Amalgamating Ensemble Machine Learning Soft Voting Classifier, SMOTE, and Pearson's Correlation Coefficient for Enhanced Malware Detection

Mustafa Jumaah
Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah 61004, Iraq
pgs.mustafa.jumaa@uobasrah.edu.iq

Ali A. Yassin
Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah 61004, Iraq
ali.yassin@uobasrah.edu.iq

Zaid Ameen Abduljabbar
Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah 61004, Iraq | Department of Business Management, Al-Imam University College, Balad 34011, Iraq
zaid.ameen@uobasrah.edu.iq (corresponding author)

Muwafaq Jawad
Directorate General of Education Basra, Ministry of Education, Basra 61001, Iraq
pgs.muwafaq.abbas@uobasrah.edu.iq

Vincent Omollo Nyangaresi
Department of Computer Science and Software Engineering, Jaramogi Oginga Odinga University of Science and Technology, Bondo 40601, Kenya | Department of Applied Electronics, Saveetha School of Engineering, SIMATS, Chennai, Tamil Nadu 602105, India
vnyangaresi@jooust.ac.ke

Ali Hasan Ali
Department of Mathematics, College of Education for Pure Sciences, University of Basrah, Basrah 61004, Iraq | Technical Engineering College, Al-Ayen University, Thi-Qar 64001, Iraq | Institute of Mathematics, University of Debrecen, Pf. 400, H-4002 Debrecen, Hungary
ali.hasan@science.unideb.hu

Received: 1 February 2025 | Revised: 23 February 2025 and 1 March 2025 | Accepted: 6 March 2025

 DOI: https://doi.org/10.48084/etasr.10420

ABSTRACT

Obfuscated malware poses a significant threat to personal and IoT devices, and traditional detection methods often face significant challenges and weaknesses in their capabilities and performance. This study proposes a malware detection approach using Machine Learning (ML) algorithms and a soft voting ensemble technique, enhanced by the Pearson's correlation coefficient for feature selection on the CIC-MalMem-2022 dataset. It addresses data imbalances with the Synthetic Minority Oversampling Technique (SMOTE) method and employs various ML classifiers. The results demonstrate improved accuracy, precision, and recall in malware detection compared to single classifiers and traditional methods. The

research model is evaluated using a confusion matrix and evaluation metrics, and achieves 99.99% accuracy rate, 99.99% classification rate, 99.99% precision rate, 99.99% recall rate and 99.99% F1 score, surpassing the results of previous studies. These results indicate that the combination of feature selection and ensemble learning can significantly improve the efficiency and security of high-performance malware prediction systems, paving the way for advanced threat mitigation strategies.

Keywords-machine learning; malware detection; SMOTE; IoT; feature selection

I. INTRODUCTION

The Internet of Things (IoT) is a vast ecosystem in which everyday objects ranging from smartphones, vehicles, watches, refrigerators, and ovens to cameras, heart monitors, smart assistants, and even buildings, such as hospitals, homes, and stores are integrated with electronics and software that are equipped to connect, communicate, and exchange data [1]. To date, billions of devices and sensors make up the vast IoT ecosystem, and as this number continues to grow, so do the attack vectors [2]. A wide range of vulnerabilities expose more devices to the Internet, increasing the risk of compromising systems in this environment [3, 4]. Unlike previous network environments, physical networks have resource-constrained capabilities and cannot run the resource-intensive algorithms required for security [5, 6]. Malware can operate in multiple forms and functions, such as a weapon in cyberattacks, a delivery mechanism that enables long-term cyberattacks, and an enabler of attacks, including the development of advanced cyberattacks and ransomware campaigns [7]. As a result, it serves as a core tool for executing cyberattacks, helping attackers infiltrate systems, compromise data, and disrupt operations, and is well integrated into cybercriminals' strategies and objectives [8]. As the frequency of intrusions increases, there is an urgent need for security measures to protect these devices from such attacks [9]. This emerging paradigm of IoT networks is primarily built on resource-constrained and low-power devices, which makes debugging and providing security cumbersome [10, 11]. As malware attacks have increased over the past few years, traditional methods have repeatedly failed to detect the latest growing threats and new forms of malware [12]. As a result, specialized security mechanisms such as robust Machine Learning (ML) algorithms are required [13], which have recently gained attention as advanced and powerful techniques for distinguishing between malware and benign files [14]. ML is an increasingly popular and effective tool for enhancing cyber defense and digital forensics [15]. A critical area of focus is malware detection, which is essential to address potential harms, including privacy violations and risks to human safety [16]. Current security measures have been inadequate in addressing these issues. Although ML algorithms can detect malware, they require further analysis of specific datasets [17], which often contain a larger number of features than required for optimal classification [18]. The large number of features imposes a burden on the system, resulting in increased processing time and diminished overall performance [19]. This highlights the need for further research to develop effective and innovative methods for detecting sophisticated and adaptive malware. However, little work has been done on discovering the best features for malware detection [20]. Extraneous features need to be removed from the feature set before being fed into the ML algorithm for further processing [21, 22]. Feature selection is a critical process because it can significantly affect the performance of the model, either negatively or positively [23].

