Journal of Computing Research and Innovation (JCRINN) Vol. 8 No.2 (2023)
https://jcrinn.com : eISSN: 2600-8793 / https://dx.doi.org/10.24191/jcrinn.v8i2.329

Comparison of Supervised Machine Learning Algorithms for Malware Detection

Mohd Faris Mohd Fuzi[1], Syamir Mohd Shahirudin[2], Iman Hazwam Abd Halim[3], Muhammad Nabil Fikri Jamaluddin[4]
[1,2,3,4] *College of Computing, Informatics, and Mathematics, Universiti Teknologi MARA Perlis Branch, Arau Campus, 02600 Arau, Perlis, Malaysia.*

*Corresponding author: * farisfuzi@uitm.edu.my*

Received Date: 31 August 2022
Accepted Date: 27 April 2023
Revised Date: 5 May 2023
Published Date: 1 September 2023

HIGHLIGHTS

- Malware is becoming more sophisticated, making it difficult to detect using malware detection software.
- Machine learning algorithm techniques have grown in popularity among researchers for analysing malware detection.
- Focus on the supervised machine learning algorithm for malware detection.
- The best algorithm for malware detection will have the highest percentage of detection accuracy.

ABSTRACT

Due to the prevalence of security issues and cyberattacks, cybersecurity is crucial in today's environment. Malware has also evolved significantly over the past few years. With the advancement of malware analysis, Machine Learning (ML) is increasingly being used to detect malware. This study's major objective is to compare the best-supervised ML algorithms for malware detection based on detection accuracy. This study includes the scripting and development of supervised ML techniques such as Decision Tree (DT), K-Nearest Neighbors (KNN), Naive Bayes, Random Forest, and Neural Networks. This study was solely concerned with the Windows malware dataset. The malware classification was determined by testing and training the supervised ML algorithms using the extracted features from the malware dataset. Then, the percentage of detection accuracy was used to compare the detection performance of all five algorithms. The detection accuracy is calculated using the confusion matrix, which includes the False Positive Rate (FPR), the True Positive Rate (TPR), and the False Negative Rate (FNR). The results indicated that the Decision Tree and Random Forest algorithms provided the best detection accuracy at 96%, followed by the K-NN algorithm at 95%. To improve the detection accuracy for future research, it is suggested that the malware dataset be enhanced using several architectures, such as Linux and Android, and use additional supervised and unsupervised machine learning algorithms.

Keywords: *supervised machine learning; malware detection; detection accuracy; machine learning algorithms*

INTRODUCTION

People nowadays use the internet for a variety of purposes, including shopping, watching videos, listening to music, and even filing taxes. It is no secret that the World Wide Web's quick expansion over the last two decades has brought us fantastic things and given us the ability to do our tasks from the comfort of our own

homes or offices. But, as with many excellent things, there is always the other side, which is not always so nice. Cybersecurity is a term that most people are not familiar with or are not interested in because it can be complicated. The basic purpose of cybersecurity is to secure the user, their data, and any other sensitive information that should be kept away from nosy eyes. However, if people nowadays are unaware of the importance of cybersecurity and neglect to take preventative precautions, they are likely to have problems.

Those issues can lead to situations that are not only difficult to resolve but also put safety in jeopardy. Examining the repercussions of situations where cybersecurity was lacking is the best way to determine why we need cybersecurity in the first place. The company cannot defend itself against data breach operations without a cybersecurity programme, making it an easy target for fraudsters. The attacker uses a dangerous weapon to harm or steal the personal information of the target user, which is called malware. The evolution of malware has changed rapidly. Malware-based cyberattacks are used in the banking sector, for example, to automate the process of penetration into the targeted organisation's IT systems (Irfan et al., 2020).

Cybersecurity practitioners require malware analysis to study and gain information about the dangerous malware nowadays. Because of the rising number of malware attacks on computers and networks, researchers are concentrating their efforts in the field of malware detection and analysis (Samy et al., 2018). Malware analysis can help cybersecurity by determining whether a suspicious file is harmful, studying its origin, method, capabilities, and impact, and assessing its impact to aid detection and prevention. Malware analysis is the study or process of determining the functionality, origin, and potential impact of a given malware sample, such as a virus, worm, trojan horse, rootkit, or backdoor. Also, malware analysis is the process of determining how a suspicious file or URL behaves and what its aim is. The analysis's output aids in detecting and mitigating the potential hazard. Machine learning is a type of data analytics that allows computers to do specific jobs without being given explicit instructions. In recent years, machine learning capabilities have been employed to create both static and dynamic malware detection algorithms.

