Malware Classification Using Machine Learning and Dimension Reduction Techniques on PE File Data

Arif Harsa Pradipta[1], Lili Ayu Wulandhari[2]
[1,2]Department of Computer Science, BINUS Graduate Program, Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

Article Info

Article history:

Received Jun 20, 2024
Revised Aug 8, 2024
Accepted Sep 9, 2024

Keywords:

Cyber Security
Machine Learning
Data Dimension Reduction
Malware Classification
Portable Executable Format

ABSTRACT

The digital transformation has enhanced efficiency, transparency, and accessibility but has also led to a notable increase in cyber incidents, including malware attacks. According to the 2022 annual report from the Honeynet Project by the National Cyber and Encryption Agency, Indonesia experienced over 370 million cyber attacks, with 800,000 of these being malware attacks. The increasing complexity of Portable Executable files further complicates accurate classification in machine learning models. This research aims to develop an effective malware detection approach using machine learning classifiers—Random Forest, XGBoost, and AdaBoost—on raw feature dataset and integrated feature dataset. Dimension reduction techniques such as Principal Component Analysis and Linear Discriminant Analysis were utilized to enhance classification efficiency. The results demonstrated that Random Forest and XGBoost consistently outperformed AdaBoost, particularly in classifying ransomware, achieving recall values ranging from 0.72 to 0.85 and F1-scores from 0.74 to 0.81 For the trojan class, both Random Forest and XGBoost achieved recall values ranging from 0.96 to 0.97, with corresponding F1-scores between 0.95 and 0.97. Both classifiers maintained high precision, recall, and F1-scores across all malware classes, even with reduced feature sets.

Corresponding Author:

Arif Harsa Pradipta,
Department of Computer Science,
Bina Nusantara University,
Jl. Raya Kb. Jeruk No.27, RT.1/RW.9, Kemanggisan, Kec. Palmerah, Kota Jakarta Barat, Daerah Khusus Ibukota Jakarta 11530, Indonesia.
Email: arif.pradipta@binus.ac.id

1. INTRODUCTION

The Electronic-Based Government System (SPBE), as stipulated in Presidential Regulation Number 95 of 2018 regarding the Electronic-Based Government System, has revolutionized public administration in Indonesia, including within a certain institution. Through the implementation of advanced and up-to-date information and communication technology, SPBE enables the institution to enhance efficiency, transparency, and accessibility in public services related to infrastructure and housing. This system allows for better data management, real-time monitoring of infrastructure projects, and more effective and efficient communication between various work units within the institution. Alongside the progress and development of digital transformation in the government system, it is undeniable that this correlates with an increase in cyber incidents in the government sector. According to the 2022 annual report from the Honeynet Project by the National Cyber and Encryption Agency, Indonesia experienced over 370 million cyber attacks, with 800,000 of these being malware attacks [1].

To understand malware, malware analysis is essential. Malware analysis is a process aimed at determining and identifying the behavior of malware in attacking a system. There are two primary techniques for analyzing malware: static analysis and dynamic analysis [2]. Static analysis involves examining the source code or executable files without running them, to identify suspicious signs or potential vulnerabilities that could be exploited. In contrast, dynamic analysis involves running the malware in a controlled environment to monitor and understand its behavior. Malware contains valuable information that can aid in classifying and identifying cyber threats, often recorded in the format of a Portable Executable (PE) header.

The Portable Executable (PE) format is used by the Windows operating system to execute programs and store the necessary information for execution. PE files consist of several main sections: the DOS header, PE header, and optional headers, as illustrated in Figure 1. The DOS header is the first part of a PE file and contains information related to backward compatibility with DOS. The PE header is a part of the Windows file structure that contains basic information about the PE file, such as the architecture type (32-bit or 64-bit). The PE header provides essential information stored in various fields, including the compatible machine, the number of sections, the size of the optional headers, the number of symbols, and the timestamp of the compilation date. The optional header, on the other hand, is an integral part of the PE file that describes the logical structure of the PE file and is considered the most critical part of a PE file header [3].

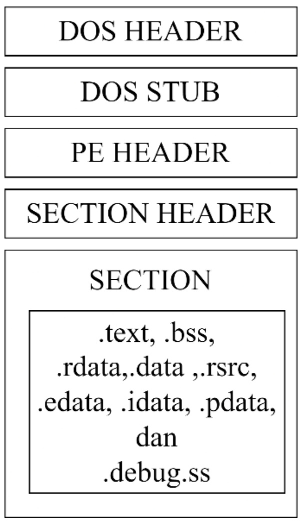


Figure 1 Structure of a Portable Executable File

Data dimensions continue to grow exponentially in size, heterogeneity, complexity, and variety [4]. High-dimensional data in machine learning models can lead to several issues in accurate classification, necessitating the use of dimension reduction techniques. These techniques provide an efficient approach by reducing the number of data dimensions before applying machine learning models [5].

Recent advancements in malware detection have employed a diverse array of machine learning techniques to improve classification accuracy. Hwang et al. [6] introduced a Deep Neural Network (DNN) model, trained on a dataset comprising 10,000 malware and 10,000 benign samples from both Windows and Linux platforms. By allocating 80% of the data for training and 20% for testing, their model achieved a notable accuracy of 94% [6]. Smmarwar et al. [7] utilized Random Forest, Decision Tree, and Support Vector Machine (SVM) classifiers on the CIC-InvesAndMal2019 dataset, reporting accuracies of 82.33% for SVM, 91.32% for Random Forest, and 91.8% for Decision Tree, demonstrating the effectiveness of these classifiers in malware detection. Further advancements were made by Jeon and Moon [8], who developed a Convolutional Recurrent Neural Network (CRNN) model to process opcode sequences. Their approach involved compressing these sequences using an opcode-level convolutional autoencoder (OCAE) and subsequently classifying them with a dynamic recurrent neural network (DRNN), achieving an impressive detection accuracy of 96.2% [8]. Matin and Rahardjo [9] combined honeypots with machine learning algorithms for malware detection. Their use of SVM and Decision Tree classifiers, validated through a 90:10 data split and 10-fold validation, yielded an accuracy of approximately 90%.

Research by Rezaei and Hamze [10] focused on analyzing PE header structures, extracting nine significant features from a dataset of 2,460 PE files, and achieved the highest accuracy of 95.59% with the Random Forest algorithm. In contrast, Maleki et al. [11]proposed a data mining technique based on static features from PE headers, achieving an exceptional accuracy of 98.26% with the Decision Tree algorithm. Penmatsa et al.[12] employed Ant Colony Optimization (ACO) to identify a minimal feature set for malware detection, optimizing data size with results of 97.15% and 92.8% for integrated and raw ClaMP datasets, respectively. In the domain of image-based feature extraction, Manavi and Hamzeh [13]utilized Convolutional Neural Networks (CNN) to classify malware from grayscale images generated from 1,024 bytes of PE headers, achieving an accuracy of 93.33%. Rezaei et al. [14]explored deep learning with raw bytes from PE file headers, attaining accuracies of 92.41% and 97.75% with two separate datasets.

Azeez et al. [15]demonstrated that ensemble learning combined with Principal Component Analysis (PCA) achieved outstanding results, with accuracies of 99.24% for AdaBoost and 98.06% for Random Forest. Dimensionality reduction techniques such as PCA and Linear Discriminant Analysis (LDA) are crucial for addressing the challenges posed by high-dimensional data. PCA reduces data complexity by transforming original features into uncorrelated principal components, while LDA projects datasets with high feature dimensions into a lower-dimensional space with effective class separation. Both techniques enhance classification model accuracy and efficiency while reducing the rate of false positives [16]. Collectively, these studies underscore the substantial progress and efficacy of various machine learning approaches in tackling the challenges of malware detection, particularly when combined with dimensionality reduction techniques like PCA and LDA.

This research aims to address this challenge by proposing a performance measurement approach for classifying malware attacks. The study will integrate feature extraction results from a honeypot with public malware datasets from MalwareBazaar, utilizing the Classification of Malwares (CLaMP) feature extractor [17]. The classification models to be employed include Random Forest, XGBoost, and AdaBoost, while dimension reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) will be applied. By combining these methods, the research seeks to enhance the understanding of how machine learning techniques, in conjunction with dimension reduction, contribute to malware classification. This approach is expected to improve the development of more effective strategies for mitigating cybersecurity threats within organizations.

