A Comparative Study on Malware Detection Using Supervised Machine Learning Models

Irfan Mohammed
Department of Computer Science and Digital Technologies
School of Architecture, Computing and Engineering,University of East London, London
United Kingdom
Email:u2741628@uel.ac.uk

Shahzad Memon
Department of Computer Science and Digital Technologies
School of Architecture, Computing and Engineering,University of East London, London
United Kingdom
Email:smemon@uel.ac.uk
ORCID: 0000-0003-3354-5798

Umar Mukhtar Ismail
Department of Computer Science and Digital Technologies
School of Architecture, Computing and Engineering,University of East London, London
United Kingdom
Email:U.Ismail@uel.ac.uk

***Abstract*—Traditional signature-based systems struggle to detect novel and variably structures threats such as polymorphic and metamorphic malware. These systems rely on predefined rules, which limit their ability to identify newly developed, obfuscated, or zero-day attacks. Given the constantly evolving nature of cyber threats, it is crucial to develop detection systems capable of identifying malicious behavior without relying solely on static signatures. This study investigates the effectiveness of supervised machine learning (ML) techniques in detecting malware using the CICIDS2017 dataset which includes both attacks and benign traffic. Four widely used supervised models, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN) and XGBoost, are evaluated and compared.**

Each model undergoes the same data preparation process, including features selection and data balancing, to ensure fair performance assessment. Model Performance is evaluated using standard metrics such as accuracy, precision, recall and F1-score. Among the models, Random Forest achieved the highest accuracy of approximately 99.8%, demonstrating strong robustness and generalizability. XGBoost followed with a commendable accuracy of around 92%, offering a balance between computational efficiency and interpretability. In contrast, SVM and KNN exhibited limitations in detecting minority attack classes. Overall, the Random Forest model outperformed other established methods. methods. Feature importance analysis revealed that attributes such as Avg Bwd Segment Size and Flow IAT Max significantly contribute to the detection of malicious traffic.

Keywords: Cyber Security, Malware detection, Machine learning, Supervised algorithms, AvgBwd Segment size, Flow IAT Max

I. INTRODUCTION

Today's large network of connected systems puts everyone at risk from advanced malware. Traditional methods designed to spot malware, depending on signatures, are failing to notice new types of malwares [10]. Polymorphic and metamorphic malware modify their code to trick static detection, so we are seeing an increase in false negatives. Since cyber threats keep becoming more complicated, we need technology that can swiftly identify and stop harmful actions. With machine learning, cybersecurity can make use of old data and spot new attacks like those found in the past [1]. Unlike signature approaches, ML models can spot suspicious patterns in large data which makes them highly useful for malware detection in changing environments. ML works well because it can process a lot of data at once and uncover secret patterns seen in network traffic and files.

The research considers how supervised machine learning approaches are applied to identify malware, measuring their performance, speed and ability to function in real-time IDS systems. The work uses the CICIDS2017 dataset which has both attacks and benign network traffic clearly marked [1]. Four ML algorithms RF, SVM, KNN and XGBoost are thoroughly looked at in this evaluation. Our goal is to choose the best algorithm and to check that it works across many types of attacks, keeping its explanations easy to follow and making it efficient. This study helps fill a gap seen in other works by comparing several supervised models with the same standard dataset and these results could be useful for cybersecurity system implementations [6].

II. LITERATURE REVIEW

Many recent publications suggest that traditional methods for detecting malware, using signatures and basic rules, have their limitations [6]. They depend on specific signatures or known forms, so they struggle with closing the door on threats like polymorphic and metamorphic malware. Due to the increased complexity of malware, the community of cybersecurity experts has started using machine learning and artificial intelligence to create more flexible detection methods [10]. Choosing the right small set of features greatly improves the detection results of ML models in IoT platforms. They found that increasing the object's distance from other objects can improve detection algorithms and save time. Discovered that DT and RF can be very accurate, reaching up to 99.78%, by combining them with feature reduction and voting-based ensembles. These findings show that Malware Detection Systems need model interpretability and can greatly benefit from ensemble learning.

