Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.059116

ARTICLE

Reverse Analysis Method and Process for Improving Malware Detection Based on XAI Model

Ki-Pyoung Ma[1], **Dong-Ju Ryu**[2] and **Sang-Joon Lee**[3,*]

[1]Security Team, Doosan Corp., Seoul, 04563, Republic of Korea

[2]Btress Corp., Hanam-Si, Gyeonggi-Do, 12902, Republic of Korea

[3]School of Business Administration, Chonnam National University, Gwangju, 61186, Republic of Korea

*Corresponding Author: Sang-Joon Lee. Email: s-lee@chonnam.ac.kr

Received: 28 September 2024 Accepted: 14 November 2024 Published: 19 December 2024

ABSTRACT

With the advancements in artificial intelligence (AI) technology, attackers are increasingly using sophisticated techniques, including ChatGPT. Endpoint Detection & Response (EDR) is a system that detects and responds to strange activities or security threats occurring on computers or endpoint devices within an organization. Unlike traditional antivirus software, EDR is more about responding to a threat after it has already occurred than blocking it. This study aims to overcome challenges in security control, such as increased log size, emerging security threats, and technical demands faced by control staff. Previous studies have focused on AI detection models, emphasizing detection rates and model performance. However, the underlying reasons behind the detection results were often insufficiently understood, leading to varying outcomes based on the learning model. Additionally, the presence of both structured or unstructured logs, the growth in new security threats, and increasing technical disparities among control staff members pose further challenges for effective security control. This study proposed to improve the problems of the existing EDR system and overcome the limitations of security control. This study analyzed data during the preprocessing stage to identify potential threat factors that influence the detection process and its outcomes. Additionally, eleven commonly-used machine learning (ML) models for malware detection in XAI were tested, with the five models showing the highest performance selected for further analysis. Explainable AI (XAI) techniques are employed to assess the impact of preprocessing on the learning process outcomes. To ensure objectivity and versatility in the analysis, five widely recognized datasets were used. Additionally, eleven commonly-used machine learning models for malware detection in XAI were tested with the five models showing the highest performance selected for further analysis. The results indicate that eXtreme Gradient Boosting (XGBoost) model outperformed others. Moreover, the study conducts an in-depth analysis of the preprocessing phase, tracing backward from the detection result to infer potential threats and classify the primary variables influencing the model's prediction. This analysis includes the application of SHapley Additive exPlanations (SHAP), an XAI result, which provides insight into the influence of specific features on detection outcomes, and suggests potential breaches by identifying common parameters in malware through file backtracking and providing weights. This study also proposed a counter-detection analysis process to overcome the limitations of existing Deep Learning outcomes, understand the decision-making process of AI, and enhance reliability. These contributions are expected to significantly enhance EDR systems and address existing limitations in security control.

KEYWORDS
Endpoint detection & response (EDR); explainable AI(XAI); SHapley Additive exPlanations (SHAP); reverse XAI; machine learning (ML)

1 Introduction

Computers and the Internet have become integral to our daily lives with the advancement of information technology (IT). However, with the increased use of these technologies, cyberattacks have grown in frequency and sophistication [1]. These attacks pose significant threats by disrupting networks and stealing sensitive information, including medical records, trade secrets, and financial data [2].

The Fourth Industrial Revolution, driven by technologies such as the Internet of Things (IoT), artificial intelligence (AI), and big data, is transforming industries into hyper-connected ecosystems [3]. IoT allows devices to exchange data, while AI is utilized for perception, reasoning, learning, and natural language processing [4]. These technologies are applied to various industries, enhancing convenience and creating new value [5]. For instance, in recent years, AI has played a crucial role in the field of prediction, particularly through machine learning (ML) and deep learning (DL) methodologies. Consequently, research is actively conducted to improve the accuracy of AI-based prediction. Studies employing eXtreme Gradient Boosting (XGBoost) [6], Long Short-Term Memory (LSTM) [7], and hybrid structures like Convolutional Neural Network-LSTM (CNN-LSTM) have shown that these approaches significantly enhance the accuracy of prediction and are regarded as effective approach for forecasting demand [8].

However, these technological advancements have also led to new security threats, including hacking and vulnerabilities that exploit these very technologies. To address these threats, AI-driven models, especially in the context of IoT-integrated environments, are increasingly being employed for security [9]. Despite these measures, cyberattacks continue to evolve, necessitating the constant enhancement of security tools.

Endpoint Detection and Response (EDR) systems are a key component of cybersecurity, designed to detect and mitigate security incidents at endpoints. Nevertheless, current EDR solutions primarily react to incidents post-attack, limiting their effectiveness in proactive threat prevention. This study emphasizes the need to enhance EDR solutions to support real-time detection and prediction by leveraging AI-based methods, particularly Explainable AI (XAI) and its application in malware detection.

This study aims to improve malware detection through a reverse analysis process, focusing on extracting critical parameters from the detection process, measuring their influence, and validating these results on real-world networks. The XGBoost model was selected for its superior performance, while XAI, specifically SHapley Additive exPlanations (SHAP), was employed to classify and interpret malware parameters. XAI was introduced because it can explain how individual features influence malware detection, ensuring that EDR systems remain transparent and actionable.

Section 1 provides an overview of the research background and its significance. Section 2 presents an analysis of related studies, focusing on AI detection models currently under active research. In Section 3, the testing process for the proposed model is described, along with the criteria for selecting the most commonly used models. The performance of XAI learning models applicable in existing EDR systems is also compared and analyzed. Furthermore, this section explains the selection process for the algorithm employed in the proposed backtracking analysis method. Section 4 examines the SHAP prediction results from the XAI model, detailing the characteristics of the inference and backtracking analysis process. Lastly, Sections 4 and 5 discuss the hypothesis established to verify the proposed experiment. By backtracking the target files and identifying common parameters in malware, potential security breaches are anticipated, with weights assigned to each parameter. Consequently, the proposed security control process is refined, allowing for predictive and preventive actions by tracking results backward using XAI.

2 Related Research

2.1 Traditional Malware Detection Methods

In earlier days, malware was predominantly designed for general computers, especially targeting the Windows operating system. Until around 2010, most malware targeted these systems due to the lack of alternative devices. However, the proliferation of mobile devices, such as smartphones, tablets, and PDAs, along with the growth of cloud computing and IoT environments, has expanded the range of platforms susceptible to malware. Consequently, malware detection and classification methods now need to operate effectively across diverse devices, platforms, and operating environments.

Fig. 1 illustrates a typical flow of malware detection processes, showing how classification algorithms and data mining techniques are applied to extract meaningful features from raw data for malware classification [3]. These selected features are trained with ML algorithms or rule-based learning techniques to separate malware from benign. Further classification can be performed to increase awareness of the malware's content and purpose by detecting the type and class of malware to which it belongs [5].

2.2 AI-Based Malware Detection Model Analysis

Recent advances in AI-based malware detection methods show promise, particularly against traditional malware types. However, these methods often fail to detect zero-day attacks, requiring continuous model enhancement [10]. Previous studies use malware detection approaches of new technologies such as behavioral, heuristic, and model confirmation-based detection and DL, cloud, mobile device, and IoT-based detection. Table 1 summarizes several AI-based detection models, listing the features they use, their algorithms, and performance results.

