Dynamic Malware Analysis Using Machine Learning Techniques

by

Swati Upadhyay

2018IS-15

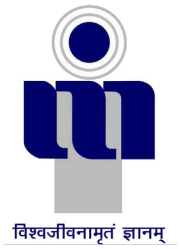
A thesis submitted in partial fulfilment of the requirements for the award of the degree of

Master of Technology

in

Information Security

2018-20



ATAL BIHARI VAJPAYEE-

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT

GWALIOR - 474015, MADHYA PRADESH, INDIA

Thesis Certificate

I hereby certify that the work, which is being presented in the report/thesis, entitled Dynamic Malware Analysis Using Machine Learning Techniques, in fulfillment of the requirement for the award of the degree of **Master of Technology**
in **Information Security** and submitted to the institution is an authentic record of my own work carried out during the period *May-2018* to *June-2020* under the supervision of Dr. Saumya Bhadauria. I also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken.

Signature: 

Name: Swati Upadhyay

Roll. No: 2018IS-15

Date: 10-June-2020

To the best of my knowldege, the statements made by Ms. Swati Upadhyay (Roll No. 2018IS-15) in the above thesis certificate is correct.

_____

Dr. Saumya Bhadauria

Date _____

Candidate's Declaration

I hereby certify that I have properly checked and verified all the items as prescribed in the check-list and ensure that my thesis is in the prop er format as specified in the guideline for thesis preparation.
I declare that the work containing in this rep ort is my own work. I understand that plagiarism is defined as any one or combination of the following:

(1) To steal and pass off (the ideas or words of another) as one's own

(2) To use (another's production) without crediting the source

(3) To commit literary theft

(4) To present as new and original idea or product derived from an existing source.

I understand that plagiarism involves an intentional act by the plagiarist of using someone else's work/ideas completely/partially and claiming authorship/originality of the work/ideas. Verbatim copy as well as close resemblance to some else's work constitute plagiarism.
I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programmes, experiments, results, websites, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.
I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report/dissertation/thesis are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable. My faculty supervisor(s) will not be responsible for the same.

Signature: 

Name: Swati Upadhyay

Roll. No: 2018IS-15

Date: 10-June-2020

Abstract

With the rapid increase in internet's usage, cyber attacks are also increasing. New types of malwares are being introduced everyday. To give protection to genuine users anti-malware products make use of signature based detection which performs static analysis of the malware. But these new forms of threat come without any prebuilt signature. To detect the ever coming novel malwares the signature database must be updated continuously and even regularly as hundreds of malwares are being designed everyday. For which new suspicious files need to be manually collected and analyzed on a regular basis which is a tedious and cumbersome job and need not to say impossible too. The proposed work aims to solve this problem by using techniques of machine learning for performing dynamic malware analysis for windows executable files. Dynamic analysis examines the behavior of the malware and not only searches for the signature code present in its body. And the algorithms of machine learning are capable of acquiring needed knowledge from the examples and performing well without human intervention. Genetic Programming along with two other algorithms i.e. Information gain attribute evaluation and Correlation attribute evaluation are used for selecting useful features of malicious and benign files after extracting them with the help of Cuckoo apparatus. Genetic Programming performs parallel searching using a data point population. So, unlike traditional methods which search using a single point, it can't get trapped in solutions containing local optima. So it brings out the most optimal features required for training the classifiers. Other two algorithms are used for the comparison purpose. The selected features are then used for training the machine learning classifiers. These classifiers are then used for malware detection.

Performance of the three attribute evaluation algorithms are compared to find out the best algorithm for feature selection as better features will better govern the performance of the classifiers. Among the classifiers also the best one is identified depending on the performance.

Dedication

Dedicated to my mentor, Dr. Saumya Bhadauria for her advice, patience and faith in my abilities.

Acknowledgments

I am highly indebted to **Dr. Saumya Bhadauria**, and obliged for giving me the autonomy of functioning and experimenting with ideas. I would like to take this opportunity to express my profound gratitude to her not only for her academic guidance but also for her personal interest in my report and constant support coupled with confidence boosting and motivating sessions which proved very fruitful and were instrumental in infusing self-assurance and trust within me. The nurturing and blossoming of the present work is mainly due to her valuable guidance, suggestions, astute judgment, constructive criticism and an eye for perfection. My mentor always answered myriad of my doubts with smiling graciousness and prodigious patience, never letting me feel like a novice by always lending an ear to my views, appreciating and improving them and by giving me a free hand in my report. It's only because of her overwhelming interest and helpful attitude, the present work has attained the stage it has.

Finally, I am grateful to our Institution and colleagues whose constant encouragement served to renew my spirit, refocus my attention and energy and helped me in carrying out this work.

Swati Upadhyay

2018IS-15

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Malware is defined as a malicious software or hardware which causes damage or harm to the user, computer or network. Malwares can steal our data, encrypt it, delete sensitive information, monitor or record user's activities in a covert manner, modify or hijack core computing functions [6].

1.1 Malware Analysis

Malware analysis is the technique used to examine the malware in order to understand its working and type which in turn helps to design suitable methods to defend it. Malware is mainly inspected by doing its static analysis or dynamic analysis. In static analysis, binary code is examined, execution paths are checked i.e. the malware is detected without executing the code while in dynamic analysis behavior of the malware is observed in a monitored and controlled environment [3].

Earlier, detection techniques are mostly based on static analysis which is a complex process. Nowadays, more and more sophisticated obfuscation techniques are produced like packing, polymorphism and encryption which can very easily bypass static analysis.

The detection process is further classified into two types, i.e. signature based detection and anomaly based detection. Signature based detection is used for static analysis. A prebuilt database is there which is used to compare malware samples for detection. Since new malwares are being designed everyday so this database needs constant updation to perform its function efficiently which

is again a tiresome and time consuming task. Whereas the anomaly based detection is based on the executable's behavior. It is less accurate but can detect zero-day attacks [9].

Table 1.1: Comparison between Signature and Anomaly based detection techniques.

