Journal of Autonomous Intelligence (2024) Volume 7 Issue 5
doi: 10.32629/jai.v7i5.1637

ORIGINAL RESEARCH ARTICLE

Unmanned aerial vehicle resilience: A new approach to fault diagnostics framework

Ashish A. Mulajkar[1], Varsha D. Jadhav[2], Dhananjay R. Dolas[3], S. Gowtham[4], Madhuri S. Bhagat[5], Harshal Patil[6], P. Satishkumar[7,*]

[1] *School of Electronics & Telecommunication Engineering, MIT Academy of Engineering, Alandi, Pune 412105, India*

[2] *Department of Artificial Intelligence and Data Science, Vishwakarma Institute of Information Technology, Pune 411048, India*

[3] *Department of Mechanical Engineering, Jawaharlal Nehru Engineering College, MGM University Aurangabad, Aurangabad 431001, India*

[4] *Department of Electrical and Electronics Engineering, K.S.R. College of Engineering, Tiruchengode 637215, India*

[5] *Department of Civil Engineering, Yeshwantrao Chavan College of Engineering, Nagpur 441110, India*

[6] *School of Computer Science & Engineering, IILM University, Greater, Noida 201306, India*

[7] *Department of Mechanical Engineering, Rathinam Technical Campus, Coimbatore 641021, India*

*** Corresponding author:** P. Satishkumar, sp.sathishkumar10@gmail.com

ABSTRACT

The unmanned aerial vehicles (UAVs) have become crucial resources for various tasks, from surveillance and tracking to environmental monitoring and responding to disasters. However, as UAV systems and their tactical surroundings get more complicated, there is a greater chance that they will malfunction or fail. A novel method for fault classifier using Genetic-Tuna Swarm Optimized Deep Neural Network (GTSO-DNN) is applied for anomaly identification. The collection of data from simulated propeller damage, Min-Max normalization for pre-processing, and Principal Component Analysis (PCA) for feature extraction are all included. A UAV model integrates dynamic and propeller models built using a Gated Recurrent Unit (GRU) network. Experimental findings demonstrate the GTSO-DNN's superior performance compared to existing methods (K Nearest Neighbour, Decision Tree, Support Vector Machines) in terms of accuracy—98.51%, precision—98.7%, recall—98.9%, and F1 score—97.2%. The GTSO-DNN efficiently locates and classifies problems, improving UAV resilience. With potential applications for improving real-time UAV safety, this comprehensive methodology enhances fault diagnosis.

Keywords: unmanned aerial vehicles; fault diagnostics; genetic-tuna swarm optimized deep neural network (GTSO-DNN); propeller model; principal component analysis (PCA); gated recurrent unit (GRU)

ARTICLE INFO

Received: 11 March 2024
Accepted: 11 April 2024
Available online: 10 July 2024

1. Introduction

In recent years, autonomous flying vehicles referred to as UAVs or drones, have revolutionized several industries, including farming, monitoring, disaster response, and logistics. Activities previously thought to be difficult or even impossible have been accomplished using these adaptable platforms. However, as UAVs continues to be integral in important applications, guaranteeing their consistent and reliable operation becomes crucial[1,2]. In the context of UAVs, the term "resilience" refers to the capability of these aircraft to adjust to, suffer, and recover from unforeseen circumstances, disruptions, or errors without affecting their operation or mission success. UAVs were built with sophisticated control systems and cutting-edge technology, yet they still face problems, including sensor failures, communication problems, bad weather, and even intentional attacks[3].

This intrinsic weakness highlights the need for developing strong solutions that improve UAV resilience. A resilient UAV can independently recognize, assess, and correct defects, minimizing the negative effects on its performance. The idea of resilience also covers proactive actions such as fault early detection, redundant application, and adaptive configuration to ensure prolonged performance in the face of difficulty. It goes beyond simple fault reaction[4]. UAV resilience was a topic with a dynamic and broad research landscape. By introducing unique procedures and techniques to strengthen the resilience of UAVs, this work seeks to contribute to this field's increasing understanding of fault diagnostics[5]. Fault Diagnostics in Unmanned Aerial Vehicle Resilience, tackles the challenge of maintaining UAV operational reliability despite unexpected issues. In an era where UAVs are increasingly vital in both military and civilian domains, dependable defect diagnostic systems are paramount. These technologies are pivotal for ensuring UAVs operate consistently and safeguarding valuable cargo, airspace security, and ground personnel[6].

The demand for reliable, resilient systems grows as the universe of UAV applications widens. This project aims to help the creation of UAVs that stay firm in the face of unexpected challenges by developing innovative fault diagnosis methodologies inside the UAV domain. The ultimate objective was to guarantee that these airborne platforms remain trustworthy and essential tools in today's technologically advanced settings[7].

This work aimed to create efficient defect diagnosis methods that will improve UAV (Unmanned Aerial Vehicle) robustness. Advanced machine learning methods will be used to examine sensor data to identify and categorize issues effectively. The research aimed to make the UAV more resilient to unforeseen problems and better able to handle them, making UAV operations safer and more dependable under different conditions.

Contribution of this research:

- This study provides a systematic approach to improving Unmanned Aerial Vehicle (UAV) robustness through defect diagnosis. It includes a thorough data collection approach focused on simulating propeller degradation, then Min-Max normalization for preprocessing. Principal Component Analysis (PCA) was utilized to feature extraction.
- The research creates a Unified Aerial Vehicle (UAV) framework by integrating dynamic and propeller models, employing a Gated Recurrent Unit (GRU) network. Additionally, it introduced a novel method utilizing a GTSO-DNNto detect anomalies and faults within UAV systems.
- The GTSO-DNN based classifier improves fault localization and classification within UAV systems, increasing overall robustness. The GTSO-DNN methodology outperforms traditional approaches in terms of accuracy, precision, recall, and F1 score, according to empirical testing.

The rest of this research has been organized as follows: In Section 2, we examine relevant surveys on UAV resilience and examine the opinions, technical papers, and research findings of many writers, as well as suggested remedies to problems with integrated and dispersed systems. Section 3 discussed the dataset, image pre-processing, feature extraction, UAV model using data generation, and GTSO-DNN used for fault classification. In Section 4, we evaluate some of the effectiveness of the suggested method. We provide some conclusions in Section 5.

2. Related works

The novel approach for fault analysis in micro aerial vehicles (MAVs), called "MAVFI"[8]. It offers an end-to-end solution by combining detection of anomalies and recovery procedures. MAVFI improved the dependability of MAV operations through integrated methodologies and algorithms, advancing drone technology. The application of deep learning techniques for identifying and locating UAV failure causes was studied in the study of Sadhu et al.[9]. The research contributes to robotics and automation focus, uses on-board implementation, and examines ways to improve UAV reliability by precisely detecting defects.