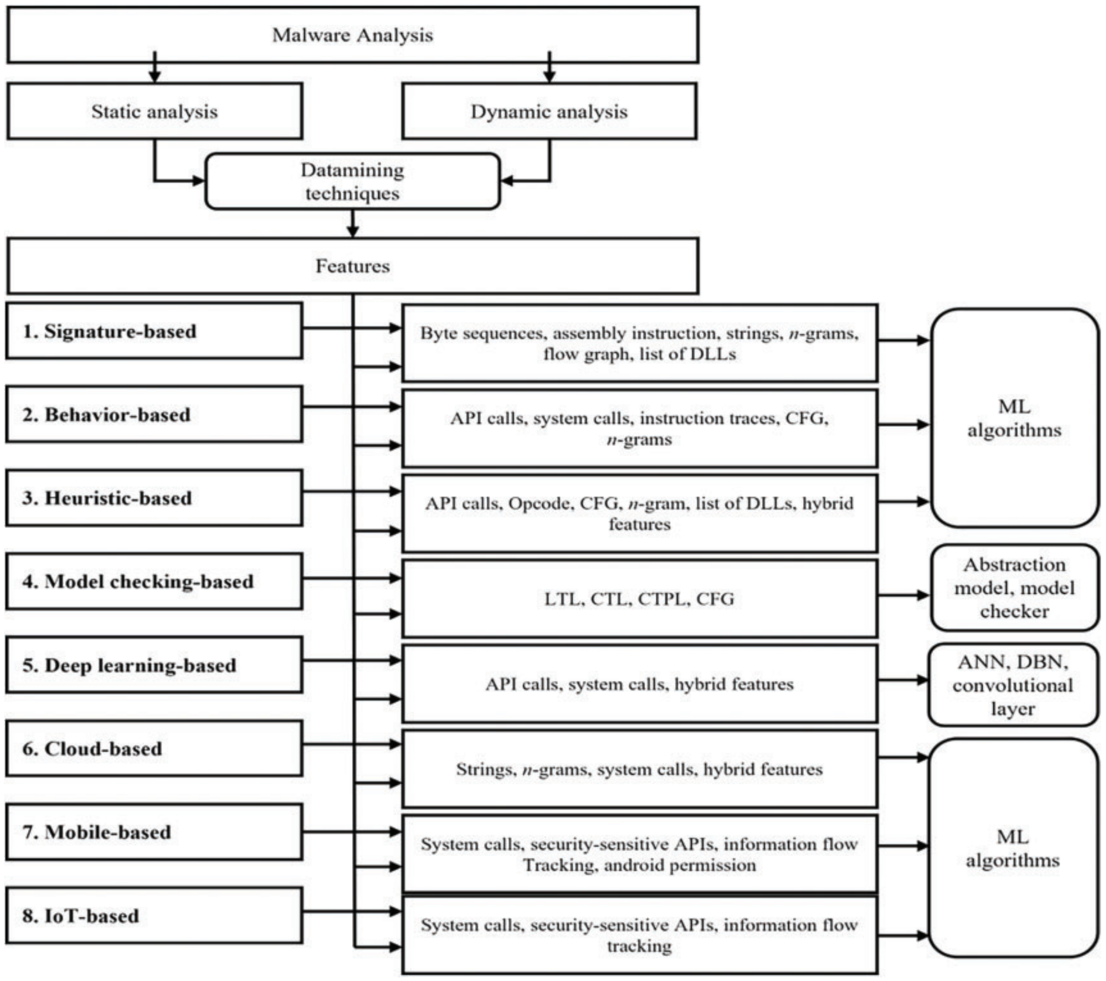

Malware Analysis
Static analysis
Dynamic analysis
Datamining techniques
Features
1. Signature-based
2. Behavior-based
3. Heuristic-based
4. Model checking-based
5. Deep learning-based
6. Cloud-based
7. Mobile-based
8. IoT-based
Byte sequences, assembly instruction, strings, n-grams, flow graph, list of DLLs
API calls, system calls, instruction traces, CFG, n-grams
API calls, Opcode, CFG, n-gram, list of DLLs, hybrid features
LTL, CTL, CTPL, CFG
API calls, system calls, hybrid features
Strings, n-grams, system calls, hybrid features
System calls, security-sensitive APIs, information flow Tracking, android permission
System calls, security-sensitive APIs, information flow tracking
ML algorithms
Abstraction model, model checker
ANN, DBN, convolutional layer
ML algorithms

Figure 1: Flow diagram of malware detection approaches and capabilities

Table 1: Summary of relevant measures for malware detection approaches

Ref.	Features in dynamic analysis	Algorithms	Dataset	Performance
[9]	File system and registry in windows. Permission monitoring in Android	–	Ransomware of 25 families	Not mentioned
[10]	File system, I/O monitoring	–	715 ransomware	DR 96.7%
[11]	File system, Access patterns, and I/O Data buffer entropy	–	148,223 general malware	DR∗ 96.3%
[12]	API calls, Registry key operations, File/Directory system	NB and SVM	582 ransomware of 11 families, and 942 goodware	ROC: 0.995

(Continued)

Table 1 (continued)

Ref.	Features in dynamic analysis	Algorithms	Dataset	Performance
[13]	HTTP traffic characteristics	–	750 CryptoWall 4.0 ransomware traffic–750 Locky ransomware traffic	DR 97%–98%
[14]	API calls	SVM	588 logs, 312 goodware, and 276 ransomware logs	CA 97.48%
[15]	Entropy analysis	–	Not mentioned	CA 92%
[16]	API calls	RF, SVM, SL, and NB	168 ransomware	CA 98.2%
[17]	Command and control (C&C) server	RF	265 ransomware-related flows	CA 87%
[18]	Portable Executable (PE) File	–	450 ransomware	CA 70%
[19]	Network traffic	DT (J48 classifier)	210 ransomware, 264 benign	F1 96.8%
[20]	Ransomware opcodes (Machine Language Instructions)	DT, RF, kNN, NB, GBDT	1787 ransomware	CA 99.3%
[21]	API calls	SVM, DT, RF, GBDT	360 ransomware, 532 general malware, and 460 benign software	CA 96.1%
[22]	API function calls, counts of the behavioral features, and counts of the memory features	–	1000 ransomware, 1000 benign software	DR 90%
[23]	Selects key features using Multi-Objective Grey Wolf Optimization and Binary Cuckoo Search algorithms	NB, RF, and SMO	582 ransomware, and 942 goodware	CA 79.3%–82.67%
[24]	Master File Table (MTF) and I/O Request Packets (IRP)	–	Logs with 2000 user activity and 2000 ransomware activity	CA 97.4%
[25]	I/O operation, LBA, and Entropy	RF, SVM, kNN, CNN	7 ransomware families	F1 0.57–0.99
[26,27]	Semantic information from logs	Bi-LSTM	Logs	CA 96.5%–99.7%

Note: ∗DR: Detection Rate, CA: Classification Accuracy.

3 How to Improve XAI-Based Malware Detection

This section provides a brief background on ML algorithms. Additionally, it describes the selected dynamic features that form the input vectors of the feature extraction tool, test setup, and ML algorithm [28].

3.1 XAI Model

Explainable AI (XAI) is critical for enhancing the transparency of AI systems, especially in high-stakes domains like cybersecurity, medicine, and defense. While traditional AI models (often seen as "black boxes") achieve high accuracy, their decision-making process is not interpretable [29]. XAI improves trust by allowing human users to understand how and why AI systems make certain predictions.

In cybersecurity, model interpretability is vital. Although deep learning (DL) models have proven effective for malware detection, their complexity and opacity pose challenges when it comes to understanding their decisions. In response, this study adopts XAI techniques, specifically SHAP, to balance the trade-off between predictive performance and model transparency [14].

3.2 Need for Preprocessing in EDR Systems

Preprocessing plays a crucial role in malware detection by filtering and refining raw data before it is processed by detection algorithms. For example, the quality of data may be improved through techniques such as data normalization, feature selection, and data augmentation, and the performance of the model may be maximized through this. It is possible to increase the efficiency of model learning by selecting important variables in the feature selection process and removing unnecessary variables. This step significantly improves detection accuracy and efficiency by eliminating irrelevant features, reducing noise, and enhancing the signal-to-noise ratio in the dataset. In the current study, dynamic API calls, access patterns, and other system-level features were used as input to XAI-based models. These preprocessing techniques were essential in boosting both detection performance and the interpretability of results.