Signature Based Detection	**Anomaly Based Detection**
Based on pattern or signature	Based on behavior or action
Highly accurate	Less accurate
Can't detect zero-day attacks	Can detect Zero-day attacks

1.2 Problem/Motivation

Computers have become a part and parcel of our daily lives. They are extensively being used everywhere i.e. in all government organisations, business, private firms, hospitals and homes. Due to their key role in our lives, it is one of the major challenge for us to defend them from malicious activities. One of such issue is malware [3]. Despite much advancement in security technology, malwares are still the most persistent threats in cyber world. Till now many techniques have been developed for analyzing malicious softwares. For each advancement in security technology a new evasion method is discovered. These evasion measures only alter the binary of malware but leave its behavior unmodified i.e. they obfuscate the code. And it is hard to design the detection rules that examine the semantics of a malware [10]. Thus, the need of the hour is to capture and analyse the additional properties that can help us to refine our security measures which in turn make the evasion process difficult.

1.3 Objectives

The proposed system is supposed to achieve the following goals:

- To detect zero-day attacks and other known malware attacks in Windows environment.
- To improve the classification process by doing better feature selection via Genetic Programming.

- To check the performance of Machine Learning classifiers (Random Forest, Naive Bayes, Nearest Neighbour, AdaBoost, Logistic Regression) in detecting malwares by comparing their sensitivity, specificity and accuracy.

The objective of the first phase is to do feature selection from the initially exracted feature set such that the selected features are capable of governing the performance of classifiers for detecting malwares.

In the next phase the selected features are then used for training machine learning classifiers. Performance of the classification algorithms is compared before and after feature selection.

The main objective is to use Genetic Programming measure to perform dynamic malware analysis and to train machine learning classifiers for detecting unknown malwares for which no pre-built signature database is present.

1.4 Research work flow

According to the research objectives, the report will describe the work flow as below:

Step 1 Malicious and Benign files in Windows portable executable format are collected.

Step 2 Using Cuckoo sandbox apparatus, features are extracted from these files in the form of JSON(JavaScript Object Notation) file format.

Step 3 Using attibute evaluation techniques for feature selection, features with maximum information are selected from the extracted feature set.

Step 4 These features are used to train the Machine Learning classifiers for malware detection (Phase II).

Step 5 Performance of the classifiers is compared by calculating their sensitivity, specificity and accuracy and the best one among them is addressed.

Chapter 2

Literature review

2.1 Background

Capturing important properties of malwares which can be used for making the evasion process difficult and can enhance the security measures is one of the major goal of malware analysts. These properties govern the behavior of the malware. Hence, are capable of distinguishing between a malware and a benign file. Machine Learning algorithms are trained to differentiate between these two classes. Features selected by the attribute evaluation algorithms are used for training the classifiers for this purpose where classification uses supervised learning to find out the class of the data item.

2.2 Key related research and Research Gaps

Use of Genetic Programming for feature selection has already been addressed by many academic works. Some of them further have used the selected features for malware detection.

The author [3] uses Cuckoo apparatus and genetic algorithm to perform dynamic analysis of malware. And the extracted features are used to train machine learning classifiers.Only feature selection is used and feature generation is not performed which can further improve the classifier's performance.

Results of Genetic Programming works efficiently on the training data but not on unseen data

or testing data. Training data is over-fitted to obtain the exact results. It works well in some cases. But for most of the real life problems of machine learning, it is not suitable. Semantic control measures [11] are proposed to solve this problem. Semantic control measures increase the generalization ability of Genetic Programming on both unseen data and training data. It is also observed that the solutions obtained after incorporating Semantic control are smaller in size than the ones evolved from Genetic Programming without using semantics.

The paper [9] provides a process for feature extraction using genetic algorithm which can be used for the detection of malware. This is done by determining a set of features that is suitable to differentiate malwares from benign files. Genetic algorithm is used for feature generation along with feature selection. It suffers from Code Bloat problem. In genetic algorithm an individual program can be of variable size. When evolution occurs, the programs become very large in size. It affects the efficiency of genetic algorithm by bringing the search process to an end.

The study [12] aims to increase the rate of detection of malwares by enhancing the accuracy of OSCP(One Side Class Perceptron) algorithm with the help of Genetic Programming. OSCP algorithm is used to keep low false positive rate. But the rate of detection of the algorithm is very low. Accuracy of the algorithm is improved by using the features evolved genetically. But still the detection rate is less when compared with that of other techniques.

The survey [10] provides a review of Machine Learning techniques used for performing malware analysis in case of portable executables in windows environment with the focus on the type of features extracted and on the algorithms of machine learning applied to process them. Current machine learning challenges and issues for malware analysis are highlighted in the survey which can be probed further to find their solutions. Appropriate malware analysis benchmarks are also defined.

The paper [5] presents a framework to avoid misclassification of new malwares by machine learning algorithms. It involves conversion of malware binaries into images and using them for training within a GAN(Generative Adversarial Networks). GAN is deployed in order to detect zero-day attacks. Conversion process of malware binaries into images is much faster than performing dynamic analysis of malware. The method proposed is also efficient against obfuscation techniques. Conversion process can be sped up and the range of malwares captured can be increased by using a more efficient code for conversion of malware binaries to images. Using another honeypot software can also serve this purpose.

In [6], Genetic Programmming is used for unknown malware detection. Various machine learning methods like Term Frequency, Fisher score and Frequency Inverse Document Frequency are used to perform the feature extraction process. These features are then used to train the classifiers. The quality of the classifiers formed by Genetic Programming is determined by using a testing set. The work proposed can be extended by adding more malicious and benign files to the existing dataset. Moreover, computational time given to Genetic Programming can be increased in order to form better classification models. Generalization ability of Genetic Programming can also be improved for getting better results.

The study [8] proposes a hybrid approach to combine feature selection and feature generation processes using a modified genetic algorithm for producing better outcomes. In cases where the original features aren't capable of performing the learning task, the performance can be improved by keeping the relevant features and taking out the irrelevant ones with the help of models used for feature selection. The search process here can be improved further by utilizing knowledge related to the search space.

In the present study, [7] malwares are visualized in the form of images. Initially the malwares are represented in the form of binary patters i.e. 0 or 1. Then they are converted into a 2-D

matrix. This matrix is visualized as an image. From this image, features are extracted with the help of Principle Component Analysis. Identification is done with the help of k-Nearest Neighbour, Support Vector Machine and Artificial Neural Network.

The author [2] presents a method to address the problem of high false positive and high false negative rates in malware detection systems due to which early malware detection becomes difficult. A combination of features taken after performing both static and dynamic malware analysis are used for training Machine Learning classifiers. These classifiers are affective for detecting Zero-day attacks.

