Study on Machine Learning Techniques for Malware Classification and Detection

Jaewoong Moon[1], Subin Kim[1], Jaeseung Song[1*], and Kyungshin Kim[2]
[1] Sejong University
209, Neungdong-ro, Gwangjin-gu, Seoul KR
[e-mail: jwmoon10@gmail.com, spdlqj99099@gmail.com, jssong@sejong.ac.kr]
[2] Convergence Technology Collaboration Directorate, Agency for Defense Development, Songpa P.O Box 132, Seoul KR
[e-mail: updatekim@add.re.kr]
[*]Corresponding author: Jaeseung Song

Received July 8, 2021; revised September 5, 2021; accepted October 8, 2021; published December 31, 2021

Abstract

The importance and necessity of artificial intelligence, particularly machine learning, has recently been emphasized. In fact, artificial intelligence, such as intelligent surveillance cameras and other security systems, is used to solve various problems or provide convenience, providing solutions to problems that humans traditionally had to manually deal with one at a time. Among them, information security is one of the domains where the use of artificial intelligence is especially needed because the frequency of occurrence and processing capacity of dangerous codes exceeds the capabilities of humans. Therefore, this study intends to examine the definition of artificial intelligence and machine learning, its execution method, process, learning algorithm, and cases of utilization in various domains, particularly the cases and contents of artificial intelligence technology used in the field of information security. Based on this, this study proposes a method to apply machine learning technology to the method of classifying and detecting malware that has rapidly increased in recent years. The proposed methodology converts software programs containing malicious codes into images and creates training data suitable for machine learning by preparing data and augmenting the dataset. The model trained using the images created in this manner is expected to be effective in classifying and detecting malware.

Keywords: Machine Learning, Artificial Intelligence, Data Augmentation, Malware detection.

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funded by the Korean Government (MSIT) under Grant 2019-0-00426 (Development of active kill-switch and biomarker-based defense system for life-threatening IoT medical devices) and IITP-2021-0-01816 (A Research on Core Technology of Autonomous Twins for Metaverse).

http://doi.org/10.3837/tiis.2021.12.003 **ISSN : 1976-7277**

1. INTRODUCTION

WHILE the recent widespread use of computers and the Internet has brought convenience and enrichment to human life, its negative side effects are constantly increasing. As network-connected software is used in every part of human life, malware (normally malicious software) attacks that take advantage of the vulnerabilities of software (such as distributed denial of service [DDoS] attacks, personal information leaks, and hacking) increasingly cause damage to users' cybersecurity. According to [1], the average daily occurrence of malware increased from 1,435 types in 2013 to 8,847 types in 2014, a six-fold increase. AVTEST, an IT security research institute, reported that the number of discovered malware per year increased from 99.71M in 2012 to 1,180.63M in 2021, an increase of about 12 times in 10 years. The reason for the rapid increase in the number of malware is that malware makers are rapidly producing similar and variant malware using automated tools that can easily duplicate and produce malware. Malware produced by an automated tool arbitrarily changes the CALL or JMP commands or modifies only the encryption code part to avoid being detected by anti-virus programs by changing its appearance while exhibiting the same malicious behavior. The number of similar and/or variant dangerous codes is fast expanding as a result of the extension of such polymorphism methodology, and the methodology of analyzing viral diagnostics tools to interfere with normal diagnosis is also evolving.

Machine learning (ML) is a technology that predicts the results and trends of how a computer will behave in the future — it provides an algorithm that analyzes and learns existing data as much as possible and runs a regression model through the process. In other words, the core of ML is to create sophisticated algorithms that can analyze data to identify and learn specific patterns. Today, ML has a significant impact on many fields of technology and science. Recent examples include applications to autonomous vehicle control, speech processing, natural language processing, and computer vision [2]. It is also being applied to various fields in real life, such as an AI security system that learns images to detect dangerous objects at airports, ports, and railroads, or intelligent Closed-Circuit Televisions (CCTVs) that detect abnormal behaviors [3].

In recent years, many attempts have been made to apply ML technology, which is used to solve complex problems in various fields, to the field of information protection. In the past, malicious cyberattacks were not as diverse as they are now, with simple attacking patterns; therefore, intrusion detection or analysis of attacks was possible simply by identifying attack patterns and matching the same patterns. However, currently, many different devices in various fields such as smartphones, -cars, -homes, -factories, and -grids are connected to networks, and many attack technologies that can bypass existing security programs have been developed, making it increasingly difficult to detect and defend against cyberattacks. As malwares and cyberattacks themselves evolve and operate intelligently with the help of artificial intelligence (AI), there is a limit to protecting systems using existing technologies, including virus intrusion detection and network attack analysis [4]. In preparation for evolving cyberattacks, it is necessary to develop technologies that can effectively learn complex and intelligent cyberattacks and perform accurate predictions by applying AI and ML technologies to the field of information security.

In particular, technology that can classify malwares with similar characteristics to detecting malware is a key component of information security. In the past, malware was mainly detected using a static analysis technique to determine the presence or absence of malicious code by identifying the structure of a code or program, or a dynamic analysis technique that detected malicious code by analyzing changes in the registry and memory while directly executing a

program. These existing analysis methods carried out the analysis while directly analyzing or executing the code, with a risk of state explosion problem or malicious code execution, as well as a limitation that the analysis was ineffective on the codes applied with obfuscation, which made code written in a programming language challenging to read. The development of a technology that can learn the characteristics of vulnerabilities or malware and predict hidden problems by utilizing the advantages of AI and ML in this complex analysis will reduce the risk burden on analysts and enable faster and more accurate analysis of the rapidly increasing number of malware [5].

