Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 11, No. 2, June 2023, pp. 570~585
ISSN: 2089-3272, DOI: 10.52549/ijeei.v11i2.4454

Malware Detection Approaches Based on Operation Codes (OpCodes) of Executable Programs: A Review

Mohammed A. Saleh[1]
[1]Department of Computer, College of Science and Arts in Ar Rass, Qassim University, Saudi Arabia

Article Info

Article history:

Received Jan 13, 2023
Revised Apr 29, 2023
Accepted Jun 25, 2023

Keywords:

OpCodes-based Malware Detection
Malware Detection Approaches
Static Malware Approaches
Dynamic Malware Approaches
Hybrid Malware Approaches

ABSTRACT

Usually, malware is analyzed in two ways: dynamic malware analysis and static malware analysis. The former collects feature datasets during the run of the malware, and involves malware API system calls, and registry, file, process, and network activities features. The latter collects feature datasets without the run of the malware, and involves OpCodes and text features. Several previous studies have addressed the review of the malware detection approaches based on various feature datasets, but none of them has addressed the review of the approaches merely based on malware OpCodes features. Therefore, this study aimed to review the malware detection approaches only based on OpCodes features and deduced that there is a positive relationship between the Study Year and the Detection Ratio. Besides, incorporating the improved deep learning (DL) in the approaches for detecting malware only based on OpCodes achieved 1.427 times greater accurate detection ratio.

Corresponding Author:

Mohammed A. Saleh,
Department of Computer, College of Science and Arts in Ar Rass,
Qassim University, Saudi Arabia,
Email: m.saleh@qu.edu.sa

1. INTRODUCTION

Computer systems are legitimate programs that play a critical role in today's modern daily life's activities that intuitively need to be protected accurately in order to perform and deliver their functions without any disruption, intervention, corruption, or any manner of undesired activities [1]. On the contrary, malicious software, or malware, are illegitimate programs that execute undesired activities on their victims, which explicitly need to be detected and prevented [2], [3], [4],. However, detecting malware depends on a precise and accurate malware analysis, which firstly is carried out prior to classifying the underlying illegitimate program, and then is detected and prevented. Particularly, malware analysis is divided into three main kinds of analyses, namely static malware analysis, [3], [5], [6], [7]–[15], dynamic malware analysis [6], [11], [16]–[19], and hybrid malware analysis [11], [20]. The advantages and disadvantages of each kind are described subsequently. Besides, each of these three kinds of malware analyses could be conducted automatically or manually (nonautomatically) [7], [10], [12], [21].

Static malware analysis extracts feature datasets of the malware without executing the malware sample [2], [3], [22]–[26], neither in a production environment nor in an experimental, or virtualized, environment. The first advantage of this kind of malware analysis is that it prevents the suspicious program from harming users at the first moment since it is fully scanned and classified before running it. The scanning and classifying process is conducted towards the raw data features of the suspicious program, and in case it passes them successfully, it can be executed, otherwise it is classified as malware and blocked immediately. Hence, this kind of malware analysis guarantees infection-free of the malware during the malware analysis process. The second advantage is that it offers immediate feature datasets that the analyst uses in order to analyze these suspicious programs. The disadvantage is that the advanced tactics of attackers could deceive the classifying and detection process by manipulating bits, assembly instructions, and API system calls of the suspicious program, which yields failing to classify it as a malware; therefore, it fails to detect this malware [27].

Conversely, the dynamic malware analysis extracts feature datasets of the malware after executing it, even for a while and a very short period of time [1], [6], [11], [17]–[20], [28]. The setting of the executed suspicious programs vary among the approaches, and all of them aim to extract and collect suitable and optimal feature datasets, which are then used for malware classification and detection. The setting of the executed suspicious programs involves running time, intervention type with a system, testing environments, etc. The advantage of this kind of malware analysis is that less prone to be decoyed by the advanced tactics of attackers because it is updated continuously to discover such a decoy [29]. The disadvantage causes huge performance overhead [27]. In addition, it rises a partial or complete malware infection in the testing environments, whether it is a production environment or an experimental (virtualized) environment [27], [28]. Furthermore, it is challenging to mimic the proper conditions, such as a vulnerable application that is exploited by the malware. It is also unclear how long the infection needs to be active before its destructive effects can be observed [30].

Hybrid malware analysis intuitively merges the static malware analysis and the dynamic malware analysis together within a single malware analysis approach [11], [20]. Automated malware analyses perform end-to-end malware analysis without any human interventions, while manual, or nonautomated, malware analyses comprises any sort of human intervention during the analysis and classification of the suspicious programs [31]–[33], [7], [9], [10], [16], [21], [34]–[36].

This study intensively reviewed the recent existing approaches that are introduced for detecting malware only based on the operation codes (OpCodes) of the executable programs, since there is a considerable necessity to achieve a comparative and comprehensible analysis of their achieved results [37]. Table 1 illustrates the acronyms list.

Table 1. The list of acronyms

1D-CNN	One-Dimensional Convolutional Neural Network Model	LLGC	Learning with Local and Global Consistency
AI	Artificial Intelligence	LMT	Logistic Model Tree
ANN	Artificial Neural Network	LR	Logistic Regression
API	Application Programming Interface	LSTM	Long Short-Term Memory
BDT	Boosted Decision Trees	MI	Mutual Information
BNB	Branch and Bound	ML	Machine Learning
BP	Back Propagation	N-grams	Continuous Sequences of N Symbols
CFG	Control Flow Graph	NB	Naïve Bayes
CFS	Correlation-Based Feature Selection	NBT	Naïve Bayes Tree
DF	Document Frequency	OpCodes	Operation Codes
DL	Deep Learning	PART	Partial Decision Tree
DT	Decision Tree	PE	Portable Executable
FS	Fisher Score	RF	Random Forest
FT	Forest Tree	STIT	Statistical Techniques and Information Theories
GR	Gain Ratio	SVM	Support Vector Machine
IDA	Interactive Disassembler	TF	Term Frequency
IG	Information Gain	AUC	Area Under the Curve
J48	C4.5 Decision Tree	WTF	Weighted Term Frequency
k-NN	K-Nearest Neighbors		

Among the widely used malware feature datasets, such as API system calls features, registry activities features, file activities features, process activities features, network activities features, operation codes (OpCodes) features, and text features, this study selected operation codes (OpCodes) features. The study chose operation codes (OpCodes) features because the review of the approaches for detecting malware only based on sample OpCodes has not been addressed before, OpCodes features immune against decoying unlike API systems call and text features [38], [39], [40] and shared in the next significant contributions:

1. To the best of our knowledge, this study has made the first attempt to provide a comparison of the approaches for detecting malware only based on sample OpCodes.
2. The study examined the improvements in the malware detection ratio over the year advances by calculating the Pearson Correlation between the "Study Year" variable and the "Detection Ratio" variable.
3. The study investigated the significance of the variables of the approaches for detecting malware only based on sample operation codes (OpCodes) by calculating the Binary Logistic Regression, which assesses the impact of the independent variables, or predictors, on the dichotomous (binary) dependent variables of the model.

The paper is structured as next. First, it broadly defines the malware analyses and states the main contributions of this study. Second, it identifies the criteria for the relevant study materials collection of malware detection approaches, and reviews the literature of the collected studies according to malware detection approaches only based on OpCodes using machine learning (ML) algorithms, deep learning (DL) algorithms, and statistical techniques and information theories (STIT). Third, it discusses and evaluates the malware detection approaches merely based on sample operation codes (opcodes) by calculating descriptive statistics and the relationship between the variables of the approaches for detecting malware only based on sample operation codes (OpCodes). Forth, it summarizes the analysis of the obtained results of the malware detection approaches and motivates recommendations for future research directions accordingly. Finally, it concludes the study.

2. LITERATURE REVIEW

The collection of the relevant study materials is critical for the literature review. In this study, the strategy of collecting and gathering the relevant literature is briefly explained in the succeeding steps.

1. In this study, the most significant information to be collected according to study review theme is identified, which focuses on approaches for detecting malware merely based on sample OpCodes.
2. The study defined the suitable search keyword, namely "an approach for detecting malware based on sample operation codes (OpCodes)".
3. The study searched various research databases, such as Science Direct [41], Web of Science [42], IEEE Xplore Digital Library [43], SpringerLink [44], and Google Scholar [45], using the mentioned keyword. As a result, it obtained 37 studies on the domain of an approach for detecting malware based on sample operation codes (OpCodes).
4. Lastly, the study preliminary reviewed the obtained 37 studies, as shown in Table 2, and categorized them into the following three categories for the sake of reviewing simplicity.
    1. The approach for detecting malware is only based on sample operation codes (OpCodes) using machine learning (ML) algorithms.
    2. The approach for detecting malware is only based on sample operation codes (OpCodes) using deep learning (DL) algorithms.
    3. The approach for detecting malware only based on sample operation codes (OpCodes) using statistical techniques and information theories (STIT).