In the field of information security, the explainability of a model in ML is inversely proportional to its predictive performance (e.g., accuracy). DL models, the most powerful and complex AI algorithm, are also the most difficult to explain. The role of XAI is to increase explainability while maintaining high performance. In this study, we propose the ability to detect and predict malware using XAI so that the results remain actionable through ML. In the past, it has been identified as a limitation that AI only informs the result when supporting decision-making and cannot logically explain on what grounds it reached this decision [29]. DL, a representative field of AI, enables judgment by extracting implicit knowledge rather than explicit knowledge from repetitive large-volume data. Although this method can improve the efficiency of risk assessment for existing malware or malicious code, it is difficult to say that it has secured fairness and accuracy [30]. Trust in the system is contingent on how the conclusions are drawn in an algorithm such as a black box. XAI is attracting attention to supplement the limitations of artificial intelligence.

3.3 Selection of XAI-Based Malware Detection Algorithm

ML is at the heart of AI and is one of the fastest-growing fields of AI. ML theories and methods are widely used to solve complex problems in engineering applications and science. ML is typically classified into supervised, unsupervised, and reinforcement learning. This study predicts risk factors using the model used in previous studies among ML models. The main parameter settings were determined using a grid search. Of the 11 ML classification models (kNN, Decision Tree, SVM, SGD, Random Forest (RF), Neural Network, Naive Bayes, XGBoost, CatBoost, AdaBoost GradientBoosting), RF, XGBoost, Catboost, AdaBoost, and NN algorithms were selected for this study. These algorithms are used for behavior-based detection and some other detection approaches. Although each algorithm has strengths and weaknesses, it cannot be concluded that one algorithm is more efficient than another. However, some algorithms may outperform others regarding data distribution, number of features, and dependencies between attributes [31].

Various machine learning algorithms were evaluated in this study, including kNN, Decision Trees, Random Forest, Support Vector Machines (SVM), and Neural Networks. After careful comparison, the XGBoost model was selected for its superior accuracy and efficiency in handling complex data sets [15]. Table 2 presents the model configurations used in this study.

Table 2: Model and parameters

Model	Parameters
XGBoost	Basic properties = number of trees = 500; learning rate = 0.300; replicable training; regularization Lambda = 1; max_depth = 6
RF	Number of trees = 10; number of attributes = 7; max_depth = 8
SGD	Loss function classification Hinge; ε = 0.10 regularization = ridge (L2) strength (α) = 0.00001; optimization with learning rate = constant initial learning rate = 0.0100, inverse scaling exponent = 0.2500 (default); number of iterations = 1000; tolerance (stopping criterion) = 0.0010
kNN	Number of neighbors = 5; Metric = Euclidean; Weight = Uniform
Tree	Min. number of instances in leaves 2, maximal tree depth = 100
NN	Neurons in hidden layers = 100, Activation Relu, Solver Adam Regularization α = 0.0002, Maximal number of iterations = 200
Gradient boosting	Basic properties 1) number of trees = 100, learning rate = 0.100, replicable training; Growth Control 1) max_depth = 3; training instances = 1.00
CatBoost	Basic Properties 1) number of trees = 100; learning rate = 0.300; regularization Lambda = 3; max_depth = 6
AdaBoost	1) Base estimators = Tree; number of estimators = 50; learning rate = 0.8000; Fixed seed for random generator = 10,000, 2) Boosting method = Classification algorithm = SAMME. R
SVM	C = 1.0
Naive Bayes	–

3.3.1 *Algorithm Selection Performance Comparison for Backtracking Analysis*

For the performance evaluation of the ML model, metrics such as area under the curve (AUC) and classification accuracy (CA) commonly used in previous studies were used. The classification performance evaluation metric uses a confusion matrix. In this study, five balanced data sets (e.g., malicious/benign) and five imbalanced data sets (e.g., malware families) are all addressed to the classification problem. AUC and CA scores were used. The performance evaluation was compared using the curve and SHAP value for visualization. In this paper, the distribution map and numerical values of the existing XAI's features are expressed as a matrix to analyze security threat factors in a method close to approximate values. It suggests a process to determine the correlation of data. Therefore, all five data sets used as the proposed performance evaluation criterion had common detection characteristics, and variables representing malware detection were distinguished.

3.3.2 Dataset

This study conducts classification prediction modeling using a total of five data sets. Data sets #1 and #2 are IEEE malware data. Data set #3 is NSL-KDD, data set #4 is Drebin, and data set #5 is the Microsoft malware classification data set. The configuration of the data sets is summarized in Table 3.

Table 3: Data set configuration

Data set	Type				Model									
	Type	Instances	Total instance	Ratio	AdaBoost		CatBoost		XGBoost		RF		SVM	
					AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Data set #1 (IEEE)	Train Test	80,000 20,000	100,000	8:2	1	1	1	1	1	1	1	1	1	1
Data set #2 (IEEE)	Train Test	35,101 8775	43,876	8:2	0.969	0.97	0.996	0.976	0.996	0.976	0.998	0.97	0.987	0.951
Data set #3 (NSL-KDD)	Train Test	536 134	670	8:2	0.969	0.97	0.996	0.976	0.996	0.976	0.998	0.97	0.987	0.951
Data set #4 (Drebin)	Train Test	29,525 7381	36,906	8:2	0.998	0.993	1	0.995	1	0.997	1	0.995	0.42	0.707
Data set #5 (Microsoft)	Train Test	68,962 17,241	86,203	8:2	0.896	0.816	0.676	0.626	0.736	0.671	0.895	0.809	-	-

The division of data to be used for ML training and prediction classification of the data set in this study reveals that data set #1 is a malware detection data set (100,000 instances), with a total of 35 features (variables). The train and test data are split 8:2, with 80,000 instances used for training and 20,000 instances for verification.

Based on the data information of data set #2, dynamic_api_call_sequence_per_malware_ 100_0_306, the instance is 43,876, and features are composed of 100 numeric types. Based on the feature descriptive statistics, the mean of the variables is distributed as t_36 124.14 and t_1 211.48. The median ranges from t_0 82 to t_1 24. The spread ranges from 0.28 for t_1 to 0.68 for t_16. The minimum values range from 0 to 2. The maximum value is 306.

Dataset #3 extracted a total of 670 instances. 42 were selected as features, 12 were categorical features, and 29 were composed of numerical variables.

Dataset #4 is the Drebin data set with 17,453 (47.29%) positive samples and 19,453 (52.71%) malware sample. It consisted of a total of 36,906 instances. It consisted of 138 variables.

Dataset #5 consists of the Microsoft BIG 2015 data set [5] with a total of 86,203 instances, with positive samples of 43,125 (50.03%) and malware sample of 43,078 (49.97%).

A supervised learning-based ML model algorithm was applied to automatically recognize malicious code through malware classification data and classify. This study used 11 ML classification models (kNN, Decision Tree, Support Vector Machine (SVM), Stochastic Gradient Descent (SGD), RF, Neural Network, Naive Bayes, XGBoost, CatBoost, AdaBoost GradientBoosting). Models with low effectiveness were excluded through learning and verification processes using the given data of this study. Furthermore, although boosting and bagging stacking were conducted, special AUC and CA were not superior, so they were excluded. The H/W components used in the experiment were 12th generation Intel(R) Core(TM) i7-1260P 2.10 GHz and X64 based processor 32.0 GB for memory, and the operating system was Win11 Pro Edition.

