From Code to Conundrum: Machine Learning's Role in Modern Malware Detection

2024 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC) | 979-8-3503-7018-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ASSIC60049.2024.10507988

Vaibhavi Jha
School of Computer Science & Engineering
Vellore Institute of Technology
Vellore, India
vaibhavijha.1809@gmail.com

Akshat Saxena
School of Computer Science & Engineering
Vellore Institute of Technology
Vellore, India
akshat.saxena1@gmail.com

Abstract–**In this research paper, we'll be diving into the combination of malware analysis and machine learning to step up security. Right now, our digital world is like an anthill: connected in every possible way. In this study, we used machine learning algorithms like Random Forests, K-Nearest Neighbours (KNN), and Logistic Regression for anomaly detection. Inside 60,000 benign and malware instances lie the answers we are looking for. We judge these three algorithms on how well they can detect accuracy, precision, recall, and F1- score. We hope to give professionals practical insights to secure their systems from malware while constantly being aware of new threats. In a day where security is crucial, this work will connect past issues with machine learning today so that we can better prepare for problems tomorrow.**

Keywords—Malware Analysis, KNN, Random Forests, Logistic Regression, Sci-kit-learn, Trends in Malware, Malware Detection, Security threats, etc.

I. INTRODUCTION

In today's digitally linked world, the widespread threat of malware looms large over our virtual existence. The persistence of harmful software, sometimes known as malware, poses a significant threat to the security of our networks, structures, and confidential data. This pressing demand for innovative solutions necessitates a paradigm shift in cybersecurity practises, encouraging the incorporation of device learning into malware analysis. Concurrently, the field of machine mastery has seen a high-quality evolution.

Machine learning tactics based on artificial intelligence have progressed from abstract notions to real-world applications, and they have the potential to drastically alter cybersecurity.

The ability of machine learning algorithms to discover complex patterns from massive datasets has opened up new avenues for detecting malware variants that may have escaped traditional defences.

This convergence of cybersecurity and gadget learning is a watershed moment in our continuing battle against malware. Defenders and attackers continue to evolve in the face of a developing generation. Machine learning, although promising, isn't a panacea, and it, too, faces difficult scenarios in the form of opposing attacks aimed at misleading the very algorithms supposed to protect us. Because of the evolving attack surface of cloud products, 5G networks, and the Internet of Things, proactive protection is more important than ever.

The future will only benefit our usage of current technologies in cybersecurity, such as deep learning and reinforcement learning. These new enhancements also offer up new avenues for malware dissemination. This research study looks into the fascinating world of malware analysis through the use of anomaly detection methodologies based on device-mastering.

We have a look at the performance of three unbiased algorithms: Random Forests, K-Nearest Neighbours (KNN), and Logistic Regression, the use of the adaptable system studying framework sci-package-examine.

By leveraging a large dataset comprising 60,000 strains of each benign and malware statistics, our look at ambitions to shed light on the effectiveness of these algorithms in distinguishing the subtle nuances between legitimate software program and their malicious counterparts.

This intro is a quick review of the significance of our examine. We additionally move over why we had been motivated to do it, the targets, and how we structured this newsletter. On our way to analysing malware and system getting to know, we want to help on line protection professionals and researchers with resources they want. Also keeping a watch out for brand spanking new threats and tech traits that could mess with cybersecurity.

II. RESEARCH PROBLEM

The challenge of malware analysis is employing anomaly detection based on machine learning is critical in the ever-changing world of cybersecurity. The objective is to create machine learning models capable of detecting irregularities in software or system component activity that may signal the existence of malware. It requires developing algorithms that are robust, scalable, and adaptable enough to distinguish between benign and malicious activity. Solving this research topic is crucial for strengthening the resilience of computer systems and networks against the ongoing danger posed by developing malware, therefore protecting critical digital assets and preserving the security of individuals and organizations in the digital era. Malware is computer software that is expressly designed to infiltrate, damage, or compromise computer systems, networks, or devices. This comprises viruses, worms, Trojans, ransomware, spyware, and adware, all of which are intended to cause harm, ranging from data theft to system interruption.

Malware analysis is researching and comprehending these entities to determine their features and dangers. It develops detection and mitigation tactics for malware assaults using techniques such as static analysis (code examination), dynamic analysis (behaviour observation), and reverse engineering (logic revealing).