Threats to validity: This study tackled the studies that encompass the following criteria: (1) propose approaches, methods, and techniques for malware detection, (2) utilize machine learning (ML) algorithms, deep learning (DL) algorithms, and statistical techniques and information theories (STIT), and (3) analyze malware samples only based on the operation code (OpCodes). It involved unlimited ML, DL, and STIT models, unlike in [46]. It identified the most significant information to be collected according to the study review theme, which focuses on the approaches for detecting malware merely based on sample OpCodes. It defined the suitable search keyword namely, "an approach for detecting malware based on sample operation codes (OpCodes)", and searched various research databases such as Science Direct [41], Web of Science [42], IEEE Xplore Digital Library [43], SpringerLink [44], and Google Scholar [45], to collect the peer-reviewed journal articles, book chapters, conference proceedings, and reports using the mentioned keyword. Initially, the study collected 348 documents and screened the title and abstracts to identify suitable articles. According to the previously stated criteria, the study excluded 144 documents that violated criteria (1) and criteria (2). Moreover, it excluded 167 documents that smashed criteria (3). Finally, the full text of 37 studies were selected for the review. The next subsections present the literature review of the acquired 37 studies that are organized according to the three previously stated categories.

2.1. The approach for detecting malware based on sample operation codes (OpCodes) using machine learning (ML) algorithms

Machine learning (ML) is a subfield of artificial intelligence (AI) that allows systems to acquire the ability to learn from experience and get better over time, all without being expressly programmed to do so [21], [31], [7], [47], [48]. Machine learning (ML) comprises of four learning types, namely supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning [49], [14], [22], [50], [51], [52], [53]. In this study, several collected studies have utilized the machine learning (ML) algorithms to detect the malware. In this category, the approach first extracted and selected the appropriate malware features and then passed them to the ML algorithm in order to detect the malware. Roughly, the collected studies have used sample OpCodes frequencies, sample N-grams OpCodes, or sample OpCodes features vectors for malware features extraction and selection, as reviewed and discussed in the next subsections.

2.1.1. Malware detection approach utilizes OpCodes frequencies for features extraction and selection

In this subsection, the malware detection approaches that employ OpCodes frequencies for features extraction and selection are presented. Authors in [54] presented a new approach for detecting advanced unknown malware with a high accuracy. Firstly, it analyzed OpCodes occurrences as features extraction through grouping the executables, which follow the rule: the difference between any malware sizes is within 5

KB. After that, it applied thirteen classifiers and then stated the top five classifiers: RF, LMT, NBT, J48, and FT. It obtained 96.28% accuracy in detecting unknown malware.

Research done by [55] used an iterative approach to determine the suitable behavioral attributes in order to gain better accuracy for classifying and identifying ransomware. It collected 150 sample reports for 10 families of ransomware. Initially, the study selected 27 attributes and then selected 24 attributes from the initial 27 attributes according to their frequencies in the dataset. After that, the iterative approach selected 20 out of 24 and 15 out of 20 attributes based on J48 results. Lastly, it used grouping to select 12 out of 15 attributes. The study verified each attribute's reduction in terms of classification accuracy to ensure identifying optimum attributes. It reduced behavioral attributes to nine attributes, but it gained worse results, so it retrained back to 12 attributes since it gives the best classification results. Finally, it applied J48, NB, and k-NN machine learning (ML) algorithms and achieved 78% of classification accuracy by using J48.

Research done by [50] extracted the OpCodes and then computed the frequency of occurrence of each opcode sequence using Term Frequency (TF). After that, it defined the Weighted Term Frequency (WTF) as the result of weighting the relevance of each OpCode when calculating the term frequency. Finally, it used the LLGC algorithm for classification. It achieved above the 80% of accuracy for merely the 10% of labelled instances.

A study in [8] proposed a method to detect unknown malware, and it consists of four steps. First, it used PE header information to divide sample categories. Then, it computed TF-IDF for each OpCodes sequence in order to choose top-K OpCodes to construct an adjacency matrix, and after that, it applied the Power Iteration algorithm for feature selection. Finally, it trained learning models like kNN and BP to detect unknown malware. The highest obtained accuracy detection of the proposed method is 98.57%, which was achieved by the Adaboost algorithm.

Research conducted by [14] proposed a model that used OpCode Extract and Count (OPEC) algorithm for feature selection, and then applied supervised learning algorithms to detect malware. The model acquired a detection accuracy of 98.7%.

Research introduced by [18] investigated optimal OpCodes set that vigorously points toward malware. It extracted the OpCodes as OpCode density histograms and then used the algorithm for features selection and malware classification, as well, and achieved a detection accuracy of 83.41%.

2.1.2. Malware detection approach utilizes N-grams OpCodes for features extraction and selection

This subsection elaborates on the malware detection approaches that harness N-grams OpCodes for features extraction and selection. Authors in [30] proposed a classification framework to detect unknown malware. First, it extracted 1000 OpCodes patterns as features with the biggest DF values. Then, it applied different methods, namely DF, GR, and FS, for feature selection. After that, it selected top 50, 100, 200, and 300 features based on each feature selection, which measures the correlation between OpCode n-grams feature and malware class. Finally, it applied and evaluated eight machine learning (ML) classifiers like SVM, LR, RF, ANN, DT, NB, BTD, and BNB. It attained more than 96% of accuracy, which is better than previous studies that utilize Byte n-gram patterns.

A study in [56] proposed multiple feature method for detecting malware based on multiple n-value OpCodes N-grams pooled sequences and multiscale grey image texture of malware. First, the method extracted multiple N-value OpCode N-grams combined sequences and selected features from them based on Information Gain (IG). In the meanwhile, it transformed sample files into grey images, generated multi scale images by using a Gaussian pyramid, and extracted features by using GLCM. Finally, it applied k-NN and RF classifiers in order to detect malware. It gained 98.85% detection accuracy.

Research in [27] proposed a model for malware detection using an ensemble approach. It generated multiple features dataset from various sizes of n-grams of OpCodes sequences to train one classifier, namely SVM, RF, or k-NN. First, it extracted n-grams of sizes array from 1 to 4, and then it vectorized them by using TF-IDF. After that, it leveraged the Information Gain (IG) to pick up 1000 maximum instructive features. Finally, it applied a particular classifier, namely SVM, RF, or k-NN to train multi features n_i-gram OpCode and n_j-gram OpCode sequences and subsequently to weight and average them using weight values and argmax() function in order to predict a final class, a benign or malware. It obtained the finest classification accuracy of 98.1%.

Research introduced by [57] proposed an early malware detection framework. It consists of three stages. The first stage is an evasive behavioral data collection stage, which collected a representative dataset according to a pre-identified list of evasive techniques for malware. The second stage extracted features based on n-gram and TF-IDF techniques and calculated correlation values between API user mode and kernel system calls mode in order to pick up the most representative features. Finally, the third stage applied an ensemble model based on Random Forest (RF) machine learning (ML) algorithm on the extracted and selected features to detect malware.

Research in [58] presented a method for malware detection based on subgraph isomorphism using blocks of OpCodes. The method first analyzed and investigated the frequencies of n-grams OpCodes to detect singular code blocks through TF/IDF, and then it used machine learning (ML) algorithms such as RF, XGBoost, DT, SVM, and KNN for learning. Finally, OpCodes sequences are transformed into a Control Flow Graph (CFG) in order to feed the database of CFGs characteristic of malware, which is used for comparing semantic and construction of known and unknown malware in order to detect and classify it. The RF algorithm achieved the finest F1 score: 0.923 for 1-grams and 0.796 for 9-grams.

A study in [59] introduced a detective mechanism based on OpCodes sequences features. First, it collected all possible k-grams for feature extraction and then applied the Information Gain (IG) selection algorithm in order to select the top representative features. Finally, it created a model and classified the unknown malware using the SVM algorithm. It gained 96.83% for malware detection accuracy.

Research by [60] proposed a method for detecting malware, which is based on Control Flow Graph (CFG) in order to extract OpCodes behaviors. It converted a CFG into a tree to form an execution tree, and the trees are concatenated to present a long execution path. Then, it used n-grams with IG and DF to select OpCode-based features. Finally, it employed KNN, DT, and SVM to classify executables. The best achieved accuracy result is 93.2% for CFG-DT.