4 Malware Traceback Analysis Method for the Proposed XAI-Applied Security Control System

This study extracted parameters that can be reversely traced using the existing XAI concept. These processes were used to identify items that affect the target parameter in the experiment. Detailed experimental results are described in Section 4.1. The data features were analyzed using the XGBoost algorithm as an XAI target. The analyzed data was retrained and applied to XAI after the ML process.

Existing XAI is when DL identifies arbitrary data as malware, using another AI tool to monitor and determine when it is classified as malware [25]. The proposed model is added to the existing XAI DL process, and you can check which features are judged to be malware in Fig. 2. As a result of experiments to confirm the hypothesis, it would be possible to attribute and analyze malware by applying reverse XAI through the XGBoost algorithm in the dataset.

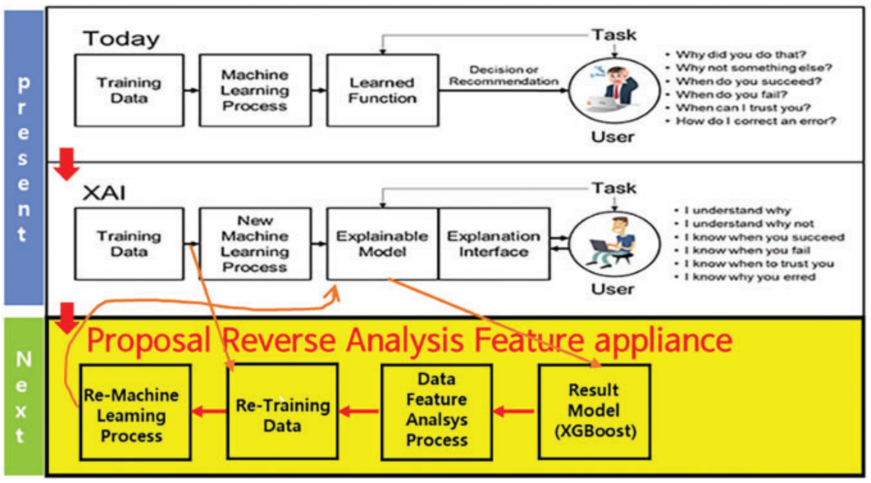


Figure 2: XAI analysis methods added to the model

Fig. 3 illustrates the basic process of existing XAI. The dotted box in this Fig. 3 at the bottom represents the analysis process. Several facts were reconfirmed through the proposed XAI model. First, XAI is useful when results are inaccurate. Second, the measure of the explanatory effect may change over time. Third, it helps measure and analyze the detection models of users and XAI systems. However, this XAI described the detection result and its contents, unlike the existing AI learning model that expressed only the result value [29]. However, this study intended to identify which parameters influenced the technology being improved through experiments. If this effect is known, it is possible to observe in detail the factors that affect the labeling or feature processes in advance.

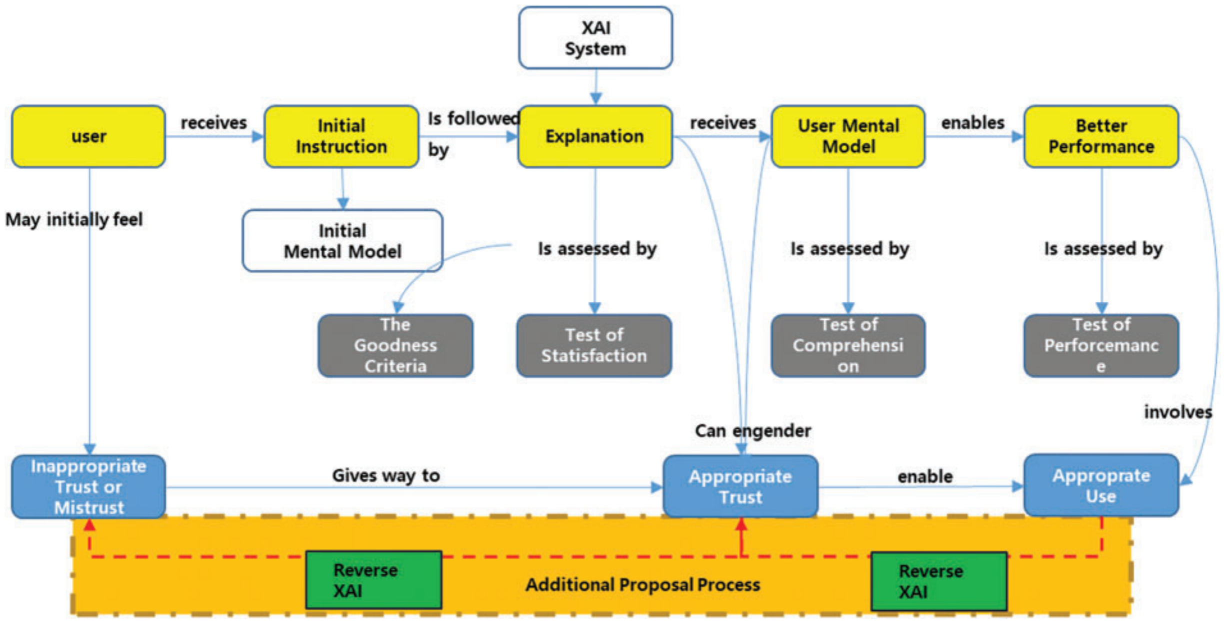


Figure 3: Reverse analysis XAI proposed model

Furthermore, it is possible to analyze the characteristics of malicious files by classifying files to be managed intensively through impact. Based on these improvements, the characteristics of malware and the threat to its files could be predicted and removed in advance. Finally, through various experiments, the XAI structure was changed to the reverse direction, and a new method was proposed to extract parameters that can represent malware. Consequently, the main variables for predicting malware detection could be identified.

4.1 XAI-Applied Malware Analysis-Detection Model Measurement Results

SHAP (SHapley Additive exPlanations) values provide the contribution of each feature to the prediction of the selected class by taking the training model and reference data for the inputs and using the provided data [32]. SHAP values are a technique used in ML to interpret a model's predictions and understand the importance of each input feature contributing to those predictions. It provides a way to attribute the predictions of a particular instance to individual features. It aims to fairly distribute "credits" or "contributions" among the input features of the model. In the context of ML, SHAP values quantify the impact of each feature by measuring the change in prediction when a feature value is included or excluded. By calculating a SHAP value for each feature, you can understand how much each feature contributes to the model output for a particular instance.

A positive SHAP value indicates that the feature contributes positively to the prediction, while a negative value indicates the opposite. The size of the SHAP value indicates the strength of the contribution. SHAP values can describe individual predictions, understand the importance of features at a global level, and evaluate the behavior of the entire model. It enables the interpretation of complex models such as ensemble methods, deep neural networks, and gradient boosting machines, often considered "black box" models. While SHAP is a powerful interpretability tool, it is computationally expensive and may slow down real-time detection in high-speed environments. This presents a limitation for real-time applications, but we propose optimizations through feature selection to mitigate this issue. In the experiment conducted using the selected model XGBoost, the horizontal axis of each graph represents the impact on the model output, measured as the contribution of each feature to the prediction results. The unit of measure is based on the SHAP value itself, which quantifies the influence of each feature on the outcome. The vertical axis lists the features analyzed for each dataset, showing their contribution distribution.

In Fig. 4, the graphs illustrate the contribution of each feature to the prediction. The red color represents the effect on the prediction result as the feature value increases, whereas the blue color represents the effect as the feature value decreases. For example, in the experimental results for Dataset #5, the higher the value (red) of the variable 'AVProductInstalled', the smaller the effect on the prediction result, while the lower the value (blue) has a greater effect. This indicates that this variable has the greatest effect in XGBoost for this particular dataset.