A detection of fault system for Fixed-Wing Unmanned Aerial Vehicles (FW-UAVs) according tothe Hybrid Deep Domain Adaptation Networks (HDDAN)and Hampel Filter (HF) was presented in research of Zhang et al.[10]. The method facilitates effective outcomes in real-world scenarios by integrating domain adaptation approaches with strong outlier identification to increase fault detection accuracy. Study of Corbetta et al.[11] offered a hybrid methodology for identifying and treating electric power train issues in UAVs. They aimed to improve the effectiveness of maintenance techniques and defect detection by integrating physics-based and data-driven methodologies.

The article of Segovia Ramírez et al.[12] used radiometric sensors built into unmanned aerial vehicles to detect and diagnose faults in photovoltaic panels. The performance of a photovoltaic system can be improved by using airborne monitoring to see solar panel flaws and increase maintenance effectiveness. A Bayesian network framework was used in study of Hossain et al.[13] and assessed the utility of UAVs in transportation and logistics. It provides insights into the potential of UAVs for effective and dependable delivery systems by conducting a thorough analysis of their performance.

The research of El Jery et al.[14] improved the resilience of autonomous armed systems-of-systems by presenting a multi-swarm cooperative reconfiguration paradigm. To dynamically rearrange the system in reaction to interruptions, it uses several swarm optimization approaches, assuring reliable operation. Study examined the potential of reconfigurable intelligent surfaces (RIS) to enhance connectivity in UAV networks[15]. It offers a technique that uses RIS to improvethe signal's quality and strength in UAV communications. The results show a considerable improvement in connection, pointing to RIS as a potentially useful technology to enhance the functioning of UAV networks.

A unique method for effective bridge inspections employing UAV and machine learning was presented in article of Perry et al.[16]. The process of assessment will be streamlined to increase accuracy and speed. They use UAVs to capture images, which were processed using machine learning techniques to look for structural flaws. Compared to conventional procedures, the results show increased inspection efficiency and accuracy. Research proposed the unmanned aerial vehicle delivery decision tool for evaluating the viability of deploying drones to distribute medical supplies[17]. They offered a mechanism for assessing drone delivery's logistical and financial implications. The results point to possible advantages for remote medical supply transport in light of variables like distance and payload.

Study of Ejaz et al.[18] outlined a novel method for effective disaster management using UAV inside an IoT framework. The objective was to improve disaster-related real-time data collecting and communication. The suggested solution effectively gathers and transmits data by integrating UAVs into the IoT platform. The outcomes show that timely information sharing and resource allocation improved catastrophe response. The article of Bronz et al.[19] described a machine learning-based solution for real-time small fixed-wing UAV malfunction detection. By spotting irregularities and errors during flight, the goal was to increase the efficiency of these UAVs. The proposed method utilizes machine learning methods for real-time sensor data analysis and problem detection. The outcomes show how the technique works to find errors and raise flight safety in general quickly.

Study of Phadke and Medrano[20] identified and analyzed robustness requirements to increase the robustness of UAV swarms. They break down those requirements and offer a technique to improve swarm resilience. The findings deepen the comprehension of UAV swarming behaviour and suggest creating more resilient and flexible systems. The research of Neves et al.[21] provided a multimodal initial fusion technique-based end-to-end method for locating offshore UAV landing platforms. For increased precision, their technique combines data from many sensors. The outcomes show the method's efficacy in delivering reliable landing platform detection under difficult circumstances.

The article of Phadke et al.[22] introduced "Drone2Drone", a framework for search and rescue intended to find missing UAV swarm agents. They suggested a technique for effective drone communication and cooperation to find and recover lost swarm agents. Utilizing the proposed framework, experimental results show enhanced search precision and successful retrieval. Study of Zhang et al.[23] focused on evaluating the load-balancing resilience of UAV swarms. Their strategy includes a suggested technique for assessing swarm performance under various loads. The study's results offer suggestions for boosting swarm robustness and efficiency, improving the effectiveness of UAV swarm operations.

The research of El Jery et al.[24] presented a unique waypoint guiding and adaptive evolution-based 3D route planning method for UAVs. Using evolutionary algorithms, the strategy improves route effectiveness and obstacle avoidance. Experimental outcomes demonstrate increased planning precision and shorter flight times, demonstrating the efficacy of the suggested approach. Study of Jerome Vasanth et al.[25] proposed a novel method for improving photovoltaic module problem diagnostics by combining UAV and deep learning-based image processing. The objective was to increase solar panel defect detection's precision and effectiveness. The suggested system blends UAV-based picture collecting with sophisticated deep-learning techniques, producing encouraging results for more accurate fault recognition and diagnosis.

3. Methodology

This section presents the dataset for fault diagnostics in unmanned aerial vehicle resilience, image pre-processing, and feature extraction. We also discuss the UAV model using data generation, and deep neural networks were used for fault classification. **Figure 1** depicted the summary of the methods.

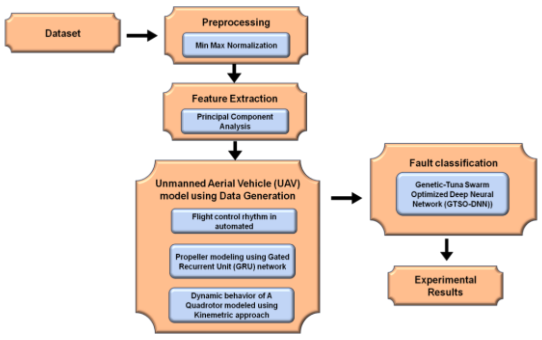
Dataset
Preprocessing
Min Max Normalization
Feature Extraction
Principal Component Analysis
Unmanned Aerial Vehicle (UAV) model using Data Generation
Flight control rhythm in automated
Propeller modeling using Gated Recurrent Unit (GRU) network
Dynamic behavior of A Quadrotor modeled using Kinemetric approach
Fault classification
Genetic-Tuna Swarm Optimized Deep Neural Network (GTSO-DNN))
Experimental Results

Figure 1. Summary of the methods.

3.1. Dataset

We collect a simulated propeller damage dataset to build a learning-based Propeller Model. We drive a propeller with a constant voltage and sensors to measure speed, torque, force, RPM, and efficiencies. This isolated system was called a load cell. The model was trained by utilizing RPM, Thrust, and Torque data produced by altering ESC signals within the system's bounds. Each of the three cases types—"Normal," "Bent," and "Crack"—was tested for five minutes at a time. 75% of the dataset comprised training data to prevent overfitting, while the remaining 25% was testing data[26].

