International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 3S, 2025
https://www.theaspd.com/ijes.php

AI Techniques for Malware Detection in Drone Communication and Security

Ankit Kumar[1], Vikas Kumar[2], Manoj Kumar[3], Raveena[4],Rachna Sharma[5]
[1,2,3] Department of Computer Application, Swami Vivekanand Subharti University, Meerut.
ankitbaliyan042@gmail.com
vikasgotsyou001@gmail.com
manoj2002199@gmail.com
[4] Department of Commerce, Vinayak Vidyapeeth Modipuram, Meerut.
raveena.kashyap16@gmail.com
[5] Department of Computer Application, SRM institute of science and Technology, NCR campus, Modinagar, Ghaziabad.
rachnas1@srmist.edu.in

Abstract

Malware detection in drone communication and security is a critical area of research given the growing use of drones in civilian and military applications. In the past few years, driven by data artificial intelligence techniques, such as Machine Learning (ML) and Deep Learning (DL) approaches, have shown promise in detecting malware by leveraging its behaviour in terms of API calls. It is anticipated that drones will play a significant part in the future linked smart cities. They will be responsible for smart city security and monitoring, transporting commodities and commerce, and acting as mobile hot points for broadband wireless access. This article includes an in-depth examination of the several ML algorithms used in malware analysis and proposed an hybrid model for the improve the performance in terms of accuracy and F1-scores. The "Hybrid With FS" model achieves the highest accuracy, nearing 80%, indicating the significant benefit of feature selection in optimizing performance.

Keywords: Communication, Drone, Malware Detection, Cybersecurity, Machine Learning.

1. INTRODUCTION

Understanding different types of malware is essential for developing effective detection techniques and ensuring strong cybersecurity defenses. Malware, which includes viruses, worms, Trojans, ransomware, spyware, adware, rootkits, and botnets, poses a growing threat by infiltrating systems, stealing data, or causing damage. Traditional signature-based detection methods struggle to keep pace, necessitating advanced techniques like machine learning (ML) for accurate detection. These AI-driven systems often lack transparency. Raising concerns about trust in automated decisions. Explainable Artificial Intelligence (XAI) offers a solution by making AI models more interpretable, helping cybersecurity professionals understand and trust detection processes, which improves threat response and model reliability.

1.1 Different Types of Malware Detection Techniques:

Malware detection techniques can be broadly categorized into traditional and advanced methods:
Signature-based Detection: This method relies on identifying known patterns or signatures of malware. While effective against known threats, it is ineffective against new or polymorphic malware.
Heuristic-based Detection: This technique analyzes the behavior of software to identify malicious intent. It helps in detecting unknown or new malware but can result in false positives.
Behavior-based Detection: This method monitors the behavior of applications or network traffic to detect malicious activity. It is highly effective against zero-day attacks and advanced persistent threats (APTs).

Anomaly-based Detection: By establishing a baseline of normal behavior, this method detects deviations that may indicate malware. Anomaly-based detection is useful for identifying unknown threats but requires constant tuning to minimize false alarms.
AI-based Detection: Leveraging machine learning and deep learning models, AI-based techniques can automatically learn and detect malware patterns from large datasets. These methods are highly scalable and adaptive but often lack transparency, making the decision-making process difficult to interpret.

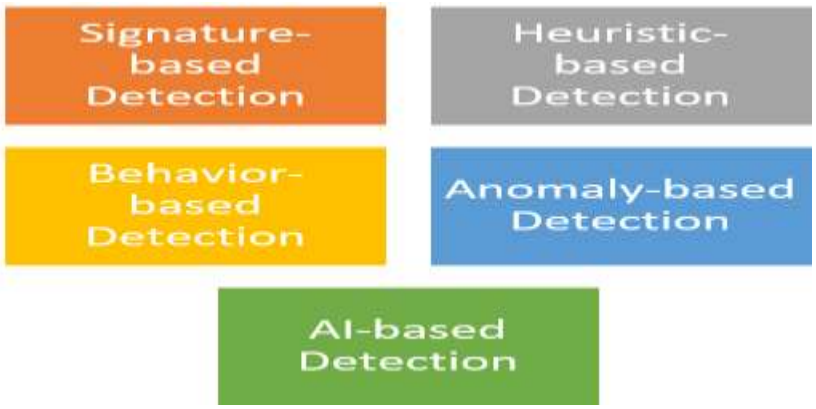

Figure 1: Different Types of Malware Detection Techniques

The key challenges in deploying AI techniques for malware detection in drone communication and security:
1.2. Resource Constraints: Drones often operate with limited computational, memory, and energy resources. Deploying AI models, particularly complex deep learning algorithms, can be infeasible on such resource-constrained platforms. Power consumption is another major limitation since onboard AI processing can quickly deplete the drone's battery. A small consumer drone might not have the processing power to execute real-time anomaly detection with deep learning models like Convolutional Neural Networks (CNNs). Running AI continuously for malware detection could lead to faster battery drainage, compromising mission longevity. the Potential Solutions Use lightweight AI models optimized for edge computing, such as MobileNet or TinyML frameworks. Offload intensive processing tasks to nearby ground stations or cloud systems (edge-cloud collaboration). Develop energy-efficient hardware specifically designed for AI applications in drones.
1.3. Real-Time Processing: Drones often require immediate malware detection to maintain mission continuity and prevent potential damage. However, traditional AI models may take longer to analyze data and provide actionable results. Network delays can exacerbate the problem, especially when drones rely on external systems for processing. A drone detecting a malware-induced hijack attempt must respond instantly to regain control or switch to a safe mode. Delays in AI-driven detection could allow the attack to succeed. Real-time communication interruptions in swarm drones could lead to cascading failures. The potential solutions Implement real-time optimization strategies, such as reducing model complexity (pruning or quantization). Utilize specialized hardware like GPUs or TPUs for faster computations. Design pre-trained models tailored for fast decision-making on specific malware categories. Integrate anomaly detection with fast, rule-based heuristics for initial screening.
1.4. Data Scarcity: AI models, particularly supervised learning approaches, require large, labeled datasets for effective training. Datasets specific to drone malware and cyber threats are often scarce, especially for novel or zero-day attacks. The highly diverse range of malware types and drone architectures adds to the complexity of obtaining comprehensive datasets. Lack of historical data on attacks targeting drone-to-drone communication makes it challenging to train robust anomaly detection systems. Zero-day malware exploits are inherently difficult to detect due to their novelty and lack of prior examples. The Potential Solutions Use data augmentation techniques to artificially expand the dataset. Implement unsupervised learning techniques to identify anomalies without requiring labeled data. Develop collaborative frameworks like federated learning to share threat intelligence

across multiple organizations while maintaining data privacy. Simulate attack scenarios in controlled environments to generate training data.