Machine learning is widely used in the field of cybersecurity, and there are several different machine learning algorithms available for research, including decision tree and logistic regression, to name but a few. Static analysis of malware involves inspection of the code at rest and is successful in the classification of malware families. The traditional method cannot manage and has become lacking in the detection of malware and weaker in the security of the protection against malware (Selamat et al., 2019). Machine learning can be used to prevent advanced malware behaviour from spreading in the system. Machine learning is an advanced form of malware analysis to make a better defence system, especially for the detection of malware (Zakaria et al., 2017). Previous researchers have developed and analysed machine learning for detection models on existing malware to get the percentage accuracy of static analysis effectively by using different supervised machine learning (Selamat et al., 2019). However, there is a lot of supervised machine learning that can be used as a malware detection model. Therefore, this study proposed a comparative analysis of supervised machine learning algorithms for malware detection and is focused on Windows-type malware. In this study, five different supervised machine-learning algorithms were used for malware detection. The algorithms were Random Forest, Neural Network, Decision Tree, K-NN, and Naive Bayes.

Moubarak and Feghali (2022) make the comparison using the average detection time for detecting malware using different machine learning methods. In their project, the authors used the detector receiver operating characteristic (ROC) curve to get the TPR and FPR classifier performance graphs. Pavithra & Josephin (2020) make a comparison of malware classification using various forms of machine learning. The author also uses various methods for malware classification from different machine learning systems. The accuracy of each model determined the effectiveness of malware detection on the specific malware dataset.

METHODOLOGY

Figure 1 shows the methodology for malware detection using supervised machine learning which consists of malware dataset, feature extraction, and malware detection model.

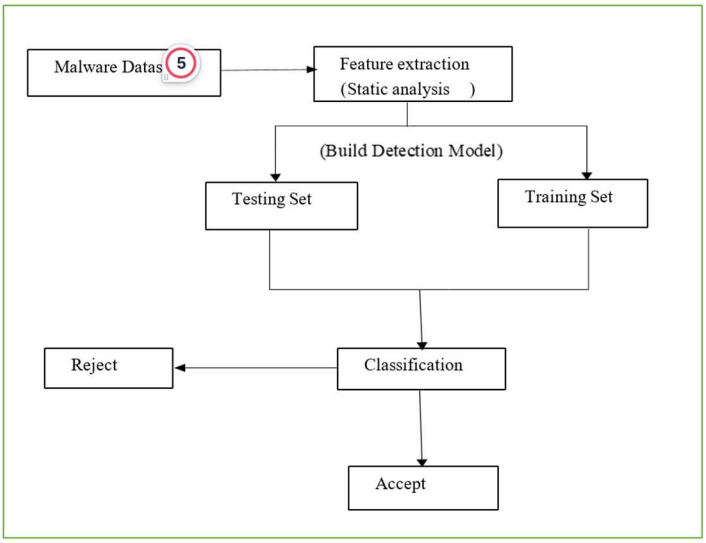

Figure 1: Methodology for malware detection using supervised machine learning.

Malware Dataset Collection

The malware dataset has been downloaded from reliable open-source malware datasets on GitHub and Kaggle. This project used a dataset that contains various samples of benign and malicious software as input for supervised machine learning. The malware dataset used was downloaded from GitHub and Kaggle, which are trusted malware datasets for research and analysis data. This malware dataset used Windows architecture because the features have been detected as suspicious as malware and benign after being called from the Windows API.

Features Extraction

Features extractions are important for making the machine learning get high accuracy detection. This is because the features will act as the input of machine learning and be trained and tested for the evaluation of the performance detection model. Eight features in this malware dataset have been detected as malware and benign features. These features have been extracted by using the malware collector and software tools named PeStudio. Figure 2 shows the feature selection that has been extracted.

AddressOfEntryPoint	MajorLinkerVersion	MajorImageVersion	MajorOperatingSystemVersion	DllCharacteristics	SizeOfStackReserve	NumberOfSections	ResourceSize	legitimate
10407	9	6	6	33088	262144	4	952	1
5354	9	6	6	33088	262144	4	952	1
56807	9	6	6	33088	262144	4	136490	1
25166	9	6	6	33088	262144	4	1940	1
70387	9	6	6	33088	262144	4	83098	1
...	...	...	...	...	...	...	...	...
123291	11	0	5	33088	1048576	5	81654	0
40000	2	6	1	32768	1048576	8	67624	0
59610	10	0	5	33088	1048576	5	22648	0
51216	2	0	1	0	1048576	8	2216	0
22731	11	0	5	33088	1048576	5	318464	0

Figure 2: Features selection that has been extracted.

