Lightweight Fault Detection in UAVs: A Machine Learning Approach with Dynamic Time Windows

Saeed Alseiari
Secure Systems Research Center - Technology Innovation Institute (TII)
Abu Dhabi, UAE
saeed.alseiari@tii.ae

Willian T. Lunardi
Secure Systems Research Center - Technology Innovation Institute (TII)
Abu Dhabi, UAE
willian.lunardi@tii.ae

Martin Andreoni
Secure Systems Research Center - Technology Innovation Institute (TII)
Abu Dhabi, UAE
martin.andreoni@tii.ae

ABSTRACT

The increasing integration of Unmanned Aerial Vehicles (UAVs) across various industries highlights the critical need for reliable fault detection mechanisms to ensure their safe and continuous operation. This paper presents a Machine Learning (ML)-based approach for efficient UAV fault detection on edge devices, addressing the limitations of traditional computationally intensive models often unreliable in real-world conditions. We introduce a scalable, lightweight fault detection model that dynamically adjusts feature extraction time windows to balance processing costs and detection accuracy. Our model employs a series of ten classifiers, each optimized for different time windows, and utilizes a hierarchical processing strategy to maintain low latency and high accuracy. Our methodology uses the "Realistic UAV Fault Dataset" to train and evaluate the model. The dataset was collected using a Holybro X500 UAV with intentional fault scenarios, providing a robust foundation for model training. Our model efficiently manages computational resources and enhances fault detection reliability by leveraging Pareto curves to determine optimal rejection thresholds. The evaluation demonstrates that our approach significantly improves detection accuracy while minimizing computational overhead, with an F1 score of 0.985, a False Positive Rate of 2.22%, and a False Negative Rate of 0.19%, while showing low increases of 8.9% and 11.6% in CPU and memory usage during analysis.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Feature selection**; • **Hardware** → *Fault models and test metrics*;

KEYWORDS

Machine Learning, UAV, Fault Detection, Time Windows, Pareto

1 INTRODUCTION

The increasing deployment of UAVs across various industries, from agriculture to logistics, underscores the critical need for reliable and secure autonomous systems. As UAVs become more integrated into daily operations, ensuring their continuous and fault-free operation is paramount[1]. Faults such as broken propellers compromise mission success and can lead to significant safety hazards and financial losses [10]. Therefore, robust fault detection mechanisms are essential to maintain operational integrity and safety standards in autonomous UAV.

This paper addresses the challenge of developing a reliable and efficient ML model for UAV fault detection on edge devices. Traditional methods often focus on detecting faults by analyzing movement related data from UAV sensors like accelerometers and gyroscopes. These methods require extensive data processing and feature extraction, which can be computationally intensive and unsuitable for resource-constrained UAV environments [7]. Additionally, the effectiveness of these models heavily depends on the quality and diversity of the training datasets, which are often difficult to obtain [2].

In our study, we build upon the findings of Katta and Viegas [4] from our research center, which explored the initial application of ML for fault detection using accelerometer, gyroscope, and audio data. While Katta and Viegas focused on foundational aspects, our work extends their approach for UAV fault detection, aiming to reduce computational costs while maintaining high detection accuracy. Unlike the previous work that utilized deep learning techniques, we treat the problem as a time series anomaly detection using lightweight ML-based methods, reducing the number of inputs and external hardware, such as a microphone. By utilizing only accelerometer and gyroscope data, we achieve better accuracy.

Our method involves a two-phase ML model with a reject option. The first phase uses an ensemble of classifiers optimized for different feature extraction time windows to classify UAV data. Low-confidence classifications are rejected and re-evaluated by classifiers with larger feature extraction windows, enhancing accuracy without compromising computational efficiency. The second phase identifies and signals unreliable classifications for operator review, facilitating continuous model improvement. By balancing detection accuracy and computational efficiency, our approach aims to be more practical for real-time UAV operations. Unlike previous models that compromise detection accuracy or require extensive computational resources, our model leverages a dynamic feature extraction process and a reject option to optimize performance. Through this mechanism, we create a scalable and reliable solution for UAV fault detection.