The "Normal" propeller type shows precise torque and thrust prediction by the network, exactly in line with the ground truth with small inaccuracies, primarily at ESC = 2000, the motor's working limit. Further outperforming the norm, the "Bent" propeller type achieves an average rate of errors for torque and thrust estimation of 1.35 and 0.685, respectively. Performance suffers in the "Crack" propeller type compared to the normal and bent forms. Between timesteps 600 and 1300, the model modestly overestimates torque, resulting in a mean error percentage of 2.45% for torque and 4.25% for thrust.

3.2. Min-Max Normalization-based data preparation

Propeller damage image normalization modifies the propeller damage image's pixel value range and distribution to improve model convergence speed, precision, and robustness. The choice of various normalization strategies impacted the performance of machine learning. The latter two methods, Z-score normalization and empirical, were picked and contrasted with the other two. Equation (1) was used to process pixel intensities during min-max normalization.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

where:

X was the data point's initial value.

X_{norm} was the normalized value of the data point.

X_{min} denotes the feature's dataset minimum value.

X_{max} represented the feature's greatest value in the dataset.

The min-max normalization was expressed more simply as Equation (2) when the propeller damage image intensity ranges from 0 to 255.

$$x_{norm} = \frac{x_i}{255} \tag{2}$$

An empirical approach was Equation (2). Every pixel's value within the propeller damage image was scaled and shifted using Z-score normalization according to the dataset's mean and standard deviation. Eq. (3) displays the precise processing technique:

$$x_{norm} = \frac{x_i - \mu}{\sigma} \tag{3}$$

where μ was the propeller damage image dataset's average, and σ represents the standard deviation. The image's original pixel intensity ranged from 0 to 255.

Only the pixel intensity of the image was scaled using the empirical method, leaving the intensity distribution and aesthetic effect untouched. Min-max normalization also scaled the image's intensity range, producing an image with a smoother distribution and a wider intensity range. The software utilizing matplotlib couldn't appropriately show intensity values beyond the scope of 0–1, which led to a major alteration in the image's visual appearance after Z-score normalization. In contrast, the data saved in the image was unaltered. A range between −3 and 3 was also added to the image's intensity levels.

3.3. Extracting features using Principal Component Analysis (PCA).

The basic idea behind PCA was building a transformed linear sequence for real information with specific linked properties. As a result, the main component load (PCL) matrix was modified to represent the actual information by a collection of novel information with few properties. It was suitable for the reduction of dimensionality procedure for data that is multi-dimensional.

For actual data, construct the observational matrix Y. Following N observations, the matrix Y is determined by the x variables $\theta_1, \theta_2, \dots, \theta_x$. Each row was a numerical estimate of the dataset's sample information, and the number n in the column refers to the number of samples that were found.

$$Y = \begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1w} \\ Y_{21} & Y_{22} & \cdots & Y_{2w} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{m1} & Y_{m2} & \cdots & Y_{mw} \end{bmatrix} \tag{4}$$

The data processing should be centralized for observing the matrix. Calculate the sample mean as well.

$$\overline{y_a} = \frac{1}{n}\sum_{b=1}^{m} y_{ba} \tag{5}$$

And Standard deviation:

$$T_a = \sqrt{\frac{1}{n}(y_{ba} - \overline{y_a})^2} \tag{6}$$

Based on equation:

$$\widetilde{y_{ba}} = \frac{y_{ba} - \overline{y_a}}{T_a}(b = 1,2, \dots, m, a = 1,2, \dots, w) \tag{7}$$

Form the standardized matrix $\overline{y_a}$ by carrying out the centralized processing of data.

Based on the equation, calculated an instance correlations matrices,

$$X = \frac{1}{n}\tilde{Y}^S\tilde{Y} \tag{8}$$

Each element in W is calculated according to Equation (9).

$$x_{ba} = \frac{\sum_{l=1}^{m}(y_{ba} - \overline{y_a})(y_{ba} - \overline{y_a})}{\sqrt{\sum_{l=1}^{m}(y_{ba} - \overline{y_a})(y_{ba} - \overline{y_a})^2 \sum_{l=1}^{m}(y_{ba} - \overline{y_a})(y_{ba} - \overline{y_a})^2}} \tag{9}$$

Find *W*'s eigenvalue and eigenvector. Calculate W's *x* feature values, which are $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_x \dots \geq 0$. So, using the equation to determine each primary component's rate of participation:

$$q_b = \frac{\lambda_b}{\lambda_1 + \lambda_2 \dots + \lambda_w}(b = 1,2, \dots, w) \tag{10}$$

Select the highest q main component that achieves 90% and results in $\lambda_{q=1}$ as PCA results. Values for the descending order features were $1,2, \dots, w$ calculated the corresponding eigenvectors, $f_1, f_2, \dots, f_o$. Select the top q vector of features for the PCL.

$$K_{w \times o} = (f_1, f_2, \dots, f_o) \tag{11}$$

Create a linear change for the information using the PCL matrix $K_{w \times o}^S$ and Equation (10) to produce new primary variables $x_1, x_2, \dots, x_q$.

$$\begin{bmatrix} x_1 \\ \vdots \\ x_o \end{bmatrix} = K_{w \times o}^S \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_o \end{bmatrix} \tag{12}$$

After the linear change, the matrix's dimensions changed from *x* to *p*, significantly reducing the volume of the obtained information.

3.4. UAV model using data generation

The proposed model for producing data from quadrotor UAVs. It consists of three primary subsystems: The control system, the propeller model, and the dynamic model. Because of their unpredictability and potential for damage, the control mechanism and brushless DC electric motor cannot be considered for this study. The Dynamic Model used three propellers' input Revolution Per Minute (RPM), which uses physical principles to determine flight data. A quadrotor with defective propellers can be simulated by first developing the Propeller Model and then integrating it into the simulation. The simulation uses preset waypoints to determine torque and thrust and computes propeller RPMs utilizing the control system. The Dynamic Model uses these numbers to determine the quadrotor kinematics when altering RPMs through an iterative feedback loop for accurate maneuvering. Subsequent subsections provide additional details.

3.4.1. Automatic flight control system

The data generative model used the PX4 autopilot controlling mechanism, an open-source program for unmanned aircraft flight control. This model uses input waypoints to determine the desired RPM and ESC signals using the autopilot, enabling accurate UAV navigation. The PX4 system uses sensors, including gyroscopes, accelerometers, magnetometers, and barometers, to determine the vehicle's status for stabilizing and autonomous control reasons.