Malware Detection Model

In this section, there are three activities involved: training, testing, and analysis. The training activities included collecting and inputting the malware dataset (80%) as training input, extracting suitable features to represent the malware file, classifying the extracted feature based on categories of malware types, and storing it as a classifier to be used in the testing phase. The testing activities included collecting and inputting a 20% malware dataset as testing input, extracting features (the same form of features as in the training phase), and comparing the features with the stored classifier to decide which type of malware category the testing malware feature belongs (accept) or does not belong to any type of malware category (reject). An analysis activity was included in the recording of the results of the testing and training phases. All these activities involved the development of scripting using the five supervised machine learning algorithms, which were Random Forest, Neural Network, Decision Tree, K-NN, and Naive Bayes.

Scripting of Machine Learning Algorithms

Five supervised machine learning algorithms were used in this scripting development: K-NN, Decision Tree, Neural Network, Random Forest, and Naive Bayes. These algorithms were used as a model for malware detection. The classifier or algorithms used are determined by the type of features, the size of the dataset, and the problem to be solved. After removing irrelevant features, these classifiers were used. Following that, these features were trained and tested on each classifier to perform classification tasks.
In this section, the scripting from each machine learning algorithm was different. The training model was based on machine learning algorithms which need to fit with the training data.

K-NN Algorithm

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
training_samples,testing_samples,training_targets,testing_target = train_test_split(samples,targets,test_size=0.2,random_state=0)
```

```
neigh = KNeighborsClassifier()
```

```
neigh.fit(training_samples,training_targets)
```

```
KNeighborsClassifier()
```

```
prediction=neigh.predict(testing_samples)
acy=100*accuracy_score(testing_target,prediction)
```

Figure 3: The scripting of the K-NN machine learning algorithm.

Decision Tree Algorithm

```
#Decision Tree Machine Learning
import sklearn
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.model_selection import train_test_split
training_samples,testing_samples,training_targets,testing_target = train_test_split(samples,targets,test_size=0.2,random_state=0)
```

```
from sklearn import tree
tree_classifier=tree.DecisionTreeClassifier()
tree_classifier.fit(training_samples,training_targets)
prediction=tree_classifier.predict(testing_samples)
accuracy=100*accuracy_score(testing_target,prediction)
```

Figure 4: The scripting of the Decision Tree machine learning algorithm.

Neural Network

```
from sklearn.model_selection import train_test_split
training_samples,testing_samples,training_targets,testing_target = train_test_split(samples,targets,test_size=0.2,random_state=0)
```

```
from sklearn.neural_network import MLPClassifier
```

```
cnn = MLPClassifier()
cnn.fit(training_samples,training_targets)
```

```
MLPClassifier()
```

```
prediction=cnn.predict(testing_samples)
accurracy=100*accuracy_score(testing_target,prediction)
```

Figure 5: The scripting of the Neural Network machine learning algorithm.

Random Forest

```
from sklearn.model_selection import train_test_split
training_samples,testing_samples,training_targets,testing_target = train_test_split(samples,targets,test_size=0.2,random_state=0)
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
cff = RandomForestClassifier()
cff.fit(training_samples,training_targets)
```

```
RandomForestClassifier()
```

```
prediction=cff.predict(testing_samples)
acuracy=100*accuracy_score(testing_target,prediction)
```

Figure 6: The scripting of the RandomForest machine learning algorithm.

Naive Bayes

```
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
training_samples,testing_samples,training_targets,testing_target = train_test_split(samples,targets,test_size=0.2,random_state=0)

gnb = GaussianNB()
gnb.fit(training_samples,training_targets)