2. RESEARCH METHOD

Our research methodology for machine learning classification in the domain of malware classification encompasses several pivotal stages aimed at harnessing the power of artificial intelligence to combat evolving cyber threats. These stages consist of (1) data collection, (2) data pre-processing, (3) feature extraction, and (4) classification modelling. Each stage plays a crucial role in the development of an effective classification system capable of accurately discerning between malicious and benign software. Figure 2 illustrates the sequential flow of these steps, delineating the systematic progression from data acquisition to model implementation. Notably, the research employs Python scripts available on GitHub, facilitating seamless access to open-source tools for both data preprocessing and feature extraction.

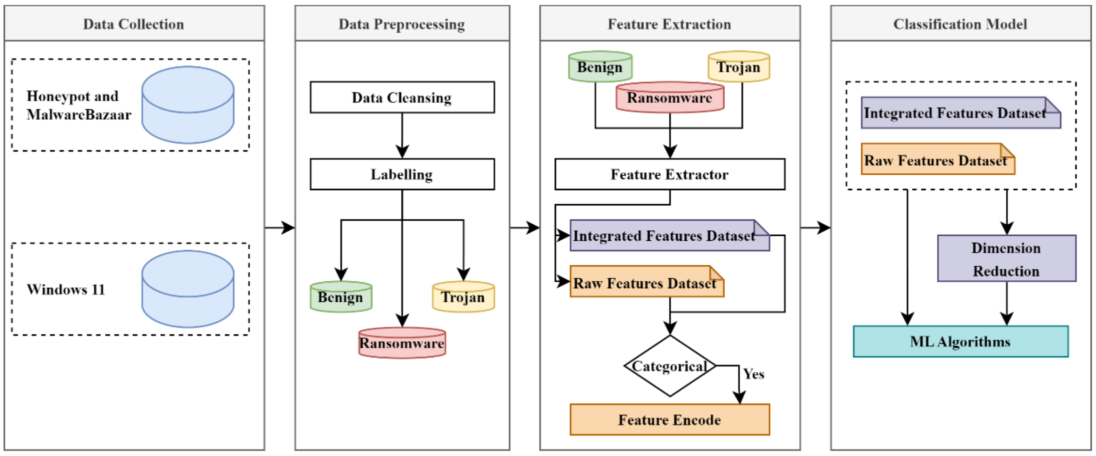


Figure 2. Research stages for malware classification

A variety of algorithms and models, including Random Forest (RF), eXtreme Gradient Boosting (XGBoost), Adaptive Boosting (AdaBoost), Principal Component Analysis (PCA), and Linear discriminant analysis (LDA), were utilized. These methods were tailored for the analysis of data obtained through honeypots and malware samples sourced from platforms such as MalwareBazaar.

2.1. Data Collection

The collection of malware samples through honeypots and MalwareBazaar serves as a crucial endeavor in cybersecurity research and threat intelligence gathering. Honeypots, specialized decoy systems designed to lure and intercept malicious activities, provide a proactive means of capturing real-world malware samples. By mimicking vulnerable systems or services, honeypots attract and trap malicious actors, allowing analysts to observe their tactics, techniques, and procedures (TTPs) firsthand. Additionally, MalwareBazaar, an online repository of malware samples contributed by security researchers and automated systems, offers a vast array of samples for analysis. Each sample undergoes rigorous validation to ensure its authenticity and relevance to current threat landscapes. Through both platforms, we gain access to diverse malware specimens, ranging from common malware families to newly emerging threats. This comprehensive dataset enables researchers to study malware behaviors, extract features for analysis, and develop robust defense mechanisms, including intrusion detection systems and antivirus solutions. Through both platforms, a total of 2042 malware samples were successfully collected, comprising 1840 trojans and 202 ransomware samples.

Collecting benign samples from Windows 11 files in various formats such as Portable Executable (.exe), system files (.sys), and dynamic link libraries (.dll) involves a targeted approach to sourcing legitimate and non-malicious files specific to the Windows 11 environment. These samples are obtained from trusted sources such as official software repositories, digital distribution platforms, or reputable software vendors providing compatibility updates or driver packages tailored for Windows 11. Additionally, benign files are validated to be malware-free using up-to-date antivirus software and are extracted from clean installations of the Windows 11 operating system or derived from well-known software applications compatible with the new platform. This ensures that the dataset reflects a realistic and secure computing environment, supporting the development and evaluation of malware detection systems in a controlled yet representative manner. To ensure an equal number of benign and malware samples, we collected 2042 benign samples from Windows 11.

2.2. Data Preprocessing

At this stage, the data or files intended for use as datasets undergo a pre-processing step, which involves cleansing by selecting suitable data for further processing. These samples, obtained from honeypots, MalwareBazaar, and Windows 11, consist of portable executables. To process these samples, we extract signatures by calculating the MD5 hash for each file. This is achieved using a Python script that opens each binary file, reads the data in chunks, updates the MD5 hash object with each chunk, and then returns the MD5 hash result in hexadecimal format. This method verifies the file's integrity and generates a unique signature for each file. To prevent duplication, each sample undergoes a labeling process where MD5 hashes are extracted, and samples with identical hash values are removed.

The collected MD5 hash samples are recorded and labeled to categorize each sample as trojan, ransomware, or benign. To accurately label these samples, we use VirusTotal's API, which scans the samples and reports the results in .csv format. The pre-processed dataset comprises a total of 4,084 samples, including 1,840 trojan samples, 202 ransomware samples, and 2,042 benign samples, as detailed in Table 1.

Table 1. The total number of samples

Dataset	Trojan	Ransomware	Benign	Total
File Samples	1840	202	2042	4084

2.3. Feature Extraction

The labeled malware and benign samples will undergo feature extraction using both raw and integrated feature extractors from ClaMP (Classification of Malware with PE headers) [17]. The extracted values will be recorded according to the feature columns specified in the header. The raw feature set is generated by retrieving data from every field within the three primary headers of a PE file, namely the DOS header, File Header, and Optional header, encompassing both standard and Windows-specific fields. This compilation results in a total of 55 features, with 19 derived from the DOS header, 7 from the File Header, and the remaining 29 extracted from the Optional header.

The integrated feature set is formulated through the amalgamation of specific raw features, resulting in a cohesive representation of pertinent characteristics. The integrated feature set consists of 69 features, with a detailed breakdown with 6 features originating from the DOS header, 37 features from the File header, 17 features from the Optional header. Furthermore, 9 derived features are incorporated, encapsulating synthesized insights derived from the interplay of various raw feature components.

Derived features are characteristics obtained by evaluating the raw values extracted from the PE header against a predefined set of rules [18]. These rules serve as criteria to validate and interpret the raw data, enabling the derivation of additional insights beyond the initial feature set. By applying logical conditions and thresholds to the raw values, derived features offer refined information about the executable file's attributes, behavior, or potential risks. This process enhances the comprehensiveness and granularity of the feature set, empowering deeper analysis and more nuanced understanding of the PE file's properties.

The extraction process of Portable Executable (PE) files yields both raw and integrated features as shown in Table 2, providing valuable insights into the structural and behavioral aspects of the analyzed files. Raw features consist of 34 categorical and 21 numerical attributes, encompassing various elements such as file header information, section characteristics, and memory allocation details. These include e_magic, Machine type, Characteristics, as well as SizeOfCode, ImageBase, and other numerical parameters essential for understanding the file's layout and execution environment. On the other hand, integrated features incorporate an expanded set of 54 categorical and 15 numerical attributes, integrating additional information derived from header characteristics, subsystem details, and packer identification. These features, including FH_char0 to FH_char14 and OH_DLLchar0 to OH_DLLchar10, offer deeper insights into the file's composition, its potential functionalities, and any obfuscation techniques employed. Furthermore, features like sus_sections, packer_type, and E_text provide context on suspicious sections, packing methods, and the presence of executable code within the file.