1.5. Adversarial Attacks: AI models are susceptible to adversarial attacks, where attackers craft inputs designed to deceive the AI system into making incorrect predictions. This vulnerability can be exploited to disable malware detection systems or mask malicious activities. An attacker could subtly alter communication packets to evade an AI-based anomaly detection system while maintaining control of the drone. Poisoning the training dataset by injecting malicious samples during the development phase could compromise the AI's reliability. The Potential Solutions are Employ adversarial training, where the AI model is exposed to adversarial samples during training to enhance robustness. Integrate explainable AI (XAI) techniques to ensure that decisions made by the AI system are interpretable and transparent. Use multi-layered defense mechanisms combining traditional signature-based approaches with AI models to reduce reliance on a single technique. Monitor AI systems continuously and update them to adapt to evolving adversarial tactics.

2. RELATED WORK

Recent advancements in malware detection leverage deep learning (DL) and machine learning (ML) techniques for robust and scalable solutions. Vinayakumar et al. (2019) proposed a big-data-driven hybrid framework integrating static and dynamic analysis, ensuring effective zero-day detection but facing dataset constraints. Rathore et al. (2018) demonstrated the superiority of Random Forests (RF) over Deep Neural Networks (DNN) using opcode frequency features, highlighting the limitations of deep learning for simpler problems. Tobiyama et al. (2016) effectively used RNN-CNN combinations for process-based malware detection, achieving high accuracy but restricted scalability. Techniques like image-based CNN classification were explored by He & Kim (2019) and Choi et al. (2017), offering robustness and fast detection but requiring significant memory and adversarial resistance. For Android-specific malware, Sabhadiya et al. (2019) reviewed DL techniques, while Kim et al. (2018) proposed a multimodal DL model combining diverse feature types for enhanced accuracy at the cost of high computational complexity. Ransomware detection received focused attention, with Fernando et al. (2020) employing Deep Belief Networks (DBNs) and Bae et al. (2020) targeting ransomware-specific file operations. Both approaches highlighted trade-offs between specialization and generalization. Alomari et al. (2023) balanced feature selection efficiency with performance, addressing dimensionality challenges. Collectively, these studies emphasize a trend towards integrating diverse features and DL techniques to balance detection accuracy, scalability, and resource efficiency.

Table 1: Overview of related work done in field of malware detection using AI techniques

Reference	Objective	Methodology	Advantages	Limitations
[11] Vinayakumar et al. (2019)	Propose a robust malware detection model using deep learning and image processing techniques.	Combines static, dynamic, and hybrid malware detection techniques in a big data environment. Evaluates deep learning and classical ML models on unbiased datasets.	Avoids feature engineering, effective zero-day malware detection, scalable framework.	Limited practical application in real-time due to dataset constraints.

[12] Rathore et al. (2018)	Detect malware using machine learning and deep learning approaches.	Applied supervised and unsupervised learning with opcode frequency as features. Compared RF, DNN, and feature reduction methods.	Demonstrated effectiveness of RF over DNN and simpler feature reduction techniques over autoencoders.	Deep learning found to be overkill; limited to opcode frequency-based features.
[13] Tobiyama et al. (2016)	Detect malware based on process behavior using deep learning.	Proposed a stepwise combination of RNN for feature extraction and CNN for classification.	Achieved AUC=0.96; effective against process-based malware detection.	Focused only on process behavior, limited scalability for diverse malware types.
[14] He & Kim (2019)	Evaluate CNN for malware detection using malware images.	Transformed malware files into image representations, classified using CNN with Spatial Pyramid Pooling.	Effective against redundant API injection; greyscale imaging improves robustness.	High memory requirements for naive SPP implementation.
[15] Choi et al. (2017)	Detect malware using malware images and deep learning.	Generated images from malware and benign files; trained a deep learning model to classify.	Fast detection without static or dynamic analysis.	Lacks exploration of adversarial resilience and scalability.
[16] Sabhadiya et al. (2019)	Android malware detection using deep learning techniques.	Reviewed various DL techniques (e.g., Maldozer, DroidDetector); proposed a DL model for malware classification.	Focused on Android-specific malware detection; provides extensive comparisons of techniques.	Limited details on feature extraction and generalizability across devices.
[17] Alomari et al. (2023)	Develop a high-performance malware detection system using DL and feature selection.	Applied correlation-based feature selection and trained LSTM and dense models.	Efficient feature reduction while maintaining high performance.	Performance degradation observed with aggressive feature reduction.
[18] Kim et al. (2018)	Propose a multimodal DL approach for Android malware detection.	Used diverse feature types with similarity-based extraction; proposed a multimodal DL model.	High accuracy leveraging multiple feature types.	Complexity in integrating multiple modalities and high computational cost.

[19] Fernando et al. (2020)	Explore ransomware detection using ML and DL techniques.	Represented malware as opcode sequences; used DBNs for classification and autoencoders for feature extraction.	DBNs outperformed traditional classifiers; effective feature reduction.	Dependency on unlabeled data for pretraining; limited to opcode sequence-based analysis.
[20] Bae et al. (2020)	Detect ransomware among malware and benign files.	Analyzed file-related operations; proposed ML methods focusing on ransomware-specific operations.	Specialized for ransomware detection; effective against zero-day ransomware.	Limited to ransomware-specific behaviors; not generalized for all malware types.