This study investigates the application of ML algorithms for malware classification, aiming to enhance their accuracy, convergence, and robustness. The CIC-MalMem-2022 dataset is utilized to identify relevant features and refine them for comprehensive analysis. This study makes fundamental contributions to the field of malware detection by:

1. Employing a combined soft voting mechanism by integrating five ML classifiers: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), and Logistic Regression (LR) to mitigate individual model bias and improve generalization.
2. Implementing Synthetic Minority Oversampling Technique (SMOTE)-based class balancing to address dataset imbalance by intentionally oversampling minority classes to ensure effective identification of underrepresented malware types.
3. Introducing the Pearson correlation coefficient into the feature selection process, which minimizes the computational burden by eliminating redundant information and preserving only 18 out of 55 features.
4. Evaluating current contributions and methodologies, emphasizing the application of ML techniques in malware detection and the development of sophisticated systems.

II. RELATED WORKS

ML has been essential in improving the identification of obfuscated malware in cybersecurity systems [24]. This section examines the significant studies that employ ML approaches to address the problems posed by advanced malware variants. Authors in [25] proposed malware detection and classification methods using memory information from the CIC-MalMem-2022 dataset, which contains both benign and malicious samples. First, they conducted optimization tests on classic ML approaches and then developed an extended Convolutional Neural Network (CNN). After memory analysis, the extended Deep CNN (DCNN) was able to detect obfuscated malware with an accuracy of 0.99. The RF model was the most accurate. However, the proposed neural network architecture classified malware families the best with 0.83 accuracy. This result suggests that the model cannot distinguish specific malware variants in complicated classification tasks. Due to its complicated architecture with numerous neurons and convolutional layers, the DCNN is computationally intensive and difficult to run on low-resource hardware.

Authors in [26] developed a deep stack model by combining the prediction outputs of weak learners, i.e., CNNs,

and feeding them as learning inputs to a meta-learner, i.e., Multilayer Perceptron (MLP). An interpretable AI-based approach was used to interpret and validate the final results. The proposed approach achieved 99.8% accuracy in analysing obfuscated Windows malware memory dump files. However, this approach cannot detect advanced polymorphic malware using the models applied. Authors in [27] introduced an improved 1D-CNN model, which can quickly sort IoT security data into groups. The suggested design has input, convolutional, self-tuning, and output layers. It uses Gaussian error linear unit activation, sealing, and normalization techniques to improve performance and reduce overfitting. The model was rigorously assessed using three benchmark datasets: CIC IoT 2023, CIC-MalMem-2022, and-CIC-IDS2017. It achieved exceptional performance metrics on the CIC-MalMem-2022 dataset, with 99.90% accuracy, 99.98% precision, 99.97% recall and 99.96% F1 score, highlighting its effectiveness in detecting and classifying diverse IoT-related attacks and malware. Despite these promising results, the model's reliance on a static training process limits its applicability, restricting its adaptability to new attack vectors and the dynamic evolution of threat landscapes. This drawback highlights a crucial area for additional research to enhance the model's robustness in real-world, dynamic security environments.

Despite previous efforts to improve malware classification, several obstacles persist, including the influence of irregular data balance on model accuracy, the absence of efficient feature selection methods, and the high computational complexity. This paper presents an approach that reduces computational dimensionality and enhances classification performance by selecting the most influential features using the Pearson correlation coefficient. In addition, the SMOTE technique is used to address the imbalance between classes, which reduces the possibility of model bias towards the dominant class. Finally, the application of the soft voting classifier improves the stability and accuracy of the results compared to the single-factor approaches used in some previous studies.

