Building on Prior Lightweight CNN Model Combined with LSTM-AM Framework to Guide Fault Detection in Fixed-wing UAVs

Aakash Kumar[1,2], Shifeng Wang[*1,2], Ali Muhammad Shaikh[3], Hazrat Bilal[3], Bo Lu[2,4], Shigeng Song[4]

Email {aakash, sf.wang}@cust.edu.cn, {alims, hbilal}@mail.ustc.edu.cn, {bo.lu, shigeng.song}@uws.ac.uk

[1]School of Optoelectronic Engineering, Changchun University of Science and Technology, Changchun 130013, People's Republic of China

[2]Zhongshan Institute of Changchun University of Science and Technology, Zhongshan 528437, People's Republic of China

[3]Department of Automation, University of Science and Technology of China, Hefei 230026, People's Republic of China

[4]School of Computing, Engineering and Physical Sciences, University of the West of Scotland, Paisley Campus, Scotland

Abstract

Fixed-wing UAVs (FW-UAVs) are empowered to handle diverse civilian and military missions, but sensor failure scenarios are constantly rising. Rapid advancement in deep learning methods currently proposes state-of-the-art solutions for fault detection of UAVs. However, most recent deep learning-based detection models suffer from model size, high computational complexity, and high-power consumption, which are challenging for small-sized FW-UAVs with limited battery backup and computational power. Therefore, to overcome these problems, this article introduces a lightweight CNN model built on prior work combined with the LSTM-AM framework to obtain accurate fault detection of FW-UAVs with low power consumption and fast computations. First, lightweight CNN architecture aims to minimize computational complexity while maintaining high accuracy in fault detection. The LSTM model merged with Attention Mechanism (AM), allows the architecture to obtain temporal dependencies and concentrate on essential features for enhanced fault detection accuracy. The combined version of lightweight CNN, LSTM, and AM commits to more reliable and efficient fault detection in FW-UAV applications, improving UAV drones' overall performance and safety.

Keywords: Lightweight CNN; FW-UAVs; LSTM; Attention Mechanism; Fault Detection

1. Introduction

In recent years, there has been a substantial advancement in unmanned aerial vehicle (UAV) technology. The UAV is so enriching that it empowers the UAV to handle complex missions in many fields of both military and civilian, such as aerial photography [1], emergency rescue [2], and traffic monitoring [3]. However, UAVs have drawbacks such as size restrictions, high accident rates, and development costs. To overcome these issues, the research community has shown a rising interest in the real-time fault detection approach. Besides, in this process, while detecting the impending faults, the airborne required maintenance and recovery measures are scheduled to prevent disasters during flight. In addition, within a wide range of UAVs, fixed-wing unmanned aerial vehicles (FW-UAVs) are a category of aircraft capable of consuming low energy while flying which rely on passive lift generated through the wings. With this ability, they can execute tough tasks requiring longer endurance and high-altitude flying. Conversely, (FW-UAVs) endure a high rate of failure because of complex systems in quite rough service environments and have a deficit of real-time decision-making abilities, which can lead to mission failure, economic losses, and possible catastrophic casualties.

Here are some potential uses, taking into account various climatic conditions and mission profiles: 1) environmental monitoring: In order to monitor the environment, such as by detecting forest fires, evaluating air quality, or keeping an eye on wildlife populations, in such scenario, FW-UAVs are first in selection. 2) precision agriculture: FW-UAVs can be utilized in precision agriculture to track crop health, pinpoint disease outbreaks, and maximize resource utilization, which guarantees that reliable data is captured for crop analysis. 3) Security and Surveillance: FW-UAVs are important in security and surveillance applications, for instance, border patrol, disaster response, and infrastructure monitoring. With these applications and various climatic conditions, faults can happen to any UAV. Therefore, it is necessary to build a fault detection system considering UAVs' size, computational power, and battery backup, improving situational awareness and decreasing the risk of compromised UAV functionality in serious scenarios. Fault

detection is crucial in guaranteeing the safe and dependable operation of unmanned aerial vehicles (FW-UAVs). By adopting Maintenance 4.0 and Maintenance 5.0 concepts [4, 5], FW-UAV fault detection, which needs worker skills and intelligent technology, may be considerably enhanced.