3. METHODOLOGY

This study employs an explainable AI (XAI) approach to build interpretable models and analyze their behavior. The methodological framework can be summarized in the following steps:

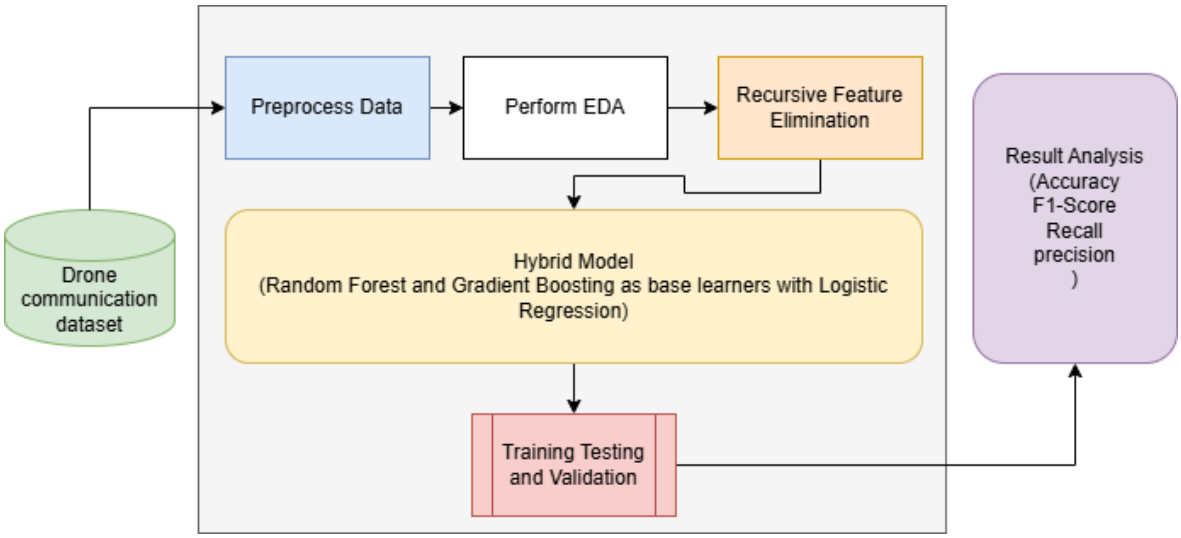

Figure 2:Proposed Methodlogy

i. **Dataset**: The dataset is collected from Kaggle reposetory from https://www.kaggle.com/datasets/datasetengineer/tokyo-drone-communication-and-security-dataset. Drone networks operating in the Tokyo metropolitan area between 2018 and 2024 provided detailed communication records, which are included in the Tokyo Drone Communication and Security Dataset. A comprehensive and accurate depiction of drone-to-drone (D2D) and drone-to-base station (D2BS) communication patterns is provided by this dataset, which documents hourly interactions between drones and base stations. This dataset, which was gathered from an area with high operating standards and a high level of technical advancement, offers important insights on network behaviors and security issues.

S.N.	Feature	Description
1	**Timestamp**	The exact time (hourly) of each recorded communication packet.
2	**Source Drone ID**	Unique identifier of the sending drone (e.g., D1, D2).

3	**Destination ID**	Identifier for the receiving drone or base station.
4	**Packet Size**	Size of the communication packet in bytes.
5	**Transmission Rate**	Rate of packet transmission (packets per second).
6	**Signal Strength**	Strength of the transmission signal, measured in dBm.
7	**Error Rate**	The ratio of corrupted packets during transmission.
8	**Encryption Status**	Binary feature indicating if data was encrypted (1) or not (0).
9	**Protocol Type**	Type of communication protocol used (e.g., TCP, UDP, Proprietary).
10	**Response Time**	Time between data transmission and acknowledgment (ms).
11	**Battery Level**	Battery percentage of the source drone during communication.
12	**GPS Coordinates**	Geographical location of the transmitting drone.
13	**Payload Type**	Type of data transmitted (e.g., Command, Data, Status).
14	**Packet Loss Rate**	Percentage of lost packets during transmission.
15	**Connection Duration**	Duration of communication sessions (seconds).
16	**Round Trip Time**	Average time for packets to reach their destination and return (ms).
17	**Hop Count**	Number of intermediary hops for packet delivery.
18	**Jitter**	Variability in packet arrival time, indicating network stability.
19	**Drone Velocity**	Speed of the drone during transmission (m/s).
20	**Signal-to-Noise Ratio**	Ratio of signal strength to background noise level.
21	**Data Throughput**	Amount of data transferred over a specific period (kbps).
22	**Port Number**	Network port number used for communication.
23	**Communication Interval**	Interval between consecutive data packets.
24	**Control Command Frequency**	Rate of control commands issued (Hz).
25	**Drone Altitude**	Altitude of the drone during transmission (meters).
26	**CPU Usage**	Percentage of CPU utilization on the transmitting drone.
27	**Memory Utilization**	Percentage of memory usage on the transmitting drone.
28	**Distance to Base Station**	Distance from the drone to its associated base station (meters).
29	**Target Label**	Categorized status of communication: Normal Communication, DDoS Attack, Malware Infection, Anomaly.

ii. **Feature Representation**: Key characteristics of the data-set are identified and converted into a feature set. This process involves the selection or engineering of features that capture the critical information needed for model training.

iii. **Feature Importance Analysis**: An analysis of the importance of different features is conducted to understand which variables contribute most significantly to the model's output. This helps in identifying key factors that influence predictions.

iv. **Interpretability Analysis**: Further analysis is performed to assess the interpretability of the model. This step evaluates how easily a human can understand the model's decisions and the rationale behind its predictions.

v. **Model Performance Evaluation**: The model's performance is evaluated using standard metrics, balancing accuracy.

Figure 3: Different types of Malware Attack

Recursive Feature Elimination (RFE) is a feature selection technique that iteratively removes features to identify the most relevant subset for a given machine learning model. It is based on assessing feature importance and eliminating the least important features in each iteration until the desired number of features is reached. Here's a detailed breakdown of the mathematics:

1. Problem Setup
Let: $X \in R^{n \times p}$: Feature matrix with n samples and p features.
$y \in R^{n}$: Target vector.
F={f1,f2,...,fp}: Set of all ppp features.
f: The chosen machine learning model (e.g., SVM, linear regression, or tree-based models).
The goal is to find a subset $F^* \subseteq F$ with r features (r<p) that are most informative for predicting y.