3.4.2. Propeller modeling using gated recurrent unit (GRU) network

The force (f) and torque (τ) generated by a propeller may often be determined by utilizing Equations (13) and (14), as stated.

$$f_j = k_f * \omega_j^2 \tag{13}$$

$$\tau_j = k_\tau * \omega_j^2 \tag{14}$$

The propeller index is represented by "j" in this section, and the rotational speed was indicated by "ω" in RPM. Experiments can be used to determine the lift constant, k_f, and the drag fixed, k_τ.

However, the previously noted linear connection in Equations (13) and (14) might no longer hold if the propeller gets bent or shattered. A workable remedy for this non-linearity entails using a machine learning strategy, like a Gated Recurrent Unit (GRU) network. This network produces thrust and torque outputs based on the RPM input. Three different propeller models are trained to accommodate other propeller conditions, including "Normal," "Bent," and "Cracked," as seen in **Figure 2**.

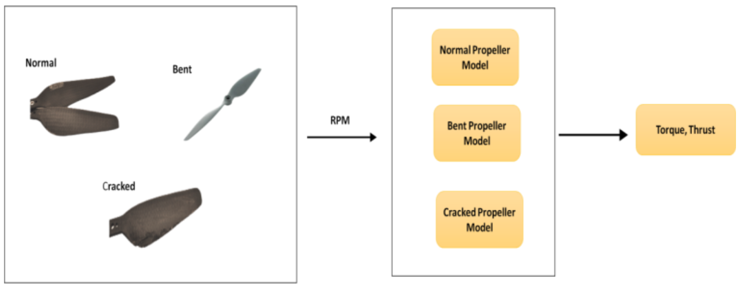

Figure 2. Training for classifying propeller conditions.

Gated recurrent unit (GRU):

The Gated Recurrent Unit (GRU) was a form of recurrent neural network (RNN) design developed to address some of the shortcomings of the conventional RNNs. RNNs are used to handle data sequences such as time-series data or natural language when the order of the elements is important. Due to the gradient-vanishing problem, in which the impact of previous inputs on the present prediction rapidly declines as the sequence gets longer, typical RNNs struggle to capture long-term dependencies.

To solve this issue and enhance RNN training, the GRU was created. The more complicated Long Short-Term Memory (LSTM) network and this gated cell architecture are similar in some ways. While being more efficient at capturing long-range relationships than LSTMs, GRUs had a more straightforward architecture. **Figure 3** depicts the GRU's basic operating concept for a single time step.

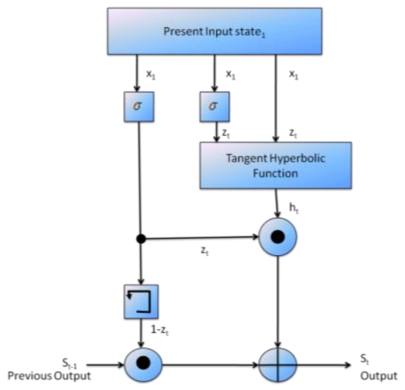

Figure 3. The GRU unit's fundamental functions in time *t*.

A GRU operates as follows, in brief:

Update gate: The update gate decides whether or not to keep any historical data. The prior concealed state and the current input were combined to create it. This gate aids the network in determining which data must be updated and which information must be remembered.

Reset gate: The reset gate determines the amount of the prior hidden state that could be ignored when determining the candidate's hidden condition that was now being considered. It regulated data flow from the previous phase to the current one.

Candidate hidden state: The reset gate and the present input value are used to calculate this new candidate's hidden state. It was a contender since the hidden state could or could not take this form.

Hidden state: The update gate determines the real hidden state for the present step by combining the prior hidden state and the candidate's hidden state. This updated hidden state captures the prior hidden and the present input state's pertinent data. The following equations govern aGRU's behavior:

$$z_t = \sigma(U_z x_t + W_z s_{t-1} + b_z) \tag{15}$$

$$r_t = \sigma(U_r x_t + W_r s_{t-1} + b_r) \tag{16}$$

$$h_t = tanh(U_h x_t + W_h(s_{t-1} \odot r_t) + b_h) \tag{17}$$

$$s_t = (1 - z_t) \odot s_{t-1} + z_t \odot h_t \tag{18}$$

where U_z, U_r, and U_h stand for the input weight matrices; W_z, W_r, and W_h are the recurrent weights; and b_z, b_r, and b_h are biases. z_t stands for the updated gate, r_t for reset gate, $x_t$stands for the current input state, h_t was the candidate activation, is the sigmoid function, and represents an element-wise multiplication. It was clear from the result function that information from past memory could be discarded and the present situation would be remembered, and vice versa, if the updated gate is close to 1. The GRU is the best contender among the others because of its innate ability to recall long-term knowledge and ignore some irrelevant data to extract the underlying relationship of a model. Consequently, it stands out among the other data-driven strategies as the greatest contender for creating the mitigation strategy.

3.4.3. The dynamical behaviour of a quadrotor modeled using a kinematic approach

Using well-established physical models, the Propeller Model provides each propeller torque and thrust to enable computation of quadrotor kinematics. Roll, pitch, and yaw control, which correspond to rotation angles relative to the quadrotor's center, create motion in a quadrotor. A dual-coordinate system was used for altitude monitoring, consisting of the inertial system fixed to the earth and the structural frame connecting at the quadrotor's center of gravity. **Figure 4** illustrates the Quadrotor's Inertia and Coordinate Frame. The angular difference between these coordinates determines the attitude behavior of the quadrotor in space.

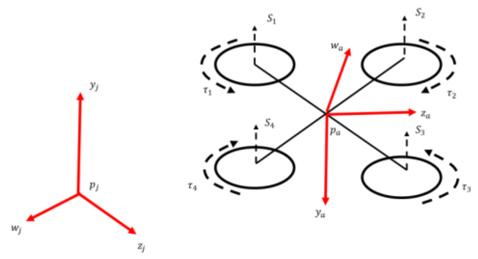

Figure 4. Inertia and coordinate frame of the quadrotor.

There are twelve states of the quadrotor, as shown below, which can be used to characterize its motion.

$$X = [\emptyset, \theta, \psi, \dot{\emptyset}, \dot{\theta}, \dot{\psi}, X, Y, Z, \dot{X}, \dot{Y}, \dot{Z}] \tag{19}$$

The orientation of the quadrotor about the inertia coordinate system was described by the roll, pitch, and yaw angles, represented as $(\emptyset, \theta, and \psi)$, and their related angular velocities $(\dot{\emptyset}, \dot{\theta}, and \dot{\psi})$. The quadrotor's actual position and motion inside the earth-fixed system were represented by the following six states (X, Y, and Z) and their related velocities $(\dot{X}, \dot{Y}, and\ \dot{Z})$. These elements characterize the quadrotor's orientation in space and placement together in an understandable manner.