The paper [13] proposes a method for malware detection in portable executable files on the basis of opcode frequency. Frequently occurring opcodes in the dataset are selected and used for training the machine learning algorithms for classification purpose. Classifiers being considered are Support Vector Machine, Boosting, Decision Tree and Random Forest.

The work [4] presents a solution to the problem of detection evasion in case of malwares. Features are extracted from dynamic and static malware analysis methods. The features are then used to form a classification model which can improve the detection rate as compared to the ones formed by either static or dynamic analysis alone and will be beneficial in case of unknown threats.

The paper [1] proposes to automate the process of malware identification by using sandbox and Machine Learning as the manual analysis takes too much time for generating signatures and is ineffective against unknown malicious codes. Machine Learning techniques are used to make behavioral analysis customized and more efficient.

2.3 Analysis

Genetic Programming is like evolution process of human beings. Survival of the fittest is being followed here. It helps to select the most optimal features which are used to train the classifiers. It is inspired by biological evolution which seeks the solution for a task defined by a user. Complete analysis of a malware is possible only by executing the code i.e. by performing dynamic analysis. For carrying out this process of information extraction from malwares, Genetic Programming comes out to be a natural choice.

2.4 Problem formulation

The major problem that has been pointed out from the study is to keep the evasion rate of malwares as low as possible. So, in order to make the detection framework more efficient, the present method has been proposed. The proposed method aims to increase the detection rate of malware and indirectly decrease the misclassification of legitimate files because wrongly classifying a benign file as a malware may affect the efficiency of a system and can cause system crash.

2.5 Conclusion

This chapter has given an overall view of the existing or previous related researches and works. It explains the present problems with the help of various studies in the field of malware analysis. It shows that the problem of evasion of malwares can be solved which is done in the present reaserch work and simultaneously decrease the occurrence of number of false positives and false negatives.

Chapter 3

Methodology

The proposed plan is to divide the entire work into two phases. In the first phase, feature selection is done from the extracted feature set using three attribute evaluation methods. The second phase uses the selected feature to train the machine learning classifiers and compares them by calculating their efficiencies.

3.1 Proposed hypothesis

Given a windows based portable executable, the system should be able to detect whether it is malicious or not. Thus, the system should be capable of ceasing the evasion of malwares. It is done by selecting the features which are actually responsible for their malicious behavior. Classifiers will be trained better in presence of better features which will enhance their performance in predicting the malwares. The methodology is divided into two phases:

In Phase I, feature selection is performed by using three attribute evaluation techniques which are Genetic Programming, information gain attribute evaluator and correlation attribute evaluator.

In Phase II, For training purpose, Machine Learning classifiers being used are Random Forest, Naive Bayes, Nearest Neighbour, AdaBoost and Logistic Regression. For avoiding overfitting, 10 fold cross validation is used in which the dataset is split into 10 parts and for each of the parts, one part acts as the testing set while the remaining 9 as training sets.

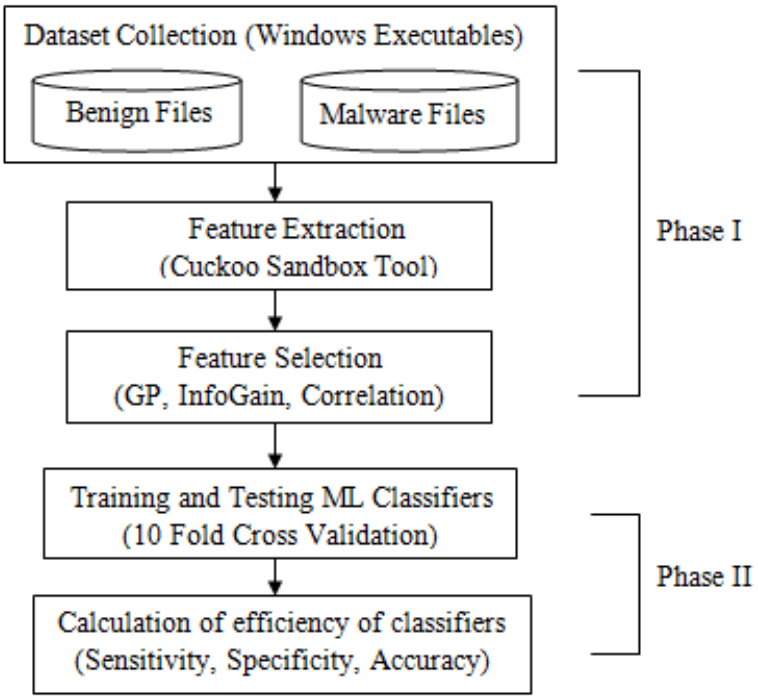


Figure 3.1: The architecture of proposed methodology

3.2 Mechanism/Algorithm

3.2.1 Dataset Collection

Dataset for this work is a huge collection of windows based portable executable samples. The sample is a combination of malwares as well as benign files. These sample executables are downloaded from online sources such as Pacific Cybersecurity, Malware DB, and GitHub repositories which allows us to clone the samples captured via Honeypots.

3.2.2 Feature Extraction

For the analysis, the cuckoo sandbox apparatus is used with the virtual box. Framework of cuckoo consists of a host machine and a guest machine. 64 bit Ubuntu 18.04 is the host machine with RAM of 4 GB. For the guest machine 32 bit windows 7 has been used inside the VMware. Sandbox's element which runs the actual investigation process is in the host machine. In the guest machine, the executables in the sample are executed individually. There is a medium to

communicate between the host machine and the guest machine called cuckoo agent. The result of the analysis after the execution is stored in a database inside cuckoo . Then a report in the form of JSON(JavaScript Object Notation) file format containing the behaviour of the sample is produced. It is stored in the host machine. This report is analysed to extract the features of malware and benign files present in the sample.

3.2.3 Feature Selection

For carrying out the process of feature selection from the extracted ones, three attribute evaluation algorithms are used.