This study examines various ML algorithms and techniques to determine the possibility of using AI and ML technology to detect malware. In particular, to improve the understanding of ML, we analyzed the process of performing ML, which led us to design a methodology for analyzing malware through ML. Because the accuracy of prediction varies greatly depending on the completeness of the ML model, the overall framework of ML behavior was summarized by investigating the classification of learning methods such as supervised, unsupervised, and reinforcement learning, the field of application of ML technology, and ML execution tools such as TensorFlow and Scikit-learn. To obtain good results, it is important to select an ML model and a utilization algorithm that is suitable for the target data. Therefore, various ML algorithms, such as support vector machine (SVM), artificial neural network (ANN), Naive Bayes classifier (NB), and Bayesian network (BN), have been examined to understand their advantages and disadvantages as well as suitable target data. Based on the results of the survey, this study proposes a methodology and a framework that can automatically classify malware by converting malware into training image data, training these data through ML, and creating models. Because malware is a software program that contains malicious code, it needs to be converted into a standardized form to allow ML to learn it. This study proposes a mechanism for converting given software programs containing malicious code into images and turning them into training data for the model. The proposed mechanism uses the fact that the same malicious code will appear in the same image pattern even if it is located in different parts of the infected malicious software. Therefore, the proposed solution assumes that the malware in the same category will have a similar pattern after being converted into an image, and ML learns this pattern and uses it to create a sophisticated model for automatically classifying programs containing malicious code. Based on previous research results, this study suggests the use of a convolutional neural network (CNN) based inception model to classify images converted from software, including malicious code.

The contributions of this study are as follows:

- A survey on various ML techniques and their procedures, classification of ML techniques according to learning methods, fields of application, and tools for ML.
- A proposed ML-based automatic malware classification scheme that supports automatic classification of software containing malicious codes using ML techniques.

This study investigates the definition of ML in Section 2, the utilization of ML technology in Section 3, and ML technology in Section 4. Section 5 introduces and explains the framework designed to convert and classify the software containing malicious codes into images. In addition, Section 6 presents the conclusion and suggestions for future studies.

2. ML DEFINITION

ML is a branch of AI and refers to a method of data analysis that automates analytics model building. ML uses a large amount of data for training and is mainly used to classify and predict data through a trained model. This section describes the ML procedure and the classification

of ML according to the learning method.

2.1 ML Procedure

The ML procedure is to develop a model from data training and perform a test using the developed model. For example, ML creates a model that can identify a puppy based on photographs of puppies, and then uses the model created in this way to test and identify photographs with a puppy in them.

Although there exist many different understandings of the ML procedure, If the process of data management and processing for learning, such as gathering and preparing data, is included in the ML stage, the ML procedure is generally classified into seven steps [6]. This study used an ML scheme consisting of seven steps. The seven steps of the ML scheme are as follows:

Step 1: Because the quality and quantity of data collected directly affect the performance of the predictive model in ML, the first step is to gather high-quality data for building the model.

Step 2: The gathered data are prepared using various preprocessing methods for building the model. For example, the dataset is augmented by preprocessing (crop or rotation for images) when the dataset is insufficient, or the noise is removed from the images. In addition, the dataset is divided into two parts: training and evaluation.

Step 3: There are many machine learning algorithms and models that have been developed to solve various problems. This step chooses the most suitable ML model for the problem to be solved. For example, some models are suitable for detecting image patterns, whereas others are more suited to dealing with video data.

Step 4: This is the heart of the ML process. In this step, part of the dataset allocated for training is used to teach/train the selected model. This step requires patience and experimentation, as the model can be built based on trial and error.

Step 5: When the training is completed, the maturity of the model is verified through evaluation. How the created model would perform in the real world can be tested through an evaluation.

Step 6: After evaluation, further improvements for learning can be considered, and the performance of the model can be improved by adjusting the parameters through hyperparameter tuning.

Step 7: Because the ultimate goal of ML is to use data to obtain answers to problems, the value of ML is realized by predicting what the given data means by using the improved model built through the previous steps.

2.2 ML Classification by Learning Method

ML can be largely divided into supervised learning, which uses labeled input and output data, and unsupervised learning, which does not. Recently, it has been classified into three categories, including reinforcement learning, owing to the development of a method involving trials and errors called deep learning (**Fig. 1**). The following is a brief introduction to the three ML methods.

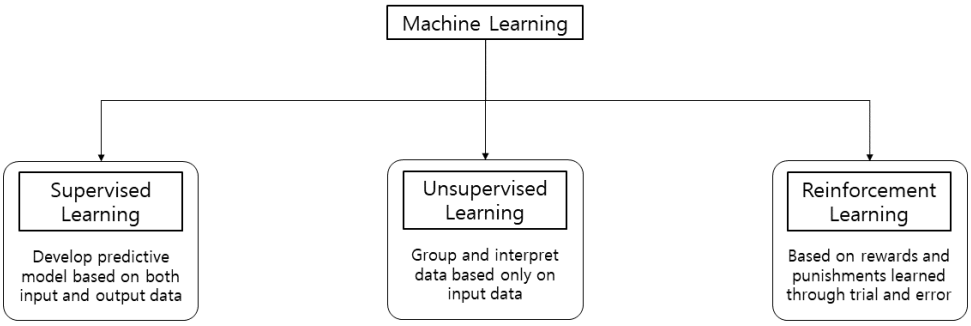

Fig. 1. ML classification

Supervised learning

Supervised learning, also known as supervised ML, is the basis of ML techniques. After making a model and training the computer with prepared sample data with predefined labeled input and output data, it predicts the expected values or determines the types of new data based on the results of training. Supervised learning methods use two types of algorithms, i.e., classification and regression. Classification algorithms predict the discrete values such as Male or Female, while regression algorithms predict the continuous values such as exchange currency, stock value, etc.

The preparation of a training dataset is essential for supervised learning. Because most datasets contain noise and missing feature values, preprocessing is required to facilitate the training. Many studies have been conducted to detect and process data outliers in data preparation and preprocessing steps [7][8]. A representative example is the inter-quartile range technique, which helps to identify outliers in continuously distributed data. Outliers are detected by dividing the dataset into quartiles and sorting them into four equal parts [9].