III. METHODOLOGY

The malware detection problem in this study is structured as a binary classification problem, where benign applications are class 0, and malicious programs are class 1. This section describes the proposed methodology, which uses a voting classification technique and several models to achieve a high level of accuracy in the classification results of benign and malicious programs. The proposed methodology is divided into three stages: stage A considers the preprocessing of the dataset, stage B considers the feature selection, stage C considers the splitting of the dataset, and stage D includes the classifier and prediction, as shown in Figure 1.

A. Data Preprocessing

The data preprocessing stage includes the management of missing values and the use of binary label datasets and several data preprocessing techniques to improve the dataset quality and ensure efficient classification with minimal operations. The main tasks of this stage include cleaning the data and encoding the labels.

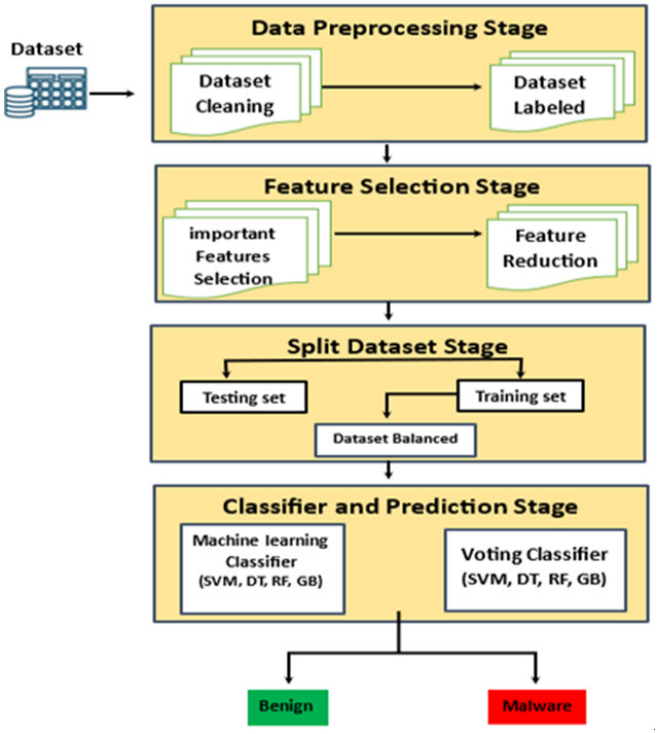

Fig. 1. Diagram of the proposed approach.

1) Data Cleaning

Data cleaning is a fundamental and necessary step before starting any data analysis process, as the data quality directly affects the results. After obtaining the CIC-MalMem-2022 dataset, which consists of 58,596 rows and 57 features, we use data preprocessing techniques, including strategies for handling missing values and removing duplicates, to properly prepare the data and analyze their impact on model performance. Initially, only 55 features are used because 2 of the 57 features in the dataset are considered insignificant. The "Category" feature is used for identification only, and the "Class" feature is a target value column that is used for labeling and then eliminated from the dataset. In addition, if a significant number of values are missing in any row or column, that specific row or column is removed to maintain data integrity and accuracy. If certain values are missing in a column or row, the average is calculated to impute the data. Moreover, 534 completely duplicate rows were removed, resulting in a final total of 58,062 rows. The dataset remains accurate and clear, eliminating the need for noise reduction or further preprocessing.

2) Label Encoder

After cleaning the data, the next step in preprocessing is to convert the category data into numerical values so that the computational models can process them. The target "Class" record was encoded by assigning a value of 1 to "malware" and a value of 0 to "benign", as shown in the Figure 2. Qualitative

attributes and variables scored on a ratio scale typically influence the dependent variable in classification analysis. Therefore, these types of variables were converted to numeric values using encoding techniques, since ML algorithms only accept numeric inputs.

	Category	pslist.nproc	pslist.avg_threads	callbacks.ngeneric	Class
0	0	45	10.555556	8	0
1	0	47	11.531915	8	0
2	0	40	14.725000	8	0
3	0	32	13.500000	8	0
4	0	42	11.452381	8	0
...	...	...	...	...	...
58591	9362	37	10.108108	8	1
58592	9282	37	9.945946	8	1
58593	9411	38	9.842105	8	1
58594	9325	37	10.243243	8	1
58595	9042	38	9.868421	8	1

Fig. 2. Label encoder on the dataset.