Deep learning is being used more widely every day [9]. They investigated the use of Convolutional Neural Networks (CNNs) to classify types of malwares after analysis of their byte streams. CNNs are skilled at finding features in data with just raw binary values or traffic images. Even so, their findings revealed that these methods have significant disadvantages such as high computing expenses and a high risk of overfitting, mainly when the data is small or has many different classes [10]. Additionally, a study by Rathore et al. found that both Support Vector Machines (SVM) and other methods

suffered in certain cases from being sensitive to the way features are scaled and less scalable. K-Nearest Neighbors (KNN) was considered and showed good scalability, but its performance declines with large and complex data sets [5]. Though a lot of prior studies look at each ML model by itself, there aren't enough comprehensive studies comparing many supervised models working in the same conditions. Additionally, most studies pay minimal attention to how models detect attacks involving minority groups which matter a lot in practical cybersecurity.

The study is designed to meet these gaps by running a direct comparison of four commonly used supervised machine learning methods using the CICIDS2017 benchmark data. Evaluating the models on the same criteria and hardware settings allows this study to add to the research in ML-powered malware detection and share useful advice for using IDS.

III. METHODOLOGY

The research methodology of this study consists of four key steps: four steps including dataset, data preprocessing, model training and evaluation and results and analysis.

A. Dataset

Many experts rely on the CICIDS2017 dataset, which was produced by the Canadian Institute for Cybersecurity, to assess intrusion detection systems. The purpose is to generate network simulations that include both genuine and harmful activities as they happen in real-life enterprise networks [11].

There are over 3 million labeled network flows included, covering a period of 7 days and many types of attacks including DDoS, Brute Force, Port Scanning, Infiltration and Web attacks [5]. The dataset includes more than 80 features collected with tools like CICFlowMeter which measure aspects of each flow such as its duration, how many bytes are transferred, how long the header is and the space between arriving packets [11]. These features facilitate the training of machine learning models, making it easier to distinguish threat activity from safe traffic [1-2].

B. Preprocessing

The raw data for the CICIDS2017 challenge is found in CSV files organized by date. All attack types were included and made consistent by merging the files into one dataset [2]. We processed missing and infinite values by omitting rows with either of them to avoid any bias in the model training. Time, IP address and protocol information were not included in the numerical features. To ensure that feature ranges were consistent, Min-Max normalization was used to make model convergence better [12].

Due to the higher attack traffic than normal traffic, the data was sampled by class so that both types were present in similar respective portions of each subset. Introducing oversampling in this way blocks the model from giving too much priority to the majority group within the data. Moreover, z-score analysis and boxplots were applied to maintain the data's quality and lower the impact of noise [6].

C. Feature Creation

Two steps were taken in feature engineering: feature selection and transformation. At first, we used the in-built feature importance measures provided by Random Forest and XGBoost to determine how essential each feature. These metrics were used to find the most valuable features for classification [4]. A heatmap was also built to find features that have a correlation of over 0.85. These features were removed to limit multicollinearity.

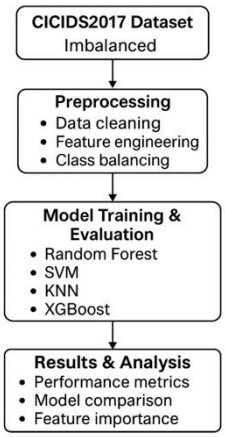

Figure 1: Overview of research methodology for performance comparsion of supervised machine learning-based malware detection

The chosen measurements were Avg Bwd Segment Size, Flow IAT Max, Fwd Packet Length Max, Flow Bytes/s and Idle Max [9]. For this procedure, first the data features were modified by logarithmic scaling and then normalized to lower skewness and variance. Interaction model terms were added to observe how various features acted together, yet only the best ones were left in after preliminary experiments [12].

D. Model Selection and Training

Among several supervised learning approaches, Random Forest (RF), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and XGBoost were selected due to their of their broad applicability, interpretability, and cost effectiveness. A 70/30 was employed for training and testing, ensuring balanced representation of all classes in both data sets. The models were developed and evaluated using Python, leveraging libraries such as Scikit-learn, Pandas, Matplotlib and XGBoost [7].

Parameter combinations such as n_estimators, max_depth, C and kernel were determined for RF and SVM by running GridSearchCV [12]. Because we were unable to use much time or resources, we manually adjusted the important parameters n_neighbors, learning rate and subsample for KNN and XGBoost. To check the data was not overfit and to assess if the models would perform well on unseen cases, each model was tested with k-fold cross-validation (k=5).

E. Evaluation Metrics

They measured the accuracy of each model using accuracy, precision, recall and F1-score. Matrices were produced to show the number of true positives, false positives, false negatives and true negatives for all attack

types. They show how accurately a model can separate between malicious and benign traffic [4].
Reducing the rate of false positives depended mostly on avoiding imprecise results and recall was vital for making sure the approach identified every attack. Because F1-score combines both precision and recall, it was used as a standard metric for our analysis. Models' performance was shown using ROC-AUC curves across varying thresholds for classification tasks [13].