Research in [17] proposed a new scheme for dynamic OpCode acquisition through QEMU binary translation mechanism. The OpCodes information is obtained from the software runtime and is used for offline analysis. The scheme used a variety of feature selection algorithms, CFS, Chi-square, IG, Symmetrical, and N-gram algorithms to extract features of the operating code information when the software is running. Then, the extracted feature subset is combined with a variety of machine learning (ML) algorithms like DT, SVM, Bayesian network, ensemble and NN algorithms to conduct cross-comparison experiments. The detection accuracy of offline malware reaches 99.85%. As well, the research proposed an online detection scheme based on the above research results called CPU built-in malware monitoring model (CBMM), which accurately identified the execution trajectory of malware under the current process, and monitored malware in real-time.

Research accomplished by [61] designed a method which applied SVM and RF classifiers to the greatest values of frequencies of OpCodes n-grams in order to detect malware and its multi families, as well. The method obtained a detection accuracy of 97%.

A study in [32] proposed a new feature which performed OpCodes n-gram shingling with control statements as stopwords while requiring a smaller feature vector and shorter training time. Random Forest (RF) algorithm is implemented for both learning the classification and achieving 99.11% of accuracy in malware detection.

Research established by [22] proposed a new method that used only single class learner to detect unknown malware. The method is proposed based on examining the frequencies of the appearance of OpCodes sequent. It used TF-IDF to weigh each OpCodes n-grams sequences, suggested labelling only malware samples, and employed the Roc-SVM algorithm for malware detection. It obtained 85% of malware detection accuracy.

Research talented by [33] used n-gram OpCodes and then applied a data segmentation technique for feature selection. Finally, it applied ML algorithms like Naïve Bayes (NB), support vector machine (SVM), partial decision tree (PART) and random forest (RF). It gained f-measure of 98% for malware detection.

A study introduced by [31] used n-OpCode up to 10-grams and then selected the most important features based on IG. Finally, it applied ML algorithms, like Naïve Bayes (NB), support vector machine (SVM), partial decision tree (PART) and random forest (RF) to classify and category malware. It obtained f-measure of 98% for malware detection.

Research accomplished by [13] extracted OpCodes and converted them into a vocabulary dataset, and then applied n-gram on each word to represent a feature. After that, it uses TF-IDF to measure the significance of every word in order to extract significant features. Finally, the obtained data set is processed with CPD to gain a feature-reduced dataset, which is then evaluated using Weka (6 DM algorithms: Ripper (JRip), C4.5 Decision Tree (J48), Support Vector Machines (SMO), and Naive Bayes (NB). The largest attained malware detection is 0.949 AUC score, which is achieved by the k-NN algorithm.

A study in [62] proposed a technique to extract the behavior of OpCodes based on Control Flow Graph (CFG), jointly with 4-gram of OpCodes sequence. After that, the technique used the k-NN algorithm to detect Trojan Ransomware, and it achieved a detection accuracy of 98.86% when k=1 (1-KK) and n=1 (1-gram) OpCodes.

Research established by [34] obtained OpCodes, then used n-gram and TD-IDF to represent terms and sequences of disassembled instructions as vectors. Finally, it applied six classifiers, namely RF, NB, LR, kNN, Linear SVM, and XGBoost. The best achieved F1 accuracy of 86% by using RF algorithm.

2.1.3. Malware detection approach utilizes OpCodes features vectors for features extraction and selection

The malware detection approaches that use OpCodes features vectors for features extraction and selection are presented in this subsection. Research conducted by [63] attempted to detect IoT-based malware. First, it extracted OpCodes from IoT-based devices and services and then preprocessed them through filtering, which involves normalizing, centering, and scaling. Finally, it applied three ML algorithms, RF, SVM, and k-NN. RF achieved the best accuracy at 98%, followed by SVM and k-NN, both with 91%.

Research presented by [26] proposed a malware detection method for OpCodes and API calls extraction in order to form a feature vector, which eventually applied NB and kNN classifiers in order to detect the malware. The proposed method acquired 95.21% of malware detection accuracy.

Research done [16] created a procedure based on learning to discriminate and classify in the Internet of Battlefield Things (IoBT) using OpCodes progression. The procedure transformed the OpCodes into a vector space and then applied a technique called Deep Eigen space learning to distinguish between malware and benign software. In addition, the procedure utilized the SVM algorithm and n-gram algorithm for robust classification.

2.2. The approach for detecting malware based on sample operation codes (OpCodes) using deep learning (DL) algorithms

Deep learning (DL) is a subfield of machine learning (ML) that imitates the structure of the human brain neural network (NN) so that the computer can act autonomously in response to unseen events. DL aids a computer model in predicting and classifying information by filtering it through layers of data [9] ,[21], [64], [48]. A number of collected studies have utilized deep learning (DL) algorithms to detect malware, as discussed in the following subsections according to whether the malware features are extracted and selected based on OpCodes frequencies, N-grams OpCodes, embedding, or images.

2.2.1. Malware detection approach utilizes OpCodes frequencies for features extraction and selection

In this subsection, the malware detection approaches that employ OpCodes frequencies for features extraction and selection are presented. Research [65] proposed a system for detecting malware based on 1D-CNN. The system took a binary file as an input and then classified it to whether malware or benign. In the meanwhile, the researchers classified the binary file into malware or benign using the TF-IDF algorithm [66] and used it as a benchmark in order to compare it with the 1D-CNN classifier. The overall accuracy of the system for detecting malware is 99.2%.

Research established by [20] proposed a hybrid solution for detecting malware. It adapted OpCode sequences as static features and network traffic as dynamic features in order to detect malware. The proposed hybrid solution achieved malware detection accuracy of 97%.

2.2.2. Malware detection approach utilizes N-grams OpCodes for features extraction and selection

This subsection demonstrates the malware detection approaches that exploit N-grams OpCodes for features extraction and selection. Authors in [2] introduced a method based on a dual branch convolutional neural network (CNN) to determinate and classify malware using multiple features fusion which consists of local fine-grained and global structure features of the visualized malware. The proposed method converted malware global structural information into a bytecode image and then extracted the OpCode semantic information of the code segment by using the n-gram feature model to produce an OpCode image. The method attained a family classification accuracy of 99.05%.

Research in [7] proposed an end-to-end model based on ID CNN to determine binary file maliciousness. First, the model extracted n-grams of OpCodes automatically. Then, the model is trained on multiple feature sets, e.g. 1-garms and 2-grams, and sequentially combined these two predictions using a weighted average ensemble. The proposed model utilized a grid search on values (0-1) for optimal prediction weights. The model attained a positive prediction of 98% using a weight parity of 0.5 for ensemble unigram and bigram OpCodes sequences.

Research conducted by [35] introduces a new classifier called SNNMAC, which is a model for classifying malware based on shallow neural networks and static analysis. First, the model extracted n-gram OpCodes sequences from a binary file using a decompiler. Then, the n-gram dataset is decreased according to the designed enhanced n-gram algorithm. Finally, the SNNMAC classifier learned from the dataset to classify the malware. The classifier attained malware classification accuracy of 99.21%.

2.2.3. Malware detection approach utilizes OpCodes embedding for features extraction and selection

In this subsection, the malware detection approaches that exploit OpCodes embedding for features extraction and selection are discussed. Research talented by [10] proposed a novel system based on deep CNN for detecting Android malware. The proposed system extracted a raw OpCodes sequence and then performed training using a pipeline technique; thus, it eliminated the need for a lot of n-grams sequences enumeration and manually engineered malware features. Therefore, it yielded better performance than n-grams based systems but less malware accuracy detection of 69%.

Research accomplished by [36] presented a malware detection system based on optimized deep CNN. It went through the embedding layer and then used the k-max pooling method to detect the malware. It gained malware accuracy detection of 99%.

Research [12] proposed a novel approach which modeled malware as a language to detect malware. It collected OpCodes by using IDA Pro software, then used word embedding technique for feature vector, and finally applied two-stage LSTM. It reached an average AUC of 98.7% for malware classification.

Research in [67] introduced a system for detecting malware based on a deep optimized deep neural network. The pipeline of the proposed detection system comprised three consecutive layers, namely the embedding layer, convolutional layer, and k-max pooling layer. The proposed system extracted OpCodes sequences from a binary file and fed them to the optimized deep neural network. It demonstrated malware detection accuracy of 99%.

Research introduced by [68] presented a method based on stacked LSTM to circumvent the time-consuming drawback of random weight initialization for neural networks (NN). The proposed method used six distinct malware datasets to extract various malware feature datasets like OpCodes, Bytecodes, and API Systems Calls. The method incorporated a model with four hidden layers; the first three of them are pre-trained layers, while the fourth layer is a dense layer as a classifier. The suggested method entailed two phases: unsupervised pre-training on training data to determine the initial weights and supervised fine-tuning of the network to distinguish between malware and benign samples. The extracted feature datasets are converted into embedding vector for OpCodes and System Calls, and one-hot vector for Bytecodes, and then are passed to the model for classification purposes to detect malware. The method achieved IoT malware detection accuracy of 99.1%.