Our evaluation of current ML-based approaches for UAV fault detection, including experiments on a new dataset focused on broken UAV propeller detection, demonstrates that the feature extraction time window is directly related to detection accuracy and processing costs. Building on these findings, we propose a reliable ML model for lightweight UAV fault detection that proactively identifies the suitable feature extraction time window for the classification task. This new model can improve true-positive rates by up to 2.22% and true-negative rates by up to 0.19%.

The remainder of the paper is organized as follows. Section 2 reviews the related works. In Section 3, the main challenges of the ML in UAV fault detection. Section 4 describes our proposed model. Finally, Section 5 concludes our work.

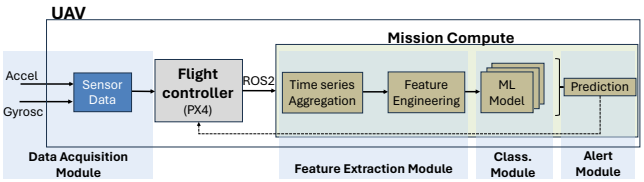


Figure 1: Traditional fault detection sequence, composed of four stages. First, the data acquisition module gathers the data from sensors. Secondly, feature engineering abstracts statistical data on processed features, and then the classification module executes different machine learning models for classifying the data. Finally, the alert module is sent to the flight controller or the operator, signaling the fault.

2 RELATED WORKS

Detection of UAV faults is crucial for ensuring their safe operation. The typical fault detection process consists of four interconnected modules as illustrated in Figure 1: Data Acquisition, Feature Extraction, Classification, and Alert. The process starts with the Data Acquisition module, which gathers high-frequency sensor data, such as readings from the gyroscope and accelerometer, generally at 100 Hz. This data is then processed in the Feature Extraction module, which compiles a behavioral feature vector from data summarized over a predefined time window. Subsequently, the Classification module uses this vector to identify any physical faults, and the results are communicated via the Alert module to signal detected issues [8].

Recent studies have explored various ML techniques for UAV fault detection. These methods typically require building an ML model through a computationally intensive training process using a training dataset that is representative enough to ensure reliable model construction. The effectiveness of these models is assessed using a test dataset, with the results serving as an indicator of the model's real-world performance [9].

Unmanned Aerial Vehicles (UAVs), being resource-constrained devices, face significant challenges in implementing fault detection processes. These challenges are compounded by the need to execute various processing tasks simultaneously [2]. Several researchers have focused on detecting specific faults like broken propellers using visual inspections performed pre- or post-mission, thus not utilizing onboard ML algorithms [3]. Others have proposed collecting audio data during flights, requiring additional hardware and complex ML algorithms that increase resource consumption, potentially jeopardizing the mission [4, 5]. Moreover, comparative studies have shown that data-driven approaches are generally more accurate than audio-driven methods [6].

The high processing costs are a significant issue with current fault detection methods. Researchers often extend the feature extraction time window to increase accuracy and employ sophisticated deep-learning architectures for classification tasks. However, these approaches are not feasible for deployment on UAVs due to the excessive computational resources they require [2, 9].

Furthermore, the reliability of fault identification in autonomous UAVs remains a challenge. Autonomous UAVs can exhibit a wide array of potential faults that are difficult to reproduce in training datasets. This makes it challenging for the deployed ML model to recognize new scenarios as they arise. Constructing a realistic training dataset is critical, and any shortcomings here can lead to an ML model that fails to achieve the accuracy observed during the testing phase when deployed in production environments [8]. Identifying an unreliable ML model can be difficult, often discovered only after manually analyzing false positive events [2].

3 CHALLENGES IN UAV FAULT DETECTION

In this section, we explore the challenges of developing a realistic UAV fault dataset and assess the performance of traditional ML models. We begin by describing our data collection process and then analyze the impact of processing costs on model accuracy, highlighting the need for optimized fault detection solutions.

(a) Holybro X500 UAV **(b) UAV Propeller**

Figure 2: Unmanned Aerial Vehicle (UAV) used on the *Realistic UAV Fault Dataset.*

Creating a realistic dataset for ML-based fault detection in autonomous UAV applications presents significant challenges. Related studies often accomplish this by using simulated environments, where data collection follows a set of predefined scenarios. However, this method does not accurately capture the complexities of real-world conditions, rendering the resulting ML model unreliable for practical applications.