2. Feature Importance Calculation
The importance of each feature is determined by a measure based on the chosen model.

For Linear Models (e.g., Linear Regression, SVM): The feature importance can be quantified using the coefficients (weights) wj of the model:
Importancej=|wj|\text{Importance}_j = |w_j|Importancej=|wj|
where wj is the weight associated with feature fj in the decision function:
y^=w1x1+w2x2+⋯+wpxp+b
For Tree-Based Models (e.g., Decision Trees, Random Forests): The feature importance I_j is calculated based on the decrease in impurity (e.g., Gini index or entropy) from splitting on feature fj:
Ij=∑t∈TΔIt·
is the reduction in impurity at node t, and T is the set of all nodes.

3. Recursive Elimination Process
The RFE algorithm proceeds iteratively as follows:
Train the Model f on the current subset of features Fk (starting with Fk=F, the full set of features).
Rank Features: Compute the importance scores Ij for all features j∈Fk
Eliminate Features: Remove the least important kkk features (usually k=1 per iteration). The updated feature set becomes: Fk+1=Fkis smallest}
Repeat Steps 1–3 until the desired number of features rrr remains: |Fk|=r
4. Stopping Criteria
RFE stops under one of the following conditions:

The desired number of features r is reached.
Model performance on a validation set begins to degrade.

5. Algorithm Summary

Input: Feature matrix X, target vector y, base model f, and the desired number of features r.
Output: Optimal feature subset F.
Steps:
Train f on the current feature set Fk.
Compute feature importance Ij for all features j.
Eliminate the k least important features.
Repeat until |Fk|=r.

The Figure 3 providers Different types of Malware Attack. The dataset used in this study comprises images of various malware and benign, sourced from publicly available datasets shown in figure 4.

Packet Size	Transmission Rate	Signal Strength	Error Rate	Encryption Status	Protocol Type	Response Time	Battery Level	...	Signal-to-Noise Ratio	Data Throughput	Port Number	Communication Interval	Control Command Frequency	Drone Altitude	CPU Usage	Memory Utilization	Distance to Base Station	Target Label
2000	1	-48.674464	0.05	1	UDP	100	85.157405	...	25.649452	43.669663	443	1	20	71.580763	57.156316	41.748626	756.361257	Normal Communication
500	5	-51.596394	0.01	1	UDP	10	66.445555	...	23.916592	61.761303	443	10	5	118.741755	34.611260	40.978094	1089.687122	Normal Communication
500	1	-44.884766	0.01	1	UDP	1000	52.206726	...	26.361278	17.959860	80	1	1	90.429526	42.410581	22.175317	1082.864013	DDoS Attack
2000	1	-58.707466	0.01	1	TCP	10	74.205084	...	15.833704	82.039414	8080	1	1	101.328301	43.501778	36.141103	1299.121206	Normal Communication
1000	1	-44.503895	0.01	1	TCP	100	72.992942	...	20.536170	41.238151	80	1	5	71.590715	41.047166	22.635897	697.142708	Anomaly/Unusual Behavior

Figure 4: Sample Malware analysis data-set.

The Figure 5 illustrates the distribution of various target labels in the dataset, which are categorized into Normal Communication, DDoS Attack, Malware Infection, and Anomaly/Unusual Behavior. Normal Communication dominates the dataset with a count exceeding 35,000, indicating that the majority of traffic is benign. In contrast, malicious activities such as DDoS Attack and Malware Infection have significantly lower counts, with DDoS Attack surpassing 7,000 and Malware Infection reaching approximately 5,000. Anomaly/Unusual Behavior appears least frequently, with fewer than 3,000 occurrences. This imbalance highlights the predominance of normal traffic compared to malicious or unusual behavior, which could influence the performance and bias of machine learning models trained on this data.

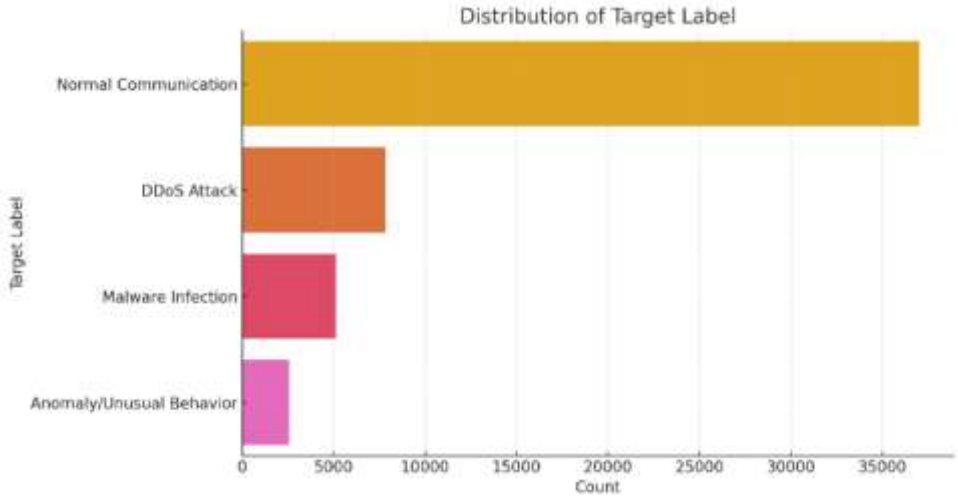

Figure 5: Target Class Labels Count

The correlation matrix displays in figure 6 shows the relationships between various features related to drone communication, performance, and environmental factors. The diagonal elements represent a perfect correlation (1.0), as a variable is always perfectly correlated with itself. All off-diagonal correlations are very close to 0, indicating negligible or no linear relationships between the features. This suggests that the variables are largely independent of one another and do not exhibit multicollinearity. Such a result implies that each feature contributes unique information to the dataset, which is valuable for machine learning or statistical models as it reduces redundancy. However, it also indicates no strong dependencies between these features that could simplify predictive modeling.