B. Feature Selection

After the data preprocessing stage, the feature selection process is performed to extract the most important and relevant features from the dataset. The identified features must have a high correlation with the "Class" and a low correlation with each other to enable the model to achieve better training results. Feature selection enriches the data or features and provides the learning procedure with rich and homogeneous information to improve the overall performance. In the proposed dataset, we select the most important features and then feed these features to train and evaluate the ML model [28].

C. Data Splitting

Data splitting produces three parts: a testing set, a validation set, and a training set. The training set instructs the model to determine whether an application is benign or malicious based on the input data. The testing set allows the model to be evaluated using standard evaluation metrics. The results of the evaluation metrics allow us to adjust or improve the approaches employed in the data preprocessing and partitioning phases, thus improving the model's performance. The implemented measures allow comparisons with ongoing investigations.

Cross-validation is a replication technique that divides a dataset into subsamples that are systematically alternated in their application. This methodology includes an outer loop for the training and testing datasets and an inner loop for the training and validation datasets. The outer loop evaluates the generalization error by computing the mean test set scores across all dataset partitions. The inner loop aims to implement the ideal model for each training set and determines hyperparameters using the validation set to improve the results [29].

D. Classification and Prediction

This section provides a concise overview of the classifiers employed in this research.

1) Voting Classifier

An ensemble classifier is a type of classifier that utilizes AI models. It combines a set of distinct models into a single system that integrates the advantages of each model to yield the most accurate predictions [30]. The proposed methodology uses five conventional ML models in modern research. RF is an ensemble method that integrates the outputs of numerous decision trees to yield a single result. SVM categorizes data points by transforming them into a high-dimensional feature space. Even if the classes are not linearly separable, the data are transformed to facilitate hyperplane delineation that optimally divides the data into two classes. GB classification is an ensemble method that integrates several weak learning models to improve prediction accuracy. This model addresses both regression and classification issues by developing a predictive model from a collection of weak models. LR is the most effective technique for predicting the occurrence of a binary outcome based on one or more independent variables. The classifier operates within a probabilistic framework, producing a probability value for each input model. It identifies the input model with the greatest likelihood as the definitive result. The methodology begins with fundamental preprocessing techniques to enhance the dataset, followed by feature selection for a voting classifier that precisely categorizes detection types as benign or malicious [31].

IV. EXPERIMENTAL EVALUATION

The experimental evaluation process was conducted utilizing Python and ML libraries within the Anaconda environment on a computer with the following specifications: 13th generation Intel(R) Core (TM) i7-1355U CPU @ 1.70 GHz, 16.0 GB RAM, 2 GB graphics card, and Windows 11 Pro operating system.

A. Dataset Description

The CIC-MalMem-2022 is an academic dataset, published by the Canadian Cybersecurity Institute for research in malware classification, especially obfuscated malware [32]. It was created by extracting features from memory dumps to address memory-based obfuscation attacks, and consists of malware from recent actual cyber incidents. It contains 58,596 entries, evenly split between 50% benign and 50% malicious instances. The malicious dumps come from three types of malwares: trojan horses, spyware and ransomware. It contains 57 features with two categories of labels: one indicating whether the samples malicious and the other indicating their malware family. Among the malicious samples, 16.19052% are trojan horse malware, 17.10014% are spyware, and 16.70933% are ransomware, as shown in Figure 3.

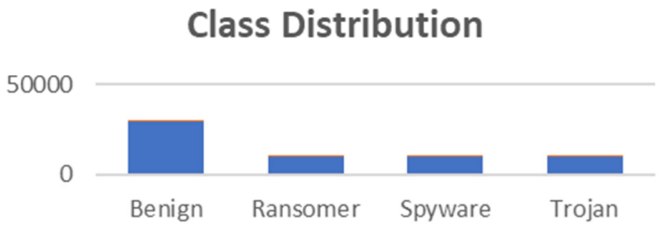

Fig. 3. Distribution of malware categories.

B. Experimental Results: Baseline

Initial experiments focused on taxonomic features and missing and duplicate values, which some classifiers cannot handle. The label encoding converted all taxonomic features into numbers, and missing or duplicate values were excluded. In addition, the initial experiments did not use validation sets or hyperparameter tuning, and the data were split into 80% training and 20% testing from a random sample. Figure 4 presents a summary of the results obtained for all classifiers, focusing on accuracy. Table I shows the experimental results obtained for each classifier, demonstrating consistently good performance of all models with minimal variation.