To address this issue, we utilize the "Realistic UAV Fault Dataset", introduced by Katta and Viegas [4] from our research center. This dataset includes data from an experiment involving a UAV with a broken propeller fault scenario. We gathered the data using a Holybro X500 UAV, equipped with an Intel UP Xtreme i7 8665UE Mission Compute (MC), managed by PX4 Autopilot software. The UAV flew

autonomously in an eight-shape pattern for approximately five minutes during each experiment. Data was collected using the Robot Operating System (ROS2) throughout the mission.

Each test involved flying the UAV under normal conditions with four intact propellers or under fault conditions, where one to four propellers were intentionally damaged by trimming approximately one centimeter from the edge of each. In total, we conducted 60 flights, amounting to over five hours of flight time—half of these in normal conditions and the other half under fault conditions. Figure 2 illustrates the UAV and the varied propeller scenarios employed during our tests. Thus, our dataset offers realistic data from UAV flights under various operational conditions, which supports the development and validation of ML techniques for fault detection.

From this dataset, we utilized a comprehensive set of features extracted from accelerometer and gyroscope data to enhance the fault detection capabilities of our machine-learning model. The 78-feature set included basic statistical measures and advanced distribution characteristics within each time window. Each feature is obtained on the three axes, x, y, and z. Specifically, we calculated the count of values, that is, the number of values in that time window, the maximum and minimum values, the mean, and the standard deviation to capture the central tendency and variability of the data. Quartile-based features such as the first, median, and third quartiles were included to provide a detailed understanding of the data distribution. Additionally, we counted the number of positive, negative, and values greater than the mean and less than the mean to describe the data's distribution further. The kurtosis was also measured to assess the "tailedness" of the distribution, indicating the presence of outliers. This diverse set of features ensured the model had a robust and detailed representation of the sensor data, facilitating accurate and reliable fault detection in UAV operations. Table 1 describes the features used in the model, applying to both accelerometer and gyroscope data across all axes x, y, and z.

Feature	Description
count	Number of values in the time window
mean	Mean value in the time window
std	Standard deviation in the time window
min	Minimum value in the time window
25%	First quartile value in the time window
50%	Median value in the time window
75%	Third quartile value in the time window
max	Maximum value in the time window
posMeanCount	Number of values greater than the mean
negMeanCount	Number of values less than the mean
posCount	Number of positive values
negCount	Number of negative values
Kurtosis	Kurtosis of the distribution

Table 1: Description of Features Used in the Model (Applies to both accelerometer and gyroscope data across all axes)

Table 2: Classification performance of the selected classifiers for UAV fault detection using a 5-second feature extraction time window.

Classifier	F-Measure (%)	FPR (%)	FNR (%)
Random Forest (RF)	98.99	0.85	0.77
Gradient Boosting (GBT)	98.53	0.85	1.51
k-Nearest Neighbors (kNN)	96.83	2.75	2.36
Decision Tree (DT)	97.95	0.58	2.67
Gaussian Naive Bayes (NB)	97.13	0.04	4.45

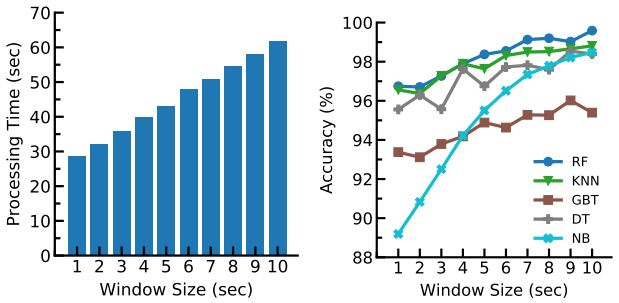
3.1 ML-based Detection of UAV Faults

Building our model leverages traditional ML techniques for fault detection in UAV applications, utilizing accelerometer and gyroscope data collected continuously at a 100 Hz frequency. We employ a sliding window approach for feature extraction, computing statistical features such as the average values from the horizontal axis accelerometer over the last 5 seconds. A total of 200 distinct feature values are extracted from this process and are then used for classification.