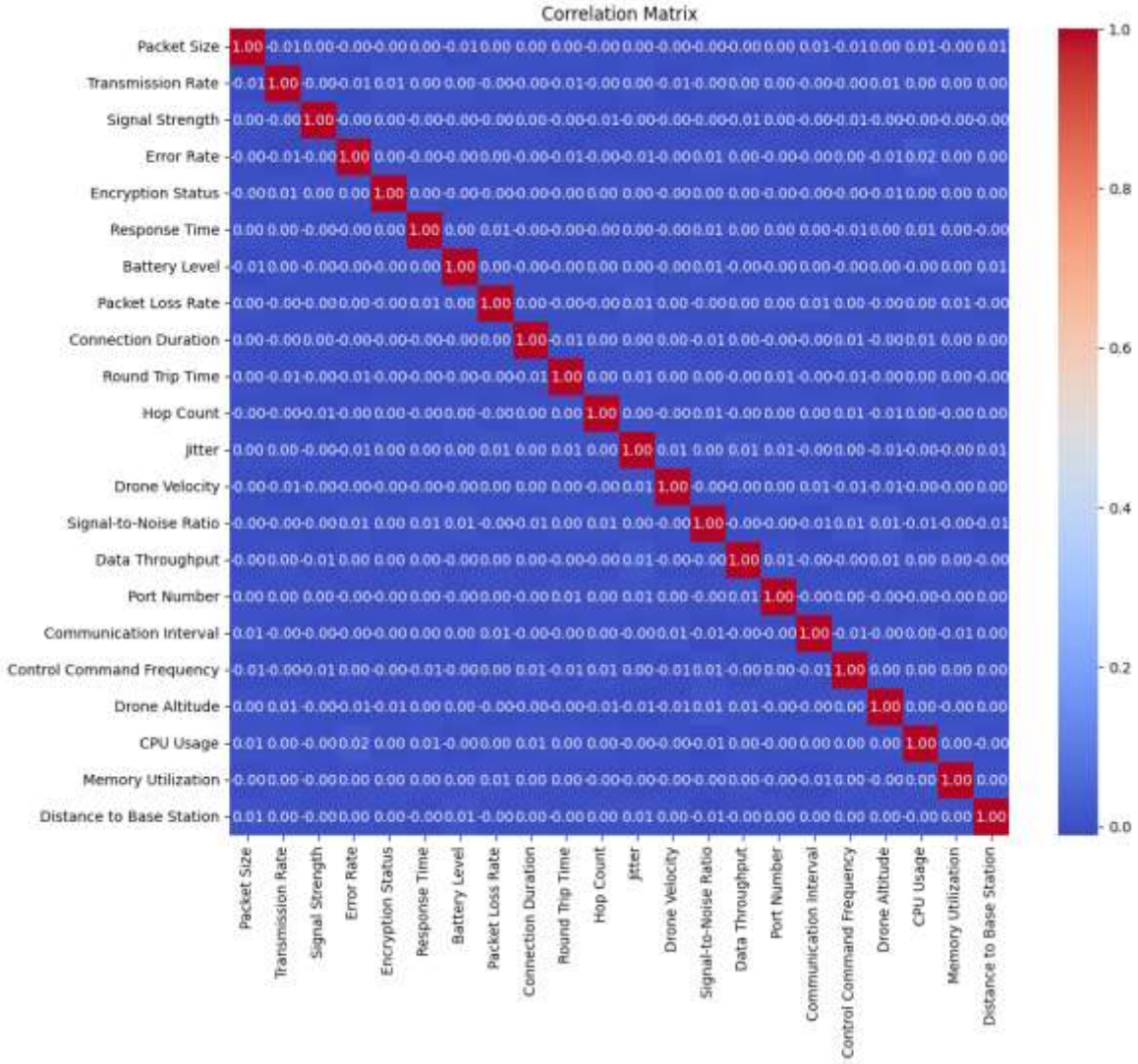

Figure 6: Corelation Matrix

4. RESULT ANALYSIS

The Figure 7 shows the top 10 features impacting drone performance, with "Memory Utilization" and "Distance to Base Station" being the most significant factors, both contributing over 7% importance. CPU usage, data throughput, and signal-to-noise ratio also play a critical role, indicating that computational and communication efficiencies heavily influence drone operations. Features such as "Drone Altitude," "Signal Strength," "Jitter," and "Drone Velocity" demonstrate moderate influence, likely impacting real-time performance and stability. "Battery Level" completes the list, reinforcing its

role in sustained operations. These insights help identify key areas for optimization in drone design and control systems.

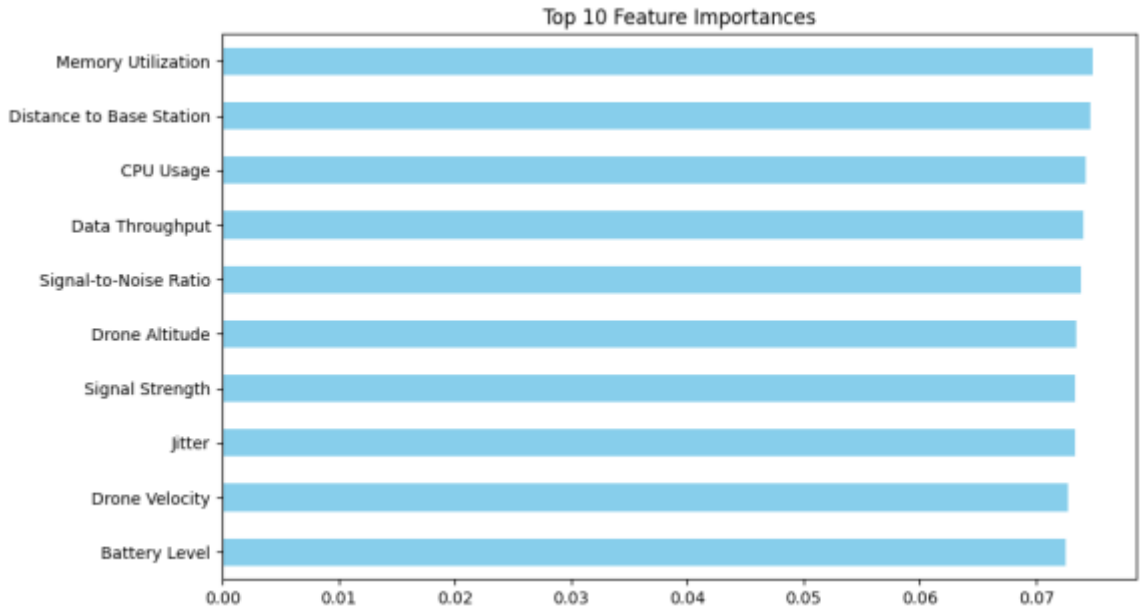

Figure 7: Ten Important Features

Table 1: Compares the performance of KNN, Decision Tree (DT), and Multi-Layer Perceptron (MLP)