4.2 Malware Traceback Security Analysis Model Applied with XAI

Fig. 3 has already shown the proposed reverse analysis model, which traces back detection results to identify which parameters contributed to a malware classification. This approach not only enhances detection accuracy but also helps system administrators understand the specific characteristics of detected malware. The reverse analysis process can also provide insights into potential preventive measures, thus addressing concerns about the lack of proactive defense mechanisms, such as Intrusion Prevention Systems (IPS) and advanced threat intelligence.

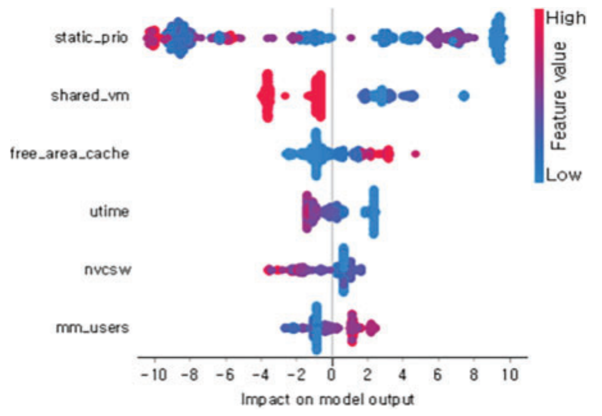
Fig. 5 is the analysis process applying the proposed model in the network for the experiment. It is configured with minimal impact on the existing network. As the experimental conditions of this study, after inputting the same data set and confirming the lowest performance result, five final models were selected, and the experiment was equally conducted within 100 repetitions of epochs in the same period.

In Fig. 5, traditional security equipment and control processes are schematically illustrated. The various analysis results from the security devices on the left of Fig. 5 are analyzed and resolved in the second area on the right. In this process, the question mark and the diamond shape in the middle determine whether an analysis is possible. Then, unique signatures represented by each file are checked and uploaded to the existing detection engine. Undiscovered threat attacks are classified as unknown attacks.

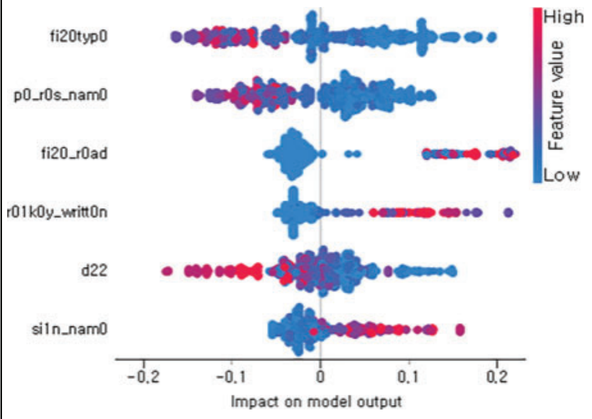
As proposed, by examining the malware detection results and process using the XAI model, various threats can be identified, including viruses, malicious codes, DoS, DDoS, information system damage, scanning, hacking traces, via hacking, information leakage, forgery and falsification, and illegal access attempts. It was classified, and an accident analysis was conducted for each type. However, it is necessary to analyze events for each detection type to detect malware. Optimized detection is possible through reverse XAI claimed in Fig. 3. In traditional security control, signature-based detection and AI engines have similar judgment processes.

In Fig. 5, the analysis process and the detection pattern are finished only when the end is at the bottom. Therefore, the proposed analysis modeling method proposes to perform the analysis process in parallel with the existing method through mirroring. Consequently, the proposed self-detection and the reverse engines will be combined, and only the results will be delivered to the controller to identify the problem from those results. Therefore, the proposed analysis modeling has the advantage of being able to perform the analysis process simultaneously with minimal impact on the existing system and network configuration.

