholds

Source: Created by the authors.

The analysis understands how threshold affects as trade-off between different performance metrics, which is crucial for determining the most appropriate threshold based on specific application goals, such as reducing false positives or increasing detection rate. Figure 5 illustrate the model classification report at different decision thresholds.

Confusion Metrics Shows excellent performance with few false positives (3,039) and false negatives (2,926)

Threshold Analysis:

The notebook explores different classification thresholds (0.4, 0.5, 0.6):

- At 0.4: Higher recall (98%) but slightly lower precision (96%)
- At 0.5: Balanced precision (97%) and recall (97%)
- At 0.6: Higher precision (98%) but slightly lower recall (96%)

Key Strengths

1. **Excellent Performance:** The model achieves near-perfect AUC score (0.9949) indicating it can almost perfectly distinguish between malware and benign files.
2. **Balanced Metrics:** Both precision and recall are high and balanced, meaning the model detects most malware while keeping false positives low.
3. **Robust Training:** Uses techniques like:
 - Early stopping to prevent overfitting
 - Regularization (L1/L2)
 - Feature and data sampling
 - Proper train/validation/test splits
4. **Threshold Analysis:** Explores different operating points showing how to trade off precision vs recall.

Comparison of Test Results

The results and effectiveness of different algorithms were compared on the EMBER 2018 dataset, using these metrics: Accuracy, F1-Score, Recall, and Precision.

A detailed comparison was conducted for malware detection, using controlled experiments on each model. These models were carefully selected to represent a variety of machine learning approaches. Results obtained on the EMBER 2018 dataset in term of accuracy compared with previous studies, presented in Table 2.

Table 2. Accuracy results for EMBER 2018 data with previous studies

Method	DataSet	Accuracy
LightGBM& XGBoost	EMBER 2018	90%
LightGBM	EMBER	97.8%
LightGBM	EMBER 2018	99.49%

Source: Created by the authors.

This discovery could significantly contribute to the development of more advanced and effective security systems to combat growing cyber threats, were achieving a top accuracy of 99.49% on the EMBER 2018 dataset.

4 Limitations and Future Works

Although the proposed deep learning-based malware detection system is effective in identifying diverse samples from different families, there are some noteworthy limitations to the research. This study classified software samples as either malicious or benign. No further classification was performed to identify malware types. The goal of the next study is to clarify the malware sample categories and evaluate our proposed architecture using additional malware datasets, such as Maling and Microsoft BIG 2015. The proposed model is capable of identifying new malware types; however, we have not yet tested it on inputs specifically designed for adversarial attacks. Additionally, it will be assessed for evasion assaults in the future. In subsequent studies, we might also examine a greater number of benign and malicious samples. The model training period fell within a reasonable range. In order to build a more reliable system, we intend to integrate different deep learning architectures with reinforcement learning. In this study, we employed a deep learning strategy for categorization. To enhance model performance and give more computational power and resources, we want to leverage emerging deep learning technologies like blockchain, cloud computing, and big data.

In two stages approach, Transformer-based language models have also been applied to handwritten text recognition with human and synthetic data [34].

The effort involved in working with static PE files is an effort to identify malware in its early stages and thus prevent it from affecting any operational environment later. Several researchers have identified that finding accurate static PE files for a problem is challenging. Some research has

encountered difficulties in extracting features from PE files and has tried various extraction methods. Although only a few have produced satisfactory results, the computational time and resource requirements are high. In this research, these issues were addressed by developing a malware detection and classification model based on static PE files of malware. This model can extract features, remove duplicates or missing information, and organize the extracted features into a standard range. To address issues related to resource and time consumption, a deep learning model was designed that only contains dense and dropout layers. The experiment was conducted using the EMBER dataset, published by Endgame Inc. Training was conducted on the EMBER 2018 dataset for ten epochs, using 2,382 features and a learning rate of 0.05. The results showed an accuracy of 98.85%. This study classifies malicious PE files based on static analysis at their initial stage, allowing for appropriate preventative measures to be implemented at subsequent stages. This research identifies the use and importance of dataset handling and various data preprocessing techniques in this field. This paper presents a systematic approach to solving classification problems, which involves stages rather than simply creating datasets and feeding them into machine learning models.