2.2.4. Malware detection approach utilizes images for features extraction and selection

This subsection debates the malware detection approaches that exploit N-grams OpCodes for features extraction and selection. Research in [21] proposed a method called MalNet which learned features automatically from raw data. It generated grayscale images and OpCodes sequence to be used for CNN and LSTM networks, respectively and took a stacking ensemble for malware classification. The proposed method gained malware detection accuracy of 99.36%.

Research conducted by [69] utilized a technique of image similarity based on the CNN approach to detect malware. It converted the executable (EXE) files into images and then applied CNN for classification. Subsequently, it converted the executable (EXE) files to OpCodes, then to images, and finally applied CNN for classification. Finally, it compared the previous two classifications. It achieved malware accuracy detection of 97.6%.

Research established by [9] presented a new approach based on deep learning and function call graph (FCG) in order to detect and classify malware. First, it produced OpCodes based on FCG and then transformed them into vector. Finally, it applied Long Short-Term Memory (LSTM) algorithm for malware classification. It attained malware accuracy detection of 97%.

2.3. The approach for detecting malware based on sample operation codes (OpCodes) using statistical techniques and information theories (STIT).

This subsection elaborates the approaches for detecting malware based on sample operational codes (OpCodes) using statistical techniques and information theories (STIT). Mutual information (MI) is a metric used in probability and information theory to quantify the degree to which one variable can be inferred from another. Research in [24] proposed a new method based on the frequency of appearance of OpCodes sequences to detect variants of malware throughout Mutual Information measure: I (x ; Y). The method achieved variant family similarity detection. It conquered malware accuracy detection of 90%.

3. EVALUATION AND DISCUSSION OF THE MALWARE DETECTION APPROACHES BASED ON SAMPLE OPERATION CODES (OPCODES)

This section evaluates, analyzes, and discusses the obtained results of the approaches for detecting malware that were reported by authors to evaluate their performance. First, it presents the descriptive statistics on the approaches for detecting malware only based on sample operation codes (OpCodes). Then, it explains the relationship between the variables of the approaches for detecting malware based on sample operation codes (OpCodes).

3.1. Descriptive statistics on the approaches for detecting malware based on sample operation codes (OpCodes)

As shown in Table 2, 25 studies out of the 37 collected studies of the approaches for detecting malware only based on OpCodes were using machine learning (ML) algorithms, which acted 67.57% of the overall studies. Therefore, this category took the majority. Besides, 11 studies out of the 37 collected studies of approaches for detecting malware merely based on OpCodes were using deep learning (DL) algorithms, which represented 29.73% of the whole studies, and this category came second. Lastly, 1 studies out of the 37 collected studies of approaches for detecting malware only based on OpCodes was utilizing statistical techniques and information theories (STIT), which denoted 2.70% of the total studies.

After extensive literature reviews, this study found that the approaches for detecting malware based on OpCodes that used machine learning (ML) algorithms have conquered the first rank due to their simple construction, easy implementation, fast computation speed, and low calculation overheads. On the contrary, they did not support an end-to-end malware detection process, which enforced the malware detector to conduct some steps of the whole malware detection process manually. In addition, the approaches for detecting malware based on OpCodes that utilized deep learning (DL) algorithms have occupied the second rank due to their complexity for implementation, low computation speed, and huge calculation overheads, despite they support end-to-end malware detection process. Therefore, the latter approaches have outperformed the former approaches in terms of full automation from end-to-end for the malware detection process. The approaches for detecting malware based on OpCodes that used statistical techniques and information theories (STIT) have been subjugated after all since they did not provide any sort of intelligence [53].

Table 2. A Comparison of the approaches for detecting malware based on sample operation codes (OpCodes)

No.	Study Reference	Study Year	Malware Detection Approaches based on			Analysis			Detection		Detection Ratio %
			ML	DL	STIT	Dynamic	Static	Hybrid	Auto matic	Manual	
1	[54]	2016	√	X	X	X	√	X	X	√	96.28
2	[30]	2012	√	X	X	X	√	X	X	√	96
3	[56]	2021	√	X	X	X	√	X	X	√	98.85
4	[55]	2018	√	X	X	√	X	X	X	√	78
5	[27]	2021	√	X	X	X	√	X	X	√	98.1
6	[57]	2021	√	X	X	√	X	X	X	√	-
7	[63]	2020	√	X	X	X	√	X	X	√	91
8	[58]	2021	√	X	X	X	√	X	X	√	92.3
9	[59]	2014	√	X	X	X	√	X	X	√	96.83
10	[60]	2014	√	X	X	X	√	X	X	√	93.2
11	[17]	2020	√	X	X	√	X	X	X	√	99.85
12	[50]	2011	√	X	X	X	√	X	X	√	80
13	[61]	2015	√	X	X	X	√	X	X	√	97
14	[18]	2016	√	X	X	X	√	X	X	√	83.41
15	[32]	2018	√	X	X	X	√	X	X	√	99.11
16	[22]	2011	√	X	X	X	√	X	X	√	85
17	[26]	2017	√	X	X	X	√	X	X	√	95.21
18	[24]	2010	X	X	√	X	√	X	X	√	90
19	[33]	2016	√	X	X	X	√	X	X	√	98
20	[31]	2016	√	X	X	X	√	X	X	√	98
21	[13]	2011	√	X	X	X	√	X	X	√	94.9
22	[8]	2019	√	X	X	X	√	X	X	√	98.57
23	[14]	2021	√	X	X	X	√	X	X	√	98.7
24	[62]	2021	√	X	X	X	√	X	X	√	98.7
25	[16]	2020	√	X	X	X	√	X	X	√	-
26	[34]	2019	√	X	X	X	√	X	X	√	86
27	[2]	2021	X	√	X	X	√	X	X	√	99.05
28	[7]	2022	X	√	X	√	X	X	√	X	98
29	[10]	2017	X	√	X	X	√	X	X	√	69
30	[36]	2020	X	√	X	√	X	X	X	√	99
31	[69]	2018	X	√	X	X	√	X	X	√	97.6
32	[9]	2020	X	√	X	X	√	X	X	√	97
33	[12]	2019	X	√	X	X	√	X	√	X	98.7
34	[21]	2018	X	√	X	X	√	X	√	X	99.36

No.	Study Reference	Study Year	Malware Detection Approaches based on			Analysis			Detection		Detection Ratio %
			ML	DL	STIT	Dynamic	Static	Hybrid	Auto matic	Manual	
35	[35]	2020	X	√	X	X	√	X	X	√	99.21
36	[65]	2019	X	√	X	X	√	X	X	√	99.2
37	[20]	2021	X	√	X	√	√	√	X	√	97
Total	-	-	25	11	1	-	-	-	-	-	-

As presented in Table 3, the approaches have utilized OpCodes frequencies for features extraction and selection represented 24% of the collected studies that used machine learning (ML) for malware detection. Besides, the approaches have employed N-grams OpCodes for features extraction and selection acted 64% of the collected studies that use machine learning (ML) for malware detection. Lastly, the approaches have used OpCodes features vectors for features extraction and selection appeared in 12% of the collected studies that use machine learning (ML) for malware detection. Figure 1 shows the percentage of each one.

Table 3. OpCodes features extraction and selection in malware detection approaches based on ML algorithms

	OpCodes frequencies	N-grams OpCodes	OpCodes features vectors	Total
No. of studies	6	16	3	25
The percentage	24%	64%	12%	100

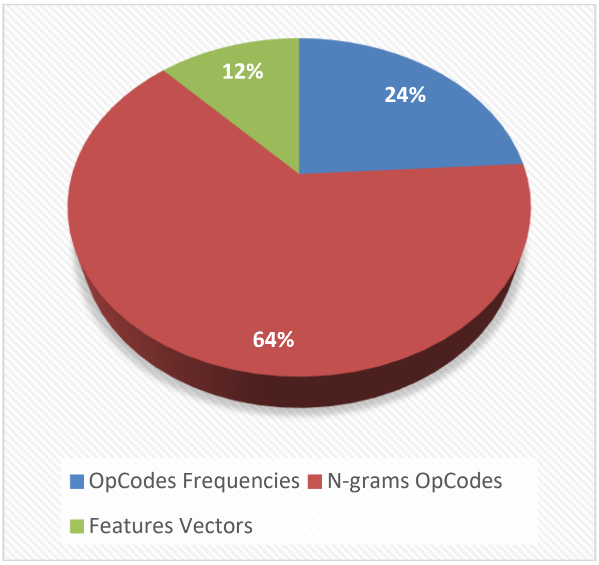

Figure 1. OpCodes features extraction and selection in malware detection approaches based on ML

Similarly, as shown in Table 4, the approaches have employed OpCodes frequencies for features extraction and selection acted 18.18% of the collected studies that use deep learning (DL) for malware detection. In addition, the approaches have utilized N-grams OpCodes for features extraction and selection equaled 27.27% of the collected studies that use deep learning (DL) for malware detection. Furthermore, the approaches have exploited OpCodes embedding for features extraction and selection denoted 27.27%. Finally, the approaches have utilized images for features extraction and selection appear in 27.27% of the collected studies that used deep learning (DL) for malware detection. Figure 2 presents the percentage of each one.