Many algorithms can be used in supervised learning, such as decision trees, k-nearest neighbors (KNN), SVMs, and neural networks (see Section IV).

Unsupervised learning

Unsupervised learning refers to a technology that obtains a certain result, which is the clustering of similar data, using sample data that a computer does not know the answers to by itself without human intervention. For example, the energy usage data for which no meaning has been found in the data analysis can be applied with an unsupervised learning algorithm to discover a pattern or answer that had not been recognized by the human eye.

Representative unsupervised learning methods include clustering and association rules. When a human does not know how to classify in a situation where too many features exist, a computer can perform classification in a new way using the clustering algorithm. Association rule is a method of finding hidden patterns in data in the absence of a target variable.

In unsupervised learning, learning is performed using unlabeled data based on patterns, structures, and knowledge. It identifies relationships, structures, associations, or hierarchies of data based on data samples provided to the system. Clusters can be formed based on similarity or distance measures, such that similar samples belong to the same cluster.

Reinforcement learning

Reinforcement learning is a method of finding patterns hidden in numerous data gathered through trials and errors during learning. The reward is the key to reinforcement learning. Computers find problems by themselves through rewards according to their actions, and in the beginning, some human intervention may be involved in providing these rewards. Deep learning is a typical method used in reinforcement learning. In fact, deep learning is also used for supervised and unsupervised learning, as deep learning implies that the neural network is built very deeply [10].

This is an optimization method for solving sequential decision problems that are mathematically modeled using the Markov decision process [11]. Learning is performed by adjusting the execution method based on the results of the algorithm. The result received by reinforcement learning is feedback on whether the algorithm performs well, not the ground truth in supervised learning. Reinforcement learning consists of three elements: an agent (a learner or a decision maker), an environment (any object with which the agent interacts), and an action (activity of the agent). In this situation, the purpose of reinforcement learning is to allow the agent to select an action that can maximize the expected reward within a certain time.

An example of reinforcement learning is autonomous vehicle control. In this case, the computer, which is the brain of an autonomous vehicle, should not receive instructions for driving the car. Because it is impossible to predict everything that will happen on the road, a reinforcement learning agent for an autonomous vehicle that can learn from systems that provide rewards and penalties can be developed, and this agent receives rewards for achieving certain goals (e.g., compliance with laws, driving safely, and achieving optimum fuel economy).

3. ML APPLICATIONS AND TOOLS

This section describes the fields of application for ML and the ML tools used. ML is being used in almost all fields related to human life, including computer science. Recently, it has been applied to various fields such as voice recognition with AI speakers that are often used at home, character recognition, virtual reality, robotics, robot surgery, network optimization, and stock price prediction.

3.1 Applications of ML

ML can also be effectively used in the field of information security. It can help companies effectively analyze various threats and respond more effectively to incidents that occur. For example, it can be used to detect intrusions into internal networks and to detect and analyze malicious codes that are planted in generic applications for distribution. Recently, cyberattacks have become more intelligent as society becomes cyber-intelligent as a whole with advancements such as smart-phones, -cars, -homes, -factories, and -grids. Efforts are being made around the world to develop new technologies that combine ML and information security to effectively respond to intelligent cyberattacks. For example, Google uses ML to analyze threats from mobile devices using Android, the mobile operating system, and uses the technology to find and remove malicious codes from infected mobile devices. Amazon, a cloud service provider, has launched an AI service that uses ML to efficiently search, classify,

and sort data stored in Amazon cloud storage.

ML technology is being used in various security fields such as zero-day vulnerability removal using ML, automation of repetitive security tasks, physical security using stranger face recognition, and threat analysis on mobile endpoints. For example, the Computer Science and Artificial Intelligence Lab of the Massachusetts Institute of Technology has developed a system that can find problems in vast amounts of corporate data. This system provides a function to check the login and activity information of millions of companies, and to automatically classify anomalies (such as external leakage of confidential data) among them and deliver them to the security team.

In addition, one of the technologies and services required in the field of information protection is a technology that can detect continuously evolving malicious codes and classify software containing similar malicious codes using ML. More effective intelligent security technology will be realized by using ML technology that can train itself to analyze the continuously increasing amount of malicious codes and security threat-related data and to sort out the data that threaten security.

3.2 ML Tools

Various tools, platforms, and software have been developed and employed to facilitate the use and development of ML services. Representative examples include TensorFlow, Scikit-learn, convolutional architecture for fast feature embedding (Caffe), and computational network toolkit (CNTK). These tools perform cumbersome tasks that AI service developers have to manually perform to train models, such as gathering and preparing data, adding features, and starting calculations.

- TensorFlow: This is an ML tool developed by Google for ML and deep learning and is an open-source library useful for large-scale and numerical ML. TensorFlow supports multi-parallel processing that can utilize multiple central processing units (CPUs) and graphics processing units (GPUs) for data processing and analysis. It also provides a data flow technology that processes data on the cloud in real-time. While TensorFlow supports a wide range of applications, it utilizes deep learning to deploy ML systems across many domains, such as speech recognition, computer vision, robotics, and natural language processing, and serves as a platform for research [12].
- Scikit-learn: This is an open-source ML package, which is a multi-purpose, integrated tool. It is a Python module that does not deal with deep learning or reinforcement learning but provides various ML methods for medium-scale supervised or unsupervised learning. It supports regression, clustering, classification, and preprocessing. Scikit-learn emphasizes ease of use, performance, documentation, and Application programming interface (API) consistency, and is distributed under a simplified Berkeley Software Distribution (BSD) license to encourage its use in both academic and commercial environments [13].
- Caffe: It is one of the first deep learning libraries developed by the Berkeley Vision and Learning Center and is the deep learning framework used by Facebook. Caffe was developed considering expression, speed, and modularity. Caffe is also a BSD-licensed C++ library with Python and MATLAB bindings for efficient training and deployment of general-purpose CNNs and other deep models in the architecture [14]. Initially developed with a focus on computer vision, it is now expanding its scope to reinforcement learning, speech recognition, and multimedia.
- CNTK: This is an open-source deep learning framework for Windows and Linux provided