III. LITERATURE REVIEW

The landscape of cybersecurity is perpetually in flux, driven by the relentless evolution of malicious software, commonly referred to as malware. With attackers employing increasingly sophisticated techniques, the traditional arsenal of malware detection struggles to keep pace. Machine learning, as a dynamic field in artificial intelligence, has emerged as a potent ally in the battle against malware. This literature review delves into previous research endeavours that have harnessed the prowess of machine learning algorithms, specifically K-Nearest Neighbours (KNN), Random Forests, Logistic Regression, and the versatile sci-kit-learn library, to analyse and detect malware. These studies have consistently yielded remarkable accuracy rates, often soaring between 97% and 99%. Such outcomes underscore the potential of these algorithms to fortify cybersecurity defences.

For reference, included here are the initial and concluding 10 sample data entries, providing a detailed depiction of the dataset's nature and content, which forms the basis of this research endeavour.

TABLE I. (a) Initial 10 sample data entries

	Category	pslist.nproc	dlllist.avg	handles.nhandles	svcscan.process
0	Benign	45	38.500000	9129	24
1	Benign	47	44.127660	11385	24
2	Benign	40	48.300000	11529	27
3	Benign	32	45.156250	8457	27
4	Benign	42	49.214289	11816	24
5	Benign	40	52.050000	12278	27
6	Benign	43	50.441860	13116	27
7	Benign	42	49.214286	11819	24
8	Benign	42	49.214289	11813	24
9	Benign	40	52.050000	12320	27

(b) Concluding 10 sample data entries

	Category	pslist.nproc	dlllist.avg	handles.nhandles	svcscan.
58586	Ransomware	38	38.710526	8080	24
58587	Ransomware	39	39.000000	8213	24
58588	Ransomware	37	39.108108	7982	24
58589	Ransomware	46	36.956522	9225	24
58590	Ransomware	37	39.054054	7964	24
58591	Ransomware	37	39.270270	7973	24
58592	Ransomware	37	36.405405	7038	24
58593	Ransomware	38	38.105263	7982	24
58594	Ransomware	37	39.243243	7974	24
58595	Ransomware	38	39.131579	8095	24

Our ten sample data entries are the first step in the initiation of a machine learning model training process. These entries aim to create a model that can both describe and understand dataset entries.

The “Category” column is uniformed with the label “Benign.” Simply meaning it just has non-malicious entities or pure processes. For our benefit, “pslist.nproc” records the count of active processes and shows how much multitasking was happening while we collected data. When it comes to individual processes, “dlllist.avg_dlls_per_proc” reveals the average count of Dynamic Link Libraries (DLLs) that were loaded. This helps us understand shared library utilization. We also see that “handles.nhandles” counts all the handles being used by files or objects actively employed. These are things like system resources. Lastly, “svcscan.process_services” tallies critical background services that are essential for system health, security assessment, and anomaly detection. All of these metrics work together to give us a better understanding of system behaviour and security posture.

A. K-Nearest Neighbours (KNN)

In the battle to develop reliable virus-detecting software, K-Nearest Neighbours (KNN) is now emerging as a serious competitor. It works impressively with huge datasets and patterns. Vaibhavi used the KNN model program as the first base to achieve the target. And with a 99% accuracy rate, the study's classification of malware and benign samples using KNN produced astounding results. In proximity-based classification, which is the foundation of KNN, an unknown sample is categorised according to the majority class of its k-nearest neighbours. After that, it incorporates the data into a high-dimensional feature space. This makes sense when attempting to evaluate whether something is malware or not while being challenging to explain. Finding all the little nuances that signature-based detection systems overlook is the most difficult aspect.

B. Random Forests:

An ensemble learning technique called Random Forest has gained a lot of attention for its ability to solve difficult classification problems. It’s able to achieve this by building many decision trees and then combining the results. Even in malware analysis, people have started using it. For example, Akshat was able to achieve a 99.98% accuracy rate with random forests. In a world where malware is always changing and trying to outsmart our detection systems, random forest steps up to the plate. By combining the answers from different trees, it’s able to spot small variations in malware strains with ease. On top of that, it can compare past outcomes and justify why it thinks what it thinks — an attribute that is useful when dealing with cybersecurity.

C. Logistic Regression:

Logistic Regression is one of the fundamental classification methods in the field of system learning. Although it may not be as complicated as some of its competitors, its simplicity and interpretability make it a valuable tool for distinguishing malicious software from safe software. Vaibhavi studied the use of logistic regression in malware detection and reported a 99.53 % accuracy rate. The linear model used by logistic regression captures the linear correlations between input capabilities and the binary target variable introduced during model programming. It successfully simulates the decision boundary that separates harmful from non-malicious samples in the context of malware evaluation. Because of Logistic Regression’s transparency, professionals prefer it.