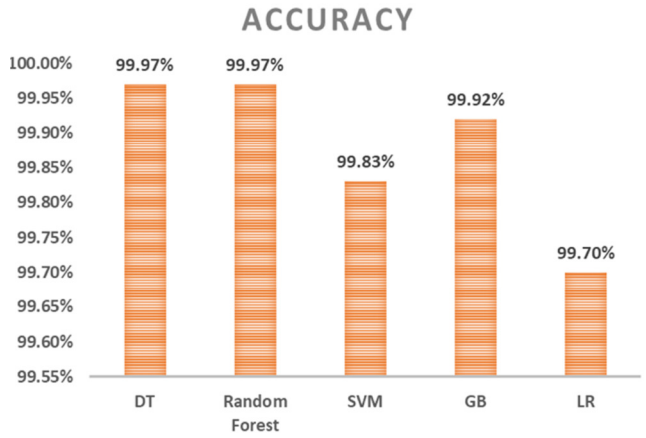

Fig. 4. Achieved accuracy of each classifier.

TABLE I. BASELINE CLASSIFIER RESULTS

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
DT	99.97	99.97	99.97	99.97
RF	99.97	99.97	99.97	99.97
SVM	99.83	99.83	99.83	99.83
GBC	99.92	99.92	99.92	99.92
LR	99.70	99.70	99.70	99.70

Despite the impressive performance of the classifiers, a soft voting classifier was used to improve model performance by combining their advantages. This approach reduces bias and increases robustness and generalization. Each classifier may show strong performance, but consolidating their outputs reduces individual errors and improves prediction reliability. The voting classifier is an effective tool for improving the stability and precision of classification models.

C. Experimental Results: Feature Selection and Voting Classifier

A feature selection was initially performed to find the most useful and important features out of 55 features using Pearson correlation analysis. As an indicator, the Pearson correlation matrix was calculated to analyze the relationships in the dataset and identify the most important features that are strongly associated with the target. Pearson correlation studies measure the strength of an association between multiple features using a correlation coefficient ranging from 1 to -1. After applying Pearson's correlation coefficient, the features with the highest importance relative to the target are as follows: 'pslist.avg_threads', 'dlllist.ndlls', 'dlllist.avg_dlls_per_proc', 'handles.nevent', 'handles.nkey', 'handles.nthread', 'handles.nsemaphore', 'handles.ntimer', 'handles.nmutant', 'ldrmodules.not_in_load', 'ldrmodules.not_in_init', 'ldrmodules.not_in_mem', 'svcscan.nservices', 'svcscan.kernel_drivers', 'svcscan.process_services', 'svcscan.shared_process_services', 'svcscan.nactive', 'callbacks.ncallbacks'.

Classification tasks can be affected by category imbalance because ML algorithms select samples from the dominant class to improve accuracy. If one class (the minority) has significantly fewer samples than the other (the majority), the model may favor the majority. As a result, the learning model will underpredict the minority class samples. To address this problem, we use SMOTE to correct for imbalances in the data between classes. The goal is to generate new instances of the minority class by interpolating between surrounding instances. SMOTE randomly selects surrounding instances from a given minority class sample Z_i and generates a new instance using (1):

$$Z_{new} = Z_i + |Z_j - Z_i| * \delta \quad (1)$$

where Z_j is the randomly selected neighbor and δ is a random value from the range [0,1].

The balanced dataset, improved by the previously outlined steps, including feature selection and the application of a training and testing partition, is then fed into the soft voting classifier to classify benign and malicious data. The soft voting classifier demonstrates excellent performance, achieving 99.99% for accuracy, precision, recall and F1 score, as shown in Figure 5. These results, summarized in Table II, show how robust and effective the model is in distinguishing between benign and malicious samples.

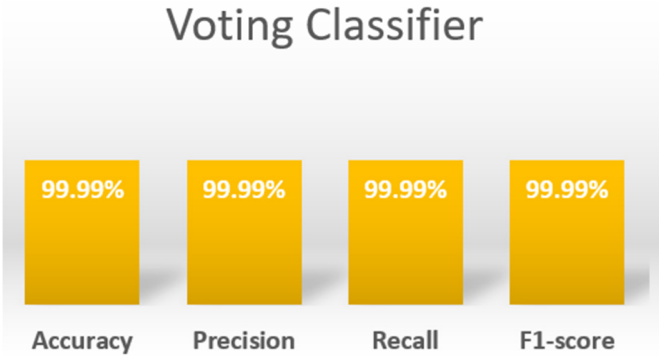

Fig. 5. Voting classifier performance results.