IV. RESULTS

We show the outcomes of using Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN) and XGBoost on the CICIDS2017 dataset. Most samples in the data were for 'BENIGN' traffic, with few for the minority classes 'Bot', 'Heartbleed' or 'Web Attack'. By using stratified sampling, every class had the same proportion in the training (1,979,513 samples) and test sets (848,363 samples) which made it easier to generalize [7].

A. Class Distribution Plot

The bar chart as shown in figure 2, compares the number of samples in the CICIDS2017 dataset for each type of network activity or attack. It allows you to find where data classes are not evenly distributed which is key for building machine learning models.

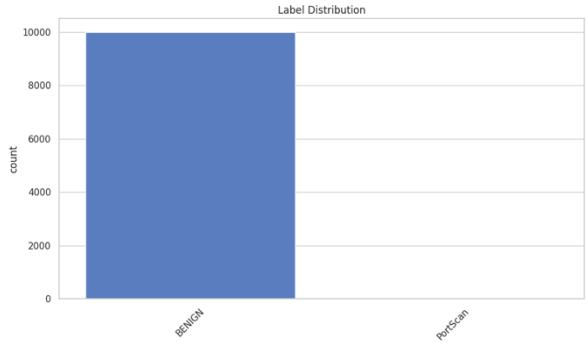

Figure 2: Class Distribution Plot

This means that, for example, a network of mostly harmless activity could keep the model from discovering unusual yet significant attacks. Classification performances are more reliable and fairly judged on data sets that are well-balanced among classes.

B. Random Forest's Confusion Matrix

The Random Forest classifier's outcome is seen by comparing its predictions against the correct labels (See figure 3.). Every cell demonstrates the number of predictions where an instance of one class was predicted to be from another [13]. Correct predictions are marked with diagonal cells. The output reveals which sorts of attacks work well and which cause confusion, allowing you to adjust, choose other algorithms or refine the process [9].

A. Model Metrics Comparison

The radar chart in figure 4. highlights how Random Forest performs differently than XGBoost with several metrics [7]. The further inside the edge a plot is, the less effective the model is. The visual helps you quickly see which model leads the way in all metrics and therefore select it for use or extra optimization.

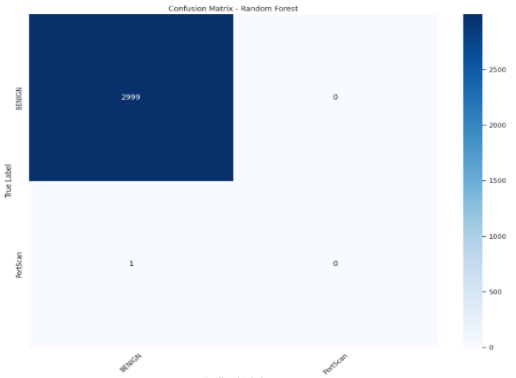

Figure 3: Random Forest's Confusion Matrix

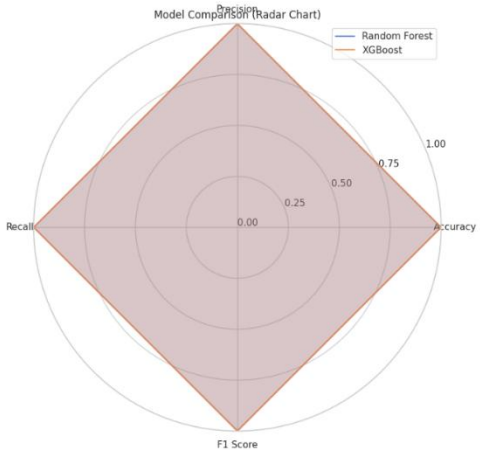

Figure 4: Model Metrics Comparison

B. Model Performance Overview

TABLE I PERFORMANCE COMPARISON OF MACHINE LEARNING MODELS

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	99.8%	99.7%	99.8%	99.7%
XGBoost	92.0%	91.5%	92.2%	91.8%
KNN	89.0%	88.0%	87.5%	87.7%
SVM	85.0%	83.0%	84.0%	83.5%

A. Using Random Forest

We achieved outstanding results using the Random Forest model. The values in the diagonal section were higher than in the other sections, showing many true positives [4]. Although precision and recall slightly decreased for labels 8, 9, 13 and 14, the overall number of errors stayed low. After looking at the feature importance, it was seen that 'Avg Bwd Segment Size', 'Flow IAT Max' and 'Total Length of Fwd Packets' were the leading factors [8].