Model Feature Selection	Accuracy	Precision	Recall	F1-Score
KNN Without FS	0.620995	0.525344	0.620995	0.564888
KNN With FS	0.626414	0.527496	0.626414	0.568182
Decision Tree Without FS	0.702006	0.514837	0.702006	0.579650
Decision Tree With FS	0.701721	0.643420	0.701721	0.579480
MLP Without FS	0.673006	0.525579	0.673006	0.577471
MLP With FS	0.702387	0.493347	0.702387	0.579594
SVM Without FS	0.702387	0.493347	0.702387	0.579594
SVM With FS	0.702387	0.493347	0.702387	0.579594
Hybrid Without FS	0.776524	0.726051	0.876524	0.778176
Hybrid With FS	0.802101	0.843499	0.799665	0.779665

The Figure 8 compares the accuracy of multiple models with and without feature selection (FS). The "Hybrid With FS" model achieves the highest accuracy, nearing 80%, indicating the significant benefit of feature selection in optimizing performance. Similarly, the "Hybrid Without FS" model also performs well but shows slight improvement with FS. Decision Tree and MLP models achieve moderate accuracy, with minimal difference between FS and non-FS cases. KNN demonstrates lower accuracy overall, highlighting its sensitivity to feature quality. SVM models show consistent performance with and without FS. This analysis suggests that hybrid models, particularly when paired with feature selection, yield superior accuracy, emphasizing the importance of selecting relevant features for enhanced model performance.

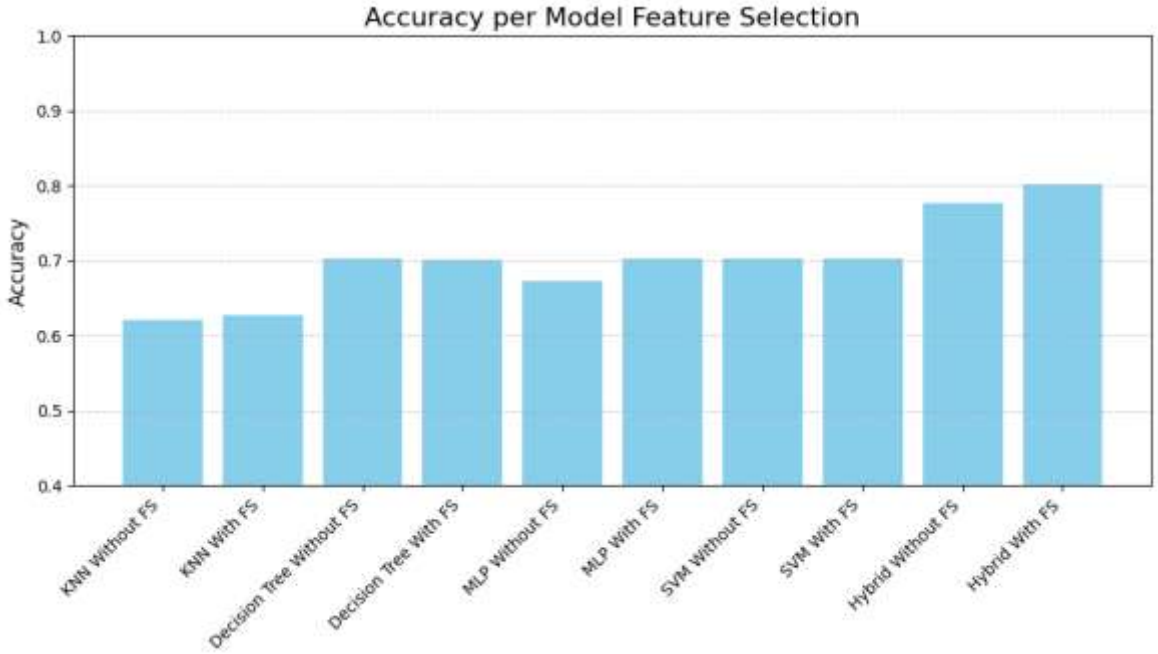

Figure 8: Accuracy per model feature selection

The Figure 9 illustrates the precision of different models with and without feature selection (FS). The "Hybrid With FS" model stands out with the highest precision, surpassing 0.8, highlighting the significant impact of feature selection on precision performance. "Hybrid Without FS" also achieves relatively high precision, but with a noticeable improvement when FS is applied. MLP with FS performs moderately well, reaching a precision above 0.6, whereas Decision Tree and KNN models show lower and comparable precision, regardless of FS. SVM models consistently demonstrate the lowest precision, around 0.5, with little to no improvement through feature selection. Overall, these results emphasize that hybrid models greatly benefit from feature selection, making them the best-performing approach for maximizing precision.