For our analysis, we evaluated five widely recognized ML classifiers: RF, GBT, kNN, DT, and NB. The RF and GBT classifiers use 100 decision trees each, with RF and DT utilizing the Gini impurity measure for determining splits, while GBT employs a learning rate of 0.1 and the Friedman mean squared error. The kNN classifier is configured with 5 neighbors, using the Euclidean metric to measure distances. These classifiers were implemented using the sci-kit-learn API version 1.1.1. To prepare the data for classification, we applied min-max normalization to the features and partitioned the dataset into training, validation, and testing sets, comprising 40%, 30%, and 30% of the data, respectively. This segmentation thoroughly evaluates each classifier's generalization ability across different data scenarios.

The classifiers' performance was rigorously evaluated based on their False-Positive Rate (FPR), False-Negative Rate (FNR), and F-Measure. Here, False-Positive (FP) represents the proportion of normal UAV samples incorrectly identified as faults, while False-Negative (FN) represents the proportion of actual fault instances that are misclassified as normal. Our initial experiment assessed the accuracy of traditional ML-based fault detection techniques for UAVs by using a feature extraction time window of 5 seconds with a 1-second sliding window for updates. Every second, a new sample to be classified is generated based on the most recent 5-second data window, as detailed in Table 2, which shows the classifiers achieving low error rates.

Despite the promising results, deploying these classifiers in an autonomous UAV setting demands significantly lower error rates to avoid critical operational failures, such as incorrect landing decisions triggered by a false positive. Ensuring the reliability of fault detection systems is crucial, given their potential impact on autonomous operations, highlighting the ongoing need for enhancements in ML techniques to handle such high-stakes environments effectively.



(a) Feature extraction window time *vs*. processing costs **(b) Feature extraction window time *vs*. f-measure**

Figure 3: Evaluation of feature extraction time window size on processing costs and classification accuracy in our dataset.

The second experiment we conducted aimed to analyze the trade-off between accuracy and processing costs associated with traditional ML techniques and to evaluate how the system's accuracy can be enhanced at the expense of increased processing costs. To explore this, we examined the effects of varying feature extraction window sizes on the performance of the selected ML classifiers. The findings, as depicted in Figure 3, illustrate the relationship between processing costs and the F-Measure across different feature extraction window sizes, highlighting how changes in the time window impact the accuracy of the ML models.

Notably, the NB classifier demonstrated a significant increase in F-Measure, improving from 0.89 in a 1-second window to 0.98 in a 10-second window, as detailed in Figure 3b. This improvement, however, comes with a corresponding increase in processing costs. On average, each additional second added to the feature extraction window increased 8.95% in processing costs, as shown in Figure 3a. This experiment underscores the critical balance between enhancing model accuracy and managing the computational resources required, especially in real-time applications like UAV fault detection. It highlights the necessity to consider the implications of extended processing times on the overall system performance and operational efficiency.

4 OUR PROPOSED MODEL

We introduce a reliable, efficient UAV fault detection model designed to overcome the limitations of high-error, computationally intensive ML models highlighted in Section 3. Our approach, depicted in Fig 4, dynamically adjusts the feature extraction time window to strike an optimal balance between processing costs and detection accuracy. At the heart of our model lies a sequence of ten trained classifiers, each configured to operate over incrementally increasing feature extraction windows from 1 to 10 seconds.

The model initiates fault detection by gathering real-time data into a queue, where it first processes features and performs classification using the briefest, 1-second window. This approach minimizes latency and computational overhead under normal operating conditions. The model discards this result if the classification from this initial window lacks sufficient confidence—falling below a set reliability threshold. It then escalates to the next time window, extending the feature extraction to 2 seconds, and reclassifies using

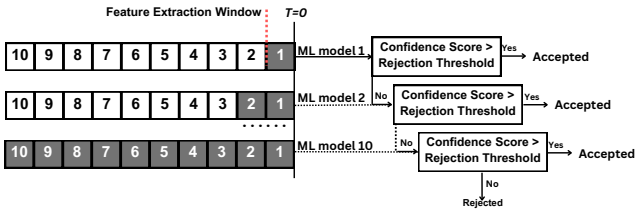


Figure 4: Proposed reliable, lightweight fault detection model for autonomous UAVs.

this expanded dataset. This hierarchical processing strategy ensures efficient use of computational resources while maintaining high detection accuracy.