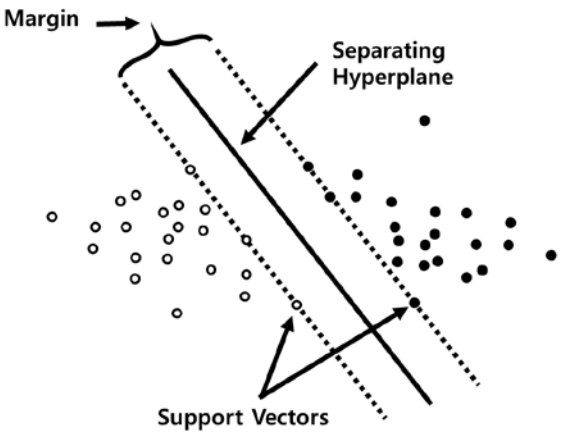

Fig. 2. Basic topology model of SVM

by Microsoft. Neural networks such as the feedforward neural network (FNN), CNN, and recurrent/long short-term memory are supported. In particular, it demonstrates exceptionally high speed in CNNs and recurrent neural networks and is optimized for CPUs. It has excellent scalability, as is designed with efficiency in consideration of optimization for parallel trains of server GPUs [15].

4. TYPES OF ML ALGORITHMS

In ML, selecting an algorithm appropriate for the condition to be examined is required to produce an effective outcome, that is, a model. This section describes various algorithms used in supervised and unsupervised learning, which are the most commonly used types of ML.

4.1 Algorithms for Supervised Learning

Well-known supervised learning algorithms include SVMs, ANNs, NB, and Bayesian networks. This section explains these algorithms for a better understanding.

SVM

SVM is a classification technique used to classify data and is based on the principle of setting a boundary in an area where points belonging to one class are gathered. Once a boundary is defined for a training dataset, classification is performed by checking which side of the boundary it belongs to when classifying a new testing dataset. Finally, defining and calculating this decision boundary is an important issue in SVM. Each data point consists of a vector, and the boundary used in the SVM is called a hyperplane. For a two-dimensional (two-attribute) SVM, the boundary is a straight or curved line, and for a three-dimensional SVM, the boundary is flat or irregular and has a complex surface.

Fig. 2 shows the basic configuration of SVM. In the figure, the support vectors represent the data points close to the decision boundary, and the margins indicate the distances between the decision boundary and the support vectors. Further, finding the optimal hyperplane is the key

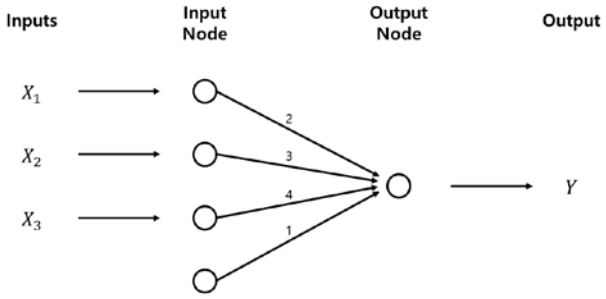

Fig. 3. ANN two-layer topology model

to SVM. In other words, a boundary that maximizes the average distance between two classes yields better classification results. At this time, the support vectors were used to find the optimal hyperplane. Nevertheless, not all data were clearly separated by the hyperplanes. When the data were linearly separated, many points were entangled in the margin, and in this case, it was preferable to find the hyperplane with the fewest points. If the data were not linearly separated despite the absence of errors, it was recommended to use a kernel function. A kernel function provides an option to transform a nonlinear space into a linear one, and most packages, including SVM, have various nonlinear kernel functions ranging from polynomials to sigmoid functions. For datasets that cannot be solved by a linear model, it is necessary to find and use the appropriate kernel function for the data to be used for SVM.

The SVM algorithm consists of three key steps: (1) find the boundary; (2) find the hyperplane that maximizes the margin or distance at each boundary; and (3) determine which part of the hyperplane the given trial record belongs to.

ANN

ANN uses mathematical relationships that closely resemble the biological processes of neurons and the ways in which the human brain perceives patterns, to model the relationship between input and output variables. In other words, it is a method for interpreting various input data using perceptrons, classification, or clustering. The use of ANN allows classifying or clustering data by recognizing specific patterns in image, sound, text, and time-series data. Currently, ANN technology is widely used in all sectors associated with human life, including finance, economy, social science, and national defense.

For example, in a remarkably simple neural network linear model, where Y is the calculated output value and X1, X2, and X3 are the input attributes, 1 is the y-intercept, and 2, 3, and 4 are the coefficients of the input attributes X1, X2, and X3, respectively. **Fig. 3** shows that this simple linear model can be expressed in topological form. In this topology, the predicted value Y is output by adding all the terms of the inputs and weights, and then applying the activation function where a node is called a building block in the terminology of a neural network. The first layer of nodes closest to the input is called the “input layer” or “input node”. The last layer is called the “output layer” or “output node”. The output layer may perform the transform function as well as the compositing function. The transform function converts the output value to the desired range. The output layer also performs an activation function. A simple two-layer topology with one input layer and one output layer is called a perceptron, as shown in the topology in **Fig. 3**. Perceptron is a FNN, where the input moves only in one direction and is

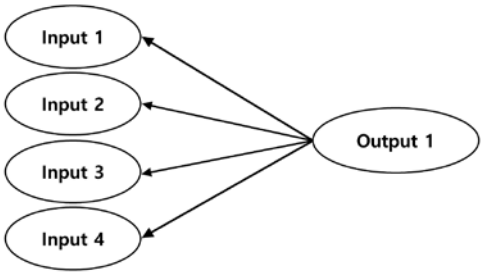

Fig. 5. Naïve Bayes model

not cycled.