Table 2. Comparison of Raw and Integrated Features

Raw Features	Data Type	Integrated Features	Data Type
e_magic	Categorical	e_cblp	Categorical
e_cblp	Categorical	e_cp	Categorical
e_cp	Categorical	e_cparhdr	Categorical
e_crlc	Categorical	e_maxalloc	Categorical
e_cparhdr	Categorical	e_sp	Categorical
e_minalloc	Categorical	e_lfanew	Numerical
e_maxalloc	Categorical	NumberOfSections	Categorical
e_ss	Categorical	CreationYear	Categorical
e_sp	Categorical	FH_char0	Categorical
e_csum	Categorical	FH_char1	Categorical
e_ip	Categorical	FH_char2	Categorical
e_cs	Categorical	FH_char3	Categorical
e_lfarlc	Categorical	FH_char4	Categorical
e_ovno	Categorical	FH_char5	Categorical
e_res	Categorical	FH_char6	Categorical
e_oemid	Categorical	FH_char7	Categorical
e_oeminfo	Categorical	FH_char8	Categorical
e_res2	Categorical	FH_char9	Categorical
e_lfanew	Numerical	FH_char10	Categorical
Machine	Categorical	FH_char11	Categorical
NumberOfSections	Categorical	FH_char12	Categorical
CreationYear	Categorical	FH_char13	Categorical
PointerToSymbolTable	Numerical	FH_char14	Categorical
NumberOfSymbols	Numerical	MajorLinkerVersion	Categorical
SizeOfOptionalHeader	Categorical	MinorLinkerVersion	Categorical
Characteristics	Categorical	SizeOfCode	Numerical
Magic	Categorical	SizeOfInitializedData	Numerical
MajorLinkerVersion	Categorical	SizeOfUninitializedData	Numerical
MinorLinkerVersion	Categorical	AddressOfEntryPoint	Numerical
SizeOfCode	Numerical	BaseOfCode	Numerical
SizeOfInitializedData	Numerical	BaseOfData	Numerical
SizeOfUninitializedData	Numerical	ImageBase	Numerical
AddressOfEntryPoint	Numerical	SectionAlignment	Categorical
BaseOfCode	Numerical	FileAlignment	Categorical
BaseOfData	Numerical	MajorOperatingSystemVersion	Categorical
ImageBase	Numerical	MinorOperatingSystemVersion	Categorical
SectionAlignment	Numerical	MajorImageVersion	Categorical
FileAlignment	Numerical	MinorImageVersion	Categorical
MajorOperatingSystemVersion	Categorical	MajorSubsystemVersion	Categorical
MinorOperatingSystemVersion	Categorical	MinorSubsystemVersion	Categorical
MajorImageVersion	Categorical	SizeOfImage	Categorical
MinorImageVersion	Categorical	SizeOfHeaders	Categorical
MajorSubsystemVersion	Categorical	CheckSum	Numerical
MinorSubsystemVersion	Categorical	Subsystem	Categorical
SizeOfImage	Numerical	OH_DLLchar0	Categorical
SizeOfHeaders	Numerical	OH_DLLchar1	Categorical
CheckSum	Numerical	OH_DLLchar2	Categorical
Subsystem	Categorical	OH_DLLchar3	Categorical
DllCharacteristics	Categorical	OH_DLLchar4	Categorical
SizeOfStackReserve	Numerical	OH_DLLchar5	Categorical
SizeOfStackCommit	Numerical	OH_DLLchar6	Categorical
SizeOfHeapReserve	Numerical	OH_DLLchar7	Categorical
SizeOfHeapCommit	Numerical	OH_DLLchar8	Categorical
LoaderFlags	Numerical	OH_DLLchar9	Categorical
NumberOfRvaAndSizes	Numerical	OH_DLLchar10	Categorical
		SizeOfStackReserve	Numerical
		SizeOfStackCommit	Numerical
		SizeOfHeapReserve	Numerical
		SizeOfHeapCommit	Numerical
		LoaderFlags	Numerical
		sus_sections	Categorical

Raw Features	Data Type	Integrated Features	Data Type
		non_sus_sections	Categorical
		packer	Categorical
		packer_type	Categorical
		E_text	Categorical
		E_data	Categorical
		filesize	Numerical
		E_file	Categorical
		fileinfo	Categorical

In the feature encoding stage, categorical variables within the PE header features will undergo transformation into numerical representations suitable for machine learning algorithms. This process ensures that categorical data, such as header flags or file characteristics, are effectively incorporated into the classification model. Techniques like frequency encoding will be employed to convert categorical variables into a format understandable by machine learning algorithms. By encoding features in this manner, the model can better interpret and utilize all available information, enhancing its ability to accurately classify malware samples. Integration of feature encoding alongside dimension reduction contributes to the refinement and optimization of the dataset, paving the way for more robust and efficient classification models in malware detection and classification tasks.

2.4. Classification Model

The research is conducted both with and without the utilization of dimension reduction techniques. Techniques such as PCA and LDA can aid in reducing the dimensions of complex PE header features into fewer dimensions while retaining crucial information. This facilitates more efficient analysis and the formation of classification models to accurately distinguish or identify malware. By employing dimension reduction, the complexity of the feature space is reduced, allowing for streamlined analysis and potentially improving the performance of the classification models. Overall, incorporating dimension reduction techniques enhances the effectiveness and accuracy of malware detection and classification systems by focusing on the most informative features while mitigating the impact of high-dimensional data.

Raw feature set and integrated feature set will be used to develop a malware classification model using a machine learning approach to classify into malware and benign classes. The data will be divided into training data (X_train and y_train) and testing data (X_test and y_test) with an 80:20 split ratio. X_train represents the values of each feature present in the PE header, while y_train denotes the class labels used for training the model. Similarly, X_test comprises the values of each feature present in the PE header, and y_test represents the testing class labels. In this stage, the choice of algorithms such as Random Forest, XGBoost, and AdaBoost will be determined for model development.

Figure 3 illustrating the sequential steps involved in the predictive modeling process. The flowchart commences with the input dataset, which undergoes initial preprocessing to address data quality issues and prepare it for subsequent analysis. Following dataset preparation, dimension reduction techniques such as PCA or LDA are applied to reduce the dimensionality of the feature space while preserving relevant information and enhancing computational efficiency. Subsequently, the reduced-dimensional data is fed into a selection of machine learning classifiers. Through iterative training and validation, these classifiers generate models optimized for the dataset under consideration. Finally, the selected model undergoes rigorous evaluation to assess the best algorithm based on its precision, recall, F1-score and accuracy.

The first approach begins with a dataset which is used to train a machine learning (ML) classifier. Following this, parameter tuning is performed to identify the best set of hyperparameters, resulting in the selection of the best model that is subsequently evaluated for its performance. The second approach enhances this process by incorporating feature selection, specifically using feature importance techniques, prior to training the ML classifier. This step aims to improve model performance by selecting the most relevant features, followed by parameter tuning, model selection, and evaluation. The third approach involves dimensionality reduction using LDA before training the ML classifier. LDA helps in reducing the number of features by projecting the data onto a lower-dimensional space, which can improve model efficiency and effectiveness. After dimensionality reduction, the steps of parameter tuning, model selection, and evaluation are carried out. Similarly, the fourth approach applies dimensionality reduction using PCA prior to training the ML classifier. PCA reduces dimensionality by transforming the features into principal components that capture the most variance in the data. This is followed by the standard steps of parameter tuning, best model selection, and evaluation to determine the model's performance.