A quadrotor's movement was fundamentally started by adjusting the torque and thrust distributed among its four propellers, which results in modifications to pitch, roll, and yaw angles. This dynamic model consists of a translation part representing *X*, *Y*, and *Z* locations and a rotational position for these angles. The source can be used to reference the detailed derivation. The quadrotor's body generates the motion Equations (20) and (21), which succinctly capture the kinematics of the quadrotor using the Newton-Euler equation.

$$m\ddot{x} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + RT_B + F_D \tag{20}$$

$$\dot{\omega} = \begin{bmatrix} \tau_{\emptyset} I - I_{xx} \\ \tau_{\emptyset} I - I_{yy} \\ \tau_{\emptyset} I - I_{zz} \end{bmatrix} - \begin{bmatrix} \dfrac{I_{yy} - I_{zz}}{I_{xx}} \omega_y \omega_z \\ \dfrac{I_{zz} - I_{xx}}{I_{yy}} \omega_x \omega_z \\ \dfrac{I_{xx} - I_{yy}}{I_{zz}} \omega_x \omega_y \end{bmatrix} \tag{21}$$

3.5. Fault classification using genetic-tuna swarm optimized deep neural network (GTSO-DNN)

3.5.1. Deep Neural Network (DNN)

A Deep Neural Network (DNN) may be created by fusing a number of Auto Encoders, each of which has two stages: encoding and decoding. This method gradually extracts features layer by layer using unsupervised learning, and then uses supervised learning to adjust the entire network parameters. Lower-level features are turned into higher-level features by a series of linear modifications applied over multiple layers, allowing for efficient feature extraction without the need for manual feature engineering. DNN updates the parameters of its network using back-propagation techniques. **Figure 5** illustrates the DNN network structure.

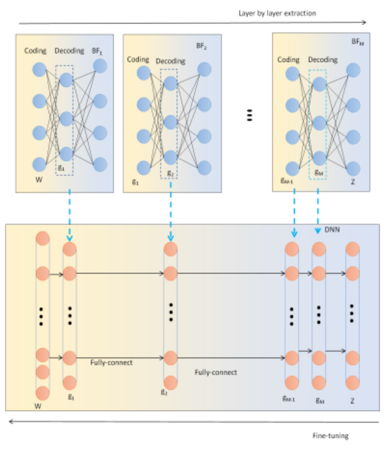

Figure 5. DNN network structure.

3.5.2. Genetic-Tuna swarm algorithm

This hybrid technique, which combines the exploratory prowess of Genetic Algorithms (GAs) with the adaptability of Tuna Swarm Algorithms (TSAs), presents intriguing potential for improved optimization in complex and dynamic problem domains. Together, they can overcome some of the drawbacks of each component, but success depends on careful parameter adjustment and problem-specific approaches.

Initialization: Use GA for initializing a population of people and TSA to initialize a swarm of tuna agents. Both random and predetermined solutions can be used to generate these populations.

Selection and crossover: Apply the GA population's individuals to the selecting and crossover (recombination) operations to provide offspring solutions.

Mutation: Applying mutation to a portion of the progeny from step 2 will provide diversity for the GA population.

Evaluation: Using the objective function, assess the fitness of the two populations GA population as well as the TSA swarm.

Communication: Make it possible for the GA group and the TSA swarm to communicate. The two populations can exchange knowledge on the top solutions or most exciting areas in the search arena.

Update and movement: Permit the TSA swarm to carry out updates and movement tasks in accordance with its regulations. Agents for the tuna can direct the GA population's migration by using information from that population.

Replacement: In order to promote diversity and maybe enhance convergence, change a section of the GA population in the best solutions discovered by the TSA swarm.

Termination: Carry out these procedures until convergence conditions are satisfied or for a predetermined number of iterations.

Deep Neural Networks (DNNs) by combining the global search capability of Genetic Algorithms (GAs) with the local exploration power of Tuna Swarm Algorithms (TSAs). DNN setups are modified over generations by GAs, whereas TSAs can spot minute changes. Information exchange between populations ensures thorough optimization, producing high-performing DNNs for various applications. Genetic-tuna swarm optimized deep neural network (GTSO-DNN) was shown in Algorithm 1.

Algorithm 1 GTSO-DNN

```
1:    population = initialize_population()
2:    def evaluate_dnn_performance(dnn_config)
3:    return performance_score
4:    for generation in range(max_generations)
5:    fitness_scores = []
6:    for dnn_config in population
7:    fitness_scores.append(evaluate_dnn_performance(dnn_config))
8:    selected_parents = ga_selection(population, fitness_scores)
9:    offspring = ga_crossover(selected_parents)
10:   offspring = ga_mutation(offspring)
11:   offspring_fitness_scores = []
12:   for dnn_config in offspring
13:   offspring_fitness_scores.append(evaluate_dnn_performance(dnn_config))
14:   population = ga_replace(population, offspring, offspring_fitness_scores)
15:   updated_population = []
16:   for dnn_config in population
17:   updated_dnn_config = tsa_movement(dnn_config)
18:   updated_population.append(updated_dnn_config)
19:   updated_fitness_scores = []
20:   for dnn_config in updated_population
21:   updated_fitness_scores.append(evaluate_dnn_performance(dnn_config))
22:   population = tsa_replace(population, updated_population, updated_fitness_scores)
23:   best_dnn_config = select_best_dnn(population, fitness_scores)
24:   deploy_optimized_dnn(best_dnn_config)
```

4. Experimental results

To set up the Genetic-Tuna Swarm Optimized Deep Neural Network (GTSO-DNN) experiment in Python, allocate a machine with at least 16GB of RAM, install required libraries (e.g., TensorFlow, scikit-learn), and create a hybrid algorithm combining Genetic Algorithms and Tuna Swarm Optimization for training deep neural networks efficiently.

Accuracy is defined as the indicator of a machine learning model's accuracy is the percentage of outcomes that were properly predicted. Better performance is indicated by higher values. Loss: A metric used to quantify the prediction error of a model while training and validation with the goal of reducing the difference among predicted and actual values. The training graph tracks these metrics over epochs, aiming for high accuracy and low loss (see **Figure 6**).