Table 4. OpCodes features extraction and selection in malware detection approaches based on DL algorithms

	OpCodes frequencies	N-grams OpCodes	OpCodes embedding	images	Total
No. of studies	2	3	3	3	11
The percentage	18.18	27.27	27.27	27.27	100

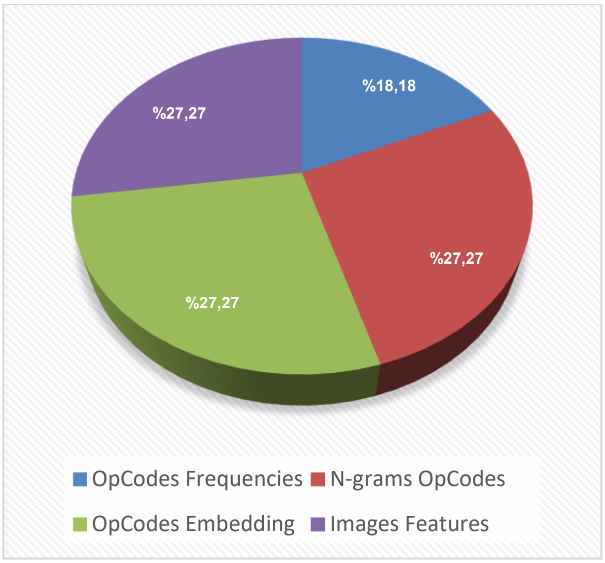

Figure 2. OpCodes features extraction and selection in malware detection approaches based on DL

Likewise, as displayed in Table 5, the approaches have taken advantage of the mutual information (MI) for features extraction and selection act represented 100% of the collected studies that use statistical techniques and information theories (STIT), as presented in Figure 3.

Table 5. OpCodes features extraction and selection in malware detection approaches based on STIT

	Mutual information (MI)	Total
No. of studies	1	1
The percentage	100	100

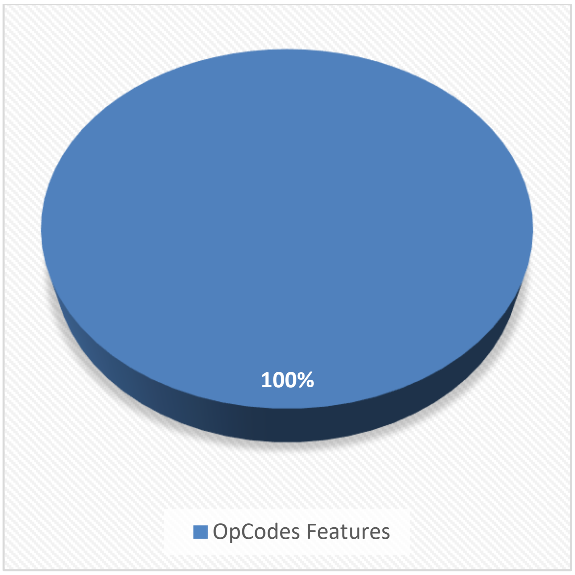

Figure 3. OpCodes features extraction and selection in malware detection approaches based on STIT

Finally, Figure 4 illustrates the average detection ratio of the approaches for detecting malware, which is calculated by dividing the total of the entire approaches detection ratios by the number of the approaches in Table 2. It equaled 86.12% for the collected studies that use machine learning (ML), 95.74% for the collected studies that employ deep learning (DL), and 90% for the collected studies that exploit statistical techniques and information theories (STIT).

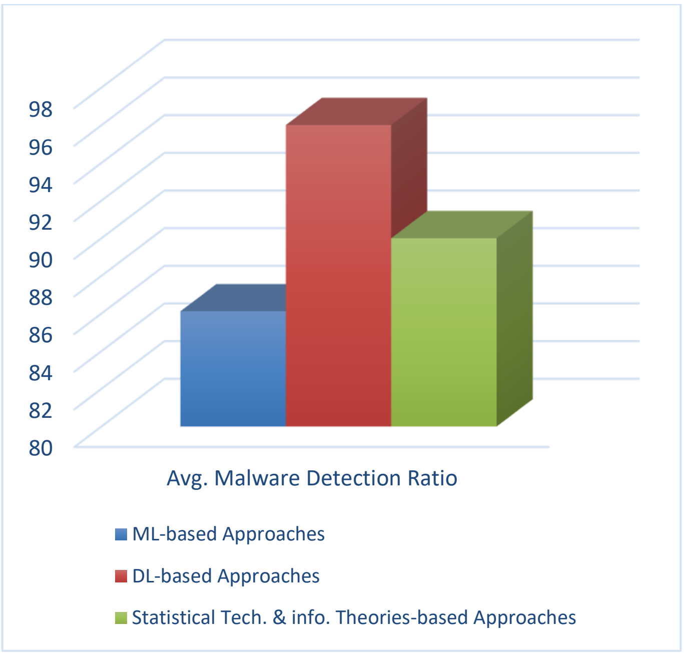

Figure 4. The average detection ratio of the approaches for malware detection

3.2. The relationship between the variables of the approaches for detecting malware based on sample operation codes (OpCodes)

First, the Pearson Correlation is calculated to measure the strength of a linear relationship between the Study Year variable and the Detection Ratio variable. The Study Year is the independent variable, while the Detection Ratio is the dependent variable, and their values are presented in Table 2. The Pearson Correlation between the Study Year variable and Detection Ratio variable is calculated according to equation (1), and it equaled 0.370, which indicates that there is a low positive correlation. This result of the correlation proved that when years advances rise, the detection ratio also rises, which means that the detection ratio of the approaches for detecting malware only based on sample operation codes (OpCodes) has been improved over years advances. Besides, the p-value equaled 0.029, which indicated that the Pearson Correlation was statistically significant.

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(1)}$$

Second, the Binary Logistic Regression model is used to assess the impact of the independent variables, or predictors, on the binary dependent variables, or outcomes that take only two values, 0 or 1. As shown in Table 2, the Study Year and Detection Ratio are the independent variables, or predictors, while the ML, DL, STIT, Dynamic, Static, Hybrid, Automatic, and Manual are the dichotomous (binary) dependent variables.

As presented in Table 6, the overall Binary Logistic Regression model was statistically significant for DL, STIT, and Dynamic dichotomous (binary) dependent variables, since their p-values in the "Model Sig." column are less than 0.05. The other five dichotomous (binary) dependent variables ML, Static, Hybrid, Automatic, and Manual, with their p-values greater than 0.05, were not significant. In addition, the Binary Logistic Regression model correctly detected 64.9%, 100%, and 86.5% cases of DL, STIT, and Dynamic dichotomous dependent variables, respectively as in the Accuracy column. Besides, the statistical significance of each predictor, namely the Study Year and Detection Ratio, is illustrated in the "Indept. Var. Sig." column, which showed that only the Study Year added statistical significance to the model since its p-value is less than 0.05, while all the others with their p-values greater than 0.05 did not add the statistical significance. Finally, the odds of using deep learning (DL) algorithms in the approaches for detecting malware based on sample operation codes (OpCodes) was 1.427 times greater over years advances, as shown in Exp (B) column. This merit indicated that adapting the improved deep learning (DL) over the years advances in the approaches for detecting malware based on sample operation codes (OpCodes) fed a more accurate detection ratio for the malware.