(1) **Genetic Programming**

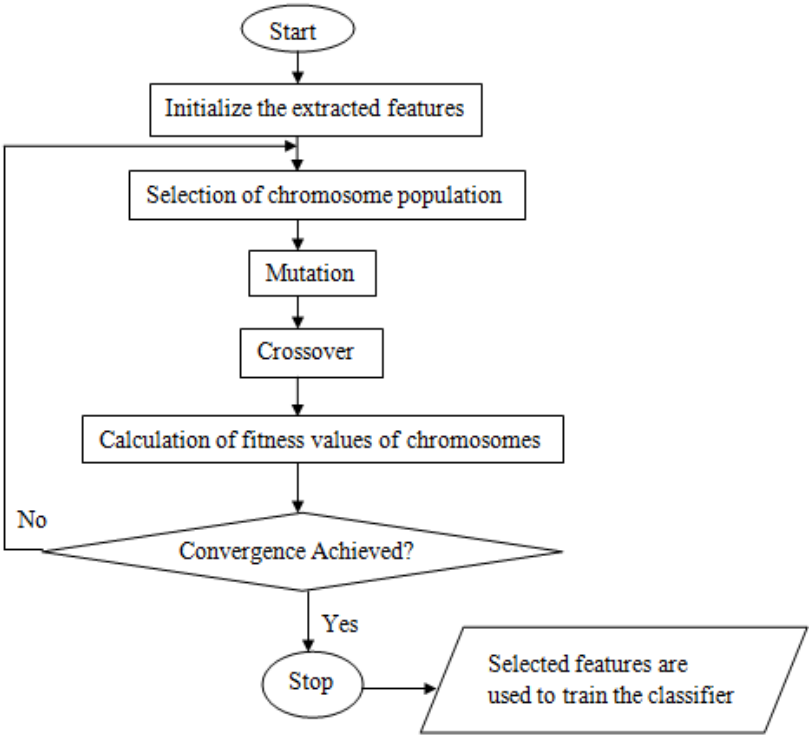


Figure 3.2: Feature selection using Genetic Programming

Features extracted by Cuckoo apparatus is given to the algorithm. Features are present in the form of chromosomes. Initially a random chromosome population is selected and passed through a fitness function. Chromosomes with better fitness are chosen to produce

the offspring. Then crossover and mutation are performed repeatedly till most optimal feature set is produced. Classifiers are trained using the selected features.

(2) **Information Gain Attribute Evaluation**

Information Gain is the amount of information gained about the class from an attribute. More the information gain less is the entropy.

$$InformationGain = Entropy_{(beforesplit)} - Entropy_{(aftersplit)}$$

$$Entropy = -\sum p(x)log_2p(x)$$

where p(x) is a fraction of examples in a given class.

After calculating the information gain associated with each feature, features with relatively higher values of information gain are selected for the training purpose.

(3) **Correlation Attribute Evaluation**

Correlation between an attribute and the class is calculated(by using Pearson's correlation formula).

For a given pair of varibles X,Y, Pearson's formula

$$\rho_{(X,Y)} = \frac{Cov_{(X,Y)}}{\sigma_X \sigma_Y}$$

where:

$Cov_{X,Y}$ = covariance between X and Y

σ_X = standard deviation of X

σ_Y = standard deviation of Y

$$Cov_{(X,Y)} = \frac{\sigma(X_i - \bar{X})(Y_i - \bar{Y})}{n-1}$$

$$\sigma_X = \sqrt{\frac{\sum(X_i - \bar{X})^2}{n-1}}$$

where:

X_i= X variable values

Y_i= Y variable values

$\bar{X}$= Mean of X

$\bar{Y}$= Mean of Y

n= sample size

Pearson's correlation coefficient shows the magnitude of linear relation or association between two variables in the form of a number between -1 to 1 where following value denotes

-1= negative correlation i.e increase in one variable's value decreases other variable's value.

0= No correlation i.e increase or decrease in one variable's value doesn't affect other variable's value.

1= positive correlation i.e increase in one variable's value increases other variable's value.

Attributes having relatively higher values of Pearson's correlation coefficient(either negative or positive) are selected for the next stage.

3.2.4 Classification

Classification uses supervised learning to find out the class of the data item. Machine Learning algorithms are trained to differentiate between the classes. Features selected by the attribute evaluation algorithms are used for training the classifiers for this purpose. In the present work, training and classification of malware is done using the following classifiers in which all of them belongs to different domains of classification algorithms.

(1) **Random Forest**

It belongs to the ensemble method of classification. The basic unit of a random forest is a decision tree. A decision tree is formed using the classification rules and is in the form of a flowchart. It is formed in such a way that the internal nodes indicate a test on a feature and the branches indicate the result of the test and class labels are indicated by the leaf nodes. Random forest constructs a number of decision trees using the training data and then calculates the mode of the output produced or the class predicted by the trees.

(2) **Naive Bayes**

It belongs to the probabilistic method of classification. It uses Bayes theorem for the classification along with the assumption of conditional independence between the attributes i.e. the presence of one attribute doesn't affect the presence of other attributes.

Bayes Theorem

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

where:

P(A/B)=Probability of happening of A given that B has occurred.

P(B/A)=Probability of happening of B given that A has occurred.

P(A)= Probability of happening of A.

P(B)= Probability of happening of B.

(3) **Nearest Neighbour**

It is a lazy learning classifier which classifies a data point depending upon the classification of its neighbours i.e. it predicts the class of a data point depending upon the vote of its nearest neighbours. The label assigned will be the one that most of its neighbour have.

(4) **AdaBoost**

AdaBoost or Adaptive Boosting is a meta algorithm. It uses boosting technique that forms a strong classifier from many weak classifiers. Boosting uses the training data to create stumps(decision trees with depth 1) one after another where each next stump rectifies the error in the previous stump. This process goes on till the maximum number of stumps are created or maximum accuracy is reached. AdaBoost initially assigns equal weights to all the training examples. With each iteration, it enhances the accuracy by increasing the weights of the misclassified examples and decreasing the weights of the correctly classified ones.

(5) **Logistic Regression**

It is a classifier that builts up a regression model for performing the classification. Logistic Regression uses sigmoid function to model the data.

Sigmoid Function

$$f(x) = \frac{1}{1 + \exp^{-x}}$$

3.3 Conclusion

This chapter discusses how the data for the experiment is collected and pre-processed. The entire experiment is divided into two phases, the selection of features appropriate for training and testing of the classifiers respectively. The computational models used are also mentioned here. The next chapter will discuss how the goodness of the results are measured to prove or disprove the superiority of one computational model over the other.