ANNs are commonly used to model nonlinear or complex relationships between the input and output variables. This requires more than one hidden layer in the topology. This layer is separate from the input or output layers and receives an input from the previous layer to apply the activation function. Then, the output value is calculated as a complex combination of input variables, as shown in **Fig. 4**.

NB algorithm

The NB algorithm is an algorithm derived from statistics and probability theory that utilizes the probabilistic relationship between basic factors (attributes) and class labels (results). At this time, the NB executes the algorithm, assuming that the properties are independent. This method assumes that the input features are independent, which is unlikely to occur in reality. NB is built based on Bayes' theorem. With X as evidence (a set of factors or attributes) and Y as an outcome (target or class label), Bayes' theorem determines Y, the probability that an outcome will occur given the value of X.

Bayesian modeling is (1) relatively easy to understand once the symbols are identified and (2) easy to implement in any programming language. Calculations to build the model are remarkably simple, and it only requires the creation of a lookup table for the probability. The advantage of the NB algorithm is that it does not require a large amount of training. As **Fig. 5** shows, the link in an NB model goes from output to input, which indirectly excludes the output and provides simplicity to the model with no interaction between the inputs [16].

4.2 Algorithms for Unsupervised Learning

Representative unsupervised learning models include K-means clustering, density-based

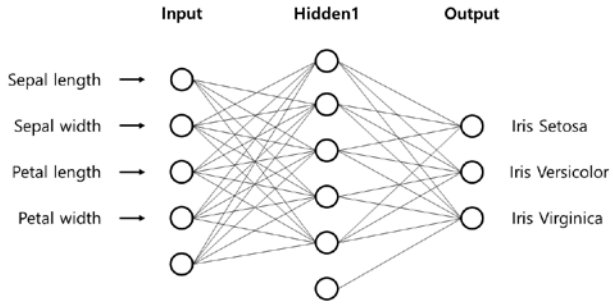

Fig. 4. ANN topology model using hidden layers

spatial clustering applications with noise (DBSCAN), expectation maximization, and hierarchical cluster analysis. This section describes k-means clustering and DBSCAN.

K-means clustering

K-means clustering is a prototype-based clustering method, and the dataset is divided into k clusters. K-means clustering is the simplest and most widely used clustering algorithm. The user specifies the number of clusters (k), and the algorithm aims to find a prototype data point for each cluster. The prototype represents the center position of the cluster, which is also called the centroid. Visually, the k-means algorithm divides the data space into k partitions, with the center of each partition being the cluster prototype. As **Fig. 6** shows, data objects existing within a compartment belong to the corresponding cluster, these compartments are called Voronoi compartments, and each prototype becomes the seed of the Voronoi compartment.

In k-means clustering, an iterative approach is used to cluster the data. The k-means clustering method is divided into the following steps:

Step 1: Resetting centroids: K arbitrary centroids are placed.

Step 2: Assigning data points: After setting the centroids, each data point is assigned to the nearest centroid to form a cluster. Here, the nearest centroid was determined using the distance scale. The distance scale methods used include the Euclidean distance scale, Manhattan scale, and Jaccard coefficient.

Step 3: Calculating the new centroid: For each cluster, a new centroid is calculated, and this centroid is also the prototype of the cluster. The newly calculated centroids represent the data points that best represent all the data points included in the cluster.

Step 4: Iteration of assigning data points and calculating centroids: All data points are assigned to new centroids by repeating the calculation of new centroids and assigning each data point to the nearest centroid (Steps 2 to 3). This process is repeated until the centroids are no longer updated.

DBSCAN

DBSCAN is a widely used density-clustering algorithm. Density is defined as the number of data points included in a unit space in an n-dimensional space, and n is the number of attributes included in a dataset. Density is related to the number of data points included in a unit space, and clusters exist when there is a relatively high-density space surrounded by a low-density space (see **Fig. 7**).

The DBSCAN algorithm creates clusters by distinguishing high-density regions from low-

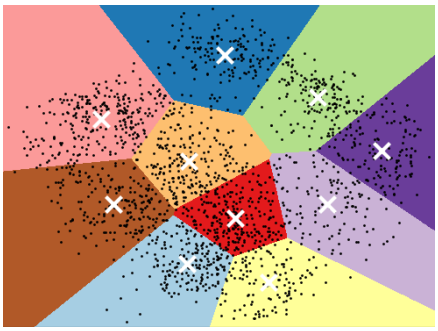

Fig. 6. K-means clustering of the dataset (Principal Component Analysis reduced data) [23]

density regions in a dataset. Because the distance is calculated as in the k-means clustering method, it is preferable to use numeric data. The DBSCAN algorithm is executed as follows:

Step 1: The high-density region is defined by setting the distance scale, e (specified distance), and n (minimum number of samples required).

Step 2: The core point meets the high-density region and assigns the region as a cluster.

Step 3: If there are data satisfying the core point in the high-density region, the previously set cluster is expanded to include the region.

Step 4: Steps 2 to 3 are repeated until no more core points can be defined within that high-density area.

Step 5: The remaining data, i.e., the data not included in any cluster, are defined as noise points.

The advantages of DBSCAN are that it is possible to set the cluster regardless of the shape or size of the cluster, and that there is no need to predetermine the number of clusters. As noise points are sorted out, it is possible to prevent noise from affecting the cluster; thereby, avoiding obtaining an erroneous cluster result due to singular values. DBSCAN also has the advantage that it can be applied to data with complex or geometric shapes as it creates clusters by connecting high-density regions. However, the amount of computation is greater than that of K-means clustering, with a longer execution time owing to the high computational complexity. The higher the dimensionality of the data, the more difficult it becomes to find or select the optimal value of the parameter that determines the minimum value. Therefore, it is difficult to generate clusters with various density distributions, and there is also a limitation in terms of accuracy [17].