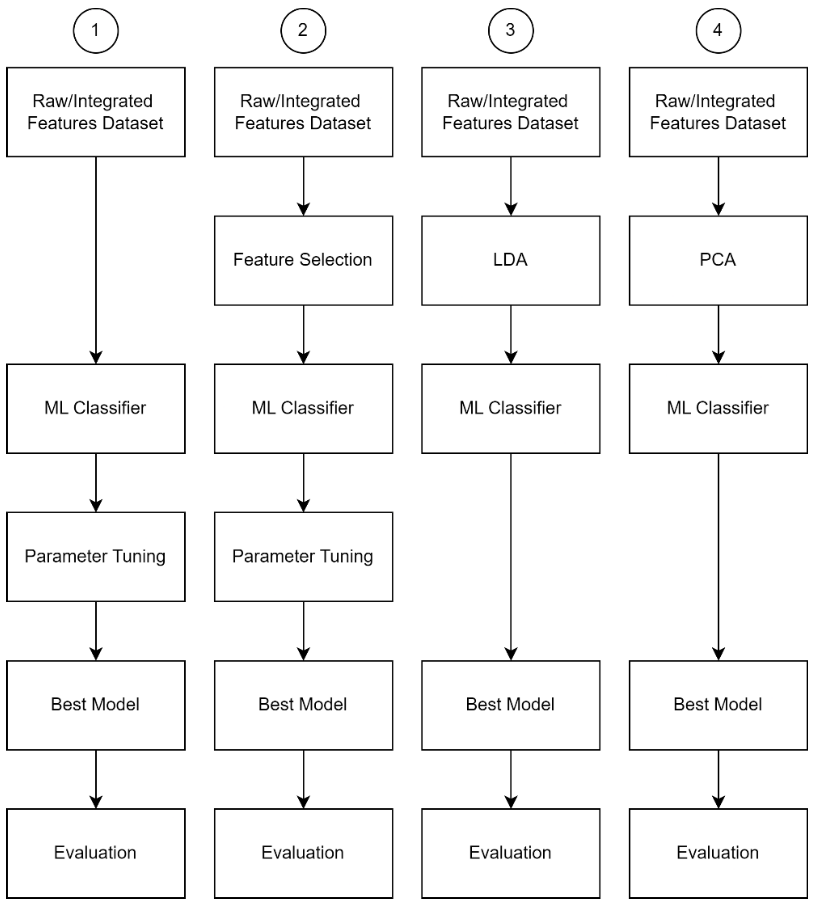


Figure 3. Classification model with and without dimension reduction technique

In this section, we detailed the research methodology employed to investigate the efficacy of various machine learning workflows. This included descriptions of the dataset, preprocessing techniques, feature selection methods, dimensionality reduction approaches, and the parameter tuning processes utilized to identify the optimal models. With a clear understanding of the experimental design and procedures, we now transition to the next section where we present and analyze the findings. In the Results and Discussion section, we will evaluate the performance of each approach, compare their outcomes, and discuss the implications of these results in the context of the research's objectives and the broader field of machine learning.

3. RESULTS AND DISCUSSION

In our experiment into malware classification, we sought to meticulously analyze and classify samples into three distinct categories: benign, trojan, and ransomware. This endeavor involved employing advanced machine learning techniques augmented by parameter tuning, feature selection, and dimensionality reduction strategies. By systematically exploring these subdomains, we aimed to enhance the accuracy and robustness of our classification models, thereby contributing to the ongoing efforts in cybersecurity research. In this results and discussion section, we delve into the outcomes of our experiments across these five pivotal areas: Machine Learning Classification and Parameter Tuning, Machine Learning Classification with Feature Selection and Parameter Tuning, Machine Learning Classification with Dimensionality Reduction using LDA, Machine Learning Classification with Dimensionality Reduction using PCA, and Experiment Summary Disscusion. Through a comprehensive analysis of our findings, we aim to elucidate the efficacy of these methodologies in accurately distinguishing between benign, trojan, and ransomware samples, ultimately fortifying digital infrastructures against malicious cyber threats.

In our exploration of these pivotal areas, we implemented and evaluated several machine learning algorithms to determine their efficacy in malware classification. Among the algorithms we selected for this task were Random Forest, AdaBoost, and XGBoost. Each of these algorithms offers unique advantages and contributes differently to our goal of accurately classifying benign, trojan, and ransomware samples. Random Forest, with its ensemble learning approach, aggregates multiple decision trees to enhance classification accuracy and robustness. AdaBoost, known for its adaptive boosting technique, iteratively adjusts weights to focus on hard-to-classify samples, thereby improving the model's performance. This technique leverages the principle of boosting by constructing a series of weak classifiers and combining them into a stronger model, specifically emphasizing the importance of difficult-to-predict data in subsequent iterations [19]. XGBoost, an optimized implementation of gradient boosting, leverages advanced regularization techniques and efficient handling of missing values to achieve high predictive accuracy and speed. Across various datasets and machine learning tasks, comparisons with several existing tree boosting systems have demonstrated that XGBoost can deliver state-of-the-art results with greater speed and lower resource consumption than other systems. Furthermore, XGBoost is capable of handling very large-scale data by utilizing out-of-core computing and distributed computing techniques [20].

In the following sections, we will delve deeper into the application and performance of these three algorithms within the context of our research, highlighting their comparative strengths and weaknesses. This detailed examination will provide insights into how each algorithm contributes to the overarching goal of enhancing malware detection and fortifying cybersecurity defenses.

3.1. Machine Learning Classification and Parameter Tuning

The performance of three different machine learning algorithms—AdaBoost, Random Forest, and XGBoost—was conducted using a raw feature dataset. The evaluation metrics considered were precision, recall, F1-score, and accuracy. The Table 3 presents the parameter tuning configurations for three machine learning algorithms: Adaboost, Random Forest, and XGBoost. For Adaboost, the parameters include algorithm, which can be either 'SAMME' or 'SAMME.R'; n_estimators, ranging from 10 to 150; and learning_rate, with values from 0.01 to 1. The algorithm parameter specifies the boosting method, with 'SAMME' used for multi-class AdaBoost and 'SAMME.R' for real AdaBoost. The n_estimators parameter controls the number of weak learners (decision trees) added to the model, with more estimators potentially improving performance but increasing computation time. The learning_rate parameter adjusts the contribution of each estimator, where smaller values require more estimators for optimal performance.

Table 3. Machine Learning Algorithm Parameters and Values

Algorithms	Parameter	Values
Adaboost	algorithm	['SAMME', 'SAMME.R']
	n_estimators	[10, 25, 50, 75, 100, 125, 150]
	learning_rate	[0.01, 0.02, 0.04, 0.06, 0.08, 0.1, 0.2, 0.4, 0.6, 0.8, 1]
Random Forest	criterion	['gini', 'entropy']
	n_estimators	[10, 25, 50, 75, 100, 125, 150]
	max_depth	[8, 16, 32, 48, 64, 80, 96, 112, 128, 144]
XGBoost	n_estimators	[10, 25, 50, 75, 100, 125, 150]
	learning_rate	[0.01, 0.02, 0.04, 0.06, 0.08, 0.1, 0.2, 0.4, 0.6, 0.8, 1]
	max_depth	[8, 16, 32, 48, 64, 80, 96, 112, 128, 144]

The models were fine-tuned with the following parameters: AdaBoost (algorithm = 'SAMME', n_estimators = 75, and learning_rate = 0.6), Random Forest (criterion = 'entropy', n_estimators = 75, and max_depth = 16), and XGBoost (learning_rate = 1, max_depth = 8, and n_estimators = 150) The results are summarized in the Table 4 below:

Table 4. Evaluation Metrics for Raw Feature Dataset in Experiment 1

Raw Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.98	0.95	0.97	0.94
	Trojan	0.90	0.98	0.94	
	Ransomware	0.89	0.41	0.56	
Random Forest	Benign	0.99	0.98	0.99	0.97
	Trojan	0.97	0.97	0.97	
	Ransomware	0.78	0.82	0.80	
XGBoost	Benign	0.98	0.99	0.99	0.97
	Trojan	0.98	0.96	0.97	
	Ransomware	0.79	0.85	0.81	

While the integrated feature dataset, the models were fine-tuned with different parameters: AdaBoost (algorithm = 'SAMME', n_estimators = 150, and learning_rate=1), Random Forest (criterion = 'gini', n_estimators = 25, and max_depth = 32) and XGBoost (learning_rate = 0.8, max_depth = 8, and n_estimators = 125). The performance metrics are summarized in the Table 5 below:

Table 5. Evaluation Metrics for Integrated Feature Dataset in Experiment 1

Integrated Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.97	0.96	0.97	0.95
	Trojan	0.93	0.96	0.95	
	Ransomware	0.81	0.67	0.73	
Random Forest	Benign	0.99	0.99	0.99	0.98
	Trojan	0.98	0.97	0.98	
	Ransomware	0.80	0.85	0.83	
XGBoost	Benign	0.98	0.99	0.99	0.97
	Trojan	0.98	0.97	0.97	
	Ransomware	0.80	0.82	0.81	

AdaBoost shows strong performance for benign and Trojan classes, achieving high F1-scores of 0.97 and 0.94 when using the raw feature dataset. However, it struggles significantly with ransomware detection, obtaining a low F1-score of 0.56. When using the integrated feature dataset, AdaBoost maintains its high performance for benign and Trojan classes, with F1-scores of 0.97 and 0.95, respectively. Despite this, it only shows a slight improvement in ransomware detection, achieving an F1-score of 0.73.