GaussianNB()

prediction=gnb.predict(testing_samples)
accuracyyy=100*accuracy_score(testing_target,prediction)
```

Figure 7: The scripting of the Naïve Bayes machine learning algorithm.

FINDINGS AND DISCUSSIONS

After the training and testing set has been done, the result will be produced by indicating the percentage detection accuracy in each detection model. By using the confusion matrix, the calculation of the percentage detection accuracy, False Positive Rate (FPR), True Positive Rate (TPR), and False Negative Rate (FNR) have also been done. The four key performance metrics True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) will be computed to assess the outcomes. The percentage of samples that were accurately identified as malware will be shown by the True Positives Rates (TPR). False Positive Rates (FPR) will indicate the proportion of samples that were incorrectly classified as malware. False Negative Rate (FNR) is the ratio of false-negative and positive which is double incorrectly classified as malware. The formulas for the performance measurements are **TPR=TP/(TP+FN), FPR=FP/(FP+TN)** and **FNR = FN / P**. While overall accuracy is the percentage of all correctly predicted outcomes, it is calculated as **Accuracy=((TP+TN))/(TP+FP+TN+FN**).

After all these five machine learning algorithms results have been collected, this part is to make an analysis result by comparing the result of the percentage detection accuracy on the same Windows malware dataset. The Random Forest has the highest percentage of detection accuracy which is 0.96% according to this table comparison. This indicates that, when compared to other machine learning algorithms, the Random Forest Machine Learning Algorithm offers the best high-accuracy detection for differentiating between malware and benign software. The reason why the Decision Tree Machine Learning Algorithm has the best percentage detection accuracy in differentiating between malware and benign software because it has the highest TPR which is 0.94% shows the highest accurately identified as malware and the lowest FPR and FNR which is 0.03 % and 0.06% respectively that shows the percentage on inaccurately identified as malware compared to the K-NN, Neural Network, Decision Tree and Naïve Bayes Machine Learning Algorithms. Figure 8 shows the detection accuracy results for each supervised machine learning.

ML Algorithm	Accuracy	TPR	FPR	FNR
Decision Tree	0.96	0.93	0.02	0.06
Naive Bayes	0.85	0.78	0.13	0.21
K-NN	0.95	0.92	0.04	0.08
Neural Network	0.85	0.83	0.15	0.17
RandomForest	0.96	0.94	0.03	0.06

Figure 8: Comparison Accuracy of Supervised Machine Learning

CONCLUSION AND RECOMMENDATION

Malware is developing with advanced and complex by the day. By comparing five different supervised machine learning algorithms, the focus of this experiment is on analyzing and measuring the detection accuracy of the supervised machine learning classifier that used static analysis to extract the features based on PE information. We were able to train machine-learning algorithms to distinguish between malicious and benign files. According to the results, the Random Forest machine learning technique is the best classifier for classifying our data with 0.96% accuracy. According to the results of this experiment, using static analysis based on PE information and selecting the relevant features of the data can also provide the best detection accuracy and accurately represent malware. Another benefit of static analysis is the accuracy percentage detection still gets higher even though the malware does not execute.

CONFLICT OF INTEREST DISCLOSURE

The authors declared that they have no conflicts of interest to disclose.

REFERENCES

Irfan, A. N., Ariffin, A., Naz, M., & Anuar, S. (2020). A Malware Detection Framework Based on Forensic and Unsupervised Machine Learning Methodologies. *ICSCA '20: 9th International Conference on Software and Computer Applications*. 194–200. https://doi.org/10.1145/3384544.3384556

Moubarak, Joanna, and Tony Feghali. (2020). Comparing Machine Learning Techniques for Malware Detection. *4th International Workshop on Formal Methods for Security Engineering*. DOI:10.5220/0009373708440851

Pavithra, J. and Josephin, J. S. F. (2020). Analyzing Various Machine Learning Algorithms for the Classification of Malwares. International Conference on Mechanical, Electronics and Computer Engineering 2020. Kancheepuram, India. Volume 993. DOI 10.1088/1757-899X/993/1/012099

Samy, Ganthan Narayana and Magalingam, Pritheega and Mohd. Ariffin, Aswami Fadillah and Mohd. Khairudin, Wafa and Md. Senan, Mohamad Firham Efendy and Yunos, Zahri.(2018). Analysis of feature categories for malware visualization. *Journal of Telecommunication, Electronic and Computer Engineering*. 10 (3-2). pp. 1-5. ISSN 2180-1843

Selamat, N. S., & Mohd Ali, F. H. (2019). Comparison of malware detection techniques using machine learning algorithm. *Indonesian Journal of Electrical Engineering and Computer Science.* http://doi.org/10.11591/ijeecs.v16.i1.pp435-440

Zakaria, W. A., Abdullah, M. F., Mohd, O. and Ariffin, A. (2017). The Rise of Ransomware. *International Conference on Software and E-Business - ICSEB 2017*. Pages 66–70. https://doi.org/10.1145/3178212.3178224