The safety of UAVs has attracted many researchers. For this purpose, many options are considered, such as hardware redundancy (HR), which is the preferred preference for enhancing aircraft reliability with the help of integrating redundant hardware for critical elements, namely the control system, to prevent system-level failure. Hardware redundancy is quite famous in aviation applications because of its high reliability. However, due to payload, size, and manufacturing costs, HR is not a practical application for UAVs. Hence, for the aim of fault detection, analytical redundancy was proposed by HR in UAVs. Fourlas et al. [6] collected fault data from onboard sensor data using structured algorithms that combine model-based and data-driven techniques. Besides, numerous studies on fault detection have been conducted, mainly divided into sensor-based, model-based, and data-driven approaches. The approaches that rely on the model usually utilize the physical model considering the state transition function. With the help of the Kalman filter (KF) or the variants, its KF builds an observer from it. So, a fault alarm is released if the observer's output is not the same as the measured value. Guo et al. [7] utilized a variant of KF along with a UAV kinematics model to discern sensor analytical redundancy. An unknown input observer was proposed in [8] to sort out the sensor fault diagnosis of a quadrotor UAV's attitude control system. Another extended variant of KF was introduced in [9] to help generate residual signals for fault detection of UAVs. However, model-based approaches have a potential drawback in the necessity of accurate physical models, which are demanding because of FW-UAVs' non-linear, dynamic, and multivariate nature.

On the other hand, an emerging technology, "Internet of Things (IoTs)," has many applications for UAVs with wireless technology. UAVs can be deployed and operated more quickly in general since there are a lot of readily available guidance, navigation, and control systems. Typically, these solutions facilitate close-quarters line-of-sight activities. However, affordable wireless access is necessary for managing a swarm of UAVs for surveillance in various terrains. Therefore, Zeeshan et al. [10] tackled the problem of UAV swarm communications by introducing a framework that provides an open-interface networking and communication solution for outreach and surveillance operations in urban settings. Further, the capacity to handle massive volumes of data in real-time and keep up with recent developments in user traffic requirements are still challenges for pure Software Defined Networks (SDNs). On the other hand, real-time and timely responses may now be obtained with fog computing. Hence, Adnan et al. [11] suggested an SDN-enabled fog computing architecture with low power and high performance. Recently, A low-power, single-hop, long-range wireless technology called LoRa has been envisaged for Internet of Things (IoT) applications using battery-operated nodes. However, the performance of pure Aloha in LoRaWAN is degraded by an increase in the number of end devices and different throughput requirements. Considering these restrictions, Zulfiqar et al. [12] did thorough, comprehensive simulations. They assessed the performance of slotted Aloha in LoRaWAN by using packet error rate (PER), delay, and energy usage of components considering different payload sizes.

Moreover, connecting many Internet of Things (IoT) devices to the cloud using edge devices is known as edge computing and is a relatively new trend. Though edge devices usually have more computational power than IoT devices, ensuring effective resource utilization is the leading research problem, especially for real-time delay-sensitive applications like military and battlefield. In order to enhance Edge-of-Things' (EoT) overall performance, Adnan et al. [13] suggested a dynamic policy-based computing method by examining heterogeneous energy usage.