Random Forest demonstrates excellent performance across all classes in the raw feature dataset. It achieves F1-scores of 0.99 for benign and 0.97 for Trojan, with comparatively better ransomware detection than AdaBoost, achieving an F1-score of 0.80. Using the integrated feature dataset, Random Forest further improves its performance, achieving almost perfect F1-scores for benign and Trojan classes. Ransomware detection is significantly enhanced, with an F1-score of 0.83, showcasing the robustness of Random Forest in handling diverse malware types.

XGBoost For the raw feature dataset shows high performance for benign and Trojan classes, achieving F1-scores of 0.99 and 0.97, respectively. Ransomware detection is slightly better than Random Forest, with an F1-score of 0.81. With the integrated feature dataset, XGBoost maintains its high performance for benign and Trojan classes, with F1-scores of 0.99 and 0.97, respectively. The ransomware detection performance remains consistent, with an F1-score of 0.81, indicating stability in its detection capabilities across different datasets.

Comparing the two datasets, it is evident that the integrated feature dataset generally enhances the performance of all algorithms, particularly for the Ransomware class, which is more challenging to classify accurately. AdaBoost, while improving for the Trojan class and achieving better recall and F1-score for Ransomware, still lags behind Random Forest and XGBoost in handling the latter. Random Forest benefits the most from the integrated features, showing marked improvement in its handling of Ransomware and a slight boost in overall accuracy. XGBoost, while showing minor improvements, maintains its strong and consistent performance across both datasets.

3.2. Machine Learning Classification with Feature Selection and Parameter Tuning

The evaluation of AdaBoost, Random Forest, and XGBoost algorithms using the raw feature dataset demonstrates varying levels of performance across different malware classes: benign, Trojan, and ransomware. The number of features used in this dataset is 13 for AdaBoost and 16 for both Random Forest and XGBoost, compared to a total of 55 features in the raw feature dataset. Using the parameters as shown in the Table 3, the models were fine-tuned with the following settings.: AdaBoost (algorithm='SAMME.R', n_estimators = 75, and learning_rate = 0.2), Random Forest (criterion = 'entropy', n_estimators = 50, and max_depth = 32), and XGBoost (learning_rate = 0.1, max_depth = 16, and n_estimators = 100). The performance metrics are summarized in the Table 6 below.

Table 6. Evaluation Metrics for Raw Feature Dataset in Experiment 2

Raw Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.98	0.96	0.97	0.94
	Trojan	0.91	0.98	0.94	
	Ransomware	0.84	0.41	0.55	
Random Forest	Benign	0.99	0.99	0.99	0.97
	Trojan	0.97	0.97	0.97	
	Ransomware	0.74	0.82	0.78	
XGBoost	Benign	0.99	0.99	0.99	0.97
	Trojan	0.96	0.97	0.97	
	Ransomware	0.76	0.72	0.74	

In contrast, the integrated feature dataset uses a total of 17 features for AdaBoost and 24 features for both Random Forest and XGBoost, compared to a total of 69 features in the integrated feature dataset.. The

models were fine-tuned with different parameters: AdaBoost (algorithm = 'SAMME', n_estimators = 150, and learning_rate = 1), Random Forest (criterion = 'gini', n_estimators = 150, and max_depth = 16), and XGBoost (learning_rate = 1, max_depth = 16, n_estimators = 10, and random_state = 42). The performance metrics for this dataset are presented in the
Table *7* below:

Table 7. Evaluation Metrics for Integrated Feature Dataset in Experiment 2

Integrated Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.96	0.95	0.96	0.94
	Trojan	0.92	0.95	0.93	
	Ransomware	0.76	0.67	0.71	
Random Forest	Benign	0.99	0.99	0.99	0.98
	Trojan	0.97	0.98	0.98	
	Ransomware	0.86	0.79	0.83	
XGBoost	Benign	0.99	0.99	0.99	0.97
	Trojan	0.97	0.97	0.97	
	Ransomware	0.76	0.72	0.74	

AdaBoost for the raw feature dataset shows strong performance for benign and Trojan classes, achieving high F1-scores of 0.97 and 0.94, respectively. However, it struggles significantly with ransomware detection, obtaining a low F1-score of 0.55. When using the integrated feature dataset, AdaBoost maintains its high performance for benign and Trojan classes, with F1-scores of 0.96 and 0.93, respectively. Despite this, it only shows a slight improvement in ransomware detection, achieving an F1-score of 0.71.

Random Forest demonstrates excellent performance across all classes in the raw feature dataset. It achieves F1-scores of 0.99 for benign and 0.97 for Trojan, with comparatively better ransomware detection than AdaBoost, achieving an F1-score of 0.78. Using the integrated feature dataset, Random Forest further improves its performance, achieving almost perfect F1-scores for benign and Trojan classes. Ransomware detection is significantly enhanced, with an F1-score of 0.83, showcasing the robustness of Random Forest in handling diverse malware types.

XGBoost performance for the raw feature dataset, shows high performance for benign and Trojan classes, achieving F1-scores of 0.99 and 0.97, respectively. Ransomware detection is moderate, with an F1-score of 0.74. With the integrated feature dataset, XGBoost maintains its high performance for benign and Trojan classes, with F1-scores of 0.99 and 0.97, respectively. The ransomware detection performance remains consistent, with an F1-score of 0.74, indicating stability in its detection capabilities across different datasets.

In summary, the integrated feature dataset generally enhances the performance of the algorithms, particularly for the Random Forest model, which shows the most notable improvement in ransomware detection. AdaBoost benefits slightly from the integrated features but still lags behind in ransomware detection. XGBoost maintains consistent performance across both datasets. These results underscore the importance of feature selection and parameter tuning in optimizing the performance of machine learning algorithms for malware detection. The integrated feature dataset, with its broader and more comprehensive set of features, provides a more robust framework for detecting various types of malware effectively.

3.3. Machine Learning Classification with Dimensionality Reduction using LDA

The results from the evaluations on both raw feature and integrated feature datasets, using LDA with n_components = 2, reveal significant insights into the performance of AdaBoost, Random Forest, and XGBoost algorithms. The performance metrics for this dataset are presented in the Table 7 and Table 8 below.

Table 8. Evaluation Metrics for Raw Feature Dataset in Experiment 3

Raw Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.94	0.94	0.94	0.91
	Trojan	0.88	0.96	0.92	
	Ransomware	0.25	0.03	0.05	
Random Forest	Benign	0.98	0.97	0.98	0.96
	Trojan	0.96	0.96	0.96	
	Ransomware	0.76	0.79	0.77	
XGBoost	Benign	0.98	0.97	0.97	0.95
	Trojan	0.95	0.96	0.96	
	Ransomware	0.77	0.77	0.77	

Table 9. Evaluation Metrics for Integrated Feature Dataset in Experiment 3

Integrated Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.95	0.96	0.96	0.94
	Trojan	0.94	0.95	0.94	
	Ransomware	0.73	0.56	0.64	
Random Forest	Benign	0.98	0.98	0.98	0.96
	Trojan	0.96	0.96	0.96	
	Ransomware	0.74	0.79	0.77	
XGBoost	Benign	0.98	0.97	0.98	0.96
	Trojan	0.95	0.96	0.96	
	Ransomware	0.74	0.74	0.74	

For the raw feature dataset, AdaBoost achieves a precision of 0.94, recall of 0.94, and an F1-score of 0.94 for the benign class, with an overall accuracy of 0.91. The Trojan class also performs well with an F1-score of 0.92, but the algorithm struggles with ransomware, showing a poor F1-score of 0.05. Random Forest performs exceptionally well, achieving an F1-score of 0.98 for the benign class and 0.96 for the Trojan class, with an overall accuracy of 0.96. It also shows relatively better performance for ransomware detection with an F1-score of 0.77. Similarly, XGBoost shows strong performance for benign and Trojan classes, with F1-scores of 0.97 and 0.96, respectively, and an F1-score of 0.77 for ransomware, achieving an overall accuracy of 0.95.