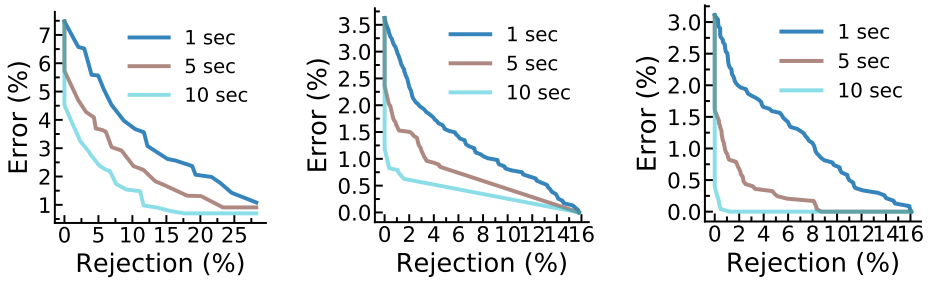
As illustrated in Fig 3b, the RF model achieves 100% accuracy at the ten-second mark. This iterative process methodically increases the feature extraction window by one second at each stage, striving to achieve classification confidence above a predetermined threshold or until it reaches the maximum limit of 10 seconds. Should the predictions remain unreliable at this extended time, they are categorized as unclassified. This indicates that the model could not confidently assess the UAV's condition with the available data. This adaptive strategy enables our system to dynamically balance computational load and accuracy, optimizing for rapid, cost-effective processing where feasible and escalating to more resource-intensive analysis only when essential for ensuring reliable fault detection.

A key advantage of our model is its efficiency in operation, conserving computational resources by allocating more extensive processing only to lower-confidence predictions that cannot be resolved in shorter time windows. Our approach significantly enhances the UAV fault detection, providing a scalable model that balances computational efficiency with reliable classification. With this advanced strategy, we aim to redefine standards for lightweight, dependable fault detection systems in autonomous UAVs, promoting safer and more effective operations in diverse conditions.

4.1 Selection of Rejection Thresholds Using Pareto Curves

In optimizing our UAV fault detection model, the Pareto curves play a pivotal role by delineating the optimal rejection thresholds for each classifier across various time windows. These curves illustrate the trade-off between error rates and rejection percentages, serving as a visual guide to strategically select thresholds that minimize error while efficiently managing computational resources. The plots for kNN, GBT and RF, classifiers, spanning feature extraction windows of 1, 5, and 10 seconds as shown in Fig 5, are crucial for this analysis. By examining the progression of these curves, we can discern the relationship between the rejection percentage—represented on the x-axis as the proportion of data classified below the confidence threshold and consequently discarded—and the error rate depicted on the y-axis as the percentage of incorrect predictions.

The curve closest to the origin represents the most favorable balance, indicating that the classifier can achieve the lowest error rate with a minimal rejection rate. We determine our rejection thresholds at these optimal points nearest to the origin for each



(a) kNN Pareto Curve (b) GBT Pareto Curve (c) RF Curve

Figure 5: The Pareto curves of three different classifiers for 1, 5, and 10 seconds.

curve, reflecting a precise balance between maintaining accuracy and reducing computational demands for each classifier and feature extraction window. By meticulously analyzing these Pareto curves and setting these critical thresholds, we ensure that our UAV fault detection system operates efficiently and leverages high-confidence predictions for quick processing. This strategy allows for a more intensive analysis only when necessary, preserving the integrity and reliability of the fault detection process while ensuring the system remains responsive and effective under varying operational demands.

4.2 Evaluation

To assess the performance of our model, we conducted an extensive testing regime involving ten distinct Random Forest (RF) classifiers, each tailored to operate over a progressively increasing feature extraction time window from 1 to 10 seconds. Based on the Pareto curve analysis outlined in Section 4.1, each classifier's rejection threshold was established to optimize performance and minimize error rates.