Nevertheless, the performance of the current DL approaches for fault detection relied on adequate failure data, high-quality labels, and rich fault types [13, 14]. These scenarios are challenging to persuade because of the complex procedures; hence, failure data of FW-UAVs is quite rare in contrast to standard data. FW-UAVs mainly comprise very complex systems and subsystems, making it hard to collect every category of faults in advance; also, flight experiments for real FW-UAVs are dangerous and quite expensive. Hence, we need to improve the DL models' performance further. More recently, the few-shot learning (FSL) approach, which is capable of training DL models with fewer training samples, has revealed contemporary performance in industrial applications [15, 16] along with language modeling [17, 18]. There are three main categories of UAVs, namely flapping wings, rotary wings, and fixed wings. Flapping-wing UAVs aim to duplicate the way that insects and birds fly. They are considered through restricted payload abilities and low endurance, while Fixed-wings UAVs are often utilized for long-range missions and high

altitudes. Finally, rotary wings are called vertical take-off and landing (VTOL) UAVs. This sort of UAV is typically engaged for tasks that comprise hovering. This paper introduces a new scheme for fault detection in fixed-wing UAVs based on combining three different AI algorithms. First, to make the trained CNNs that can be used in systems with insufficient storage and low computational processors, the filter-level pruning algorithms [19] are applied to save computational complexity and space. In addition, CNNs help in obtaining inter-channel spatial information. Second, as we know, CNNs are not sensitive to time series data. However, LSTM can help fuse time features of time series data. Therefore, we added LSTM layers to assist in obtaining inter- and intra-channel temporal information. Finally, to further enhance the accuracy of the model, an AM block is added to help the model accurately detect faults.

2. Related Work

The dominance of UAVs makes them quite valuable for substituting human tasks that probably be problematic and challenging. However, the criteria for performance and reliability are also elevating. We can say that fault detection is inevitable despite fast-growing technological advances. Because of the wide range of integrated subsystems, they are most likely to fail in uncertain scenarios and events that can happen in their operating environment [20]. A crucial feature that stages further conflicts and concerns in fault diagnosis is that every flight and mission assignment is incorporated into the vehicle's embedded control systems. However, any intervention through the ground operator is often confined and perhaps lacking or delayed. Consequently, it is pivotal for the UAV to trace its activities to tackle any faults before catastrophic outcomes.

This article considers three main approaches in the literature to highlight fault detection for UAVs: signal-based, model-based, and data-driven. Surprisingly, vibration signals are not often considered detection features in signal-based approaches. Practically any UAV has embedded accelerometers built-in in its flight controller; hence, the vibration signal data is present without requiring any extra sensor. In recent years, the usage of acceleration signals to monitor the health of rotating motors [21] and machine elements, e.g., bearings, clutches, and milling cutters, has been widely highlighted in the literature [22–24]. Together with fault detection methods based on AI using electric readings [25–27]. Evidently, in the scenario of UAVs, the accelerometers are employed on the frame rather than on the single actuator as in big, industrial rotating machinery; this increases the difficulty of fault detection since several actuators' vibrations combine with sensor noise and other disturbances (such maneuvering's aerodynamic forces) to produce acceleration readings that are simultaneously affected.

On the other hand, model-based methods, for instance, "Thau Observers" [28], Linear Parameter Varying (LPV) proportional-integral observers [29], and banks of non-linear disturbance observers [30] are solidly supported by the scholarly literature. Moreover, model-based techniques are not only limited to fault detection, but it is also possible to isolate and estimate faults for many fault classes, given certain conditions. The unavailability of numerous application factors, such as inertia, lift and drag coefficients, and friction, makes model-based techniques challenging and requires a substantial modeling effort. When employing model-based techniques, the unpredictability in the brushed motors' performance entirely masks any perceived loss of efficacy caused by the chipping of the blade [31]. Conversely, signal-based techniques don't need assumptions about the faults or complicated system models. Assuming that an AI-based fault detection system, consisting of signal-based techniques for feature generation and a classification tool, may be implemented if a sufficient quantity of data is available, including all kinds of faults of practical importance [21]. Numerous AI-based techniques for UAVs have been proposed recently. Ai et al. [32] introduced a random forest to classify sensor faults, and Liang et al. [33] gave various variants of Kernel Principal Component Analysis (KPCA) to capture sensor and actuator faults. In contrast, Park et al. [34] considered applying deep neural networks to isolate and detect actuator failure.