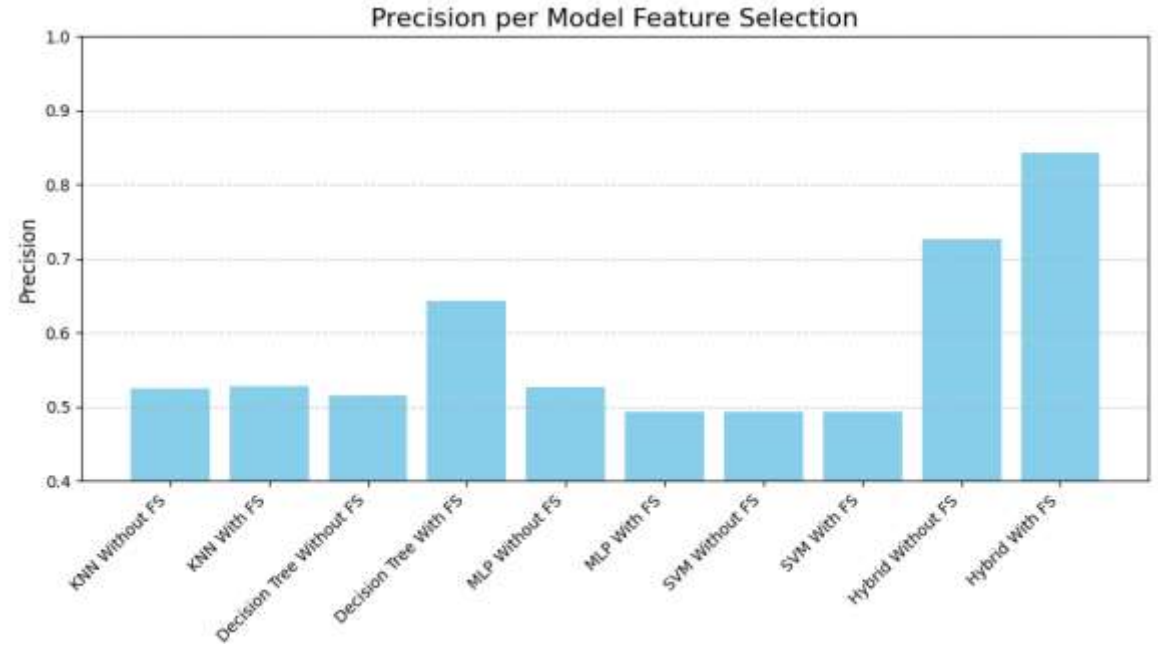

Figure 9: Precision per model feature selection

The Figure 10 illustrates the recall performance of various machine learning models, both with and without feature selection (FS). Models include KNN, Decision Tree, MLP, SVM, and a Hybrid approach. The Hybrid model without FS achieves the highest recall at approximately 0.9, followed by the Hybrid model with FS at 0.8. Other models, including KNN, Decision Tree, MLP, and SVM, show relatively consistent recall values around 0.6 to 0.7, with minimal variation between the inclusion and exclusion of feature selection. The results suggest that feature selection does not significantly impact most models' recall, except for the Hybrid model, where it slightly reduces performance but still maintains a strong recall. This indicates that the Hybrid approach is particularly effective in this setup.

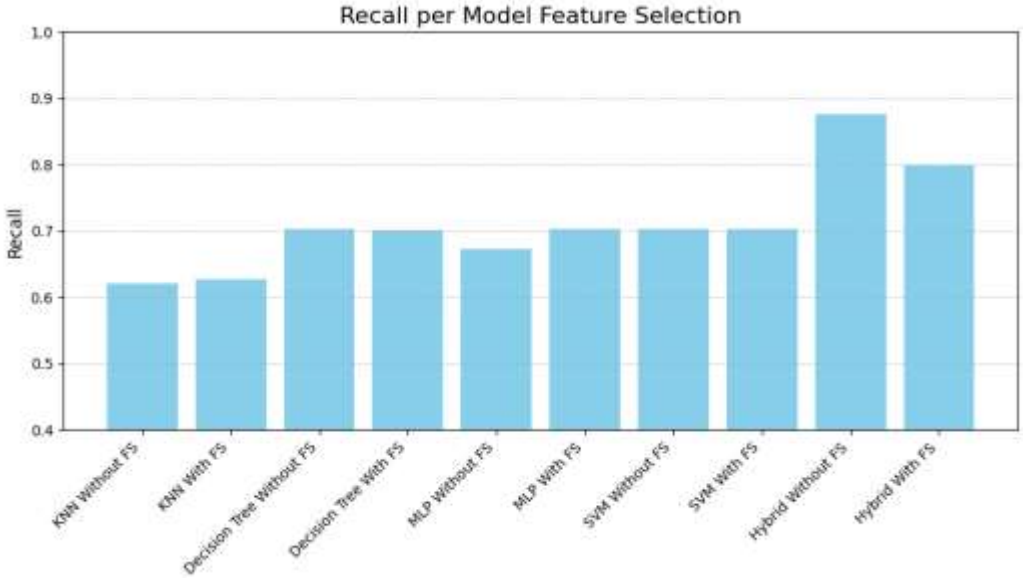

Figure 10: Model per feature selection

The bar chart presents the F1-Score performance for different machine learning models, with and without feature selection (FS). Similar to the recall results, the Hybrid model achieves the highest F1-Score, reaching approximately 0.78, regardless of FS. Other models, such as KNN, Decision Tree, MLP, and SVM, display lower F1-Scores, ranging consistently between 0.55 and 0.6. The inclusion of feature selection does not show a significant impact on the performance of these models. These findings suggest that while the Hybrid model significantly outperforms others in terms of F1-Score, the contribution of feature selection remains minimal across all models.