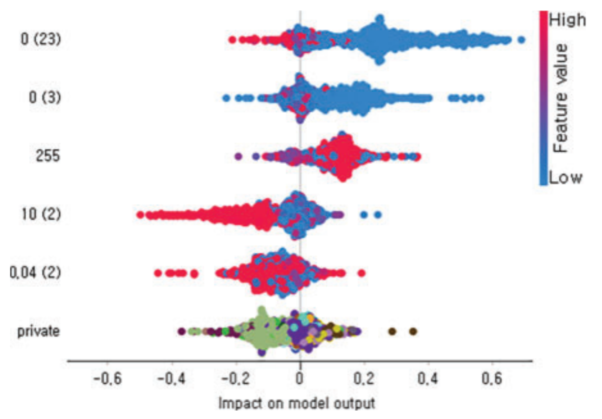
XGBoost data 1 SHAP



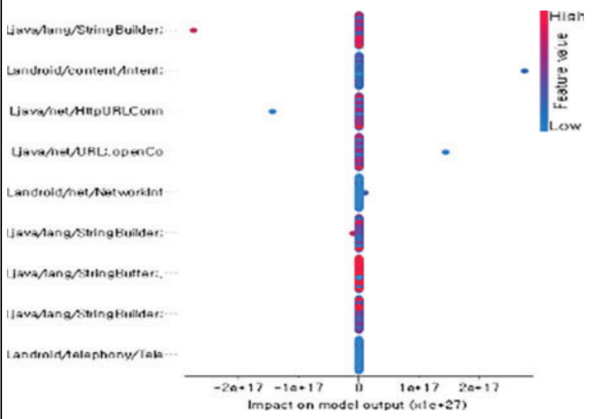
XGBoost data 2 SHAP



XGBoost data 3 SHAP



XGBoost data 4 SHAP



XGBoost data 5 SHAP

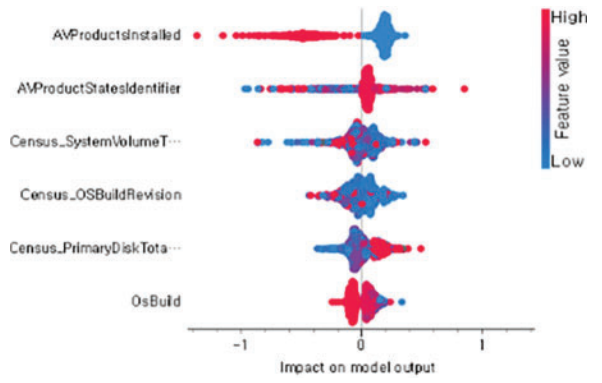


Figure 4: SHAP value of XGBoost by variable in data set

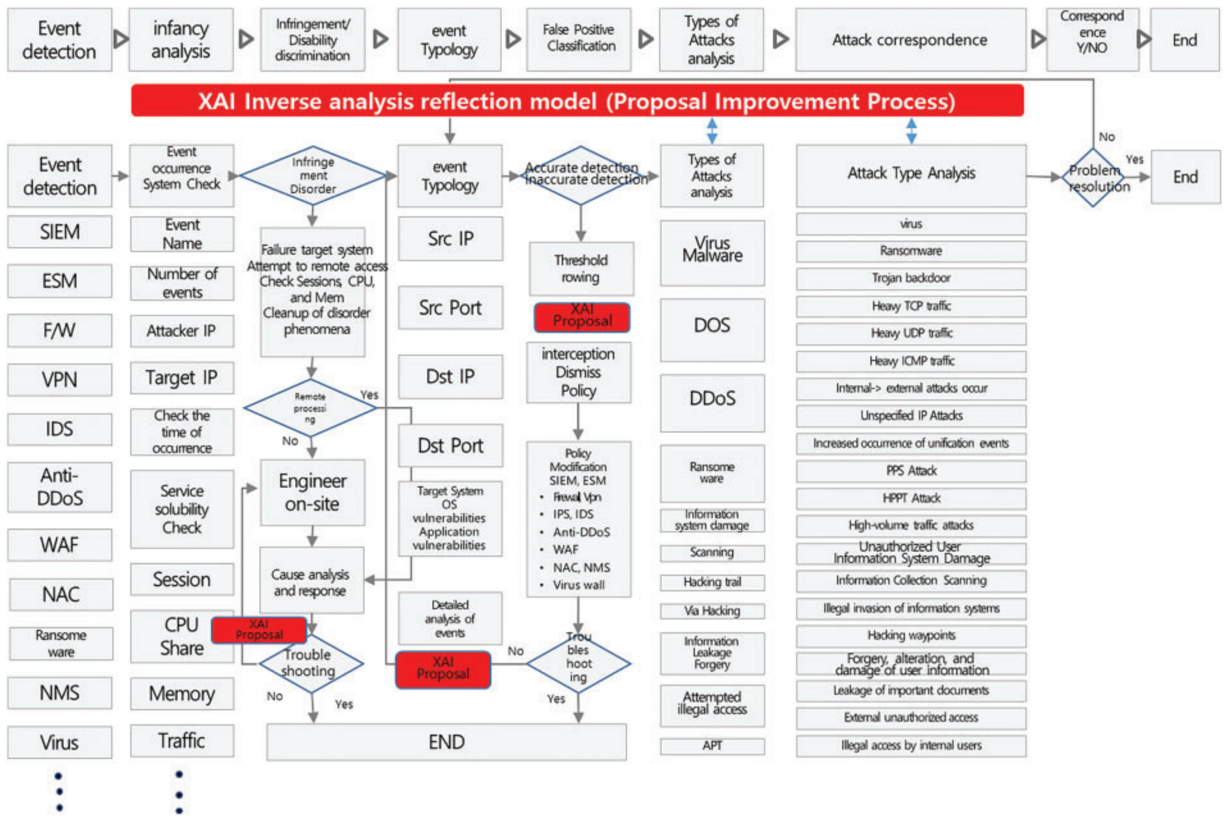
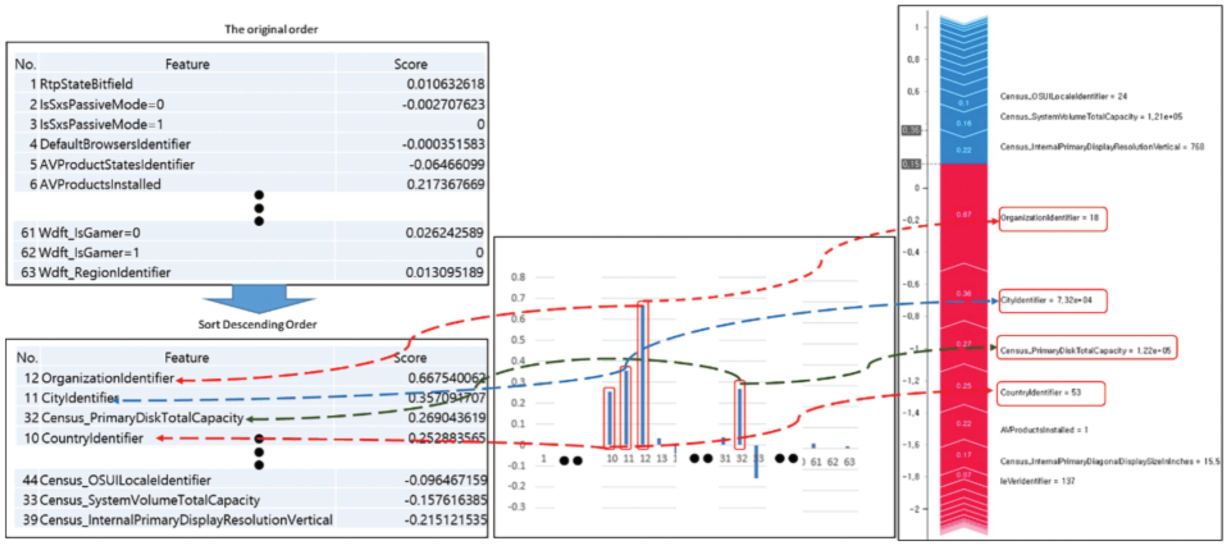

Event detection
infancy analysis
Infringement/ Disability discrimination
event Typology
False Positive Classification
Types of Attacks analysis
Attack correspondence
Correspondence Y/NO
End
XAI Inverse analysis reflection model (Proposal Improvement Process)
Event detection
Event occurrence System Check
Infringement Disorder
event Typology
Accurate detection Inaccurate detection
Types of Attacks analysis
Attack Type Analysis
Problem resolution
No
Yes
End
SIEM
ESM
F/W
VPN
IDS
Anti-DDoS
WAF
NAC
Ransome ware
NMS
Virus
Event Name
Number of events
Attacker IP
Target IP
Check the time of occurrence
Service solubility Check
Session
CPU Share
Memory
Traffic
Failure target system Attempt to remote access Check Sessions, CPU, and Mem Cleanup of disorder phenomena
Remote processing
Yes
No
Engineer on-site
Cause analysis and response
XAI Proposal
Trouble shooting
No
Yes
END
Src IP
Src Port
Dst IP
Dst Port
Target System OS vulnerabilities Application vulnerabilities
Detailed analysis of events
XAI Proposal
Threshold rowing
XAI Proposal
interception Dismiss Policy
Policy Modification SIEM, ESM
Frewal Vpn
IPS, IDS
Anti-DDoS
WAF
NAC, NMS
Virus wall
Troubleshooting
No
Yes
Virus Malware
DOS
DDoS
Ransome ware
Information system damage
Scanning
Hacking trail
Via Hacking
Information Leakage Forgery
Attempted illegal access
APT
virus
Ransomware
Trojan backdoor
Heavy TCP traffic
Heavy UDP traffic
Heavy ICMP traffic
Internal-> external attacks occur
Unspecified IP Attacks
Increased occurrence of unification events
PPS Attack
HPPT Attack
High-volume traffic attacks
Unauthorized User Information System Damage
Information Collection Scanning
Illegal invasion of information systems
Hacking waypoints
Forgery, alteration, and damage of user information
Leakage of important documents
External unauthorized access
Illegal access by internal users

Figure 5: Security analysis modeling using XAI

Fig. 6 is a drawing of the result obtained after sending the actual data using Fig. 5. Fig. 6 is expressed to improve readability because it is not easy to read the SHAP results described above. As depicted in Fig. 6, the part that affects the prediction of the XGBoost model the most among the test data is OrganizationIdentifier, and the value of 0.67 was confirmed. Next is CityIdentifier, with an influence of 0.36. The third is CountryIdentifier. In Fig. 6, 63 parameter values on the left were extracted and used in common. These parameters are values commonly used by malware in the data set and general files. The features and scores at the bottom left are numerical values representing the experimental results. The twelfth parameter, OrganizationIdentifier, was detected the most, and this parameter was predicted to represent a threat. In existing general files, this parameter was not significant and was detected most frequently in malware—the value of this most frequently detected parameter would be meaningful.

Based on verifying the actual data, OrganizationIdentifier significantly influences malware. In Fig. 6, the middle graph expresses the risk and frequency from 0 to 1. The closer to 0, the smaller the impact; the closer to 1, the greater the possibility of malicious files by parameter detection. The bar graph on the right in Fig. 6 represents the cumulative value. Based on the SHAP value of the dataset, OrganizationIdentifier (organization room identification), which has the highest value among variables, has the greatest effect, and CityIdentifier, Census_PrimaryDiskTotalCapacity, and CountryIdentifier values affect malware determination.