5. UTILIZATION OF ML TECHNOLOGY FOR INFORMATION PROTECTION

According to a report (titled “The malicious use of artificial intelligence”) published in February, 2017 at a workshop; [18], attended by 27 organizations including Oxford University and OpenAI in the UK, various AI technologies are currently having a significant impact on

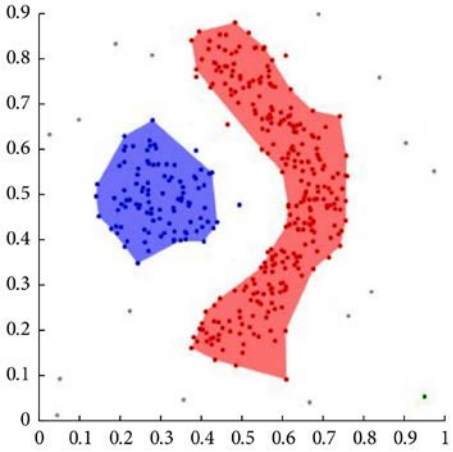

Fig. 7. Basic concept of DBSCAN [24]

cybersecurity, and will cause more severe problems in the future. The six ways in which this AI will affect security are:

1) Dual-use: AI is often used to provide benefits to human life, but it can also be used in harmful ways depending on who develops it and for what purpose.
2) Vulnerability: Due to the structural characteristics of AI, there are vulnerabilities associated with it; such as data poisoning, adversarially-generated datasets, and defects in the intelligent systems.
3) Efficiency: AI efficiency is increased owing to AI algorithm cloning and hardware performance enhancement.
4) Outstanding performance: AI can execute tasks that humans cannot.
5) Anonymity: AI algorithms freely perform human-related tasks.
6) Rapid diffusion: AI algorithms and research results are readily available online.

As such, AI technologies are being used by criminals, terrorists, and hackers for harmful purposes. In the current situation where the technology used in society is becoming highly intelligent, the number of cases of AI abuse that harms information protection will continue to increase. To solve this problem, it is necessary to prepare for various cyber security incidents that are becoming more intelligent by taking advantage of AI. Accordingly, this section investigates how ML technologies are being used for information protection.

5.1 Integrated Security Monitoring and Control

The security monitoring and control system used in smart cities processes an average of billions of security control events per day. Some of these events occur in daily routines, but several events occur due to intelligent cyberattacks. There is a limit to the functions of the existing integrated security monitoring and control system to prevent cyberattacks using various unpredictable forms of AI.

An intelligent integrated security monitoring and control system equipped with ML technology could be an alternative to deal with these threats. Real time infringement handling and energy optimization could be achieved by interlocking various information generated from various systems constituting a smart city (i.e., information communication, energy, and autonomous vehicles) on a centralized or distributed platform and by analyzing large amounts of data through ML. It even allows the detection of unknown and abnormal threats and prevents them in advance.

5.2 Network Intrusion Detection

Network intrusion detection technology refers to detecting and responding to the movement of unauthorized users illegally accessing information assets or using information resources in advance. Currently, most network intrusion detection products use technology to detect and block attacks based on signatures, such as virus detectors. This method has limitations in real time management and the blocking of network intrusions, which have recently become more advanced, automated, and intelligent. It is possible to determine whether the network packets collected in real-time are normal or abnormal by training an ML model using numerous normal network data and network breach data.

5.3 Malware Analysis

According to a 2020 report published by Malwarebytes Labs [19], 50 million malware were discovered worldwide in 2019 alone, showing a gradual increase in the rate of occurrence in

various industries. Similarly, a 2020 report released by the Korea Internet & Security Agency (KISA) [20], noted that information leakage occurred in most among malware types (35.7%), followed by account information leakage accounting for 30.6%, Distributed denial of service (DDoS) accounting for 9.7%, remote control, ransomware, and virtual currency mining, which indicated that a variety of malware was continuously being issued. Some of these malware were newly created, but there were also many variant malware that attackers have created by modifying existing malware to avoid antivirus software programs.

ML could be used in the analysis of malware by training the model based on data such as the abnormal behavior model of the malware and various IP lists used in the attacks and using the model to determine the presence or absence of abnormal software behavior, including malicious code. However, even in this case, effective results might not be obtained because of difficulties in accessing malicious code and securing sufficient malicious code learning data.

Therefore, research on various methods to use ML more effectively for malicious code analysis is underway. The next section proposes one such method — a scheme for classifying software programs containing malicious codes by converting malware into images and using them as training data.

6. MALWARE CLASSIFICATION METHODOLOGY USING ML

Although there are various existing methodologies for classifying malware, research on technologies that can detect malware more effectively is being conducted owing to several limitations. Methods using AI and ML are also included in such endeavors. This section reviews existing methodologies for malware classification and proposes a malware classification scheme using ML.

6.1 Pattern-based Malware Detection Technique

Approaches to existing malware detection technologies can be broadly classified into static and dynamic analyses. The static analysis identifies malicious patterns or defects by analyzing the meaning of the code and exploring the control flow of the executable file without executing the source code. It provides complete coverage with a given source code, but in general, it is difficult to perform static analysis owing to code obfuscation, and there are restrictions on utilization as source code access to software is not always allowed. Executable files must be decompressed and decrypted before analysis, and even the analysis can be hindered by unwieldy complexity issues.

However, dynamic analysis is a method of analyzing behavior patterns by executing codes in a virtual environment and checking changes after operating it. As a result of the dynamic analysis, executable behavior reports are generated based on execution traces. It is more efficient than static analysis, and in particular, there is no need to decompress or decrypt the executable file, and it can be applied even in situations with restricted access to the source code. However, there is a scalability problem associated with dynamic analysis because it requires a significant amount of time and resources. In addition, some malicious behavior may not be observed, as the environment does not meet the trigger conditions under which the malware is actually executed.