Table 6. The Binary Logistic Regression between the independent variables and dependent variables

Dichotomous (Binary) Dep. Var.	$\aleph^2$ (df)	Model Sig.	Accuracy	Predictor	B	Indept. Var. Sig.	Exp (B)
ML	$\aleph^2 (2) = 3.897$	0.143	70.3	Study Year	-0.164	0.197	0.848
				Detection Ratio	-0.031	0.334	0.969
DL	$\aleph^2 (2) = 8.085$	0.018	64.9	Study Year	0.355	0.047	1.427
				Detection Ratio	0.032	0.315	1.033
STIT	$\aleph^2 (2) = 9.195$	0.010	100	Study Year	-29.449	0.987	0.000
				Detection Ratio	0.112	1.000	1.119
Dynamic	$\aleph^2 (2) = 7.511$	0.023	86.5	Study Year	0.628	0.083	1.874
				Detection Ratio	-0.013	0.421	0.987
Static	$\aleph^2 (2) = 5.798$	0.055	89.2	Study Year	-.510	0.139	0.601
				Detection Ratio	0.016	0.309	1.016
Hybrid	$\aleph^2 (2) = 2.246$	0.325	97.3	Study Year	1.197	0.373	3.312
				Detection Ratio	0.030	0.809	1.030
Automatic	$\aleph^2 (2) = 4.063$	0.131	91.9	Study Year	0.162	0.666	1.175
				Detection Ratio	0.610	0.396	1.841
Manual	$\aleph^2 (2) = 4.063$	0.131	91.9	Study Year	-0.162	0.666	0.851
				Detection Ratio	-0.610	0.396	0.543

4. RECOMMENDATIONS AND FUTURE DIRECTIONS

This study conducted a comprehensive review of the approaches for detecting malware only based on sample operation codes (OpCodes) and drew useful insights towards them. As mentioned earlier, this study focused on the malware OpCodes features and dropped the other malware features like API system calls features such in [5][38][39][40][70] and text features such as in [38][39][40][71][72] due to their limitations, since the former could be decoyed when the evader uses his own developed OpCodes instructions written from the ground up instead of uses of the formal API system calls. As well, it dropped the latter because the garbag of text that could be injected into the malware, which evades detection, too. The following section discussed and summarized the analysis of the obtained results and recommended future directions:

1. There was a positive relationship, which equaled 0.370, between the "Study Year" variable and "Detection Ratio" variable that proved when the years advances rise, the detection ratio also rises, which meant that the detection ratio of the approaches for detecting malware only based on sample operation codes (OpCodes) has been improved over years advances.
2. Adapting the improved deep learning (DL) over the years advances in the approaches for detecting malware only based on sample operation codes (OpCodes) provided 1.427 times greater accurate detection ratio for the malware over years advances. Therefore, this study recommends utilizing improved deep learning (DL) algorithms and incorporating them into the approaches for detecting malware in future works.
3. The average detection ratio of the approaches for detecting malware equaled 86.12% for the collected studies that used machine learning (ML), 95.74% for the collected studies that employed deep learning (DL), and 90% for the collected studies that exploited statistical techniques and information theories (STIT).
4. The collected studies of the approaches for detecting malware only based on OpCodes that used machine learning (ML) algorithms acted 67.57% of the overall studies; therefore, this category took the majority.
5. The collected studies of the approaches for detecting malware only based on OpCodes that used deep learning (DL) algorithms represented 29.73% of the overall studies; hence, this category came second.
6. The collected studies of the approaches for detecting malware only based on OpCodes that used statistical techniques and information theories (STIT) acted 2.70% of the overall studies.
7. The approaches for detecting malware that have utilized OpCodes frequencies for features extraction and selection represented 24% and 18.18% of the collected studies that used machine learning (ML) for malware detection and use deep learning (DL), respectively.
8. The approaches for detecting malware that have utilized employed N-grams OpCodes for features extraction and selection represented 64% and 27.27% of the collected studies that used machine learning (ML) for malware detection and use deep learning (DL), respectively.
9. The approaches for detecting malware that have used vectors of features for features extraction and selection appeared in 12% of the collected studies that used machine learning (ML).
10. The approaches for detecting malware that have exploited OpCodes embedding and images for features extraction and selection denoted 27.27% and 27.27 of the collected studies that used deep learning (DL) for malware detection, respectively.
11. The approaches for detecting malware that have taken advantage of the mutual information (MI) for features extraction and selection act represent 100% of the collected studies that used statistical techniques and information theories (STIT).

12. The most spread approaches for detecting malware were using machine learning (ML) algorithms. It is due to their simple construction, easy implementation, cost-effective performance, and rapid computation. In contrast, most of them extracted malware feature datasets manually, which caused a negative impact on the overall malware classification and detection. Accordingly, this study recommends improving the approaches that were using machine learning (ML) algorithms to extract malware feature datasets automatically so that they help to avoid human intervention and boost malware detection.
13. Moreover, the most spread approaches for detecting malware were using machine learning (ML) algorithms that extracted and selected malware feature datasets statically, not dynamically, which lacked this significant malware feature datasets source. Therefore, this study recommends carrying out several extra studies for improving dynamic feature datasets extraction and selection.
14. There were quite infrequent proposed approaches for malware detection that integrated and incorporated together machine learning (ML) algorithms and deep learning (DL) algorithms within one approach, despite each one has novel advantages. Hence, this study recommends bridging this gap by proposing innovative and improved approaches that utilize both learning algorithms, whether are machine learning (ML) algorithms or deep learning (DL) algorithms.
15. As presented in Table 2, the reported detection ratio results of the reviewed studies still need to be enhanced so that the approach provides a higher detection ratio. Therefore, this study recommends improving the malware detection ratio.
16. The open issues of the introduced malware detection approaches based on OpCodes of the collected studies vary among improving detection accuracy, reducing features vector dimension, integrating and incorporating static and dynamic analysis, adapting automatic malware detection, and promoting end-to-end malware detective solutions.

5. CONCLUSION

Malicious software, or malware for short, poses a threat to computer systems, which need to be analyzed, detected, and eliminated. Malware analysis typically takes one of two forms: dynamic malware analysis and static malware analysis. The former includes malware APIs, registry activities, file activities, process activities, and network activities as features collected in a dataset while the malware is being executed. The latter entails gathering a dataset of properties, including Operational Codes (OpCodes) and text, without running the malware itself. Several prior studies, on the other hand, addressed and reviewed malware detection approaches based on numerous features, but none of them has addressed and analyzed approaches based only on malware OpCodes. As a result, the goal of this article is to review malware detection approaches only based on malware OpCodes. The review explored, demonstrated, and compared the existing approaches for detecting malware based solely on their OpCodes and eventually provided a comprehensive comparative perspective on them.

This study bridged the gap between the approaches for malware detection, and OpCodes feature datasets. In addition, this study found that there was a positive relationship between the Study Year variable and "Detection Ratio variable, which meant that the detection ratio of the approaches for detecting malware only based on sample operation codes (OpCodes) has been improved over years advances. The average detection ratio of the approaches for detecting malware equaled 86.12% for the collected studies that used machine learning (ML), 95.74% for the collected studies that employed deep learning (DL), and 90% for the collected studies that exploited statistical techniques and information theories (STIT). Adapting the improved deep learning (DL) over the years advances in the approaches for detecting malware only based on sample operation codes (OpCodes) provided 1.427 times greater accurate detection ratio for the malware over years advances. Besides, this study found that 67.57% of the entire collected studies were the approaches for detecting malware only based on OpCodes that used machine learning (ML) algorithms. As well, it deduced that 29.73% of the overall studies were the approaches for detecting malware only based on OpCodes that used deep learning (DL) algorithms, and 2.70% of the whole studies were the approaches for detecting malware only based on OpCodes that used statistical techniques and information theories (STIT). Finally, the study ended with insightful recommendations for future research directions.

REFERENCES

[1] K. Shaukat, T. M. Alam, I. A. Hameed, W. A. Khan, N. Abbas, and S. Luo, "A Review on Security Challenges in Internet of Things (IoT)," *2021 26th International Conference on Automation and Computing: System Intelligence through Automation and Computing, ICAC 2021*, no. July, 2021, doi: 10.23919/ICAC50006.2021.9594183.

[2] X. Zhu, J. Huang, B. Wang, and C. Qi, "Malware homology determination using visualized images and feature fusion," *PeerJ Comput Sci*, vol. 7, pp. 1–22, 2021, doi: 10.7717/peerj-cs.494.

[3] Z. Chen, X. Zhang, and S. Kim, "A Learning-based Static Malware Detection System with Integrated Feature," 2021, doi: 10.32604/iasc.2021.016933.

[4] O. 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] K. Iwamoto and K. Wasaki, "Malware classification based on extracted API sequences using static analysis," *Proceedings of the Asian Internet Engineeering Conference on - AINTEC '12*, no. June, pp. 31–38, 2012, doi: 10.1145/2402599.2402604.

[6] G. Canfora, F. Mercaldo, and C. A. Visaggio, "Mobile malware detection using op-code frequency histograms," *SECRYPT 2015 - 12th International Conference on Security and Cryptography, Proceedings; Part of 12th International Joint Conference on e-Business and Telecommunications, ICETE 2015*, pp. 27–38, 2015, doi: 10.5220/0005537800270038.

[7] P. N. Yeboah and H. B. Baz Musah, "NLP Technique for Malware Detection Using 1D CNN Fusion Model," *Security and Communication Networks*, vol. 2022, 2022, doi: 10.1155/2022/2957203.