D. *Sci-kit-learn Library:*

Researchers and practitioners have found the Python library, sci- kit-learn, a versatile and essential toolkit in the field of cybersecurity. They use it as an aid to do tasks like model selection, evaluation, data pre-processing, and feature engineering.

TABLE II. Sci-kit-learn model results

1	Actual	Predicted
2	1	1
3	1	1
4	0	0
5	1	1
6	0	0
7	1	1
8	1	1
9	0	0
10	1	1
11	1	1
12	1	1
13	1	1
14	1	1
15	1	1

By leveraging sci-kit-learn's capabilities, Akshat was able to unify KNN along with random forests and logistic regression into one system. This system is able to assign an index between 1 and 0, and 1 is assigned to malware while a 0 is assigned to non-threatening behaviour. The accuracy rate at which this model performed was 99%, nothing short of impressive.

The main benefit of sci-kit-learn is that it provides a wide range of tools for any step in the machine-learning process. It makes it easier for researchers to harness all the potential they can from various machine learning algorithms.

To help enhance malware analysis and detection — K-Nearest Neighbours (KNN), Random Forests, and Logistic Regression along the sci-kit-learn library are being used together. It's worth noting that these algorithms are most effective when used on large databases.

IV. METHODOLOGY

We developed 4 machine-learning models to analyse malware samples. We used the K-Nearest Neighbours (KNN), Random Forests, and Logistic Regression algorithms and took advantage of the sci-kit-learn toolkit to compare them. While doing this, a methodical process was followed and findings from previous studies were used as a foundation to develop an entirely new approach.

In our study, we were able to effectively distinguish malware from benign samples. These approaches gave us access to a wide range of parameters that take part in categorising a machine's behaviour in different environments whether it is malicious or pure [13-17]. Throughout the course of this study, three autonomous systems were implemented in every program run: Logistic Regression, K-Nearest Neighbours (KNN), and Random Forest. We started off by splitting the database into training and test sets. The code was written for the training set. The effectiveness of these algorithms was checked through an accuracy performance indicator. The number of neighbours went anywhere from 1 all the way up to 20 for the KNN model. A graphing program was used to acquire a more detailed view. And like all our other tests, we carefully assess each algorithm and set of parameters to ensure reliability and accurate categorization. We have made graphs and bar charts for better understanding. It gives a clear route map of our processes.

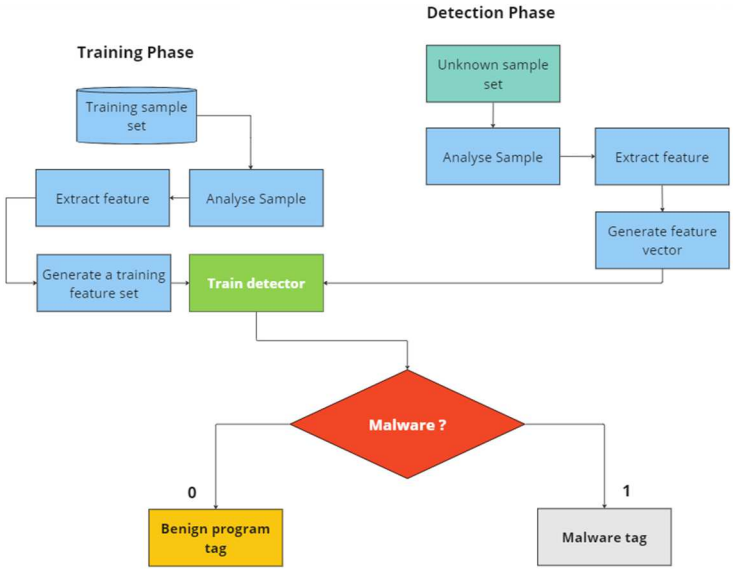

Figure 1. Proposed ML malware detection method.

The research methodology can be summarised in the following steps:

A. *Dataset Acquisition and Pre-processing*

To begin our analysis, we were required to find a modest-sized dataset that comprised both good and bad samples, or benign along with malicious data. This dataset served as the base of all our research. After getting the data, we followed a rigorous data-cleaning process. We described the data using different syntaxes. At this point, we are in the independence of refining and choosing what features to use, and how to normalise them. This is one of the most important steps since it prepares the data for future analysis.