Lastly, data-driven approaches, particularly deep-learning-based fault detection systems, are the subject of another field of fault detection research [35, 36]. They help to reduce the dependency on physical models and expert expertise by enabling automatic feature extraction capabilities to discover fault information from raw signal data. By offering a variety of well-known deep learning models, such as deep belief networks (DBNs), convolutional neural networks (CNNs), and recurrent neural networks, among others, intelligent UAV fault identification based on flight data is thus achieved. Chen et al.[37] used a model that outclassed classical shallow neural networks using a DBN-based data-driven approach for early sensor fault identification of rotorcraft. Guo et al.[38] suggested using CNN to extract

features from time-frequency images created by converting the residual signals of UAV sensor failures, with encouraging outcomes. In contrast, Cui et al.[39] employed the LSTM model for intelligent fault identification and FW-UAV health management, which worked perfectly on real-time prediction. In comparison, Ahmad et al. [40] introduced an embedded framework built on the LSTM model that identifies fault detection, classification, and prediction for FW-UAVs. References [41, 42] provide comparable work based on DL models.

Nonetheless, adequate fault data, excellent labels, and various fault types are necessary to create the most current deep learning-based UAV fault identification models [43, 44], which are problematic to meet because of the intricate requirements for FW-UAVs. The FW-UAV fault data is relatively rare compared to standard data, as these UAVs often operate in normal conditions. Since FW-UAVs are complicated systems with many components, it is not feasible to anticipate every type of fault in advance. Further, flight experiments with real FW-UAVs are dangerous and expensive, and fault data from the simulation framework are not very close to real-time situations. Therefore, we need to improve the deep learning models' performance in this case since the fault identification of FW-UAVs is a typical small data problem. In addition, the large model size, high computational complexity, and high-power consumption of deep learning-based detection models pose significant challenges for small-sized FW-UAVs with constrained processing capacity and battery backup.

A substantial amount of research has been done to tackle the abovementioned problems by compressing the deep learning models, such as model pruning, which has shown advanced state-of-the-art performance in fault detection [45], image classification [46], and image segmentation [47]. Notably, the term "prune" can be used as a lightweight model with fewer parameters than the original unpruned model; employing that lightweight model can save computations and minimize high power usage; such advantage encouraged us to explore lightweight model-based fault detection solutions to small FW-UAVs, which always has a limited battery backup, low computation power, and limited storage for unpruned model implementation. Thus, these problems limit the required tasks for UAVs in any field. Therefore, in this article, we propose a lightweight CNN model combined with LSTM and AM for fault detection on small FW-UAV applications and enabling more reliable real-time fault detection scenarios more efficiently. Finally, to our knowledge, such models have never been applied to recognize/detect the faults in UAVs based on sensor data with information on wind speed. This article introduces a new deep-learning model to identify and detect faults in fixed-wing UAVs.

3. Understanding Faults in Fixed-Wing UAVs:

As Cai et al. [48] mentioned, based on the lift generation, there are two significant kinds of UAVs available: Fixed-Wing UAVs that can fly applying generation of lift through their wing's movement within the air, whereas Rotary-Wing UAVs produce lift by pushing air toward down with multiple or single rotor blades. Therefore, this article only focused on Fixed-Wing drones, complex, intelligent machines built of structures, sensors, subsystems, and actuators. Among these, the actuators are very important for confirming orientation, angular velocity, and aerodynamics; however, these parameters are also likely to collapse in extreme environments.

3.1. Analysis of Actuator Faults:

The faults in actuators generally faced in UAVs are: 1. Lock in place, 2. Effectiveness partial loss, 3. Float, 4. Hard-over failure [49]. In faults like lock in place, the control surface remains fixed or halts at a position that might or might not be zero or neutral. In this state, the other working actuators not only recompense for the absence of the required control load of the faulty actuator but also abandon the unrequired control load generated if the actuator halts at any place other than zero. Physical harm can produce effectiveness partial loss due to physical damage to the control loader.