During the testing phase, designed to emulate real-world operational conditions, we employed a test dataset where the initial classification was attempted using the 1-second time window classifier. If the prediction at this stage failed to achieve the confidence level mandated by the predefined rejection threshold, the model would escalate to the 2-second classifier, continuing this sequential increase up to the 10-second classifier as required. This methodical approach ensures that each prediction is refined and reassessed, enhancing the accuracy of the fault detection process.

The performance metrics derived from this evaluation reveal the robustness of our model. The F1 score of 0.985 highlights a good balance between the precision in identifying true positives and the efficacy in minimizing false negatives. Regarding error rates, the Random Forest classifiers demonstrated a FPR of 2.22%, indicating a minimal misclassification of non-fault conditions as faults. More impressively, the FNR was recorded at a mere 0.19%, showcasing the model's high reliability in detecting fault conditions. This robust performance underscores the effectiveness of our model in providing reliable and precise fault detection under conditions that closely simulate actual UAV operations.

Figure 6 illustrates the distribution of samples classified at each time window, notably excluding the 1-second window, where all initial classifications occur, with subsequent samples advancing

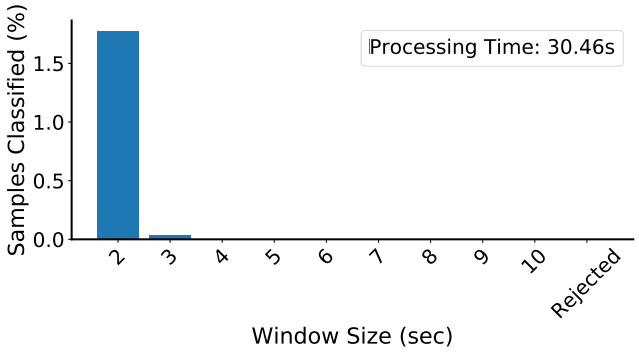


Figure 6: Percentage of points classified by each time window and the processing time of the model

to longer windows only if rejected. The "Rejected" category denotes instances where the model's classifications did not meet the confidence threshold at any stage of the various time windows. This setup simulates the operational environment by conducting feature extraction and model classification on a continuous stream of unprocessed test data in a fixed-size queue, ensuring that the experiment closely replicates real-world conditions on a drone.

The overall processing time for this experiment was 30.46 seconds, demonstrating the model's efficiency and lightweight design. Within this framework, 1.78% of points initially rejected by the 1-second window were successfully classified at the 2-second window, while only a minuscule 0.04% of points required the 3-second window for classification. These results underscore the efficacy of our model in managing computational resources effectively, minimizing processing time by accurately classifying most samples during the earliest stages. Tailored rejection thresholds for each time window ensure that extensive data analysis is employed judiciously, optimizing processing power without compromising the detection capabilities of the UAV fault detection system.

Figure 7 illustrates the system's performance by comparing CPU usage, RAM usage, and energy consumption in idle and active analysis states. During analysis, the system's CPU usage peaks at around 8.9%, which is only less than. one order of magnitude from its minimal usage when idle. Similarly, RAM usage increases only 11%, reflecting the demands of active processing and memory allocation. Despite these demands, energy consumption remains

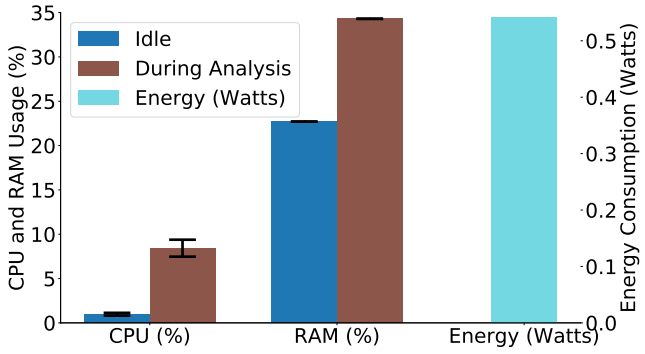


Figure 7: Percentage of points classified by each time window and the processing time of the model

low, up to 0.6 watts during analysis. This minimal increase underscores the lightweight nature of the solution. The system's ability to manage resources efficiently and handle intensive analysis tasks with such low power usage highlights its suitability for applications in autonomous systems. Maintaining low energy consumption while performing demanding computational tasks emphasizes the system's efficiency as a lightweight solution.