No.	Feature	Score
1	RtpStateBitfield	0.010632618
2	IsSxsPassiveMode=0	-0.002707623
3	IsSxsPassiveMode=1	0
4	DefaultBrowsersIdentifier	-0.000351583
5	AVProductStatesIdentifier	-0.06466099
6	AVProductsInstalled	0.217367669
●		
61	Wdft_IsGamer=0	0.026242589
62	Wdft_IsGamer=1	0
63	Wdft_RegionIdentifier	0.013095189

No.	Feature	Score
12	OrganizationIdentifier	0.667540062
11	CityIdentifier	0.357091707
32	Census_PrimaryDiskTotalCapacity	0.269043619
10	CountryIdentifier	0.252883565
●		
44	Census_OSUILocaleIdentifier	-0.096467159
33	Census_SystemVolumeTotalCapacity	-0.157616385
39	Census_InternalPrimaryDisplayResolutionVertical	-0.215121535

Figure 6: Malware detection parameter extraction results

In combining with the proposed EDR process based on XAI, it is possible to simultaneously verify the process and result of notification contents from the user side and confirm the exact reason for the percentage (e.g., statistical values and similar values) of the existing AI for the detection result. It was an experiment. Therefore, rather than improving the engine of the existing AI EDR system, we optimized the XGBoost model selected through performance comparison among various models most used in XAI. It was configured as a back end of the existing legacy equipment and analyzed data simultaneously. Consequently, it was possible to identify problems in the process that could not be resolved because of the nature of the existing endpoint control, and it is expected to serve as a basis for more clearly identifying questions about AI results.

5 Conclusion

In this study, the approach to generating malware datasets and features, as well as the methodology for malware detection and classification, was verified based on previous research. By using the same ML classification algorithm using the open data set of the previous study, the running time was reduced, and an ML model with excellent performance was selected. Open data sets were used for the data set, objectivity was secured, and comparative analysis was performed using common parameters.

Five models were trained using the same data set to support the claims of this study. The XGBoost model was selected with an average of 94.2% of malware results. The reverse analysis process of XAI was performed using the selected XGBoost model. Moreover, various variables and parameters were detected and analyzed. Afterward, because of checking the detected parameters, the part that was determined to be the most threatening factor in malware and impacted prediction was identified as OrganizationIdentifier. The value is +0.67 based on 1. The closer this value is to 1, the more serious the threat is relative to other parameters an important variable to determine whether it is malware. The second threat parameter is CityIdentifier, with an impact of 0.36. The third is Census_PrimaryDiskTotalCapacit.

What we learned through the proposed XAI inverse model is that we were able to grasp the visibility and properties of data that could not be analyzed with the well-explained results of the existing XAI or the results from the general AI model. This experiment is expected to enable an intensive analysis of the file in the future inflow using network traffic of malware. Moreover, it will be a parameter adjustment value that can be predicted in advance when the file is decomposed and operated at the endpoint or when an unknown attack is attempted. This experiment is expected to contribute significantly to solving the cause analysis using only the strengths of the existing XAI and AI models by tracing back the result analysis area and AI detection results presented as limitations in traditional security control.

This study presents an enhanced method for malware detection using Explainable AI (XAI) integrated with the XGBoost model. By employing SHAP values, we provide transparency in feature influence, making the model's predictions interpretable and actionable in real-world scenarios. One of the critical concerns in cybersecurity is the balance between detection accuracy and interpretability. This study successfully demonstrates that by employing reverse analysis techniques, malware detection can be both accurate and transparent, thus addressing key limitations in traditional EDR systems. Additionally, by identifying critical parameters that contribute to malware detection, this study opens up new avenues for proactive security measures, including enhanced intrusion prevention strategies.

Moreover, this study identifies certain challenges such as handling large datasets and the computational expense of SHAP values. Future research should focus on optimizing these processes to enable real-time application in large-scale, high-speed environments. Reducing the complexity of SHAP while maintaining its explanatory power is crucial for improving EDR system efficiency. This study contributes significantly to the field by offering a transparent, explainable approach to malware detection, which could be a foundation for future proactive security systems that not only respond to threats but also prevent them.

With deeper investigation, it is possible to secure the visibility of the endpoint by collecting various information from the endpoint and analyzing the behavior based on the collected information, AI (ML), and Indicator of Compromise (IOC). It will be possible to filter known and unknown attacks in advance with technologies such as detection. Many existing research papers have emphasized the superiority of detection accuracy of EDRs using specific datasets, but this paper has shown that malware detection accuracy has been improved by applying machine learning to five accessible datasets. In addition, it has significance as the first research paper to apply an explanable AI technique that can examine the reliability of artificial intelligence.

While existing research papers typically focus on quantitative performance in terms of malware detection accuracy, they often lack data on time costs. Consequently, this study also faces the limitation of not presenting improvements in time costs. Further research should explore user-friendly technologies and time-cost advancement to improve complex reporting and results within existing security systems. Additionally, if the processes and improvements proposed in this study can address the "why" behind detection rates—providing statistically similar results within the existing system—it may be possible to elucidate the "black box" process inherent in DL. Increased transparency and reliability of results are expected to be achieved.

Acknowledgement: The authors thank the editors and reviewers for their comments, which have helped a lot to improve the quality of the manuscript.

Funding Statement: This work was partly supported by Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2024-RS-2022-00156287, 50%) and supported by the MSIT (Ministry of Science and ICT), Republic of Korea, under the Convergence Security Core Talent Training Business Support Program (IITP-2024-RS-2022-II221203, 50%) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation).

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Ki-Pyoung Ma, Sang-Joon Lee; data collection: Ki-Pyoung Ma, Dong-Ju Ryu; analysis and interpretation of results: Ki-Pyoung Ma, Dong-Ju Ryu, Sang-Joon Lee; draft manuscript preparation: Ki-Pyoung Ma, Sang-Joon Lee. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: All datasets are publicly available and accessible from the links below: https://www.kaggle.com/datasets/mathurinache/mtakdd19, https://www.kaggle.com/code/wailinnoo/intrusion-detection-system-using-kdd99-dataset, https://www.kaggle.com/code/adepvenugopal/malware-datasets/data?select=malware+analysis.csv, https://github.com/Juan-Herrera-Silva/Paper-SENSORS, https://ieee-dataport.org/open-access/155m-api-import-dataset-malware-analysis, https://raw.githubusercontent.com/mburakergenc/Malware-Detection-using-Machine-Learning/master/data.csv, https://www.kaggle.com/datasets/hassan06/nslkdd, https://www.kaggle.com/c/malware-classification, https://www.kaggle.com/code/kerneler/starter-malware-analysis-datasets-api-78c79941-1 (accessed on 30 October 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] G. Mehta, P. Das, and V. Tripathi, "Challenges in malware detection and effecting areas: Survey," in *Advances in Information Communication Technology and Computing. Lecture Notes in Networks and Systems*, V. Goar, M. Kuri, R. Kumar, and T. Senjyu, Eds., Singapore: Springer, 2022, vol. 392. doi: 10.1007/978-981-19-0619-0_9.

[2] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments," *Energy Rep.*, vol. 7, pp. 8176–8186, 2021. doi: 10.1016/j.egyr.2021.08.126.

[3] W. C. Lin, S. W. Ke, and C. F. Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors," *Knowl. Based Syst.*, vol. 78, pp. 13–21, 2015. doi: 10.1016/j.knosys.2015.01.009.

[4] Ö. A. Aslan and R. Samet, "A comprehensive review on malware detection approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020. doi: 10.1109/ACCESS.2019.2963724.

[5] S. A. Roseline, S. Geetha, S. Kadry, and Y. Nam, "Intelligent vision-based malware detection and classification using deep random forest paradigm," *IEEE Access*, vol. 8, pp. 206303–206324, 2020. doi: 10.1109/ACCESS.2020.3036491.