On the other hand, the fault-like float is made when the actuator floats in the airflow and cannot generate any control moment. Finally, in a fault-like hand-over, the actuator jams at one of the position bounds. The actuator control faults can be done for each control through the mathematical equation given below:

$$u_{\text{app}} = cu_{\text{cal}} + k \tag{1}$$

Here, the terms c and k are referred to as constants and used for a specific control conFigureuration; however, constants can be changed or settled to other constants if faults occur in the control system. Further, u_{cal} shows the control asked by the controller and u_{app} denotes the control that is essentially given due to the saturation limit. With no failure, c and k are initialized to 1 and 0, respectively, and the u_{app} and u_{cal} are equal to each other. To freeze the actuator can be done by setting c and k to 0 and constant value, respectively, in which the surface of control is frozen. Therefore, the model suffers from hard-over and lock-in-place faults. Further, this model can also suffer effectiveness loss in which the control surface is lost or damaged, such as only half of the part is left. Suppose that the effectiveness of control is decreased by half; then c can be at 0.5, and k stays at 0 value. If c approaches zero, the float faults can also be encountered while k remains at the 0 value. Considering the control surface as a vector, therefore, the model can be considered as follows:

$$\begin{bmatrix} u_{\text{app1}} \\ u_{\text{app2}} \end{bmatrix} = \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix} \begin{bmatrix} u_{\text{cal1}} \\ u_{\text{cal2}} \end{bmatrix} + \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} \tag{2}$$

Two control surfaces are described, where script 1 shows the right wing and script 2 shows the left wing of the UAVs. Though deep learning models are famous for extracting features, their data-demanding hunger poses considerable challenges for fault diagnosis of UAVs with insufficient data, and few researchers have produced an efficient solution to address this issue. In this article, we study the fault diagnosis of Fixed-Wing UAVs as a detection and classification problem. We present the building on prior research, a lightweight CNN-LSTM based on the AM algorithm. The dataset applied is an open-source real flight dataset produced from a small Fixed-Wing UAV as an input variable, and the lightweight CNN model is used to extract dynamic variations in data. In this lightweight CNN-LSTM based on the AM algorithm, a high-dimensional feature is built as input of LSTM, and the feature's internal dynamic changes are learned. AM helps to allocate different weights to the implicit conditions of LSTM by mapping parameters and weight learning, and it can power the influence of robust information. Further, the Exponential Weighted Moving Average (EWMA) method uses a cumulative sum approach to recognize the operation state and detect time delay in UAVs.

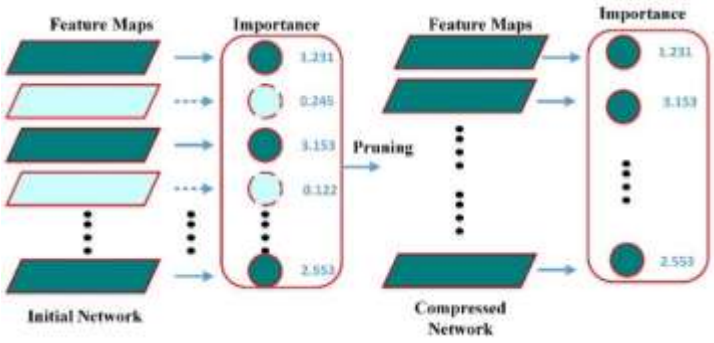

Figure 1. The architecture of the unpruned model (left) and pruned model (right)

In the initial step, the data on the operational status for different flights and targets is collected from the flight test system. The unprocessed data is generally multivariate time series flight data with features, for instance, longitude, control commands, and flight speed. A correlation analysis method is applied, and the features with a more significant correlation with the faults are selected as input features. Further, signal variables can be measured with different sensors, giving different ranges and units. Additionally, many variables possess negative values, generally showing the UAV's direction while flying [50]. We used the Min-Max normalization function with a range of (-1, 1) to rectify these variables without damaging their actual value. In this fashion, we get normalized variables for each fault. Finally, our introduced model has lightweight CNN-LSTM, which receives a sample as input. With the normalized variables, our model can generate the same and distinct variables in a similar portion. Before going into further insights into the proposed model, we create a compressive one-tabular overview of the experimentation process, including the motivation behind the dataset used and training methodology, as shown in Table 1.