Chapter 4

Experiments and results

Before moving to the results, a brief discussion of the basis of results has been done as under:

4.1 Analysis of Outcomes

The entire data can be split into 10 parts and in the 9:1 ratio of train and test data for each of those parts. The goodness of outcome will be based on the parameters which are ROC curve area, sensitivity, specificity and accuracy. Here TP and FP implies True and False Positive respectively, TN and and FN implies True and False Negative respectively.

Confusion Matrix

It is a table to represent the classifier's performance on testing data. Confusion matrix shows the number of False Positives, True Positives, False Negatives and True Negative obtained after the classification. Using them, we calculate three quantities to determine the performance of the classifier.

Table 4.1: Confusion Matrix

	Predicted Class (Negative)	**Predicted Class (Positive)**
Actual Class (Negative)	True Negative(TN)	False Postive(FP)
Actual Class (Positive)	False Negative(FN)	True Positive(TN)

$$Senstivity = \frac{TP}{TP + FN} \tag{1}$$

$$Specificity = \frac{TN}{TN + FP} \tag{2}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Sensitivity (True Positive Rate): It is the ratio of number of correctly classified malwares to the total number of malwares in the dataset.

Specificity (True Negative Rate): It is the ratio of number of correctly classified benign files to the total number of benign files in the dataset.

Accuracy: It is the ratio of number of correctly classified files to the total number of files in the dataset.

ROC Curve

A Receiver Operating Characteristic curve is a curve plotted with the FPR(False Positive Rate) on x-axis and TPR(True Positive rate) on y-axis. ROC curve area is the two-dimensional area covered by the ROC curve. This area indicates how much capable a model is in distinguishing between the classes.

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

4.2 Results and Discussion

Following experimental results are obtained:

Table 4.2 shows the accuracy obtained by various classifiers before and after the feature selection process by using three of the methods. In case of each of the classifier, this area gets increased after feature selection. Hence, proving that the selected features are improving the performance of the classifiers over the dataset. For Random Forest highest increase is shown by correlation attribute evaluator and for Naive Bayes, information gain attribute evaluator turns out to be the best method. While for the remaining three classifiers i.e Nearest Neighbour, AdaBoost

Table 4.2: Area under ROC curve for five classifiers

Algorithm	**Before Selection**	**After Feature Selection**		
		InfoGain	**Correlation**	**GP**
Random Forest	0.9277	0.9435	0.9573	0.9440
Naive Bayes	0.9563	0.9704	0.9636	0.9615
Nearest Neighbour	0.9375	0.9519	0.9376	0.9825
AdaBoost	0.8025	0.8348	0.8373	0.9320
Logistic Regression	0.9132	0.9561	0.9642	0.9666

and Logistic Regression, Genetic Programming is giving the best results. Among the classifiers, the largest change in area under the ROC curve is shown by AdaBoost followed by Logistic Regression and Random Forest while the least change is shown by Naive Bayes. On the basis of this data, the finest behaviour is depicted by Naive Bayes classifier as it covers the maximum area under the ROC curve in most of the cases followed by Nearest Neighbour and Logistic Regression. While the worst performance is of AdaBoost.