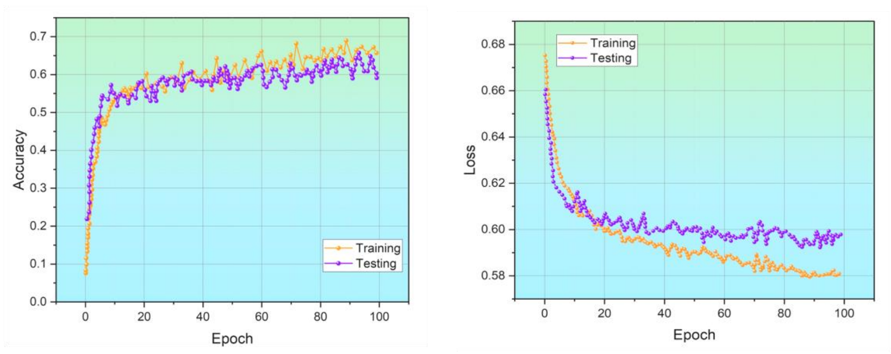

Figure 6. Outcome of accuracy and loss.

The effectiveness of the suggested and current methods was assessed in terms of accuracy, precision, recall, and F1 score. K-Nearest Neighbor (KNN), Decision Tree (DT), and Support Vector Machines (SVM)[27] were existing processes compared to the proposed method.

Accuracy gauges a categorization model's general accuracy. It was the proportion of correctly foreseen fault circumstances (including true positives and negatives) to all instances in the dataset. Accuracy in UAV fault diagnostics refers to the capability of the model to differentiate across normal and faulty situations. It was typically represented as a percentage using this Equation (22) to calculate the accuracy.

$$Accuracy = \frac{TP + TN}{Totalsamples} \tag{22}$$

- TP-It stands for perfect anticipated accuracy or accuracy above calibration.
- TN-It was the calibration level's negative predictive value.
- When the anticipated samples match the precise values and calibration-level at the identical level.
- FN-When the projected samples were at various levels than the precise values, which were in the calibration level.

Figure 7 and **Table 1** illustrate the accuracy value. Compared to existing methods KNN—88.72%, DT—92.81%, and SVM—93.55%, our proposed method was superior GTSO-DNN—98.51%. Compared to existing approaches, the suggested method GTSO-DNN showed significant improvements in fault diagnostics in unmanned aerial vehicle resilience.

Precision focuses on how well the model predicts favorable outcomes. The true positives (faults accurately anticipated) ratio to the total number of occurrences indicated as positive (including true positives and false positives) was known as the true positive to incorrect positive ratio. A model is more probable to be accurate when it predicts a fault if it has a high degree of precision. Precision is focused on avoiding erroneous alerts when identifying flaws in the setting of UAV resilience by using this equation (23) to calculate the precision.

$$Precision = \frac{Trueposivitepredic tion}{Truepositiveprediction + Falsenegativeprediction} \tag{23}$$

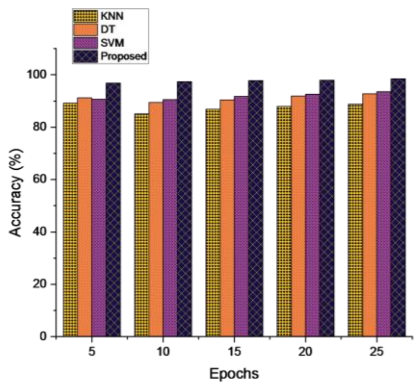

Figure 7. Outcome of accuracy.

Table 1. Values of accuracy.

Epochs	Accuracy (%)			
	KNN	**DT**	**SVM**	**GTSO-DNN [Proposed]**
5	89.19	91.25	90.7	96.8
10	85.08	89.50	90.54	97.4
15	86.89	90.45	91.84	97.8
20	87.96	91.9	92.6	98
25	88.72	92.81	93.55	98.51

Figure 8 and **Table 2** illustrate the accuracy value. Compared to existing methods KNN—98%, DT—91%, and SVM—94%, our proposed method was superior GTSO-DNN—98.7%. Compared to existing approaches, the suggested method GTSO-DNN showed significant improvements in fault diagnostics in unmanned aerial vehicle resilience.

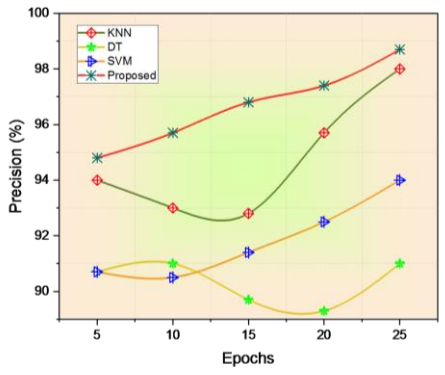

Figure 8. Outcome of precision.

Table 2. Values of precision.

Epochs	Precision (%)			
	KNN	**DT**	**SVM**	**GTSO-DNN [Proposed]**
5	94	90.7	90.7	94.8
10	93	91	90.5	95.7
15	92.8	89.7	91.4	96.8
20	95.7	89.3	92.5	97.4
25	98	91	94	98.7

The capacity of a model to correctly recognize positive cases (flaws) out of every actual positive instance is measured by recall. It was the proportion of real positives to all real positive events (including real positives and false negatives). The model effectively catches most of the actual defects when the recall is high. The recall was essential for preventing genuine failures requiring UAV resilience attention by using this equation (24) to calculate the recall.

$$Recall = \frac{Trueposivite prediction}{Truepositive prediction + Falsenegative prediction} \tag{24}$$

Figure 9 and **Table 3** illustrate the accuracy value. Compared to existing methods KNN—98%, DT—88.5%, and SVM—94%, our proposed method was superior GTSO-DNN-98.9%. In comparison to existing approaches, the suggested method GTSO-DNN showed significant improvements in fault diagnostics in unmanned aerial vehicle resilience.

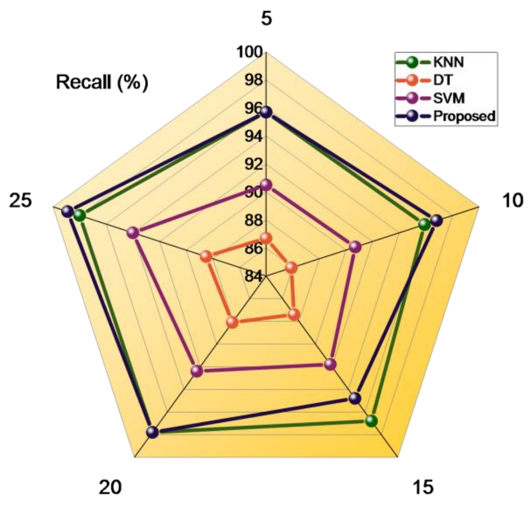

Figure 9. Outcome of recall.