Physics and distillation approaches demonstrated via the standard predictive modeling of complex & complex nonlinear structures, physics-informed (PINNs) learning approaches might be determined to include physical drawbacks into malware detection, [35], [36].

5 Conclusion

The desired outcome of this research initiative is for determining malware in Windows Portable Executable (PE) files using the technique of the use of machine learning. The LightGBM standard technique and then the EMBER 2018 dataset have been employed to create an online malware early identification model. 2,382 characteristic that were carried out from Windows Personal Environment (PE) files and processed was for a typical model testing and training are accounted for in the standard dataset.

The argued LightGBM-based model can be distinguished between unsafe and benign files with a sense of an accuracy of 99.49% coming after preprocessing, coursework, optimization,

additionally parameter improved performance. The system takes into account the usage habits linked to both benign and malicious PE files using the Application Programming Interface import particular characteristics.

The estimated results show that the proposed solution may be capable to be a successful static grouping of malware solution.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

The authors wrote, reviewed and edited the content as needed and they have not utilised artificial intelligence (AI) tools. The authors take full responsibility for the content of the publication.

References:

- [1] N. Z. Gorment, A. Selamat, and O. Krejcar, "A Recent Research on Malware Detection Using Machine Learning Algorithm: Current Challenges and Future Works Nor," 2021. doi: 10.1007/978-3-030-90235-3_41.
- [2] N. S. Sani, M. I. Esa, and B. A. Musawi, "SS symmetry Feature Selection," 2023.
- [3] S. Riaz et al., "Malware Detection in Internet of Things (IoT) Devices Using Deep Learning," 2022. doi: 10.3390/s22239305.
- [4] O. Aslan and R. Samet, "A Comprehensive Review on Malware Detection Approaches," 2020. doi: 10.1109/ACCESS.2019.2963724.
- [5] R. Mitchell and I.-R. Chen, "Behavior Rule Specification-Based Intrusion Detection for Safety Critical Medical Cyber Physical Systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 12, no. 1, pp. 16–30, 2015, doi: 10.1109/TDSC.2014.2312327.
- [6] F. Aboaoja, A. Zainal, F. Ghaleb, and B. A. Al-rimy, "Malware Detection Issues, Challenges, and Future Directions: A Survey," *MDPI*, vol. 12, no. 8482, p. 29, 2022, doi: 10.3390/app12178482 Academic.
- [7] Ö. Aslan, "Separating Malicious from Benign Software Using Deep Learning Algorithm," *Electron.*, vol. 12, no. 8, 2023, doi: 10.3390/electronics12081861.
- [8] P. P. Kundu, L. Anatharaman, and T. Truong-Huu, "An Empirical Evaluation of Automated Machine Learning Techniques for Malware Detection," in *Proceedings of the 2021 ACM Workshop on Security and Privacy Analytics*, in IWSPA '21. New York, NY, USA:

- Association for Computing Machinery, 2021, pp. 75–81. doi: 10.1145/3445970.3451155.
- [9] D. Moon, J. Lee, and M. Yoon, “Compact feature hashing for machine learning based malware detection,” *ICT Express*, vol. 8, no. 1, pp. 124–129, 2022, <https://doi.org/10.1016/j.icte.2021.08.005>.
- [10] M. Chowdhury, A. Rahman, and R. Islam, “Malware Analysis and Detection Using Data Mining and Machine Learning Classification BT - International Conference on Applications and Techniques in Cyber Security and Intelligence,” J. Abawajy, K.-K. R. Choo, and R. Islam, Eds., Cham: Springer International Publishing, 2018, pp. 266–274.
- [11] K. Singal, N. Kandhoul, and S. K. Dhurander, “Evolutionary LightGBM-Based Intrusion Detection System for IoT Networks,” *Int. J. Commun. Syst.*, vol. 38, no. 5, p. e70031, Mar. 2025, <https://doi.org/10.1002/dac.70031>.
- [12] Y.-M. Yang, K.-C. Chang, and J.-N. Luo, “Hybrid Neural Network-Based Intrusion Detection System: Leveraging LightGBM and MobileNetV2 for IoT Security,” *MDPI*, vol. 17, p. 19, 2025, doi: 10.3390/sym17030314.
- [13] S. A. H. Ayubkhan, W.-S. Yap, E. Morris, and M. B. K. Rawthar, “A practical intrusion detection system based on denoising autoencoder and LightGBM classifier with improved detection performance,” *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 6, pp. 7427–7452, 2023, doi: 10.1007/s12652-022-04449-w.
- [14] M. Wang, K. Shen, C. Tai, Q. Zhang, Z. Yang, and C. Guo, “Research on fault diagnosis system for belt conveyor based on internet of things and the LightGBM model,” *Discip. Innov. team Liaoning Tech. Univ.*, p. 17, 2023, doi: 10.1371/journal.pone.0277352.
- [15] H. S. Anderson and P. Roth, “EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models,” 2018, [Online]. <http://arxiv.org/abs/1804.04637> (Accessed Date: January 10, 2026).
- [16] Vinayakumar, R., & Soman, K. P. (2018). DeepMalNet: Evaluating shallow and deep networks for static PE malware detection. *ICT Express*, 4(4), 255–258. <https://doi.org/10.1016/j.icte.2018.10.006>.
- [17] M. S. Akhtar and T. Feng, “Malware Analysis and Detection Using Machine Learning Algorithms,” 2022. doi: 10.3390/sym14112304.
- [18] Y. Gao, H. Hasegawa, Y. Yamaguchi, and H. Shimada, “Malware Detection Using LightGBM With a Custom Logistic Loss Function,” *Japan Sci. Technol. Agency*, vol. 10, p. 14, 2022, <https://doi.org/10.1109/ACCESS.2022.3171912>.
- [19] J. Singh and J. Singh, “A survey on machine learning-based malware detection in executable files,” *J. Syst. Archit.*, vol. 112, p. 101861, 2021, <https://doi.org/10.1016/j.sysarc.2020.101861>.
- [20] K. Sethi, S. K. Chaudhary, B. K. Tripathy, and P. Bera, “A novel malware analysis for malware detection and classification using machine learning algorithms,” in *Proceedings of the 10th International Conference on Security of Information and Networks*, in SIN ’17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 107–113. doi: 10.1145/3136825.3136883.
- [21] K. Sethi, R. Kumar, L. Sethi, P. Bera, and P. K. Patra, “A Novel Machine Learning Based Malware Detection and Classification Framework,” in *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2019, pp. 1–4. doi: 10.1109/CyberSecPODS.2019.8885196.
- [22] M. Boryanka T, A. Ibrahim S, K. Habeebah A., and T. Jesse J., “An Improved Light GBM using Bayesian Optimization for Vulnerability Exploitation Prediction,” *Sci. Res. Rev.*, no. 1595–9074, pp. 49–62, 2024, doi: 10.70882/josrar.2024.v1i1.17.
- [23] S. S. Lad and A. C. Adamuthe, “Improved Deep Learning Model for Static PE Files Malware Detection and Classification,” *Comput. Netw. Inf. Secur.*, vol. 2, pp. 14–26, 2022, doi: 10.5815/ijcnis.2022.02.02.
- [24] E. Rahm and H. H. Do, “Data Cleaning: Problems and Current Approaches,” *IEEE Data Engineering Bulletin*, vol. 23, no. 4, pp. 3–13, 2000.
- [25] M. R. Machado, S. Karray, and I. T. De Sousa, “LightGBM: An effective decision tree gradient boosting method to predict customer loyalty in the finance industry,” 2019. doi: 10.1109/ICCSE.2019.8845529.