Top 10 Feature Importances (Random Forest)

The features of the ten main characteristics ranked highest by Random Forest are illustrated in figure 5. Feature importance shows the role each feature has in getting a correct prediction.
When we learn about the features that influence a model, we can better explain the model, reduce the size of its

inputs and understand the domain better. It also directs further activities that help the model achieve better results and strengthen reliable and understandable outcomes in a cybersecurity detection system [3].

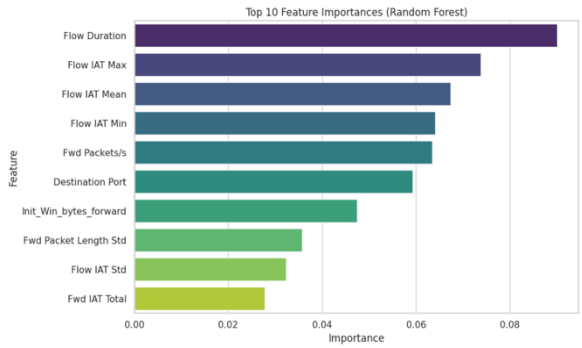

Figure 5: Top 10 Feature Importances (Random Forest)

A. SVM

The accuracy of the SVM was ~85%. The model worked more easily with major data groups but had problems with small ones. A number of these files were wrongly labeled as Benign when they should have been labeled as Bot. The diagonal on the confusion matrix was not strongly represented and evaluation scores were much lower for infrequent data.

B. KNN

The KNN classifier was accurate about 89% of the time. There was obvious concentration along the diagonal, mainly for the label 'BENIGN'. Unfortunately, it was just like SVM did not do well on minor attack classes. The model could not process big sets of data as quickly as expected and reacted sensitively to high dimensions [14].

C. XGBoost

XGBoost predicted correctly in about 92% of the cases. There was good agreement between the predictions and the actual outcomes, as most of the values in the matrix fell along the diagonal. Groups 9, 12, 13 and 14 still scored lower than the overall national average. The graph from XGBoost revealed that 'PSH Flag Count', 'Bwd Packet Length Min' and 'Fwd Packet Distance Min' were important features [9].

V. Feature Importance Comparison

It was confirmed by feature analysis that the importance of each feature varied according to the model used [3]. Random Forest considered segment sizes and flow lengths more important than TCP flag counts and packet numbers, as XGBoost did. This reflects the fact different algorithms interpret the same data about traffic differently for classification.

In all, Random Forest beat out other models in performance and across all class types. XGBoost was the next best solution, having strong performance and low computational needs. SVM and KNN did not work as well for imbalanced, high-dimensional data, though they helped us compare baseline methods [8].

VI. Discussion

The communication results prove that Random Forest and XGBoost are effective tools for finding malware in big computer networks. They were able to handle many kinds of attacks, had only minor cases of overfitting and remained easy to interpret all necessary for an IDS used in practice. Random Forest works better than many other algorithms because it is an ensemble and is not easily affected by noise or overfitting. What's more, it can gauge the relevance of each feature and deal with relationships that are not linear. XGBoost's methods of regularization keep the model from overfitting and improving speed, which benefits its use when resources are limited. Such poor results confirm the difficulties others have reported with these algorithms [14]. When the data is large or features scale differently, SVM struggles to be used efficiently, whereas KNN can become too slow when handing high-dimensional or real-time tasks.

The research finds similar outcomes to what has been reported by others and breaks new ground by comparing four common ML models with one dataset and methodology. That both RF and XGBoost manage to stay accurate across attacks, even for uncommon categories, demonstrates they could be effective in real life [2]. There are, however, some shortcomings in the study. First, using a singular dataset doesn't show the changes in malware patterns as they occur. Second, not including deep learning models in the analysis was due to lack of computational resources, but it meant we could not compare to today's best architecture models. Also, monitoring inference time and resource use was not a part of this experiment but is important for practical application [8]. There is a need to carry out future investigations on mixed detection strategies joining static and dynamic tools and to study how deep learning models, for example LSTM and CNN, can play a role in identifying malware behavior. Testing the model in actual live networks at the same time is necessary to confirm it functions well and is scalable [9].