Table 3. Values of recall.

Epochs	Recall (%)			
	KNN	DT	SVM	GTSO-DNN [Proposed]
5	95.7	86.7	90.5	95.7
10	95.9	85.9	90.7	96.8
15	96.8	87.4	91.8	94.8
20	97.8	88.1	92.4	97.8
25	98	88.5	94	98.9

The harmonic average of recall and precision is the F1 score. It evaluates a model's efficiency, considering incorrect positives and erroneous negatives. While there was a disparity between typical and fault circumstances, it was very helpful. Equation (5) was used to determine the F1 score. The F1 score in UAV failure diagnostics provides a broad overview of how well the model performs under both scenarios by using this equation (25) to calculate the F1 score.

$$F1 - score = \frac{2 * (precision * recall)}{precision + recall} \tag{25}$$

Figure 10 and **Table 4** illustrate the accuracy value. Compared to existing methods KNN—98%, DT—86%, and SVM—93%, our proposed method was superior GTSO-DNN-97.2%. In comparison to existing approaches, the suggested method GTSO-DNN showed significant improvements in fault diagnostics in unmanned aerial vehicle resilience.

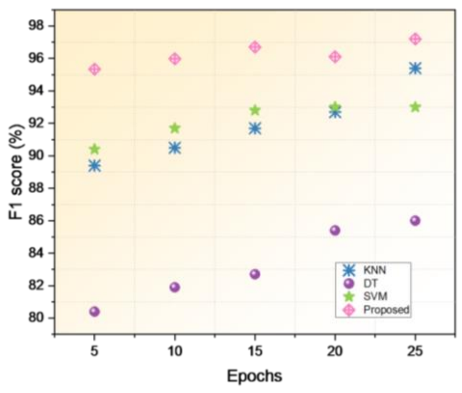

Figure 10. Outcome of F1 score.

Table 4. Values of F1 score.

Epochs	**F1 score (%)**			
	KNN	**DT**	**SVM**	**GTSO-DNN [Proposed]**
5	89.4	80.4	90.4	95.34
10	90.5	81.9	91.7	95.98
15	91.7	82.7	92.8	96.7
20	92.7	85.4	93	96.1
25	98	86	93	97.2

Discussion

High-dimensional feature spaces, frequent in intricate fault diagnostics scenarios, might cause K Nearest Neighbor's (KNN) performance to decline. The 'k' parameter must be carefully chosen for best results because it was dependent on outliers and noise, making it less resilient in processing noisy UAV sensor data. Particularly when dealing with sophisticated fault patterns in UAV systems, decision tree (DT) models were prone to overfitting. They need help to capture associations between variables that were not connected with the tree's structure, which limits their ability to diagnose faults accurately, and they can produce overly complicated trees that translate poorly to new data. Large datasets, typical in real-world UAV diagnostics, provide scaling challenges for Support Vector Machines (SVM). Choosing the proper kernel functions and tuning the hyperparameters can take time, which could result in subpar performance. SVMs have the potential to be computationally costly, which could make it difficult for UAV resilience scenarios to detect faults in real-time. Based on its distinctive combination of genetic algorithms and tuna swarm optimization, the Genetic-Tuna Swarm Optimized Deep Neural Network (GTSO-DNN) provides higher fault diagnostics capabilities in Unmanned Aerial Vehicle (UAV) resilience, enabling effective feature selection, robust model training, and accurate fault detection, thus improving the overall reliability and performance of UAVs in difficult environments.

5. Conclusion

In the rapidly advancing landscape of Unmanned Aerial Vehicles (UAVs), their indispensable role in diverse operations underscores the need for robust resilience. This work introduces a comprehensive methodology addressing Fault Diagnostics in UAV Resilience. A novel method for fault classifier using Genetic-Tuna Swarm Optimized Deep Neural Network (GTSODNN) was applied for anomaly identification. Employing simulated propeller damage data, Min-Max normalization for preprocessing, and Principal Component Analysis (PCA) for feature extraction, the approach integrates dynamic and propeller models using a Gated Recurrent Unit (GRU) network. Experimental findings demonstrate the GTSODNN's superior performance compared to existing methods in terms of accuracy-98.51%, precision-98.7%, recall-98.9%, and

F1 score-97.2%. This GTSODNN model proficiently detects and categorizes anomalies, elevating UAV resilience. This research holds significant potential for real-time safety enhancements, showcasing a robust methodology for bolstering UAV fault diagnosis capabilities. The intricacy of real-world settings may be absent in this study because it mostly uses simulated data. Minimal training data under uncommon fault conditions may also impact the GTSODNN's performance. Future studies could use real-flight data to strengthen model robustness and circumvent restrictions. Problem-solving diagnosis could be improved across multiple UAV platforms and unanticipated issues by combining multi-sensor data and investigating transfer learning algorithms.

Author contributions

Conceptualization, PS and VDJ; methodology, DRD; software, SG; validation, MSB, PS and HP; formal analysis, HP; investigation, PS; resources, PS; data curation, HP; writing—original draft preparation, PS; writing—review and editing, DRD; visualization, VDJ; supervision, PS; project administration, AAM; funding acquisition, AAM. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

I extend my heartfelt gratitude to Rathinam Technical Campus, Coimbatore, Tamilnadu, India, for their invaluable support and resources, which significantly contributed to the success of this research. The academic environment and facilities provided by the institution played a crucial role in the completion of this study. I am deeply thankful for the encouragement and guidance received from the esteemed faculty members, whose expertise and insights enriched the quality of my research. Rathinam Technical Campus has been an indispensable part of my academic journey, and I am profoundly appreciative of their continued commitment to fostering research and education.

Conflict of interest

The authors declare no conflict of interest.