TABLE II. EXPERIMENTAL RESULTS OBTAINED FOR THE PERFORMANCE OF THE VOTING CLASSIFIER.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
Voting classifier	99.99%	99.99%	99.99%	99.99%

The effectiveness of the voting classifier was evaluated using the LR, SVM, GB, RF, and DT models, and the results were compared with previous studies that used the CIC-MalMem-2022 dataset and a train-test-split dataset methodology, as shown in Table III. Despite the slight variation in the results, the proposed method outperforms the previous methods by integrating the strengths of different models into the soft voting framework, improving its ability to handle different data patterns and reducing errors. Although the proposed method in [27] achieved 99.9% accuracy using 1D-CNN, our approach outperforms it by its dynamic adaptability. Unlike static deep learning architectures, the soft voting classifier combines diverse classifiers, allowing it to adapt to evolving malware patterns. By reducing the number of features from 55 to 18, it reduces training time and computational efficiency. In addition, SMOTE integration ensures balanced learning by mitigating class imbalance.

TABLE III. COMPARISON OF THE PROPOSED METHOD WITH PREVIOUS METHODS.

Reference	Method	Dataset	Accuracy (%)
[25]	DCNN	CIC-MalMem-2022	99
[26]	CNNs and MLP	CIC-MalMem-2022	99.8
[27]	1D-CNN	CIC-MalMem-2022	99.9
Proposed method	Soft voting classifier (DT, RF, SVM, GB, LR)	CIC-MalMem-2022	99.99

The proposed method achieves an impressive accuracy rate of 99.99%, however, this result may raise concerns about possible overfitting, indicating that the model may have been over-adapted to the training data, thus undermining its performance on new data. Several procedures were used to mitigate this effect, including cross-validation, removal of extraneous features using Pearson correlation, and application of SMOTE to preserve data balance. Future investigations could explore approaches such as deep learning using convolutional networks to enhance generalization and improve model performance on new inputs.

V. CONCLUSION

This paper presents a methodology for malware identification using machine learning (ML) models, and explores the complex correlations between multiple parameters of malware datasets through comprehensive data analysis and ML methods. Through correlation analysis, we identify critical feature pairs that reveal essential relationships and provide key insights for improving malware detection systems. The combined soft voting mechanism that integrates five ML classifiers: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), and Logistic Regression (LR) demonstrates effective performance in distinguishing benign from malicious cases. The proposed strategy achieves 99.99% accuracy, F1 score, recall, and precision, outperforming many traditional and state-of-the-art malware detection techniques. This paper introduces an innovative classification approach that reduces computational complexity while maintaining performance. In addition, the data imbalance problem is addressed using the Synthetic Minority Oversampling Technique (SMOTE), which enhances the model's ability to classify rare malware more efficiently than previous studies. Although the proposed approach achieved high performance, there are several aspects that can be improved in future research, including testing the model in real operating environments to ensure its compatibility with modern security systems and improving its speed and real-time performance. In addition to SMOTE, other techniques such as dynamic data augmentation can be explored to improve the balance of training samples and increase the effectiveness of the detection system in combating cyber threats.

REFERENCES

[1] M. Lombardi, F. Pascale, and D. Santaniello, "Internet of Things: A General Overview between Architectures, Protocols and Applications," *Information*, vol. 12, no. 2, Feb. 2021, Art. no. 87, https://doi.org/10.3390/info12020087.

[2] M. Shafiq, Z. Gu, O. Cheikhrouhou, W. Alhakami, and H. Hamam, "The Rise of 'Internet of Things': Review and Open Research Issues Related to Detection and Prevention of IoT-Based Security Attacks," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, Aug. 2022, Art. no. 8669348, https://doi.org/10.1155/2022/8669348.

[3] A. E. Omolara *et al.*, "The internet of things security: A survey encompassing unexplored areas and new insights," *Computers & Security*, vol. 112, Jan. 2022, Art. no. 102494, https://doi.org/10.1016/j.cose.2021.102494.

[4] K. Aldriwish, "A Deep Learning Approach for Malware and Software Piracy Threat Detection," *Engineering, Technology & Applied Science Research*, vol. 11, no. 6, pp. 7757–7762, Dec. 2021, https://doi.org/10.48084/etasr.4412.