5 CONCLUSION

This paper presented a reliable and efficient UAV fault detection model that leverages a tiered classification system guided by selected rejection thresholds. The model showed a good performance, as evidenced by an F1 score of 0.985, alongside low false positive and negative rates of 2.22% and 0.19%, respectively. These results highlight the model's ability to balance high detection accuracy with minimal computational costs, making it well-suited for real-time applications in resource-constrained UAV environments. A key achievement of this study is the innovative approach of dynamically adjusting feature extraction time windows, which optimizes the trade-off between processing speed and detection accuracy. The model effectively manages computational resources while maintaining robust fault detection capabilities by progressively increasing the feature extraction window and employing a reject option for low-confidence classifications. This strategy ensures that most faults are accurately identified in the earliest stages, significantly reducing processing time and enhancing the overall efficiency of the fault detection system.

This research has limitations. The dataset was recorded exclusively indoors, which may affect the model's generalizability to outdoor conditions. Additionally, the model is tuned to detect faults related to slightly broken propellers, which are less common than other faults like crashes or wear and tear. Future research should include outdoor data and a broader range of fault types to enhance the model's applicability and robustness for UAV fault detection.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to Eduardo K. Viegas and SivaPrasad Nandyala for their valuable discussions and insightful suggestions, which greatly improved this work.

REFERENCES

[1] Martin Andreoni Lopez, Michael Baddeley, Willian T Lunardi, Anshul Pandey, and Jean-Pierre Giacalone. 2021. Towards secure wireless mesh networks for UAV swarm connectivity: Current threats, research, and opportunities. In *17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 319–326.

[2] Mohamad Hazwan Mohd Ghazali, Kelvin Teoh, and Wan Rahiman. 2022. Real-time data collection technique for UAVs propeller inspection. In *Control, Instrumentation and Mechatronics: Theory and Practice*. Springer, 170–179.

[3] Mohamed Salim Harras, Shadi Saleh, Batbayar Battseren, and Wolfram Hardt. 2023. Vision-based Propeller Damage Inspection Using Machine Learning. *Embedded Selforganising Systems* 10, 7 (2023), 43–47.

[4] Sai Srinadhu Katta and Eduardo Kugler Viegas. 2023. Towards a reliable and lightweight onboard fault detection in autonomous unmanned aerial vehicles. In *International Conference on Robotics and Automation (ICRA)*. IEEE, 1284–1290.

[5] Wansong Liu, Chang Liu, Seyedomid Sajedi, Hao Su, Xiao Liang, and Minghui Zheng. 2024. An audio-based risky flight detection framework for quadrotors. *IET Cyber-Systems and Robotics* 6, 1 (2024), e12105.

[6] Mohamad H Mohd Ghazali, Dinagaran Tamilselvam, and Wan Rahiman. 2024. Comparative analysis between data-driven and visual-based approach in drone propeller imbalance detection. *Journal of Vibration and Control* (2024), 10775463241234158.

[7] Radosław Puchalski and Wojciech Giernacki. 2022. UAV fault detection methods, state-of-the-art. *Drones* 6, 11 (2022), 330.

[8] Md Habibur Rahman, Mohammad Abrar Shakil Sejan, Md Abdul Aziz, Rana Tabassum, Jung-In Baik, and Hyoung-Kyu Song. 2024. A Comprehensive Survey of Unmanned Aerial Vehicles Detection and Classification Using Machine Learning Approach: Challenges, Solutions, and Future Directions. *Remote Sensing* 16, 5 (2024), 879.

[9] Chia-Ming Tsai, Chiao-Sheng Wang, Yu-Jen Chung, Yung-Da Sun, and Jau-Woei Perng. 2021. Multi-sensor fault diagnosis of underwater thruster propeller based on deep learning. *Sensors* 21, 21 (2021), 7187.

[10] Benkuan Wang, Xiyuan Peng, Min Jiang, and Datong Liu. 2020. Real-time fault detection for UAV based on model acceleration engine. *IEEE Transactions on Instrumentation and Measurement* 69, 12 (2020), 9505–9516.