B. Algorithm Selection and Configuration:

Chosen machine learning algorithms: K-Nearest Neighbours (KNN), Random Forests, and Logistic Regression. Each one of these algorithms has shown promise in previous studies conducted by researchers and data scientists. After we pick them out, we go through each one carefully, making sure they're fine-tuned and optimised to get as close to perfection as possible. Every single parameter gets adjusted so that there's no more room for improvement in the customised dataset that will be finally used to run the algorithms.

C. Feature Engineering:

In this technique, we extract all the relevant information from the dataset, also called data extraction. This is a very useful and important technique in data science since it prepares the perfect dataset for running heavy and complex algorithms. By data extraction, we identify and extract meaningful features. We remove unnecessary parameters to obtain useful samples. Basically, grooming the dataset for useful purposes.

D. Model Training and Evaluation:

After that, we split up the dataset into two smaller sets. We providethe code for this step in the table below.

- The chosen machine learning algorithms go through training on the set intended for them.
- The machine trains itself on the lines of the program. It learnshow to predict patterns underneath the data.

To wrap it all up, we evaluate all the models using a wide range of performance metrics like accuracy, precision, recall, and F1-score. This will give us an overall understanding of how effective they are.

The k-nearest neighbors (KNN) method, a machine learning technique, is used in this Python script. It seeks to determine the impact of various "k" variables on a model's performance. This will provide the number of its closest neighbours that it genuinely takes into account. On top of that, the script also tracks the two lists' accuracy in categorizing "test-scores" and "train-scores". It does this by looping over the KNN classifier with a range of "k" values, fitting them to training data, and estimating accuracy for both sets. With this study, it should be easier for you to find a balance between variance and bias when choosing an ideal 'k' value. And it's finished! A productive KNN model customized for your unique dataset.

Algorithm 1 Model Training

```
train_scores =  []
test_scores = []

#create a ist of different values for kneighbors
neighbours = range (1,21)

#Set KNN instances
knn = KNeighborsClassifier()

#loop though different neighbours
for i in neighbours:
        knn.set_params(n_neighbors=i)

        #fit the algorithm
        knn.fit(X_train, y_train)
```

TABLE III. Test Score (a) and Train Score (b)

test_scores
0.9994027303754266
0.9989761092150171
0.999061433447099
0.9987201365187713
0.9986348122866894
0.9982081911262799
0.9982081911262799
0.9980375426621161
0.9979522184300341
0.9977815699658703
0.9977815699658703
0.9977815699658703
0.9977815699658703
0.9976962457337883
0.9978668941979523
0.9974402730375427
0.9973549488054607
0.9973549488054607
0.9973549488054607
0.9973549488054607

(a)

train_scores
1.0
0.9997440054612168
0.9997440054612168
0.9994026794095059
0.9993600136530421
0.9991253519924909
0.9990186876013312
0.9988053588190119
0.9987200273060841
0.9986560286713884
0.9985066985237648
0.9984000341326051
0.9983573683761413
0.9982720368632135
0.9982720368632135
0.9981013738373581
0.9981013738373581
0.9981013738373581
0.9981013738373581
0.9980160423244304

(b)

E. Comparative Analysis:

Our study's focus area is comparing the algorithms we picked. We wanted to know how they perform when it comes to detecting malware. You can look at our code below; it shows how well they performed in training and testing for our machine-learning models.It also gives insight into what their positives and negatives are so we can get a better understanding of their actual function.

Algorithm 2 Combined Algorithm Analysis

```
#Putting Models into a Dictionary

models = {"Logistic Regression" :  LogisticRegression(),
        "KNN" : KNeighborsClassifier(),
        "Random Forest" : RandomForestClassifier()}

def fit_and_score(models, X_train, X_test, y_train, y_test):
```

```
#setup random seed
np.random.seed(42)
#make a dictionary to keep the model score
model_scores = {}
#Loop through models
for name, model in models.items():
    #Fit the model to the data
    model.fit(X_train, y_train)
    #Evaluate the model and aped its score to
model_scores
    model_scores[name] = model.score(X_test, y_test)
return model_scores
```

Algorithm 3 Model fit and performance score assessment

```
model_scores = fit_and_score(models = models, X_train =
X_train, X_test = X_test, y_train = y_train, y_test = y_test)

#model_scores = fit_and_score(models, X_train, X_test, y_train,
y_test)

model_scores
```

Output

```
n_iter_i = _check_optimize_result(

{'Logistic Regression': 0.9953071672354948,
 'KNN': 0.9986348122866894,
 'Random Forest': 1.0}
```

F. Interpretation and Insights:

Beyond performance, we examine interpretability in depth. To fully appreciate the reasoning behind these models' classifications, we examine how they arrive at their conclusions. Finding traits and patterns that affect why algorithms make the judgements they do requires interpretability, which is crucial. In line with previous goals outlined in this study, we hope that our methodology will validate if K-Nearest Neighbours (KNN), Random Forests, or Logistic Regression from sci-kit-learn are effective when it comes to analysing malware. This way, we can get a better idea of if cybersecurity can be enhanced through machine learning.