In the integrated feature dataset, AdaBoost shows improved performance across all classes. The benign class achieves an F1-score of 0.96, and the Trojan class achieves an F1-score of 0.94, with an overall accuracy of 0.94. Importantly, ransomware detection significantly improves, with an F1-score of 0.64. Random Forest maintains high performance with an F1-score of 0.98 for the benign class and 0.96 for the Trojan class, and an F1-score of 0.77 for ransomware, showing an overall accuracy of 0.96. XGBoost also shows consistent performance, achieving an F1-score of 0.98 for the benign class and 0.96 for the Trojan class, with a slightly lower F1-score of 0.74 for ransomware, maintaining an overall accuracy of 0.96.

These results highlight the robustness and reliability of Random Forest and XGBoost across both datasets, while also demonstrating how the integrated feature dataset, enhanced by LDA, significantly boosts AdaBoost's performance in ransomware detection. The use of integrated feature and dimensionality reduction with LDA proves beneficial, particularly for AdaBoost, indicating its potential for improved malware detection capabilities.

3.4. Machine Learning Classification with Dimensionality Reduction using PCA

The evaluation of the raw feature dataset and the integrated feature dataset, both reduced in dimensionality using PCA, provides critical insights into the performance of the machine learning algorithms AdaBoost, Random Forest, and XGBoost. For the raw feature dataset, consisting of 55 features, PCA was applied with n_components=22, resulting in a cumulative variance of 90.53%. For the integrated feature dataset, consisting of 69 features, PCA was applied with n_components=30, resulting in a cumulative variance of 90.11%. The performance metrics for these algorithms on both datasets are summarized and compared in Table 10 and Table 11 below.

Table 10. Evaluation Metrics for Raw Feature Dataset in Experiment 4

Raw Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.96	0.94	0.95	0.93
	Trojan	0.92	0.95	0.93	
	Ransomware	0.69	0.56	0.62	
Random Forest	Benign	0.98	0.98	0.98	0.96
	Trojan	0.96	0.96	0.96	
	Ransomware	0.76	0.82	0.79	
XGBoost	Benign	0.98	0.97	0.97	0.96
	Trojan	0.96	0.96	0.96	
	Ransomware	0.76	0.82	0.79	

The raw feature dataset results show that Random Forest and XGBoost achieve the highest overall accuracy of 0.96 and F1-scores for benign and Trojan classes. However, ransomware detection remains a challenge, particularly for AdaBoost, which has a notably lower F1-score of 0.62 compared to Random Forest of 0.79 and XGBoost 0.79.

Table 11. Evaluation Metrics for Integrated Feature Dataset in Experiment 4

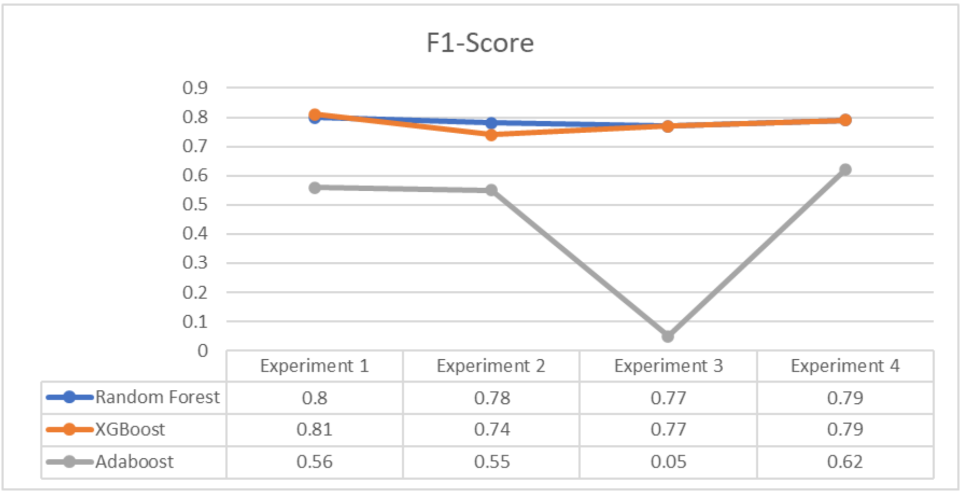
Integrated Features Dataset					
Algorithms	Class	Precision	Recall	F1-Score	Accuracy
AdaBoost	Benign	0.98	0.96	0.97	0.93
	Trojan	0.93	0.94	0.94	
	Ransomware	0.52	0.59	0.55	
Random Forest	Benign	0.98	0.97	0.98	0.96
	Trojan	0.95	0.97	0.96	
	Ransomware	0.78	0.74	0.76	
XGBoost	Benign	0.98	0.97	0.98	0.96
	Trojan	0.96	0.96	0.96	
	Ransomware	0.73	0.77	0.75	

The integrated feature dataset results indicate that AdaBoost, while effective for benign and Trojan detections, significantly underperforms in ransomware detection with an F1-score of 0.55. Both Random Forest and XGBoost maintain high performance for benign and Trojan classes with F1-scores close to 0.98 and 0.96, respectively. However, they show a slight decline in ransomware detection compared to the raw feature dataset, with Random Forest achieving an F1-score of 0.76 and XGBoost 0.75.

The comparative analysis of both datasets highlights several key findings. First, PCA effectively reduces dimensionality while retaining significant variance, facilitating efficient model training and evaluation. For the raw feature dataset, Random Forest and XGBoost consistently achieve high accuracy and F1-scores across all classes, particularly excelling in ransomware detection compared to AdaBoost. The integrated feature dataset further demonstrates the robustness of Random Forest and XGBoost, though with a minor decrease in ransomware detection performance. AdaBoost, despite showing high performance for benign and Trojan classes, remains less effective for ransomware detection across both datasets.

3.5. Experiment Summary Disscusion

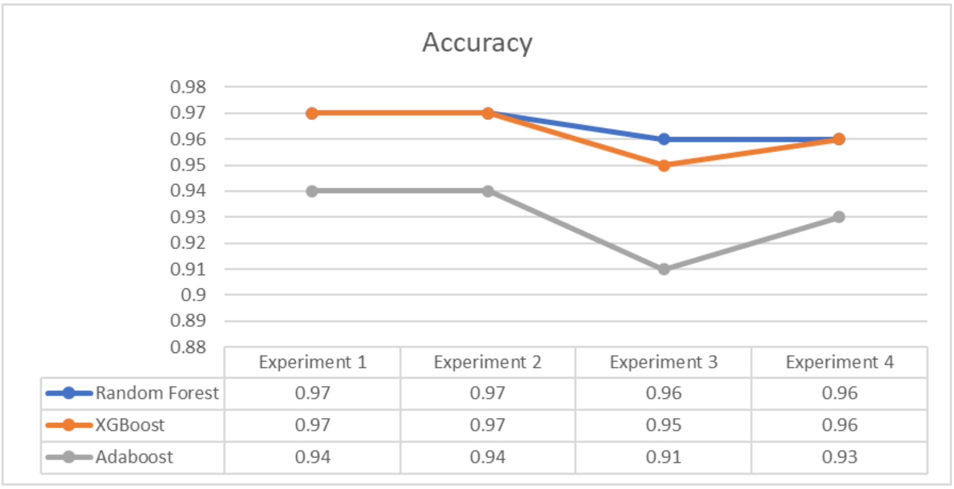
The evaluation of the raw feature dataset across four experiments reveals that AdaBoost generally performs less effectively compared to Random Forest, XGBoost, particularly when focusing on F1-score metrics. For the benign and trojan classes, AdaBoost achieves relatively high F1-scores, ranging from 0.92 to 0.97. However, its performance on the ransomware class is notably weak, with the F1-score reaching only 0.62 at best in Experiment IV, as reflected in Table 10. This significant drop in performance for the ransomware class highlights AdaBoost's difficulty in handling more complex classifications within this dataset. In contrast, Random Forest and XGBoost demonstrate more robust and consistent performance across all classes, as indicated in Table 4 and Table 6. Random Forest achieves high F1-scores for the benign (0.98-0.99) and trojan (0.96-0.97) classes, with the ransomware class showing F1-scores ranging from 0.77 to 0.80. Similarly, XGBoost maintains strong performance, with F1-scores for the benign class between 0.97 and 0.99, and for the trojan class between 0.96 and 0.97. For the ransomware class, XGBoost shows F1-scores between 0.74 and 0.81, indicating better handling of this complex class compared to AdaBoost. Among the four experiments, Experiment I stands out as the best model for the raw feature dataset, with XGBoost showing slightly better performance on the ransomware class (F1-score 0.81), as seen in Table 4 and Figure 4, suggesting its effectiveness in managing the variability within the dataset.