[6] R. Cai, S. Xie, B. Wang, R. Yang, D. Xu and Y. He, "Wind speed forecasting based on extreme gradient boosting," *IEEE Access*, vol. 8, pp. 175063–175069, 2020. doi: 10.1109/ACCESS.2020.3025967.

[7] H. L. Shifare, R. Doshi, and A. Ved, "Comparative analysis of short-term load forecasting using machine learning techniques," in *Int. Conf. Adv. Netw. Technol. Intell. Comput.*, Cham, Springer Nature Switzerland, 2023, pp. 117–133. doi: 10.1007/978-3-031-64070-4_7.

[8] Y. Wang, Y. Hao, B. Zhang, and N. Zhang, "Short-term power load forecasting using SSA-CNN-LSTM method," *Syst. Sci. Control Eng.*, vol. 12, no. 1, pp. 1–10, 2024. doi: 10.1080/21642583.2024.2343297.

[9] M. P. Zavarsky and D. Lindskog, "Experimental analysis of ransomware on Windows and Android platforms: Evolution and characterization," *Procedia Comput. Sci.*, vol. 94, pp. 465–472, 2016. doi: 10.1016/j.procs.2016.08.072.

[10] A. B. Kardile, "Crypto ransomware analysis and detection using process monitor," The Univ. of Texas at Arlington, Arlington, TX, USA, 2017.

[11] E. Kirda, "UNVEIL: A large-scale, automated approach to detecting ransomware (keynote)," in *IEEE 24th Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Klagenfurt, Austria, 2017. doi: 10.1109/SANER.2017.7884603.

[12] D. Sgandurra, L. Muñoz-Gonz£lez, R. Mohsen, and E. C. Lupu, "Automated dynamic analysis of ransomware: Benefits, limitations and use for detection," 2016. doi: 10.48550/arXiv.1609.03020.

[13] K. Cabaj, M. Gregorczyk, and W. Mazurczyk, "Software-defined networking-based crypto ransomware detection using HTTP traffic characteristics," *Comput. Electr. Eng.*, vol. 66, pp. 353–368, 2018. doi: 10.1016/j.compeleceng.2017.10.012.

[14] Y. Takeuchi, K. Sakai, and S. Fukumoto, "Detecting ransomware using support vector machines," in *Proc. 47th Int. Conf. Parallel Process. Companion*, New York, NY, USA, ACM, 2018, pp. 1–6. doi: 10.1145/3229710.3229726.

[15] S. Jung and Y. Won, "Ransomware detection method based on context-aware entropy analysis," *Soft Comput.*, vol. 22, pp. 6731–6740, 2018. doi: 10.1007/s00500-018-3257-z.

[16] Z. -G. Chen, H. -S. Kang, S. -N. Yin, and S. -R. Kim, "Automatic ransomware detection and analysis based on dynamic API calls flow graph," in *Proc. Int. Conf. Res. Adapt. Convergent Syst.*, ACM, 2017, pp. 196–201. doi: 10.1145/3129676.3129704.

[17] G. Cusack, O. Michel, and E. Keller, "Machine learning-based detection of ransomware using SDN," in *Proc. 2018 ACM Int. Workshop Secur. Softw. Defined Netw. Netw. Funct. Virtual.*, Tempe, ACM, 2018, pp. 1–6. doi: 10.1145/3180465.3180467.

[18] K. P. Subedi, D. R. Budhathoki, and D. Dasgupta, "Forensic analysis of ransomware families using static and dynamic analysis," in *Proc. 2018 IEEE Secur. Priv. Workshops (SPW)*, San Francisco, CA, USA, IEEE, 2018, pp. 180–185. doi: 10.1109/SPW.2018.00033.

[19] O. M. K. Alhawi, J. Baldwin, and A. Dehghantanha, "Leveraging machine learning techniques for windows ransomware network traffic detection," in *Cyber Threat Intelligence. Advances in Information Security*, A. Dehghantanha, M. Conti, and T. Dargahi, Eds., Springer, 2018, vol. 70, pp. 93–106. doi: 10.1007/978-3-319-73951-9_5.

[20] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli and A. K. Sangaiah, "Classification of ransomware families with machine learning based on N-gram of opcodes," *Future Gen. Comput. Syst.*, vol. 90, pp. 211–221, 2019. doi: 10.1016/j.future.2018.07.052.

[21] M. M. Hasan and M. M. Rahman, "RansHunt: A support vector machines based ransomware analysis framework with integrated feature set," in *Proc. 2017 20th Int. Conf. Comput. Inform. Technol. (ICCIT)*, Dhaka, Bangladesh, IEEE, 2017, pp. 1–7. doi: 10.1109/ICCITECHN.2017.8281835.

[22] T. Lu, L. Zhang, S. Wang, and Q. Gong, "Ransomware detection based on V-detector negative selection algorithm," in *Proc. 2017 Int. Conf. Secur., Pattern Anal., Cybern. (SPAC)*, Shenzhen, China, IEEE, 2018, pp. 531–536. doi: 10.1109/SPAC.2017.8304335.

[23] F. Khan, C. Ncube, L. K. Ramasamy, S. Kadry, and Y. Nam, "A digital DNA sequencing engine for ransomware detection using machine learning," *IEEE Access*, vol. 8, pp. 119710–119719, 2020. doi: 10.1109/ACCESS.2020.3003785.

[24] I. A. Bello *et al.*, "Detecting ransomware attacks using intelligent algorithms: Recent development and next direction from deep learning and big data perspectives," *J. Ambient Intell. Humaniz. Comput.*, vol. 12, pp. 8699–8717, 2020. doi: 10.1007/s12652-020-02630-7.

[25] M. Almgren, V. Gulisano, and F. Maggi, *Detection of Intrusions and Malware, and Vulnerability Assessment*, Cham: Springer; 2015, vol. 9148. doi: 10.1007/978-3-319-20550-2.

[26] M. Hirano, R. Hodota, and R. Kobayashi, "RanSAP: An open dataset of ransomware storage access patterns for training machine learning models," *Forensic Sci. Int.: Digit. Investig.*, vol. 40, 2022, Art. no. 301314. doi: 10.1016/j.fsidi.2021.301314.

[27] K. C. Roy and Q. Chen, "DeepRan: Attention-based BiLSTM and CRF for ransomware early detection and classification," *Inform. Syst. Front.*, vol. 23, pp. 299–315, 2021. doi: 10.1007/s10796-020-10017-4.

[28] A. B. Arrieta *et al.*, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fus.*, vol. 58, pp. 82–115, 2020. doi: 10.1016/j.inffus.2019.12.012.

[29] B. H. M. van der Velden, H. J. Kuijf, K. G. A. Gilhuijs, and M. A. Viergever, "Explainable artificial intelligence (XAI) in deep learning-based medical image analysis," *Med. Image Anal.*, vol. 79, 2022, Art. no. 102470. doi: 10.1016/j.media.2022.102470.

[30] V. Belle and I. Papantonis, "Principles and practice of explainable machine learning," *Front. Big Data*, vol. 4, 2021, Art. no. 688969. doi: 10.3389/fdata.2021.688969.

[31] J. A. Herrera-Silva and M. Hernández-Álvarez, "Dynamic feature dataset for ransomware detection using machine learning algorithms," *Sensors*, vol. 23, no. 3, 2023, Art. no. 1053. doi: 10.3390/s23031053.

[32] N. Aslam *et al.*, "Interpretable machine learning models for malicious domains detection using explainable Artificial Intelligence (XAI)," *Sustainability*, vol. 14, no. 12, pp. 1–22, 2022. doi: 10.3390/su14127375.