Comparison of ROC curves

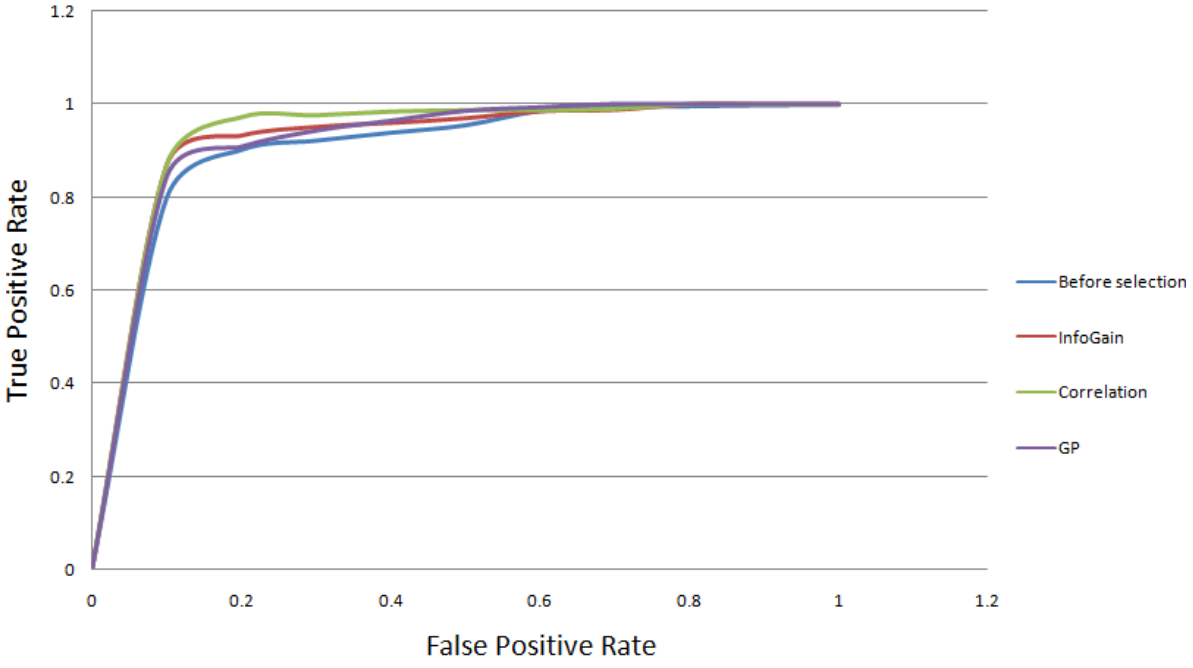


Figure 4.1: ROC Curves for Random Forest

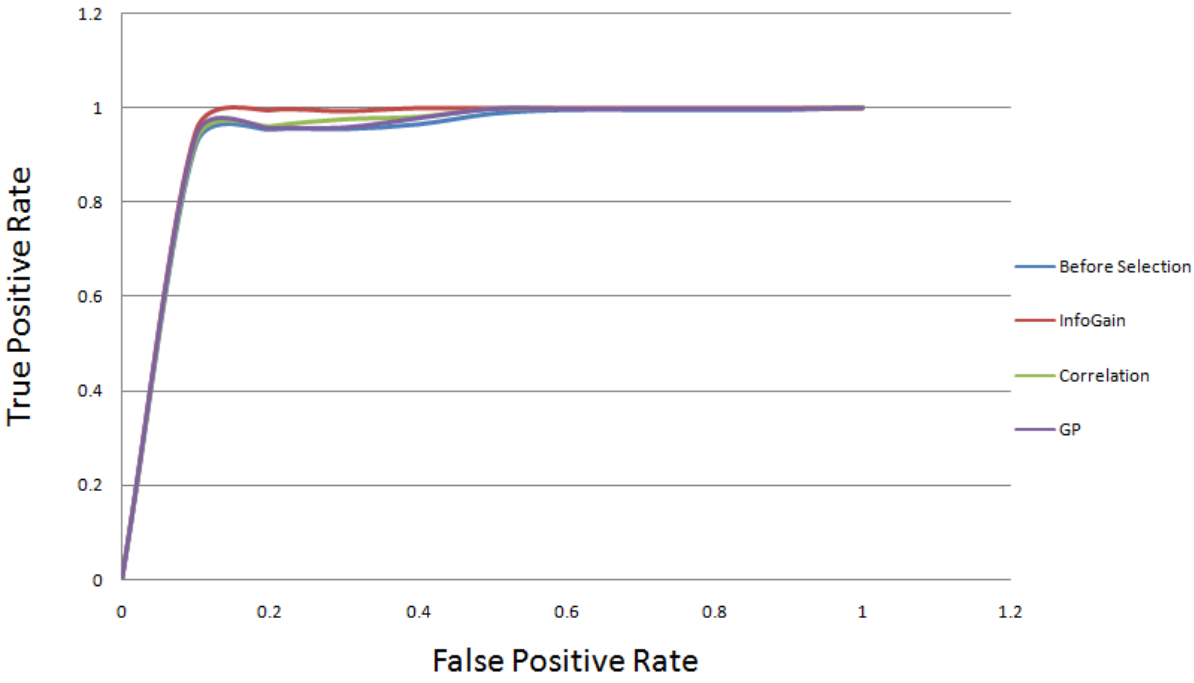


Figure 4.2: ROC Curves for Naive Bayes

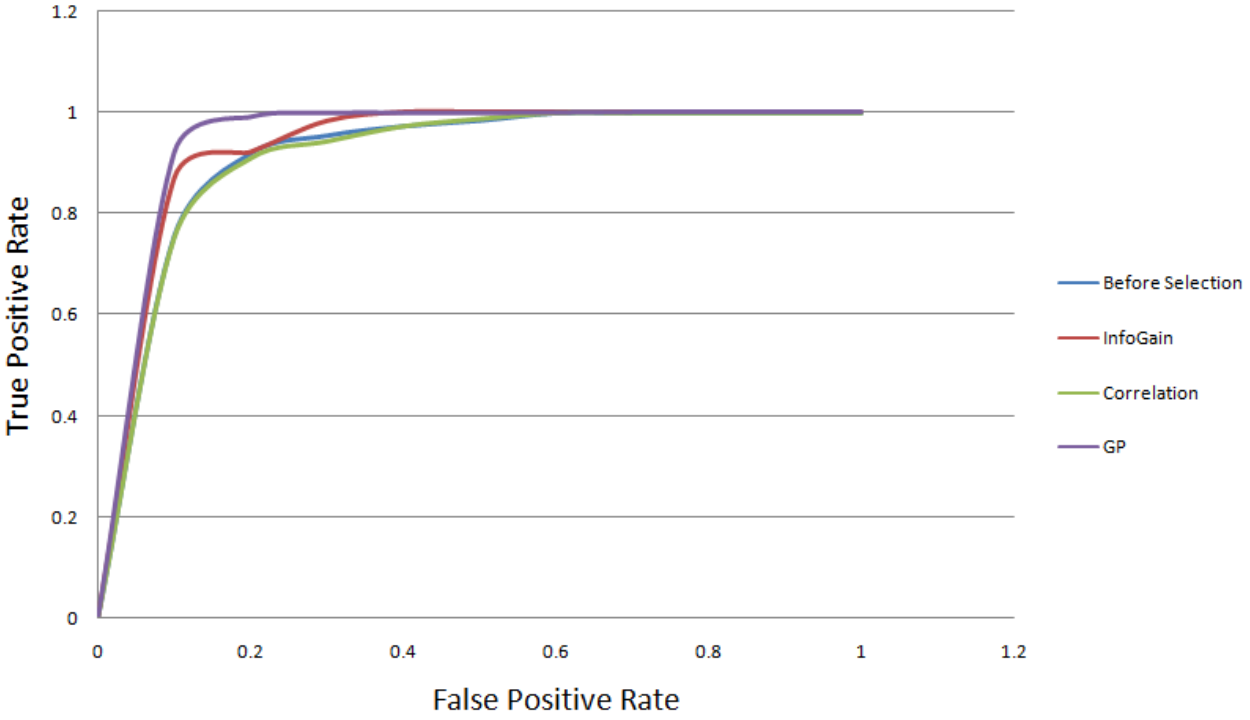


Figure 4.3: ROC Curves for Nearest Neighbour

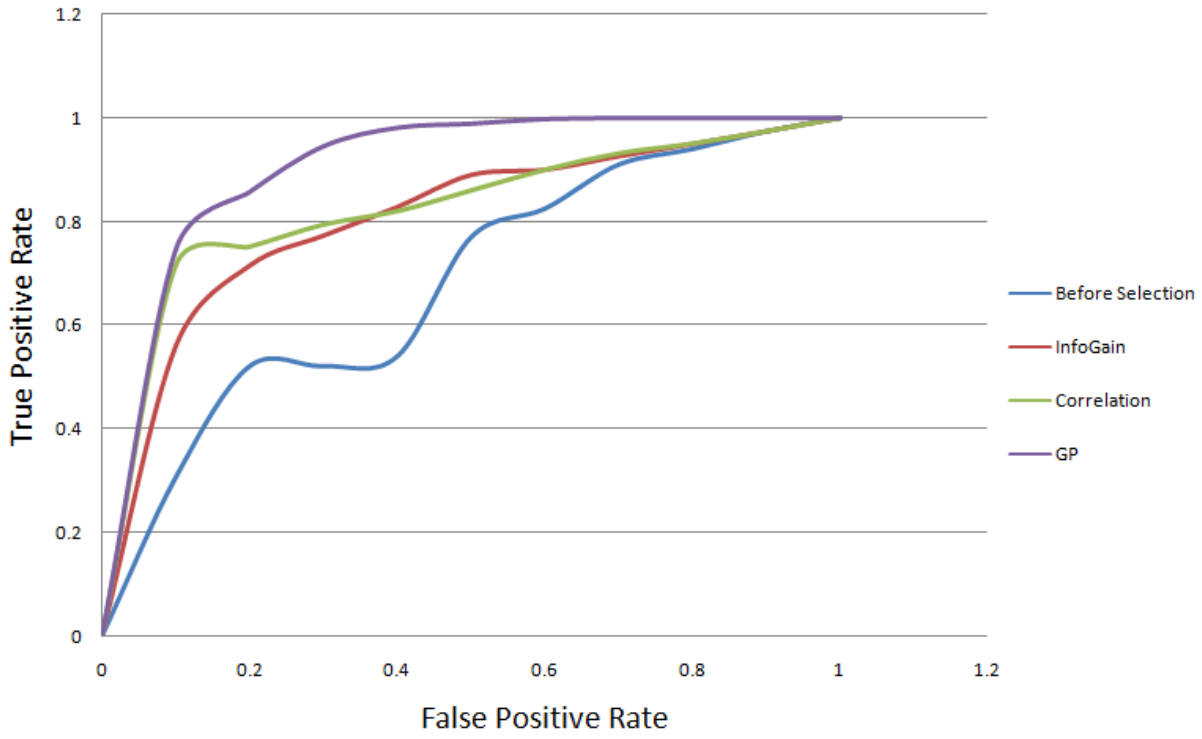


Figure 4.4: ROC Curves for AdaBoost

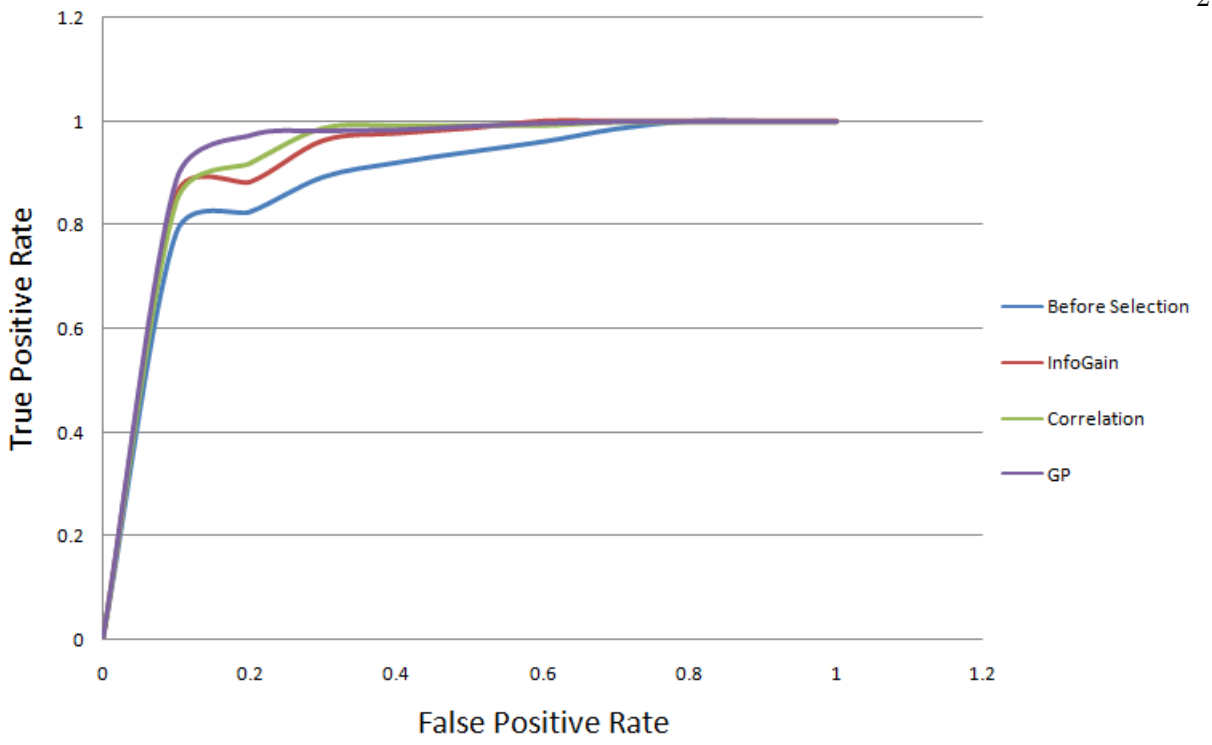


Figure 4.5: ROC Curves for Logistic Regression

Fig. 4.1, 4.2, 4.3, 4.4, 4.5 show the ROC curves obtained on training five Machine Learning classifiers i.e. Random Forest, Naive Bayes, Nearest Neighbour, AdaBoost and Logistic Regression before and after doing feature selection. Feature selection is done using three different feature selection methods i.e. Information gain attribute evaluation, Correlation attribute evaluation and Genetic Programming.

Table 4.3: Performance Metrics of Random Forest Classifier

Performance Metrics	**Before Feature Selection(%)**	**After Feature Selection(%)**		
		InfoGain	**Correlation**	**GP**
Sensitivity	90.7%	89.9%	89.8%	85.2%
Specificity	77.7%	87.7%	87.5%	89.2%
Accuracy	84.2%	88.8%	88.7%	87.2%

Table 4.4: Performance Metrics of Naive Bayes Classifier

Performance Metrics	**Before Feature Selection(%)**	**After Feature Selection(%)**		
		InfoGain	**Correlation**	**GP**
Sensitivity	89.9%	87.6%	94.6%	92.1%
Specificity	92.2%	95.2%	89.2%	94.7%
Accuracy	91.1%	91.4%	91.9%	93.4%

Table 4.5: Performance Metrics of Nearest Neighbour Classifier

Performance Metrics	**Before Feature Selection(%)**	**After Feature Selection(%)**		
		InfoGain	**Correlation**	**GP**
Sensitivity	80.9%	85.7%	86.2%	89.3%
Specificity	87.9%	90.5%	86.7%	93.6%
Accuracy	84.4%	88.1%	86.5%	91.5%