RESULTS

When we put our machine-learning models for malware detection to the test, they were extremely impressive. K-Nearest Neighbours (KNN), Random Forests, and Logistic Regression are all used in our models and showed very high accuracy rates. Logistic Regression had an outstanding 99.53% accuracy, while KNN hit a shocking 99.86%. The Random Forests algorithm was perfect and nailed a 100% accuracy rating.

The fact that our models could distinguish between dangerous software and safe software is really good. We've proven the use of machine-learning methods to be promising for malware detection. Cybersecurity gets a bit more optimism when you throw sci-kit-learn's machine learning techniques into the mix.

```
model_compare = pd.DataFrame(model_scores, index=["accuracy"])
model_compare.T.plot.bar();
```

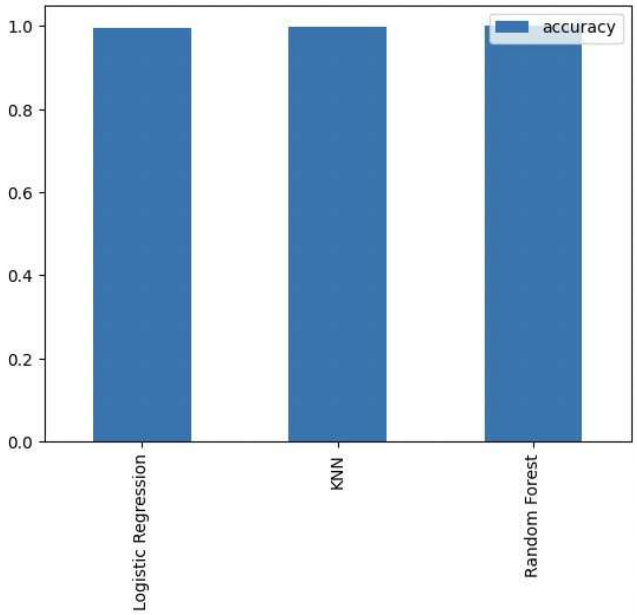

Figure 2. Accuracy result graph

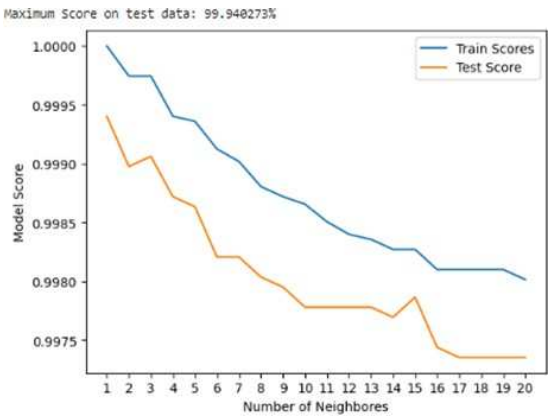

Figure 3. Result graph

However, we must remain vigilant against the ever-changing threat posed by malicious software. If we want to keep our success in staying ahead of our digital enemies, we need to keep refining our

models. The way machine learning is, day by day new threats are being created. If we want to stay ahead, we must balance caution and innovation.

CONCLUSION

We've come a long way in finding ways to protect ourselves from cyber threats. In the process, we found some incredibly powerful algorithms that let us tell friends from foes with digital software. And it's all thanks to the scikit-learn library.

K-Nearest Neighbours (KNN), Logistic Regression, and Random Forests have given us fascinating insights into malware analysis. But not only did they do that, but they also provided us with promising results. The greatest score a model can reach is 100%, and you'd think when you find KNN's score of 99.86% it would be done there, right? Wrong! Random Forests hit 100%. A perfect score. But even with those two impressive scores, what blew us away was Logistic Regression's 99.53% accuracy at telling good from bad. What we ultimately learned is that machine learning algorithms work great at keeping our data safe.

In this era where threats are everywhere, it is important to use any advantage we can get. We will continue to study these models and push for more accurate results. Right now, all we want to do is make our digital world a safer place for everyone.