Abbreviations

MAVs	Micro Aerial Vehicles
PCA	Principal Component Analysis
UAV	Unmanned aerial vehicles
DNN	Deep Neural Network
HF	Hampel Filter
MAVFI	Micro Aerial Vehicle Fault Isolation
GTSO-DNN	Genetic-Tuna Swarm Optimized Deep Neural Network
RIS	reconfigurable intelligent surfaces
FW-UAVs	Fixed-Wing Unmanned Aerial Vehicles
GRU	Gated Recurrent Unit
HDDAN	Hybrid Deep Domain Adaptation Networks
RNN	Recurrent Neural Network
RPM	Revolution Per Minute
LSTM	Long Short-Term Memory

References

1. Yaqot M, Menezes BC. Unmanned Aerial Vehicle (UAV) in Precision Agriculture: Business Information Technology Towards Farming as a Service. In: Proceedings of the 2021 1st International Conference on Emerging

Smart Technologies and Applications (eSmarTA).
2. Iqbal U, Barthelemy J, Perez P. Emerging role of unmanned aerial vehicles (UAVs) for disaster management applications. In: Nanotechnology-Based Smart Remote Sensing Networks for Disaster Prevention. Elsevier; 2022.
3. Madni AM, Erwin D, Sievers M. Constructing Models for Systems Resilience: Challenges, Concepts, and Formal Methods. Systems. 2020; 8(1): 3. doi: 10.3390/systems8010003
4. Chen M, Wang H, Chang CY, et al. SIDR: A Swarm Intelligence-Based Damage-Resilient Mechanism for UAV Swarm Networks. IEEE Access. 2020; 8: 77089-77105. doi: 10.1109/access.2020.2989614
5. Debele Y, Shi HY, Wondosen A, et al. Deep Learning-Based Robust Actuator Fault Detection and Isolation Scheme for Highly Redundant Multirotor UAVs. Drones. 2023; 7(7): 437. doi: 10.3390/drones7070437
6. Wang G, Gu C, Li J, et al. Heterogeneous Flight Management System (FMS) Design for Unmanned Aerial Vehicles (UAVs): Current Stages, Challenges, and Opportunities. Drones. 2023; 7(6): 380. doi: 10.3390/drones7060380
7. Bondyra A, Kołodziejczak M, Kulikowski R, et al. An Acoustic Fault Detection and Isolation System for Multirotor UAV. Energies. 2022; 15(11): 3955. doi: 10.3390/en15113955
8. Hsiao YS, Wan Z, Jia T, et al. April. Mavfi: An end-to-end fault analysis framework with anomaly detection and recovery for micro aerial vehicles. In: 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE; 2023.
9. Sadhu V, Zonouz S, Pompili D. On-board Deep-learning-based Unmanned Aerial Vehicle Fault Cause Detection and Identification. In: Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA).
10. Zhang Y, Li S, He Q, et al. An Intelligent Fault Detection Framework for FW-UAV Based on Hybrid Deep Domain Adaptation Networks and the Hampel Filter. International Journal of Intelligent Systems. 2023; 2023: 1-19. doi: 10.1155/2023/6608967
11. Corbetta M, Jarvis KJ, Schuet S. Hybrid Modeling of Unmanned Aerial Vehicle Electric Powertrain for Fault Detection and Diagnostics. In: AIAA AVIATION 2023 Forum. AIAA; 2023.
12. Segovia Ramírez I, Das B, García Márquez FP. Fault detection and diagnosis in photovoltaic panels by radiometric sensors embedded in unmanned aerial vehicles. Progress in Photovoltaics: Research and Applications. 2021; 30(3): 240-256. doi: 10.1002/pip.3479
13. Hossain NUI, Sakib N, Govindan K. Assessing the performance of unmanned aerial vehicle for logistics and transportation leveraging the Bayesian network approach. Expert Systems with Applications. 2022; 209: 118301. doi: 10.1016/j.eswa.2022.118301
14. El Jery A, P S, Mahmood Salman H, et al. Comparison of different approaches for numerical modeling of nanofluid subcooled flow boiling and proposing predictive models using artificial neural network. Progress in Nuclear Energy. 2023; 156: 104540. doi: 10.1016/j.pnucene.2022.104540
15. Al-Abiad MS, Javad-Kalbasi M, Valaee S. Effectiveness of Reconfigurable Intelligent Surfaces to Enhance Connectivity in UAV Networks. ArXiv. 2023; arXiv:2308.10788.
16. Perry BJ, Guo Y, Atadero R, et al. Streamlined bridge inspection system utilizing unmanned aerial vehicles (UAVs) and machine learning. Measurement. 2020; 164: 108048. doi: 10.1016/j.measurement.2020.108048
17. Eichleay M, Evens E, Stankevitz K, et al. Using the Unmanned Aerial Vehicle Delivery Decision Tool to Consider Transporting Medical Supplies via Drone. Global Health: Science and Practice. 2019; 7(4): 500-506. doi: 10.9745/ghsp-d-19-00119
18. Ejaz W, Azam MA, Saadat S, et al. Unmanned Aerial Vehicles enabled IoT Platform for Disaster Management. Energies. 2019; 12(14): 2706. doi: 10.3390/en12142706
19. Bronz M, Baskaya E, Delahaye D, et al. Real-time Fault Detection on Small Fixed-Wing UAVs using Machine Learning. In: Proceedings of the 2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC).
20. Phadke A, Medrano FA. Towards Resilient UAV Swarms—A Breakdown of Resiliency Requirements in UAV Swarms. Drones. 2022; 6(11): 340. doi: 10.3390/drones6110340
21. Neves FS, Claro RM, Pinto AM. End-to-End Detection of a Landing Platform for Offshore UAVs Based on a Multimodal Early Fusion Approach. Sensors. 2023; 23(5): 2434. doi: 10.3390/s23052434
22. Phadke A, Medrano FA, Chu T, et al. Drone2Drone (D2D): A Search and Rescue Framework Module for Finding Lost UAV Swarm Agents. In: Proceedings of the 2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE).
23. Zhang P, Wu T, Cao R, et al. UAV Swarm Resilience Assessment Considering Load Balancing. Frontiers in Physics. 2022; 10. doi: 10.3389/fphy.2022.821321
24. El Jery A, P S, Mahmood Salman H, et al. Comparison of different approaches for numerical modeling of nanofluid subcooled flow boiling and proposing predictive models using artificial neural network. Progress in Nuclear Energy. 2023; 156: 104540. doi: 10.1016/j.pnucene.2022.104540
25. Jerome Vasanth J, Naveen Venkatesh S, Sugumaran V, et al. Enhancing Photovoltaic Module Fault Diagnosis with Unmanned Aerial Vehicles and Deep Learning-Based Image Analysis. International Journal of Photoenergy. 2023; 2023: 1-17. doi: 10.1155/2023/8665729
26. Tong JJ, Zhang W, Liao F, et al. Machine Learning for UAV Propeller Fault Detection based on a Hybrid Data Generation Model. arXiv. 2023; arXiv:2302.01556.

27. Altinors A, Yol F, Yaman O. A sound based method for fault detection with statistical feature extraction in UAV motors. Applied Acoustics. 2021; 183: 108325. doi: 10.1016/j.apacoust.2021.108325