	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Random Forest	0.8	0.78	0.77	0.79
XGBoost	0.81	0.74	0.77	0.79
Adaboost	0.56	0.55	0.05	0.62

Figure 4. Comparison of Ransomware F1-scores Using the Raw Feature Dataset.

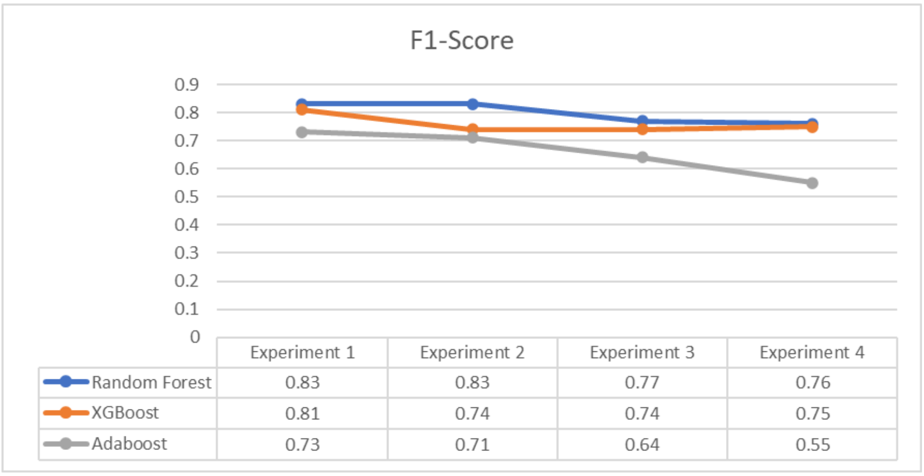
When evaluating the accuracy of the raw feature dataset across the four experiments, it becomes evident that AdaBoost underperforms compared to Random Forest and XGBoost. The accuracy for AdaBoost fluctuates, showing its limitations in handling the variability of the dataset. Specifically, the accuracy for AdaBoost across the experiments ranges from 0.91 to 0.94, with Experiment 1 and Experiment 2 achieving the highest accuracy of 0.94, as indicated in Figure 5. This is significantly lower than the performance of Random Forest and XGBoost, which consistently demonstrate higher accuracy levels.



	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Random Forest	0.97	0.97	0.96	0.96
XGBoost	0.97	0.97	0.95	0.96
Adaboost	0.94	0.94	0.91	0.93

Figure 5. Comparison of Accuracy Using the Raw Feature Dataset.

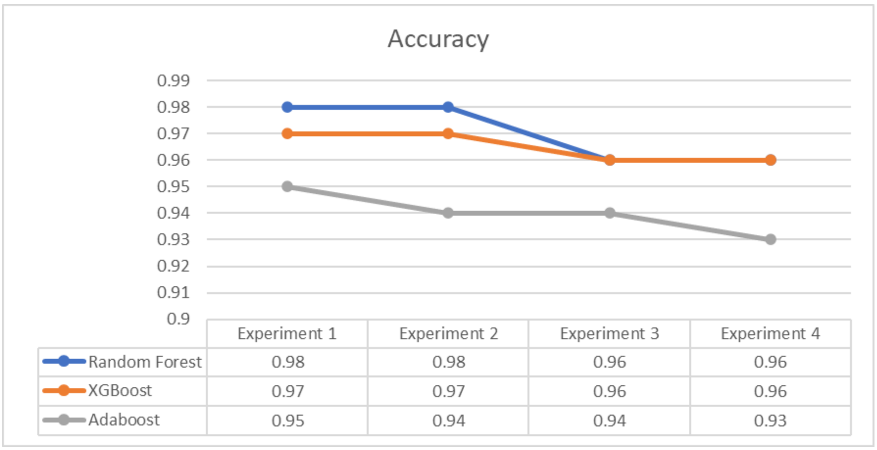
For the integrated feature dataset, the trend of AdaBoost underperforming compared to Random Forest and XGBoost persists. Although AdaBoost shows moderate improvements, it still lags in performance, especially for the ransomware class as illustrated in Figure 6, with its F1-score peaking at 0.73. Random Forest and XGBoost again demonstrate superior performance with integrated features, maintaining high F1-scores for the benign (0.97-0.99) and trojan (0.96-0.98) classes, and F1-scores for the ransomware class ranging from 0.76 to 0.83. Experiment I with Random Forest is identified as the best model for the integrated feature dataset, achieving the highest F1-score for the ransomware class (0.83), demonstrating its effective utilization of integrated features for robust classification, as shown in Table 5.



	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Random Forest	0.83	0.83	0.77	0.76
XGBoost	0.81	0.74	0.74	0.75
Adaboost	0.73	0.71	0.64	0.55

Figure 6. Comparison of Ransomware F1-scores Using the Integrated Feature Dataset.

In the integrated feature dataset, the accuracy trend remains consistent with that observed in the raw feature dataset. AdaBoost, despite showing some improvements as illustrated in Figure *7*, still lags behind, with accuracy values ranging from 0.93 to 0.95. The highest accuracy for AdaBoost is 0.95, achieved in Experiment I (Table *5*). Random Forest again demonstrate superior achieving an accuracy of 0.96 to 0.98. The highest accuracy for Random Forest in the integrated feature dataset is recorded in Experiment I and Experiment 2 at 0.98, as shown in Table *5* and Table *7*, making it the most effective model for integrated features.



	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Random Forest	0.98	0.98	0.96	0.96
XGBoost	0.97	0.97	0.96	0.96
Adaboost	0.95	0.94	0.94	0.93

Figure 7. Comparison of Accuracy Using the Raw Feature Dataset.

The application of dimension reduction in Experiments III and IV affects the algorithms differently. While dimension reduction can simplify models, reduce computational costs, and mitigate overfitting, it may also result in the loss of important information, potentially degrading model performance. For AdaBoost, the reduced feature set does not sufficiently address its inherent limitations, particularly with the ransomware class, as seen in Table 8 and Table 10. Random Forest and XGBoost show slight variability in F1-scores, suggesting trade-offs between model simplicity and classification efficacy. Random Forest and XGBoost generally maintain high performance, but the slight drop in F1-scores for the ransomware class indicates that some informative features might have been lost during reduction. The primary drawback of dimension reduction is the potential loss of crucial features that contribute to accurate classification, particularly for complex classes like ransomware. The trade-off lies in balancing the benefits of reduced complexity and computational efficiency against the need for maintaining high classification accuracy. In these experiments, it appears that the models can handle the reduced feature space reasonably well, but careful tuning and validation are essential to optimize this balance effectively.

Table 12. Comparative Performance of Various Machine Learning Techniques in Malware Detection.