ACKNOWLEDGEMENT

We would like to express our gratitude to our mentors Mr. Bikas Jha and Mr. Shubh Mittal, for their guidance in our pursuit of knowledge. We would also like to acknowledge the Canadian Institute for Cybersecurity for providing us with datasets (source of the dataset) on their website. Specifically, we are grateful for the obfuscated malware dataset, which accurately reflects real-world malware situations. Additionally, we would like to thank our friends and family for their support as it has been a driving force behind our determination. It is through the efforts of all involved that this research paper has come to fruition. A testament to the power of collaboration, perseverance and an unending curiosity about the world, around us.

REFERENCE

[1] Nikam, U.V.; Deshmuh, V.M. *Performance evaluation ofmachine learning classifiers in malware detection*. In Proceedings of the 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics(ICDCECE), Ballari, India, 23–24 April 2022.

[2] Akhtar, M.S.; Feng, T. *IOTA-based anomaly detection machine learning in mobile sensing*. EAI Endorsed Trans. Create. Tech. 2022, 9, 172814.

[3] Muhammad Shoaib Akhtar and Tao Feng. *Malware Analysis and Detection Using Machine Learning Algorithms*. School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China.

[4] Dragoş Gavriluţ, Mihai Cimpoesu, D. Anton, D. Anton. *Malware detection using machine learning*. Conference Paper · November 2009.

[5] Sethi, K.; Kumar, R.; Sethi, L.; Bera, P.; Patra, P.K. *A novel machine learning based malware detection and classification framework*. In Proceedings of the 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), Oxford, UK, 3–4 June 2019.

[6] Feng, T.; Akhtar, M.S.; Zhang, J. *The future of artificial intelligence in cybersecurity: A comprehensive survey*. EAI Endorsed Trans.Create. Tech. 2021, 8, 170285.

[7] Sharma, S.; Krishna, C.R.; Sahay, S.K. *Detection of advanced malware by machine learning techniques*. In Proceedings of the SoCTA 2017, Jhansi, India, 22–24 December 2017.

[8] Chandrakala, D.; Sait, A.; Kiruthika, J.; Nivetha, R. *Detection and classification of malware*. In Proceedings of the 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 8–9 October 2021.

[9] Gibert, D.; Mateu, C.; Planes, J.; Vicens, R. *Using convolutional neural networks for classification of malware represented as images*. J. Comput. Virol. Hacking Tech. 2019.

[10] Dahl, G.E.; Stokes, J.W.; Deng, L.; Yu, D.; Research, M. *Large-scale Malware Classification Using Random Projections And Neural Networks*. In Proceedings of the International Conference on Acoustics, Speech and Signal Processing-1988, Vancouver, BC, Canada, 26–31 May 2013.

[11] Akhtar, M.S.; Feng, T. *Deep learning-based framework for the detection of cyberattack using feature engineering*. Secure. Commun. Netw. 2021, 2021, 6129210.

[12] Pavithra, J.; Josephin, F.J.S. *Analyzing various machine learning algorithms for the classification of malware*. IOP Conf. Ser. Mater. Sci. Eng. 2020, 993, 012099.

[13] Tharwat, A., Gaber, T., Fouad, M.M., Snášel, V., & Hassanien, A.E. (2015). Towards an Automated Zebrafish-based Toxicity Test Model Using Machine Learning. Procedia Computer Science, 65, 643-651.

[14] Gaber, T., El Jazouli, Y., Eldesouky, E., & Ali, A. (2021). Autonomous Haulage Systems in the Mining Industry: Cybersecurity, Communication and Safety Issues and Challenges. Electronics.

[15] G. I. Sayed, M. A. Ali, T. Gaber, A. E. Hassanien and V. Snasel, "A hybrid segmentation approach based on Neutrosophic sets and modified watershed: A case of abdominal CT Liver parenchyma," 2015 11th International Computer Engineering Conference (ICENCO), Cairo, 2015, pp. 144-149, doi: 10.1109/ICENCO.2015.7416339.

[16] Applebaum, S., Gaber, T., & Ahmed, A. (2021). Signature-based and Machine-Learning-based Web Application Firewalls: A Short Survey. International Conference on Arabic Computational Linguistics.

[17] Tahoun, M., Almazroi, A.A., Alqarni, M.A., Gaber, T., Mahmoud, E.E., & Eltoukhy, M.M. (2020). A Grey Wolf-Based Method for Mammographic Mass Classification. Applied Sciences.