Because existing methods analyze and diagnose known malicious code samples or patterns to determine whether the corresponding code and behavior occur during static or dynamic analysis of a target program, it is impossible to respond to newly generated or tampered

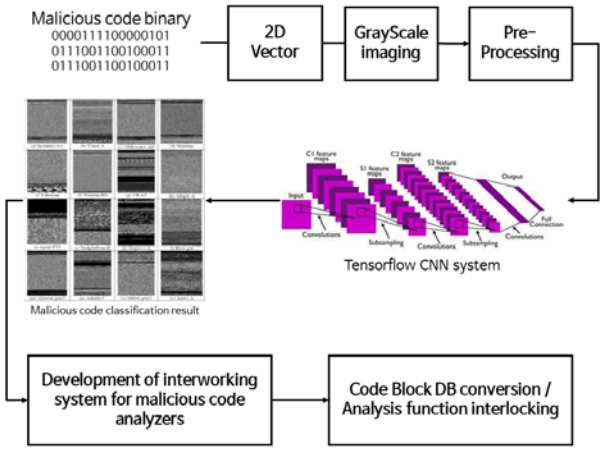

Fig. 8. Structure of malicious code classifier and the process of the malicious code classification scheme: (a) structure of malicious code classifier; (b) process of the malicious code classification scheme

malicious code attacks. As such, existing methods have limitations, and research is being conducted to classify malware by taking advantage of AI.

6.2 Methodology for Malware Image Conversion and Classification

As mentioned earlier, one of the fields where ML exhibits the best performance includes pattern analysis and classification of images and videos. AI/ML technologies provide sufficiently accurate real time performance to identify criminal faces from real time video and image data collected from numerous surveillance cameras installed in smart cities [21].

Based on the fact that the current AI/ML technologies provide good performance on image detection, this paper introduces an idea to convert malicious software (i.e., malware) into an image. This concept makes a malware classification problem into an image recognition problem. The concept of applying AI/ML to malware detection now gains some popularity among security experts [26, 27]. Therefore, this paper proposes a high-level methodology that defines a process of classifying malware using AI/ML image detection technologies.

The conversion of a software program containing malicious code into images to obtain a high-quality malicious code dataset for model training would enable malware classification and real time detection using ML. In other words, if the malware binary file is expressed as an image, the problem of classifying malware can be viewed as a problem of classifying images. Existing classification technology requires the decomposition or execution of malicious codes, but the method using images does not require such a process. In particular, in terms of performance, the classification accuracy was expected to be further improved (see **Fig. 8**). In addition, it would be more adaptable to widely used obfuscation techniques, such as section encryption. This requires key technical support.

Conversion of Malware into Image

This technique transforms software containing malicious code into an image. It is most appropriate to visualize the bytecode of a malware as a grayscale image. What is needed in learning an image is a pattern that shows edges, etc. A grayscale image is composed of only

brightness information without color information, providing the advantage of being able to check the patterns of images faster and more efficiently than a color image. As for the malware family group and the variant family group generated from the same malicious code, similar patterns in the layout and texture of the converted images appear. Motivation and classification methods that use standard image features could be applied based on such visual similarities. In contrast to static and dynamic analyses, such classification does not require code decomposition or execution.

Preprocessing and augmentation for ML

After converting the malware into a grayscale image, a preprocessing process was required to convert it into an environment that could perform best in ML. For example, a study at the University of California, Santa Barbara (UCSB) proposed a method of converting the images into a rectangular shape of the same width [22], whereas other research groups [25] converted the images into a square shape, which was then zoomed in/out to the appropriate size, as in the case of MNIST of ML. For image data in general, an image dataset with the same label can be enlarged using the data augmentation technique when it is difficult to create a model with the desired level of capability due to the insufficient volume of the training dataset. Here, the dataset volume was augmented by cropping, enlarging, or rotating specific parts of the image data.

Neural network system for image classification

A CNN specialized for image classification could be used to classify the converted images from a malware dataset. As mentioned above, CNN is a technique for classifying images by applying a filtering technique to an artificial neural network. Owing to their high performance, CNNs are widely used in computer vision fields such as autonomous vehicles and face recognition. In the ML stage, selecting the model that best fits the purpose is an essential factor in creating a model that provides the desired output. Therefore, the CNN appears to be suitable for the proposed scheme of imaging malware for classification. Because TensorFlow, newly released by Google, provides various libraries, including CNN, a model suitable for classifying malware will be easily developed using these tools.

7. CONCLUSION

AI is being applied in several fields and is undergoing rapid development as a result of recent advances in computer performance. Since the number of intelligent and malicious cyberattacks has increased in recent years, many studies have been conducted on the use of AI and ML technologies in the information security field. Because the frequency and complexity of such attacks exceed the limits of human processing capabilities, it is necessary to respond to them by incorporating AI into the information security field.

This study reviewed ML and malware classification techniques by using ML and proposing a scheme for imaging the malware binary file and classifying the converted images using a CNN-based inception library, which is an ML algorithm specialized for image analysis. Using the proposed methodology of this study, further research will be conducted to implement the proposed methodology and verify its effectiveness through experiments using real malware datasets.

References

[1] AV-TEST GmbH, https://www.av-test.org/en/statistics/malware/

[2] T. M. M. M. I. Jordan, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, Issue 6245, pp 255-260, Jul. 2015. Article (CrossRef Link)

[3] K. W. Kug, “Cases of application by artificial intelligence technology and industry,” *IITP*, 2019.

[4] Z.-K. Zhang, “IoT Security: Ongoing Challenges and Research Opportunities,” in *Proc. of 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, pp 230-234, Nov. 2014. Article (CrossRef Link)

[5] D. L. JS Luo, “Binary malware image classification using machine learning with local binary pattern,” in *Proc. of IEEE International Conference on Big Data*, pp 4664-4667, Dec. 2017. Article (CrossRef Link)

[6] I. S. Oh, "Machine Learning," in Seoul, KOREA: Hanbit, 2017.