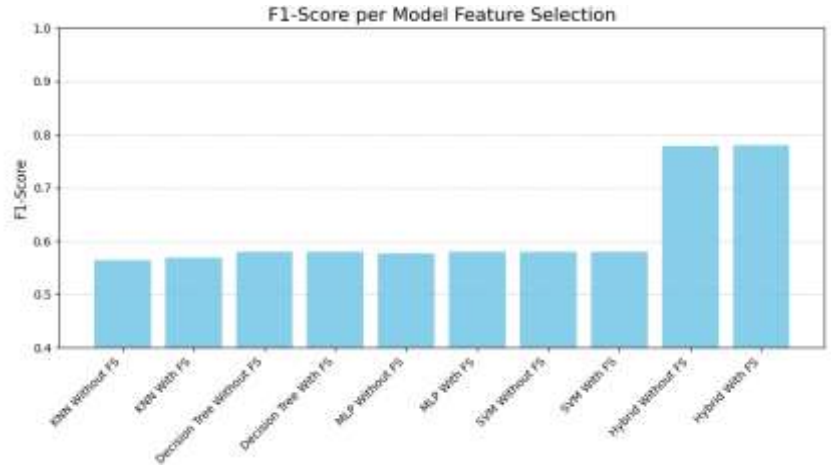

Figure 11: F1-Score per model feature selection

5. CONCLUSION

While AI-based techniques for malware detection and analysis have gained traction, traditional non-AI methods remain indispensable. Techniques like static and dynamic analysis, system call monitoring, and reverse engineering are essential for understanding malware behavior in detail, especially when dealing with sophisticated threats. The analysis of model performance across accuracy, precision, recall, and F1-score reveals that the Hybrid model outperforms all other models significantly, both with and without feature selection (FS). Notably, the Hybrid model with FS achieves the highest accuracy (80.2%), precision (84.3%), and F1-score (0.779665), indicating strong predictive power and balanced performance. KNN shows slight improvements in all metrics when FS is applied, but its overall performance remains the lowest among all models. Decision Tree, MLP, and SVM models exhibit very similar results, with FS having minimal impact on their metrics. The Decision Tree achieves slightly better precision compared to others. Feature Selection has a minimal effect on most models except for the Hybrid model, where it enhances precision and maintains high accuracy and F1-score. In conclusion, while traditional models like KNN, Decision Tree, MLP, and SVM perform adequately, the Hybrid model stands out as the most effective, with FS further optimizing its performance, particularly in precision and accuracy. This suggests that the Hybrid approach is better suited for achieving high-quality results in this scenario.

REFERENCES

[1] Pachhala, N., Jothilakshmi, S., & Battula, B. P. (2021, October). A comprehensive survey on identification of malware types and malware classification using machine learning techniques. In 2021 2nd international conference on smart electronics and communication (ICOSEC) (pp. 1207-1214). IEEE.

[2] Qamar, A., Karim, A., & Chang, V. (2019). Mobile malware attacks: Review, taxonomy & future directions. Future Generation Computer Systems, 97, 887-909.

[3] Alenezi, M. N., Alabdulrazzaq, H., Alshaher, A. A., & Alkharang, M. M. (2020). Evolution of malware threats and techniques: A review. International journal of communication networks and information security, 12(3), 326-337.

[4] Yan, S., Ren, J., Wang, W., Sun, L., Zhang, W., & Yu, Q. (2022). A survey of adversarial attack and defense methods for malware classification in cyber security. IEEE Communications Surveys & Tutorials, 25(1), 467-496.

[5] Alzahrani, A. I., Ayadi, M., Asiri, M. M., Al-Rasheed, A., & Ksibi, A. (2022). Detecting the presence of malware and identifying the type of cyber attack using deep learning and VGG-16 techniques. Electronics, 11(22), 3665.

[6] Ferdous, J., Islam, R., Mahboubi, A., & Islam, M. Z. (2023). A State-of-the-Art Review of Malware Attack Trends and Defense Mechanism. IEEE Access.

[7] Rudd, E. M., Rozsa, A., Günther, M., & Boult, T. E. (2016). A survey of stealth malware attacks, mitigation measures, and steps toward autonomous open world solutions. IEEE Communications Surveys & Tutorials, 19(2), 1145-1172.

[8] Divya, S. (2013). A survey on various security threats and classification of malware attacks, vulnerabilities and detection techniques. International Journal of Computer Science & Applications (TIJCSA), 2(04).

[9] Adhikari, P.P., Mall, P.K., Mishra, A., Srivastava, S. (2024). KDSR: Hybrid Machine-Learning Solution for Intrusion Detection in Fog Computing Environment. In: Woungang, I., Dhurandher, S.K., Singh, Y.J. (eds) Proceedings of the NIELIT's International Conference on Communication,

Electronics and Digital Technology. NICEDT 2024. Lecture Notes in Networks and Systems, vol 1022. Springer, Singapore. https://doi.org/10.1007/978-981-97-3601-0_28

[10] Mall, P. K., & Singh, P. K. (2023). Credence-Net: a semi-supervised deep learning approach for medical images. International Journal of Nanotechnology, 20(5-10), 897-914.

[11] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. IEEE access, 7, 46717-46738.

[12] Rathore, H., Agarwal, S., Sahay, S. K., & Sewak, M. (2018). Malware detection using machine learning and deep learning. In Big Data Analytics: 6th International Conference, BDA 2018, Warangal, India, December 18–21, 2018, Proceedings 6 (pp. 402-411). Springer International Publishing.

[13] Tobiyama, S., Yamaguchi, Y., Shimada, H., Ikuse, T., & Yagi, T. (2016, June). Malware detection with deep neural network using process behavior. In 2016 IEEE 40th annual computer software and applications conference (COMPSAC) (Vol. 2, pp. 577-582). IEEE.

[14] He, K., & Kim, D. S. (2019, August). Malware detection with malware images using deep learning techniques. In 2019 18th IEEE international conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering (TrustCom/BigDataSE) (pp. 95-102). IEEE.

[15] Choi, S., Jang, S., Kim, Y., & Kim, J. (2017, October). Malware detection using malware image and deep learning. In 2017 international conference on information and communication technology convergence (ICTC) (pp. 1193-1195). IEEE.

[16] Sabhadiya, S., Barad, J., & Gheewala, J. (2019, April). Android malware detection using deep learning. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1254-1260). IEEE.

[17] Alomari, E. S., Nuiaa, R. R., Alyasseri, Z. A. A., Mohammed, H. J., Sani, N. S., Esa, M. I., & Musawi, B. A. (2023). Malware detection using deep learning and correlation-based feature selection. Symmetry, 15(1), 123.

[18] Kim, T., Kang, B., Rho, M., Sezer, S., & Im, E. G. (2018). A multimodal deep learning method for android malware detection using various features. IEEE Transactions on Information Forensics and Security, 14(3), 773-788.

[19] Fernando, D. W., Komninos, N., & Chen, T. (2020). A study on the evolution of ransomware detection using machine learning and deep learning techniques. IoT, 1(2), 551-604.

[20] Bae, S. I., Lee, G. B., & Im, E. G. (2020). Ransomware detection using machine learning algorithms. Concurrency and Computation: Practice and Experience, 32(18), e5422.