[5] R. Sridharan and S. Domnic, "Network policy aware placement of tasks for elastic applications in IaaS-cloud environment," *Cluster Computing*, vol. 24, no. 2, pp. 1381–1396, Jun. 2021, https://doi.org/10.1007/s10586-020-03194-z.

[6] M. A. Mohammed, M. A. Hussain, Z. A. Oraibi, Z. A. Abduljabbar, and V. O. Nyangaresi, "Secure Content Based Image Retrieval System Using Deep Learning," *Basrah Researches Sciences*, vol. 49, no. 2, pp. 94–111, Dec. 2023, https://doi.org/10.56714/bjrs.49.2.9.

[7] J.-P. A. Yaacoub, H. N. Noura, O. Salman, and A. Chehab, "Robotics cyber security: vulnerabilities, attacks, countermeasures, and recommendations," *International Journal of Information Security*, vol. 21, no. 1, pp. 115–158, Feb. 2022, https://doi.org/10.1007/s10207-021-00545-8.

[8] S. Abdelkader *et al.*, "Securing modern power systems: Implementing comprehensive strategies to enhance resilience and reliability against cyber-attacks," *Results in Engineering*, vol. 23, Sep. 2024, Art. no. 102647, https://doi.org/10.1016/j.rineng.2024.102647.

[9] I. H. Sarker, A. I. Khan, Y. B. Abushark, and F. Alsolami, "Internet of Things (IoT) Security Intelligence: A Comprehensive Overview, Machine Learning Solutions and Research Directions," *Mobile Networks and Applications*, vol. 28, no. 1, pp. 296–312, Feb. 2023, https://doi.org/10.1007/s11036-022-01937-3.

[10] G. Sharma, S. Vidalis, N. Anand, C. Menon, and S. Kumar, "A Survey on Layer-Wise Security Attacks in IoT: Attacks, Countermeasures, and Open-Issues," *Electronics*, vol. 10, no. 19, Oct. 2021, Art. no. 2365, https://doi.org/10.3390/electronics10192365.

[11] A. Al-Marghilani, "Comprehensive Analysis of IoT Malware Evasion Techniques," *Engineering, Technology & Applied Science Research*, vol. 11, no. 4, pp. 7495–7500, Aug. 2021, https://doi.org/10.48084/etasr.4296.

[12] Gopinath and S. C. Sethuraman, "A comprehensive survey on deep learning based malware detection techniques," *Computer Science Review*, vol. 47, Feb. 2023, Art. no. 100529, https://doi.org/10.1016/j.cosrev.2022.100529.

[13] M. Aqeel, F. Ali, M. W. Iqbal, T. A. Rana, M. Arif, and Md. R. Auwul, "A Review of Security and Privacy Concerns in the Internet of Things

(IoT)," *Journal of Sensors*, vol. 2022, no. 1, Sep. 2022, Art. no. 5724168, https://doi.org/10.1155/2022/5724168.

[14] N. K. Gyamfi, N. Goranin, D. Ceponis, and H. A. Čenys, "Automated System-Level Malware Detection Using Machine Learning: A Comprehensive Review," *Applied Sciences*, vol. 13, no. 21, Nov. 2023, Art. no. 11908, https://doi.org/10.3390/app132111908.

[15] E. Nowroozi, A. Dehghantanha, R. M. Parizi, and K.-K. R. Choo, "A survey of machine learning techniques in adversarial image forensics," *Computers & Security*, vol. 100, Jan. 2021, Art. no. 102092, https://doi.org/10.1016/j.cose.2020.102092.

[16] M. Soni and D. K. Singh, "New directions for security attacks, privacy, and malware detection in WBAN," *Evolutionary Intelligence*, vol. 16, no. 6, pp. 1917–1934, Dec. 2023, https://doi.org/10.1007/s12065-022-00759-2.

[17] D. Gibert, C. Mateu, and J. Planes, "The rise of machine learning for detection and classification of malware: Research developments, trends and challenges," *Journal of Network and Computer Applications*, vol. 153, Mar. 2020, Art. no. 102526, https://doi.org/10.1016/j.jnca.2019.102526.