[7] Z. Y. I Muhammad, “SUPERVISED MACHINE LEARNING APPROACHES: A SURVEY,” *ICTACT Journal on Soft Computing*, vol. 5, pp. 946-952, May. 2015. Article (CrossRef Link)

[8] H Paulheim, R Meusel, “A decomposition of the outlier detection problem into a set of supervised learning problems,” *Machine Learning*, vol. 100, pp 509-531, Jun. 2015. Article (CrossRef Link)

[9] B. P. B. S. HP Vinutha, “Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset,” *Information and Decision Sciences*, vol. 701, pp 511-518, Apr. 2018. Article (CrossRef Link)

[10] R. F. P. G. AI Károly, “Unsupervised clustering for deep learning: A tutorial survey,” *Acta Polytechnica Hungarica*, vol. 15, pp 29-53, Aug. 2018. Article (CrossRef Link)

[11] S. Y. Jang, H. J. Yoon, N. S. Park, “Research Trends on Deep Reinforcement Learning,” *ETRI*, vol. 34, Issue 4, pp 1-14, Aug. 2019. Article (CrossRef Link)

[12] M. Abadi, “TensorFlow: learning functions at scale,” in *Proc. of ICFP 2016: Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, vol. 51, pp 1-1, Sep. 2016. Article (CrossRef Link)

[13] G. V. A. G. F Pedregosa, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, 12, 2825-2830, Oct. 2011. Article (CrossRef Link)

[14] E. S. J. D. Yangqing Jia, “Caffe: Convolutional Architecture for Fast Feature Embedding,” in *Proc. of the 22nd ACM international conference on Multimedia*, pp. 675-678, Nov. 2014. Article (CrossRef Link)

[15] A. A. Frank Seide, “CNTK: Microsoft's Open-Source Deep-Learning Toolkit,” in *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp 2135, Aug. 2016. Article (CrossRef Link)

[16] Z. Y., I Muhammad, “Supervised Machine Learning Approaches: A Survey,” *ICTACT Journal on Soft Computing*, vol. 5, Issue. 03, pp 946-952, Apr. 2015. Article (CrossRef Link)

[17] J. H. Kim, N. Aziz, “An Enhanced DBSCAN Algorithm to Consider Various Density Distributions for Educational Data,” *KACE*, vol. 22, pp 41-44, Jan. 2018. Article (CrossRef Link)

[18] BRUNDAGE, Miles, et al, “The malicious use of artificial intelligence: Forecasting, prevention, and mitigation,” *arXiv preprint arXiv:1802.07228*, Feb. 2018. Article (CrossRef Link)

[19] Malwarebytes Labs, 2020 State of Malware, 2020, [Online] Available: https://www.malwarebytes.com/resources/files/2020/02/2020_state-of-malware-report-1.pdf

[20] KISA, “KISA Cyber Security Issue Report : Q3 2020,” pp 1-54, Oct. 2020. Article (CrossRef Link)

[21] S. W. LEE, J. Y. PARK, S. W. LEE, “Low resolution face recognition based on support vector data description,” *Pattern Recognition*, vol. 39, Issue. 9, pp. 1809-1812, Sep. 2006. Article (CrossRef Link)

[22] NATARAJ Lakshmanan, MANJUNATH, B. S, “SPAM: Signal processing to analyze malware [applications corner],” *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp 105-117, Mar. 2016. Article (CrossRef Link)

[23] “scikit-learn.org,” [Online]. Available: https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html.

[24] A. Sharma, “Advances in Computational Imaging: Theory, Algorithms, and Systems,” *Mathematical Problems in Engineering*, vol. 2017, pp 9, Feb. 2017. Article (CrossRef Link)
[25] C. Shorten, T.M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *J Big Data*, 6, no. 60, pp 1-48, Jul. 2019. Article (CrossRef Link)
[26] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang and F. Iqbal, "Malware Classification with Deep Convolutional Neural Networks," in *Proc. of 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pp 1-5, Feb. 2018. Article (CrossRef Link)
[27] J. Zhang, Z. Qin, H. Yin, L. Ou and Y. Hu, "IRMD: Malware Variant Detection Using Opcode Image Recognition," in *Proc. of 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 1175-1180, Dec. 2016. Article (CrossRef Link)

Jaewoong Moon is a professor at Sejong University and has been an expert in information protection and software for 25 years. He majored in information protection in 2019 at Sejong Cyber University Graduate School and completed his doctorate in the Department of Information Protection at Sejong University in 2021. Starting as an engineer, he conducted security consulting, research and development, and system design in the field of cybersecurity, and founded an information protection company in 2002 and operated the company as a CEO for about 16 years.

Subin Kim is a B.S. candidate in the department of computer and information security with the college of software convergence, Sejong University, Seoul, Korea. Her research interests include information security, deep learning, and artificial intelligence.

Kyungshin Kim is a Principal Researcher, leading the Computer Security Development Team in the Agency for Defense Development of Korea. He received a PhD at Department of Computer Engineering of Kyunghee Univ in Korea, and holds B.S and M.S in the Electronic Engineering of Kumoh Institute of Technology and in Electronic Engineering from Yonsei University. Now he focuses on Machine Learning for Malware Detection and Security information and event management (SIEM) in Network Security Monitoring Systems.

JaeSeung Song is an associate professor, leading the Software Engineering and Security Lab (SESLab), in the Computer and Information Security Department at Sejong University. He received a PhD at Imperial College London in the Department of Computing, United Kingdom. He holds B.S. and M.S. in Computer Science from Sogang University. His research interests span the areas of software engineering, software testing, networked systems and security, with a focus on the design and engineering of reliable IoT/M2M platforms, particularly in the context of semantic IoT data interoperability, secure software patch techniques, blockchain IoT, and edge computing.