Work	Classification Method	Accuracy
Hwang et al. [6]	Deep Neural Network (DNN)	94%
Smmarwar et al. [7]	Random Forest, Decision Tree, SVM	91.32% (RF), 91.8% (DT), 82.33% (SVM)
Jeon & Moon [8]	Convolutional Recurrent Neural Network (CRNN)	96.20%
Matin & Rahardjo [9]	SVM, Decision Tree	90%
Rezaei & Hamze [10]	Random Forest	95.59%
Maleki et al. [11]	Decision Tree	98.26%
Penmatsa et al. [12]	Ant Colony Optimization (ACO)	97.15% (Integrated), 92.8% (Raw)
Manavi & Hamzeh [13]	Convolutional Neural Networks (CNN)	93.33%
Rezaei et al. [14]	Deep Learning	92.41%, 97.75%
Azeez et al. [15]	Ensemble Learning with PCA	99.24% (AdaBoost), 98.06% (RF)
Proposed Method	Adaboost, Random Forest, XGBoost and Dimension Reduction (PCA and LDA)	95%(Adaboost), 98% (RF), 97% (XGBoost)

In the context of comparing our results with existing literature, the highest accuracy achieved in this study was obtained using Random Forest with an integrated feature dataset, reaching an impressive 98.0% accuracy, as highlighted in Table 5. This performance is on par with, and in some cases exceeds, the accuracies reported in prior works. Despite this, the robustness and consistency of Random Forest across different feature sets in our experiments suggest its suitability for complex classification tasks, especially in scenarios where high accuracy and reliable detection of complex malware classes like ransomware are critical.

4. CONCLUSION

The implementation of the Electronic-Based Government System (SPBE) has significantly advanced public administration in Indonesia, enhancing efficiency, transparency, and accessibility in infrastructure and housing services. However, this digital transformation has been accompanied by a rise in cyber incidents, notably malware attacks. This research addresses the critical need for effective malware detection methods by evaluating the performance of various machine learning classifiers on malware datasets, including data collected from a honeypot at a specific institution and the MalwareBazaar dataset. This research addressed the need for effective malware detection methods by evaluating the performance of machine learning classifiers—

Random Forest, XGBoost, and AdaBoost—using both raw and integrated feature datasets. The analysis incorporated dimension reduction techniques, specifically PCA and LDA, to improve classification efficiency.

The evaluation of raw feature datasets across four experiments indicated that Random Forest and XGBoost consistently outperformed AdaBoost, particularly in classifying ransomware. While AdaBoost showed high recall and F1-scores for benign and trojan classes, its performance on ransomware was notably poor. Conversely, Random Forest and XGBoost demonstrated robust performance, with high precision, recall, and F1-scores across all malware classes, particularly excelling in the ransomware class. For integrated feature datasets, the trend of AdaBoost underperforming persisted, although slight improvements were observed. Random Forest and XGBoost maintained their superior performance, with Random Forest achieving the highest accuracy and robust classification metrics across all classes. This underscores the effectiveness of these models in leveraging integrated features for improved malware detection.

The application of dimension reduction techniques revealed a balance between model simplicity and classification accuracy. While dimension reduction simplified models and reduced computational costs, it sometimes led to the loss of critical features, affecting the classification of complex malware types such as ransomware. Nonetheless, Random Forest and XGBoost managed to maintain high performance, demonstrating their robustness even with reduced feature sets.

In conclusion, the research highlights the complexity of developing efficient malware detection methods and emphasizes the importance of selecting robust machine learning models. Random Forest and XGBoost, particularly when combined with integrated features, offer a promising approach for malware classification, delivering high accuracy and consistent performance across various malware types. Future research should focus on optimizing these models further and exploring additional feature extraction and integration techniques to enhance malware detection capabilities in the ever-evolving landscape of cyber threats.

ACKNOWLEDGMENTS

The authors thank to Bina Nusantara University for supporting this research.

REFERENCES

[1] Badan Siber dan Sandi Negara, “Laporan Tahunan Honeynet Project Tahun 2022,” 2023.

[2] Y. Alosefer, “Analysing web-based malware behaviour through client honeypots,” Cardiff University, 2012.

[3] M. S. Yousaf, M. H. Durad, and M. Ismail, “Implementation of portable executable file analysis framework (PEFAF),” in *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, IEEE, 2019, pp. 671–675.

[4] L. Gao, J. Song, X. Liu, J. Shao, J. Liu, and J. Shao, “Learning in high-dimensional multimedia data: the state of the art,” *Multimed Syst*, vol. 23, no. 3, pp. 303–313, 2017, doi: 10.1007/s00530-015-0494-1.

[5] S. Ayesha, M. K. Hanif, and R. Talib, “Overview and comparative study of dimensionality reduction techniques for high dimensional data,” *Information Fusion*, vol. 59, pp. 44–58, 2020.

[6] C. Hwang, J. Hwang, J. Kwak, and T. Lee, “Platform-independent malware analysis applicable to windows and linux environments,” *Electronics (Switzerland)*, vol. 9, no. 5, May 2020, doi: 10.3390/electronics9050793.

[7] S. K. Smmarwar, G. P. Gupta, and S. Kumar, “A hybrid feature selection approach-based Android malware detection framework using machine learning techniques,” in *Cyber Security, Privacy and Networking: Proceedings of ICSPN 2021*, Springer, 2022, pp. 347–356.

[8] S. Jeon and J. Moon, “Malware-Detection Method with a Convolutional Recurrent Neural Network Using Opcode Sequences,” *Inf Sci (N Y)*, vol. 535, pp. 1–15, Oct. 2020, doi: 10.1016/j.ins.2020.05.026.

[9] I. M. M. Matin and B. Rahardjo, “Malware detection using honeypot and machine learning,” in *2019 7th international conference on cyber and IT service management (CITSM)*, IEEE, 2019, pp. 1–4.

[10] T. Rezaei and A. Hamze, “An efficient approach for malware detection using PE header specifications,” in *2020 6th International Conference on Web Research (ICWR)*, IEEE, 2020, pp. 234–239.

[11] N. Maleki, M. Bateni, and H. Rastegari, “An improved method for packed malware detection using PE header and section table information,” *International Journal of Computer Network and Information Security*, vol. 11, no. 9, p. 9, 2019.

[12] R. K. V. Penmatsa, A. Kalidindi, and S. K. R. Mallidi, “Feature reduction and optimization of malware detection system using ant colony optimization and rough sets,” *International Journal of Information Security and Privacy (IJISP)*, vol. 14, no. 3, pp. 95–114, 2020.

[13] F. Manavi and A. Hamzeh, “A new method for ransomware detection based on PE header using convolutional neural networks,” in *2020 17th International ISC Conference on Information Security and Cryptology (ISCISC)*, IEEE, 2020, pp. 82–87.

[14] T. Rezaei, F. Manavi, and A. Hamzeh, “A PE header-based method for malware detection using clustering and deep embedding techniques,” *Journal of Information Security and Applications*, vol. 60, p. 102876, 2021.

[15] N. A. Azeez, O. E. Odufuwa, S. Misra, J. Oluranti, and R. Damaševičius, “Windows PE malware detection using ensemble learning,” in *Informatics*, MDPI, 2021, p. 10.

[16] G. T. Reddy *et al.*, "Analysis of dimensionality reduction techniques on big data," *Ieee Access*, vol. 8, pp. 54776–54788, 2020.
[17] Ajit Kumar, "ClaMP (Classification of Malware with PE headers)," 2020, *Mendeley Data, V1*. doi: 10.17632/xvyv59vwvz.1.
[18] A. Kumar, K. S. Kuppusamy, and G. Aghila, "A learning model to detect maliciousness of portable executable using integrated feature set," *Journal of King Saud University-Computer and Information Sciences*, vol. 31, no. 2, pp. 252–265, 2019.
[19] R. E. Schapire, "Explaining adaboost," in *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*, Springer, 2013, pp. 37–52.
[20] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

BIOGRAPHY OF AUTHORS

Arif Harsa Pradipta earned his Bachelor of Computer Science degree from the Faculty of Computer Science at the University of Indonesia. He is currently pursuing a Master of Computer Science at Bina Nusantara University. Arif has a keen interest in cybersecurity and machine learning. With professional experience as a backend developer, he brings both practical and technical perspectives to his work and research. For any academic or professional inquiries related to his areas of expertise, Arif can be reached at arif.pradipta@binus.ac.id.

Lili Ayu Wulandhari is a computer engineering lecturer at Bina Nusantara University. She specializes in machine learning, data science, and text mining. She successfully obtained her master's and Ph.D. degrees in computer science from the Malaysian University of Technology. Lili possesses a decade of experience as an academic as well as expertise as a professional data scientist. She has successfully applied AI and machine learning techniques in several domains and data types, including text, image, and financial data. She is actively involved as both the chairman and a member of the campus's internal and national grants. In the past five years, the research has resulted in over 10 articles that have been distributed in respected national and international journals.