[18] M. J. J. Ghrabat, G. Ma, I. Y. Maolood, S. S. Alresheedi, and Z. A. Abduljabbar, "An effective image retrieval based on optimized genetic algorithm utilized a novel SVM-based convolutional neural network classifier," *Human-centric Computing and Information Sciences*, vol. 9, no. 1, Aug. 2019, Art. no. 31, https://doi.org/10.1186/s13673-019-0191-8.

[19] R. J. Mohammed *et al.*, "A Robust Hybrid Machine and Deep Learning-based Model for Classification and Identification of Chest X-ray Images," *Engineering, Technology & Applied Science Research*, vol. 14, no. 5, pp. 16212–16220, Oct. 2024, https://doi.org/10.48084/etasr.7828.

[20] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, and H. Liu, "A Review of Android Malware Detection Approaches Based on Machine Learning," *IEEE Access*, vol. 8, pp. 124579–124607, 2020, https://doi.org/10.1109/ACCESS.2020.3006143.

[21] M. S. Khalefa *et al.*, "Deep Sentiment Analysis System with Attention Mechanism for the COVID-19 Vaccine," *TEM Journal*, vol. 13, no. 2, pp. 1470–1480, May 2024, https://doi.org/10.18421/TEM132-61.

[22] H. M. Jasim *et al.*, "Provably Efficient Multi-Cancer Image Segmentation Based on Multi-Class Fuzzy Entropy," *Informatica*, vol. 47, no. 8, pp. 77–88, Sep. 2023, https://doi.org/10.31449/inf.v47i8.4840.

[23] M. J. J. Ghrabat, G. Ma, Z. A. Abduljabbar, M. A. Al Sibahee, and S. J. Jassim, "Greedy Learning of Deep Boltzmann Machine (GDBM)'s Variance and Search Algorithm for Efficient Image Retrieval," *IEEE Access*, vol. 7, pp. 169142–169159, 2019, https://doi.org/10.1109/ACCESS.2019.2948266.

[24] N. Usman *et al.*, "Intelligent Dynamic Malware Detection using Machine Learning in IP Reputation for Forensics Data Analytics," *Future Generation Computer Systems*, vol. 118, pp. 124–141, May 2021, https://doi.org/10.1016/j.future.2021.01.004.

[25] A. Mezina and R. Burget, "Obfuscated malware detection using dilated convolutional network," in *2022 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops*, Valencia, Spain, 2022, pp. 110–115, https://doi.org/10.1109/ICUMT57764.2022.9943443.

[26] H. Naeem, S. Dong, O. J. Falana, and F. Ullah, "Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification," *Expert Systems with Applications*, vol. 223, Aug. 2023, Art. no. 119952, https://doi.org/10.1016/j.eswa.2023.119952.

[27] B. Taşcı, "Deep-Learning-Based Approach for IoT Attack and Malware Detection," *Applied Sciences*, vol. 14, no. 18, Sep. 2024, Art. no. 8505, https://doi.org/10.3390/app14188505.

[28] M. Rostami, K. Berahmand, E. Nasiri, and S. Forouzandeh, "Review of swarm intelligence-based feature selection methods," *Engineering Applications of Artificial Intelligence*, vol. 100, Apr. 2021, Art. no. 104210, https://doi.org/10.1016/j.engappai.2021.104210.

[29] J. Allgaier and R. Pryss, "Cross-Validation Visualized: A Narrative Guide to Advanced Methods," *Machine Learning and Knowledge Extraction*, vol. 6, no. 2, pp. 1378–1388, Jun. 2024, https://doi.org/10.3390/make6020065.

[30] Y. Zhang, H. Zhang, J. Cai, and B. Yang, "A Weighted Voting Classifier Based on Differential Evolution," *Abstract and Applied Analysis*, vol. 2014, no. 1, May 2014, Art. no. 376950, https://doi.org/10.1155/2014/376950.

[31] M. A. Khan *et al.*, "Voting Classifier-Based Intrusion Detection for IoT Networks," in *Advances on Smart and Soft Computing: Proceedings of ICACIn 2021*, Casablanca, Morocco, 2021, pp. 313–328, https://doi.org/10.1007/978-981-16-5559-3_26.

[32] T. Carrier, P. Victor, A. Tekeoglu, and A. H. Lashkari, "Malware memory analysis (CIC-MalMem-2022)." Canadian Institute for Cybersecurity, UNB, 2022. [Online]. Available: https://www.unb.ca/cic/datasets/malmem-2022.html.