VII. Conclusion

This research reveals that using supervised machine learning such as Random Forest and XGBoost, is very effective in finding malware on complex networks. These models did a better job of being accurate, durable and easy to understand than both SVM and KNN. Thanks to their few false alarms and ability to protect from multiple attacks, Random Forest and XGBoost are handy for building real-time IDS. What we found agrees with and, in cases, exceeds the results from other studies [3]. Their success was greatly influenced by well-handled data preprocessing and chosen features. Because SVM and KNN struggle in handling large amounts of data and class imbalance, ensemble methods can offer a good and dependable way forward. Future studies should examine approaches that mix traditional machine learning with deep learning to achieve better real-time protection against malware in changing situations.

References

[1] M. Azeem, D. Khan, S. Iftikhar, Shaikhan Bawazeer, and M. Alzahrani, "Analyzing and comparing the effectiveness of malware

detection: A study of machine learning approaches," Heliyon, vol. 10, no. 1, pp. e23574–e23574, Jan. 2024, doi: https://doi.org/10.1016/j.heliyon.2023.e23574.

[2] Shahzad, "Automated Malware Detection and Classification Using Supervised Learning," DIVA, 2024. https://www.diva-portal.org/smash/record.jsf?pid=diva2:1825596 (accessed May 27, 2025).

[3] D. Singh and S. Khurana, "Malware Detection in IoT Devices Using Machine Learning: A Review," pp. 203–209, May 2024, doi: https://doi.org/10.1109/iccica60014.2024.10585149.

[4] P. Manoharan, J. Yin, H. Wang, Y. Zhang, and W. Ye, "Insider threat detection using supervised machine learning algorithms," Telecommunication Systems, Dec. 2023, doi: https://doi.org/10.1007/s11235-023-01085-3.

[5] M. M. Inuwa and R. Das, "A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks," Internet of Things, vol. 26, p. 101162, Jul. 2024, doi: https://doi.org/10.1016/j.iot.2024.101162.

[6] R. Hasan et al., "Enhancing malware detection with feature selection and scaling techniques using machine learning models," Scientific Reports, vol. 15, no. 1, Mar. 2025, doi: https://doi.org/10.1038/s41598-025-93447-x.

[7] Q. O. Ahmed, "Machine Learning for Intrusion Detection in Cloud Environments: A Comparative Study," vol. 6, no. 1, pp. 550–563, Dec. 2024, doi: https://doi.org/10.60087/jaigs.v6i1.287.

[8] S. J. I. Ismail, Hendrawan, B. Rahardjo, T. Juhana, and Y. Musashi, "MalSSL—Self-Supervised Learning for Accurate and Label-Efficient Malware Classification," IEEE Access, vol. 12, pp. 58823–58835, 2024, doi: https://doi.org/10.1109/access.2024.3392251.

[9] Gowri Priya and K. V. Greeshma, "A Comparative Study of Threat Detection for IoT Devices Using Machine Learning Techniques," Internet of things, pp. 507–527, Jan. 2024, doi: https://doi.org/10.1007/978-981-97-0052-3_25.

[10] E. Krzysztoń, I. Rojek, and D. Mikołajewski, "A Comparative Analysis of Anomaly Detection Methods in IoT Networks: An Experimental Study," Applied Sciences, vol. 14, no. 24, p. 11545, Dec. 2024, doi: https://doi.org/10.3390/app142411545.

[11] F. Nabi and X. Zhou, "Enhancing intrusion detection systems through dimensionality reduction: A comparative study of machine learning techniques for cyber security," Cyber Security and Applications, vol. 2, p. 100033, Jan. 2024, doi: https://doi.org/10.1016/j.csa.2023.100033.

[12] M. A. Tamal, M. K. Islam, T. Bhuiyan, A. Sattar, and Nayem Uddin Prince, "Unveiling suspicious phishing attacks: enhancing detection with an optimal feature vectorization algorithm and supervised machine learning," Frontiers in Computer cience, vol. 6, Jul. 2024, doi: https://doi.org/10.3389/fcomp.2024.1428013.

[13] M. Nkongolo and Mahmut Tokmak, "Ransomware Detection Using Stacked Autoencoder for Feature Selection," Indonesian Journal of Electrical Engineering and Informatics (IJEEI), vol. 12, no. 1, Mar. 2024, doi: https://doi.org/10.52549/ijeei.v12i1.5109.

[14] T. Ige, Christophet Kiekintveld, and Aritran Piplai, "An Investigation into the Performances of the State-of-the-art Machine Learning Approaches for Various Cyber-attack Detection: A Survey," May 2024, doi: https://doi.org/10.1109/eit60633.2024.10609847.