[8] Z. Sun *et al.*, "An OpCODE sequences analysis method for unknown malware detection," *ACM International Conference Proceeding Series*, vol. Part F1482, pp. 15–19, 2019, doi: 10.1145/3318236.3318255.

[9] W. Niu, R. Cao, X. Zhang, K. Ding, K. Zhang, and T. Li, "Opcode-level function call graph based android malware classification using deep learning," *Sensors (Switzerland)*, vol. 20, no. 13, pp. 1–23, 2020, doi: 10.3390/s20133645.

[10] N. McLaughlin *et al.*, "Deep android malware detection," *CODASPY 2017 - Proceedings of the 7th ACM Conference on Data and Application Security and Privacy*, pp. 301–308, 2017, doi: 10.1145/3029806.3029823.

[11] M. Mofe, A. L. Rwajah, and R. Rastogi, "Malware Materials Detection by Clustering the Sequence using Hidden Markov Model," *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 10, pp. 1227–1237, 2021, doi: 10.17762/turcomat.v12i10.4316.

[12] R. Lu, "Malware Detection with LSTM using Opcode Language," 2019.

[13] R. K. Shahzad, N. Lavesson, and H. Johnson, "Accurate adware detection using opcode sequence extraction," *Proceedings of the 2011 6th International Conference on Availability, Reliability and Security, ARES 2011*, pp. 189–195, 2011, doi: 10.1109/ARES.2011.35.

[14] O. P. Samantray and S. N. Tripathy, "An opcode-based malware detection model using supervised learning algorithms," *International Journal of Information Security and Privacy*, vol. 15, no. 4, pp. 18–30, 2021, doi: 10.4018/IJISP.2021100102.

[15] V. Sihag, A. Mitharwal, M. Vardhan, and P. Singh, "Opcode n-gram based malware classification in android," *Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability, WS4 2020*, no. May 2021, pp. 645–650, 2020, doi: 10.1109/WorldS450073.2020.9210386.

[16] B. Panduri, M. Vummenthala, S. Jonnalagadda, G. Ashwini, N. Nagamani, and A. Akhila, "Dynamics and an efficient malware detection system using opcode sequence graph generation and ml algorithm," *E3S Web of Conferences*, vol. 184, pp. 3–5, 2020, doi: 10.1051/e3sconf/202018401009.

[17] J. Zhang and Y. Wen, "Malware Detection Based on Opcode Dynamic Analysis," *ICST Transactions on Security and Safety*, vol. 7, no. 26, p. 170239, 2020, doi: 10.4108/eai.22-6-2021.170239.

[18] P. O'kane, S. Sezer, and K. McLaughlin, "Detecting obfuscated malware using reduced opcode set and optimised runtime trace," *Secur Inform*, vol. 5, no. 1, 2016, doi: 10.1186/s13388-016-0027-2.

[19] B. Kinholkar, "Study of Dataset Feature Filtering of OpCode for Malware Detection Using SVM Training Phase," *International Journal of Science and Research (IJSR)*, vol. 4, no. 12, pp. 474–479, 2015, doi: 10.21275/v4i12.nov151981.

[20] M. R. Norouzian, P. Xu, C. Eckert, and A. Zarras, "Hybroid: Toward Android Malware Detection and Categorization with Program Code and Network Traffic," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13118 LNCS, pp. 259–278, 2021, doi: 10.1007/978-3-030-91356-4_14.

[21] J. Yan, Y. Qi, and Q. Rao, "Detecting Malware with an Ensemble Method Based on Deep Neural Network," *Security and Communication Networks*, vol. 2018, 2018, doi: 10.1155/2018/7247095.

[22] I. Santos, F. Brezo, B. Sanz, C. Laorden, and P. G. Bringas, "Using opcode sequences in single-class learning to detect unknown malware," *IET Inf Secur*, vol. 5, no. 4, pp. 220–227, 2011, doi: 10.1049/iet-ifs.2010.0180.

[23] A. A. Khare, "MALWARE DETECTION IN INTERNET OF THINGS USING OPCODES AND MACHINE LEARNING," George Mason, 2020.

[24] I. Santos *et al.*, "Idea: Opcode-Sequence-Based Malware Detection," in *ESSoS'10: Proceedings of the Second international conference on Engineering Secure Software and Systems*, 2010, pp. 35–43. doi: 10.1007/978-3-642-11747-3_3.

[25] B. P. Kinholkar, "Signature Base Method Dataset Feature Reduction of Opcode Using Pre- Processing Approach," *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)*, pp. 6813–6819, 2015, doi: 10.17762/ijritcc.v3i12.5147.

[26] T. Ahn, S. Oh, and Y. Kwon, "Malware Detection Method using Opcode and windows API Calls," *The Journal of the Institute of Internet, Broadcasting and Communication (한국인터넷방송통신학회논문지)*, vol. 17, no. 6, pp. 11–17, 2017, doi: 10.7236/JIIBC.2017.17.6.11.

[27] P. N. Yeboah, S. K. Amuquandoh, and H. B. B. Musah, "Malware Detection Using Ensemble N-gram Opcode Sequences," *International Journal of Interactive Mobile Technologies*, vol. 15, no. 24, pp. 19–31, 2021, doi: 10.3991/IJIM.V15I24.25401.

[28] S. Egunjobi, S. Parkinson, and A. Crampton, *Classifying Ransomware Using Machine Learning Algorithms*, vol. 11872 LNCS. Springer International Publishing, 2019. doi: 10.1007/978-3-030-33617-2_5.

[29] M. Almousa, S. Basavaraju, and M. Anwar, "API-Based Ransomware Detection Using Machine Learning-Based Threat Detection Models," *2021 18th International Conference on Privacy, Security and Trust, PST 2021*, 2021, doi: 10.1109/PST52912.2021.9647816.

[30] A. Shabtai, R. Moskovitch, C. Feher, S. Dolev, and Y. Elovici, "Detecting unknown malicious code by applying classification techniques on OpCode patterns," *Secur Inform*, vol. 1, no. 1, pp. 1–22, 2012, doi: 10.1186/2190-8532-1-1.

[31] B. J. Kang, S. Y. Yerima, K. McLaughlin, and S. Sezer, "N-opcode analysis for android malware classification and categorization," *2016 International Conference on Cyber Security and Protection of Digital Services, Cyber Security 2016*, pp. 13–14, 2016, doi: 10.1109/CyberSecPODS.2016.7502343.

[32] M. Hassen, M. M. Carvalho, and P. K. Chan, "Malware classification using static analysis based features," *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017 - Proceedings*, vol. 2018-Janua, pp. 1–7, 2018, doi: 10.1109/SSCI.2017.8285426.

[33] B. Kang, S. Y. Yerima, S. Sezer, and K. McLaughlin, "N-gram Opcode Analysis for Android Malware Detection," *International Journal on Cyber Situational Awareness*, vol. 1, no. 1, pp. 231–255, 2016, doi: 10.22619/ijcsa.2016.100111.

[34] E. Cunningham, O. Boydell, C. Doherty, B. Roques, and Q. Le, "Using text classification methods to detect malware," in *27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science*, 2019, vol. 2563, pp. 95–103.

[35] P. Yang, H. Zhou, Y. Zhu, L. Liu, and L. Zhang, "Malware classification based on shallow neural network," *Future Internet*, vol. 12, no. 12, pp. 1–17, 2020, doi: 10.3390/fi12120219.

[36] D. Li, L. Zhao, Q. Cheng, N. Lu, and W. Shi, "Opcode sequence analysis of Android malware by a convolutional neural network," *Concurr Comput*, vol. 32, no. 18, pp. 1–18, 2020, doi: 10.1002/cpe.5308.

[37] K. Shaukat *et al.*, "MAC Protocols 802.11: A Comparative Study of Throughput Analysis and Improved LEACH," *17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2020*, no. August, pp. 421–426, 2020, doi: 10.1109/ECTI-CON49241.2020.9158097.

[38] F. Mira, "A Review Paper of Malware Detection Using API Call Sequences," in *2nd International Conference on Computer Applications & Information Security (ICCAIS)*, 2019, pp. 1–6. doi: doi: 10.1109/CAIS.2019.8769564.

[39] A. Cannarile, F. Carrera, S. Galantucci, A. Iannacone, and G. Pirlo, "A study on malware detection and classification using the analysis of API calls sequences through shallow learning and recurrent neural networks," *CEUR Workshop Proc*, vol. 3260, pp. 124–134, 2022.

[40] A. Wolsey, "The State-of-the-Art in AI-Based Malware Detection Techniques: A Review," *arXiv preprint arXiv:2210.11239*, pp. 1–18, 2022.