Dataset Used	Training Methodology and Hardware	Motivation
FW-UAV Fault Dataset	Flight test system	The public real flight dataset was obtained from a small FW-UAV that met the requirements of the flight test system. Flight records from several days in July 2020 were recorded, and various faults of loss of control effectiveness were manually injected into the small FW-UAV by altering the GCS signal and adjusting the controller onboard. As a result, we have collected different datasets of different days with different wind speeds.
Lightweight CNN Model	Efficient Computation, Small Model Size, Low Power Consumption	Employed pruned CNN architecture for efficient computation and low power consumption on FW-UAV systems with limited battery backup and computational resources. The design of these models aims to retain appropriate performance with fewer parameters and less complexity. Because of the chosen lightweight CNN model, the fault detection system can be installed on FW-UAV devices with limited resources without sacrificing operational effectiveness or real-time speed.
Exponential Weighted Moving Average	Cumulative Sum Approach	The EWMA is integrated with LSTM by feeding the estimated baseline or predicted behavior into the LSTM model as an additional input. At each time step, the EWMA method's baseline values are concatenated with the original data or utilized as another feature input. This integration allows the LSTM model to learn from raw data and the smoothed baseline, improving its fault detection capability.
LSTM-based Temporal Analysis	Sequence Modeling	FW-UAV fault dataset temporal dependencies are modeled using LSTM. In time-series data, fault detection relies heavily on the sequential patterns and long-term relationships that this model captures. The model can efficiently assess the temporal dynamics of FW-UAV faults because of LSTM's capacity to maintain the memory of previous states.
Attention Mechanism	Improved Feature Selection	We employed an Attention Mechanism in the architecture to improve feature selection during fault detection. The attention mechanism permits the model to allocate different weights to different segments of the input sequence, attentive to the most informative bits. This enhances the model's capacity to identify and focus on the most essential characteristics, increasing the model's detection accuracy and decision-making interpretability.

Table 1: Comprehensive Overview

4. Lightweight CNN Versus CNN-original

4.1. Importance of Pruning

Convolutional neural networks (CNNs) are sophisticated deep learning algorithms broadly employed in several real-world AI applications, such as image segmentation, fault detection, and personal chatbots. However, the advanced CNNs models such as VGG16 and GoogLeNet are not memory-friendly because of billions of trainable weights, and these models have high requirements for FLOPs. Further, these complex architectures have high inference costs, mainly when applied with mobile devices or embedded sensors where battery usage is high, and processing capacity may be constrained. As a result, it is critical to minimize the size of deep CNN architectures, which have low power consumption and computational cost yet good accuracy in real-world scenarios. Model compression has received significant interest in both industry and academics.

4.2. Pruning Strategy

Correlation-based filter pruning (CFP) is built and inserted between the network's two subsequent convolutional layers to identify the irrelevant filters from a pre-trained network. To determine the correlation score between the first two successive feature maps of $i-th$ convolutional layer, we put these two feature maps into the correlation weight module. One of these feature maps should be eliminated because the higher correlation score indicates information duplications between them. After that, we can prune all their incoming and outgoing connections to eliminate the feature maps that include redundant data. Following this, all of the $i-th$ layer's unnecessary filters are removed, along with feature maps used as input for the subsequent layer and the matching channels for each filter in that layer.