- [26] G. Kirubavathi and W. R. Anne, "Behavioral based detection of android ransomware using machine learning techniques," *Int. J. Syst. Assur. Eng. Manag.*, vol. 15, no. 9, pp. 4404–4425, 2024, doi: 10.1007/s13198-024-02439-z.
- [27] Mohammed Lateef H, Abduljawad Al-Shibly M, Hadi Ali AL-Jumaili A *et al.* Intelligent Cloud Resource Usage Potentially to Improve Task Scheduling by the use of Artificial Intelligence [version 2; peer review: 1 approved, 1 not approved]. *F1000Research* 2026, 15:366 <https://doi.org/10.12688/f1000research.177203.2>.
- [28] J.-C. Huang, K.-M. Ko, M.-H. Shu, and B.-M. Hsu, "Application and comparison of several machine learning algorithms and their integration models in regression problems," *Neural Comput. Appl.*, vol. 32, no. 10, pp. 5461–5469, 2020, doi: 10.1007/s00521-019-04644-5.
- [29] Y. Liu, Z. Yu, C. Chen, Y. Han, and B. Yu, "Prediction of protein crotonylation sites through LightGBM classifier based on SMOTE and elastic net," *Anal. Biochem.*, vol. 609, p. 113903, 2020, <https://doi.org/10.1016/j.ab.2020.113903>.
- [30] N. A. Mohammed *et al.*, "Recognizing Phishing in Emails by Using Natural Language Processing & Machine Learning Techniques," 2025 *3rd International Conference on Cyber Resilience (ICCR)*, Dubai, United Arab Emirates, 2025, pp. 1-7, doi: 10.1109/ICCR67387.2025.11292212.
- [31] Y. Xu, W. Cai, L. Wang, and T. Xie, "Shock and Vibration - 2021 - Xu - Intelligent Diagnosis of Rolling Bearing Fault Based on Improved Convolutional Neural.pdf," no. 1205473, p. 8, 2021, doi: 10.1155/2021/1205473 *Research*.
- [32] X. Liu, H. Duan, W. Huang, R. Guo, and B. Duan, "Classified Early Warning and Forecast of Severe Convective Weather Based on LightGBM Algorithm," *Atmos. Clim. Sci.*, vol. 11, no. 2160–0422, pp. 284–301, 2021, doi: 10.4236/acs.2021.112017.
- [33] Hierarchical Agglomerative Clustering Algorithm Based Real-Time Event Detection from Online Social Media Network AJ Frhan, Pdfs. *Semanticscholar. Org* 13, 215-222.
- [34] Al-Hitawi MAS and Gyöngyösy NM. Enhancing Transformer-Based Language Models for Hungarian Handwritten Text Recognition. *F1000Research* 2026, 15:181 <https://doi.org/10.12688/f1000research.176408.1>.
- [35] Jasim, O.H., Al-Hitawi, M.A.S., Fattah, M.Y. *et al.* Physics-informed neural networks for predicting collapse potential in chemically stabilized unsaturated gypseous soils: interpretable machine learning framework. *Model. Earth Syst. Environ.* 12, 219 (2026). <https://doi.org/10.1007/s40808-026-02853-1>.
- [36] Selective Arabic Cyberbullying Detection with Multi-View Modeling, Risk Triage, and Human Deferral. (2026). *Al-Noor Journal of Engineering Management and Computer Science*, 2(4), 25-36. <https://doi.org/10.71229/er3kch93>.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

This manuscript's conception, technique, execution, experimentation, analysis, and writing were all entirely the work of AMJAD JUMAAH FRHAN.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0

https://creativecommons.org/licenses/by/4.0/deed.en_US