[41] "Science Direct." https://www.sciencedirect.com

[42] "Web of Science." jcr.clarivate.com

[43] "IEEE Xplore Digital Library." https://ieeexplore.ieee.org/Xplore/home.jsp

[44] "SpringerLink." https://link.springer.com/

[45] "Google Scholar." https://scholar.google.com/

[46] K. Shaukat *et al.*, "Performance comparison and current challenges of using machine learning techniques in cybersecurity," *Energies (Basel)*, vol. 13, no. 10, 2020, doi: 10.3390/en13102509.

[47] S. Kamran *et al.*, "The Impact of Artificial intelligence and Robotics on the Future Employment Opportunities," *Trends in Computer Science and Information Technology*, vol. 5, pp. 050–054, 2020, doi: 10.17352/tcsit.000022.

[48] M. Ibrar *et al.*, "A Machine Learning-Based Model for Stability Prediction of Decentralized Power Grid Linked with Renewable Energy Resources," *Wirel Commun Mob Comput*, vol. 2022, 2022, doi: 10.1155/2022/2697303.

[49] M. Bat-Erdene, H. Park, H. Li, H. Lee, and M. S. Choi, "Entropy analysis to classify unknown packing algorithms for malware detection," *Int J Inf Secur*, vol. 16, no. 3, pp. 227–248, 2017, doi: 10.1007/s10207-016-0330-4.

[50] I. Santos, B. Sanz, C. Laorden, F. Brezo, and P. G. Bringas, "Opcode-sequence-based semi-supervised unknown malware detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6694 LNCS, pp. 50–57, 2011, doi: 10.1007/978-3-642-21323-6_7.

[51] I. Santos, B. Sanz, C. Laorden, F. Brezo, and P. G. Bringas, "Opcode-Sequence-Based Semi-supervised Unknown Malware Detection," in *Computational Intelligence in Security for Information Systems*, 2011, pp. 50–57. doi: 10.1007/978-3-642-21323-6_7.

[52] Y. Wang, J. W. Stokes, and M. Marinescu, "Neural Malware Control with Deep Reinforcement Learning," *Proceedings - IEEE Military Communications Conference MILCOM*, vol. 2019-Novem, 2019, doi: 10.1109/MILCOM47813.2019.9020862.

[53] K. Shaukat *et al.*, "A Review of Time-Series Anomaly Detection Techniques: A Step to Future Perspectives," *Advances in Intelligent Systems and Computing*, vol. 1363 AISC, no. April, pp. 865–877, 2021, doi: 10.1007/978-3-030-73100-7_60.

[54] A. Sharma and S. K. Sahay, "An effective approach for classification of advanced malware with high accuracy," *International Journal of Security and its Applications*, vol. 10, no. 4, pp. 249–266, 2016, doi: 10.14257/ijsia.2016.10.4.24.

[55] H. Daku, P. Zavarsky, and Y. Malik, "Behavioral-Based Classification and Identification of Ransomware Variants Using Machine Learning," *Proceedings - 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications and 12th IEEE International Conference on Big Data Science and Engineering, Trustcom/BigDataSE 2018*, pp. 1560–1564, 2018, doi: 10.1109/TrustCom/BigDataSE.2018.00224.

[56] M. Xu, H. Tong, C. Jin, and Y. Wang, "Malicious Code Detection Method Based on Multiple Features," *2021 IEEE 4th International Conference on Electronics and Communication Engineering, ICECE 2021*, pp. 8–15, 2021, doi: 10.1109/ICECE54449.2021.9674573.

[57] F. A. Aboaoja, A. Zainal, F. A. Ghaleb, and B. A. S. Al-Rimy, "Toward an Ensemble Behavioral-based Early Evasive Malware Detection Framework," *2021 International Conference on Data Science and Its Applications, ICoDSA 2021*, pp. 181–186, 2021, doi: 10.1109/ICoDSA53588.2021.9617489.

[58] A. Menelet, C. Bichot, A. Menelet, and C. B. Characterization, "Characterization of Android malware based on opcode analysis," *hal-03192097*, 2021.

[59] Q. Jerome, K. Allix, R. State, and T. Engel, "Using opcode-sequences to detect malicious Android applications," *2014 IEEE International Conference on Communications, ICC 2014*, pp. 914–919, 2014, doi: 10.1109/ICC.2014.6883436.

[60] Y. Ding, W. Dai, S. Yan, and Y. Zhang, "Control flow-based opcode behavior analysis for Malware detection," *Comput Secur*, vol. 44, no. 2007, pp. 65–74, 2014, doi: 10.1016/j.cose.2014.04.003.

[61] G. Canfora, A. De Lorenzo, E. Medvet, F. Mercaldo, and C. A. Visaggio, "Effectiveness of opcode ngrams for detection of multi family android malware," *Proceedings - 10th International Conference on Availability, Reliability and Security, ARES 2015*, pp. 333–340, 2015, doi: 10.1109/ARES.2015.57.

[62] D. Stiawan, S. M. Daely, A. Heryanto, N. Afifah, M. Y. Idris, and R. Budiarto, "Ransomware detection based on opcode behaviour using k-nearest neighbours algorithm," *Information Technology and Control*, vol. 50, no. 3, pp. 495–506, 2021, doi: 10.5755/j01.itc.50.3.25816.

[63] F. S. Ahmed, N. Mustapha, A. Mustapha, M. Kakavand, and C. F. M. Foozy, "Preliminary analysis of malware detection in opcode sequences within iot environment," *Journal of Computer Science*, vol. 16, no. 9, pp. 1306–1318, 2020, doi: 10.3844/jcssp.2020.1306.1318.

[64] H. Il Kim, M. Kang, S. J. Cho, and S. Il Choi, "Efficient Deep Learning Network with Multi-Streams for Android Malware Family Classification," *IEEE Access*, vol. 10, pp. 5518–5532, 2022, doi: 10.1109/ACCESS.2021.3139334.

[65] A. Sharma, P. Malacaria, and M. H. R. Khouzani, "Malware detection using 1-dimensional convolutional neural networks," *Proceedings - 4th IEEE European Symposium on Security and Privacy Workshops, EUROS and PW 2019*, pp. 247–256, 2019, doi: 10.1109/EuroSPW.2019.00034.

[66] U. Javed, K. Shaukat, I. A. Hameed, F. Iqbal, T. M. Alam, and S. Luo, "A Review of Content-Based and Context-Based Recommendation Systems," *International Journal of Emerging Technologies in Learning*, vol. 16, no. 3, pp. 274–306, 2021, doi: 10.3991/ijet.v16i03.18851.

[67] L. Zhao, D. Li, G. Zheng, and W. Shi, "Deep neural network based on android mobile malware detection system using opcode sequences," *International Conference on Communication Technology Proceedings, ICCT*, vol. 2019-Octob, pp. 1141–1147, 2019, doi: 10.1109/ICCT.2018.8600052.

[68] A. N. Jahromi, S. Hashemi, A. Dehghantanha, R. M. Parizi, and K. K. R. Choo, "An Enhanced Stacked LSTM Method with No Random Initialization for Malware Threat Hunting in Safety and Time-Critical Systems," *IEEE Trans Emerg Top Comput Intell*, vol. 4, no. 5, pp. 630–640, 2020, doi: 10.1109/TETCI.2019.2910243.

[69] R. Kumar and R. U. Khan, "Opcode and Gray Scale Techniques for Classification of Malware Binaries," 2018.

[70] J. Yuan, S. Zhou, L. Lin, F. Wang, and J. Cui, "Black-box adversarial attacks against deep learning based malware binaries detection with gan," *Frontiers in Artificial Intelligence and Applications*, vol. 325, pp. 2536–2542, 2020, doi: 10.3233/FAIA200388.

[71] M. J. Choi, J. Bang, J. Kim, H. Kim, and Y. S. Moon, "All-in-One Framework for Detection, Unpacking, and Verification for Malware Analysis," *Security and Communication Networks*, vol. 2019, 2019, doi: 10.1155/2019/5278137.

[72] R. Burks, K. A. Islam, Y. Lu, and J. Li, "Data Augmentation with Generative Models for Improved Malware Detection: A Comparative Study," *2019 IEEE 10th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2019*, pp. 0660–0665, 2019, doi: 10.1109/UEMCON47517.2019.8993085.

BIOGRAPHY OF AUTHORS

Mohammed A. Saleh is an Assistant Professor at Qassim University in Saudi Arabia. He received a B.Sc. (Honor) in Mathematical and Computer Science from the Faculty of Mathematical and Computer Science at the University of Gezira in Sudan. He obtained an M.Sc. (First Class) in Information Security from the Faculty of Computer Science and Information Systems and a Ph.D. in Information Security (Computer Science) from the Faculty of Computing at the University of Technology (UTM) in Malaysia. His research interests include Malware Analysis and Artificial Intelligence in Cybersecurity, and he is the author of a couple of journal articles.