Table 4.6: Performance Metrics of AdaBoost Classifier

Performance Metrics	**Before Feature Selection(%)**	**After Feature Selection(%)**		
		InfoGain	**Correlation**	**GP**
Sensitivity	49.2%	60.5%	71.6%	76.3%
Specificity	98.6%	98.2%	90.0%	85.3%
Accuracy	73.8%	79.3%	80.8%	80.8%

Table 4.7: Performance Metrics of Logistic Regression Classifier

Performance Metrics	**Before Feature Selection(%)**	**After Feature Selection(%)**		
		InfoGain	**Correlation**	**GP**
Sensitivity	80.5%	85.8%	87.5%	85.0%
Specificity	86.1%	85.8%	86.7%	93.6%
Accuracy	83.3%	85.8%	87.1%	89.3%

Tables 4.3, 4.4, 4.5, 4.6, 4.7 show Performance Metrics of Random Forest, Naive Bayes, Nearest

Neighbour, AdaBoost and Logistic Regression classifiers using equations (1),(2) and (3) before and after feature selection which is done using three methods i.e. Information gain attribute evaluation, Correlation attribute evaluation and Genetic Programming. Random Forest shows increase in specificity and accuracy when classified with selected features but decrease in sensitivity. It gives best performance with Information gain attribute evaluator. For Naive Bayes, there is increase in all three quantities with GP while a slight fall in sensitivity and specificity with InfoGain and Correlation evaluator respectively. So, it shows best behaviour with GP. Nearest Neighbour shows rise in all three parameters with InfoGain Evaluator and GP but a slight drop in specificity with Correlation Evaluator. AdaBoost shows considerable rise in sensitivity and accuracy but a little fall in specificity with three of the methods. Logistic Regression gives the same behaviour with the only difference being fall in specificity in case of InfoGain evaluator instead of the other two. So, for Nearest Neighbour and Logistic Regression also GP turns out to be the best feature selection method. AdaBoost shows nearly the same performance with Correlation evaluator and GP.

4.3 Overall conclusion

On the basis of the results obtained above, among the three feature selection methods being considered, Genetic Algorithm turns out to be the best one for feature selection as it produces optimized results while preserving the time required for training. Better feature knowledge increases the detection rate of malware in case of unknown samples along with the known ones. Among the classifiers, Naive Bayes gives the finest result on the basis of these three parameters due to its ability to converge more quickly than any other discriminative model when its assumption (Conditional Independence) holds. This assumption makes it highly scalable i.e. it can learn even when the training data is less and the amount of features related to each individual in the sample is large. Therefore, it is insensitive to non relevant features. Naive Bayes is followed by Nearest Neighbour and Random Forest and the worst performance is of AdaBoost. Data imbalance can be the reason for the deficient performance of Adaboost.

Chapter 5

Discussions and conclusion

5.1 Conclusion

This study presents a brief introduction about the malwares, malware analysis methods and malware detection techniques. Malware detection process is classified into two phases i.e. obtaining useful features and training classifiers by using them. Feature selection and training both greatly affect the evasion rate of malwares. It has been empirically shown that Genetic Programming stands superior to other methods for feature selection as it selects the most optimal features which are obtained after running the algorithm for many generations. Classifiers fed with the output obtained by GP give the best accuracy as compared to their accuracy with other two selection methods.

5.2 Future scope

In future, the work can be extended by enriching the data set with more benign files and malware codes to improve the performance of Genetic Programming and classifiers. Further, increasing the number of runs of Genetic Programming will be beneficial as well. With every run, features are selected. After considering the result of each run, features appearing in more than half of the runs are selected for further processing. With more runs, it would be easy to select better features which are in turn crucial for the classification process.

Proposed framework only focuses on detecting the malwares and not on classifying them. Therefore, it can be extended to classify malwares into their respective families.

Bibliography

[1] C. A. B. d. Andrade, C. G. d. Mello, and J. C. Duarte. Malware automatic analysis. In BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence, pages 681–686. IEEE, 2013.

[2] E. Gandotra, D. Bansal, and S. Sofat. Zero-day malware detection. In Sixth International Symposium on Embedded Computing and System Design (ISED), pages 171–175. IEEE, 2016.

[3] A. Irshad, R. Maurya, M. K. Dutta, R. Burget, and V. Uher. Feature optimization for run time analysis of malware in windows operating system using machine learning approach. In 42nd International Conference on Telecommunications and Signal Processing (TSP), pages 255–260. IEEE, 2019.

[4] A. Jain and A. K. Singh. Integrated malware analysis using machine learning. In 2nd International Conference on Telecommunication and Networks (TEL-NET), pages 1–8. IEEE, 2017.

[5] J. Kargaard, T. Drange, A. Kor, H. Twafik, and E. Butterfield. Defending it systems against intelligent malware. In IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), pages 411–417. IEEE, 2018.

[6] T. A. Le, T. H. Chu, Q. U. Nguyen, and X. H. Nguyen. Malware detection using genetic programming. In Seventh IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), pages 1–6. IEEE, 2014.

[7] B. N. Narayanan, O. Djaneye-Boundjou, and T. M. Kebede. Performance analysis of machine learning and pattern recognition algorithms for malware classification. In IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS), pages 338–342. IEEE, 2016.

[8] O Ritthof, Ralf Klinkenberg, Simon Fischer, and Ingo Mierswa. A hybrid approach to feature selection and generation using an evolutionary algorithm. In UK Workshop on Computational Intelligence, pages 147–154. ResearchGate, 2002.

[9] P. Srivastava and M. Raj. Feature extraction for enhanced malware detection using genetic algorithm. In Int. J. Eng. Technol., volume 7, pages 444–449. SPC, 2018.

[10] D. Ucci, L. Aniello, and R. Baldoni. Survey of machine learning techniques for malware analysis. In Computers & Security, volume 81, pages 123 – 147. Elsevier, 2019.

[11] N. Q. Uy, N. T. Hien, N. X. Hoai, and M. O'Neill. Improving the generalisation ability of genetic programming with semantic similarity based crossover. In Genetic Programming, pages 184–195. Springer Berlin Heidelberg, 2010.

[12] C. Vatamanu, D. Gavrilut, R. Benchea, and H. Luchian. Feature extraction using genetic programming with applications in malware detection. In 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pages 224–231. IEEE, 2015.

[13] A. Yewale and M. Singh. Malware detection based on opcode frequency. In International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), pages 646–649. IEEE, 2016.