As models continue to advance, broader and deeper models are being built. Generally, larger models do a great job in prediction tasks but require massive hardware to support. For instance, a model network called VGG16 has more than 138M parameter weights, which requires approximately 15.3B FLOPs during inference [51, 52]. Setting up such huge models for real-time fault detection data requires high hardware costs for fixed-wing UAVs, which already have a high price. Thus, it is essential to research how to reduce model size with negligible change in final accuracy to implement them in embedded devices such as UAVs. The significant problems because available CNNs cannot be enforced on embedded devices are the large model size, high run-time memory, and enormous computational complexity.

Therefore, to overcome this problem, the model pruning algorithm can be used in different scenarios, such as filter-level pruning, channel-level pruning, and weight-level pruning. Kumar et al. [19] presented a combination of L1-norm and Capped- L1-norm to measure the importance of the feature maps to prune insignificant feature maps, as illustrated in Figure. 1. Kumar et al. [53] show that Pearson correlation can be used to measure the vital information between two consecutive feature maps, with high correlation means duplication of data, In the end, the model size will be decreased with low computational cost and run-time memory by using just one feature map for additional training. This experiment uses the fine-tuned thin model with similar accuracy to the original model, which permits our network to be implemented on UAVs, which usually have low memory and require low computational complexity. The layer-wise comparison example of the unpruned model (VGG16 model trained on the CIFAR10 dataset) with the pruned model is given in Table 2. The degree to which the model has been compressed or simplified through pruning is shown by the percentage of pruned parameters and FLOPs. Greater percentages signify a more substantial decrease in parameters and calculations. Furthermore, the tradeoffs between the un-pruned and pruned models revolve around size of model, computational complexity, performance, and generalization. Reducing the size of the model and increasing computational efficiency are two advantages of pruning, however accuracy may be slightly compromised.

Model Layer	**Total Parameters**	**Total FLOPs**	**Pruned Parameters (%)**	**Pruned FLOPs (%)**
Conv 1.1	2K	0.1B	65.6	58.6
Conv 1.2	37K	1.85B	83.5	46.9
Conv 2.1	74K	0.9B	92.7	62.9
Conv 2.2	148K	1.85B	65.5	65.2
Conv 3.1	295K	0.9B	95.7	40.5
Conv 3.2	590K	1.85B	89.6	36.9
Conv 3.3	590K	1.85B	96.5	57.9
Conv 4.1	1M	0.9B	89.9	47.2
Conv 4.2	2M	1.85B	94.8	64.2
Conv 4.3	2M	1.85B	94.8	52.4
Conv 5.1	2M	462.5M	98.6	55.6
Conv 5.2	2M	462.5M	95.6	47.9
Conv 5.3	2M	462.5M	97.7	70.1
FC 1	103M	103M	96.9	60.8
FC 2	17M	17M	85.8	44.9
FC 3	4M	4M	90.5	27.2
Total	138M	15.3B	89.7	52.4

Table2. Layer-wise Model Comparison

Further, the spatial features are obtained using lightweight CNN by source data, and finally, processed data are used as input data into the AM-based LSTM model for time series prediction. We parameterized the training set as $T = \{(x_i, y_i) \mid i = 1, \cdots, m\}$, where x_i shows input data samples, $x_i \in R^p$, in which p shows the total number of features. Further, $y_i \in \{0,1\}$ represents the label of the $i-th$ sample data. The sample data is fed into a lightweight CNN model to generate high-level features from the original features.

5. LSTM (Long- and Short-Term Memory Model)

Zhao et al. [54] mentioned that CNN models are not subtle to the time feature of time series data; however, LSTM can better capture time features of the status of different parts. Therefore, the dataset's spatial and temporal properties can be extracted using LSTM and lightweight CNN. The light-weighed CNN output is considered as LSTM input.

LSTM central diagram is shown in Figure. 2, which has an input gate, output gate, and forget gate. The abandoned information is controlled by forget gate f_t, which is shown as follows:

$$f_t = \sigma(\mathrm{W_f} \cdot [\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_f}) \tag{3}$$

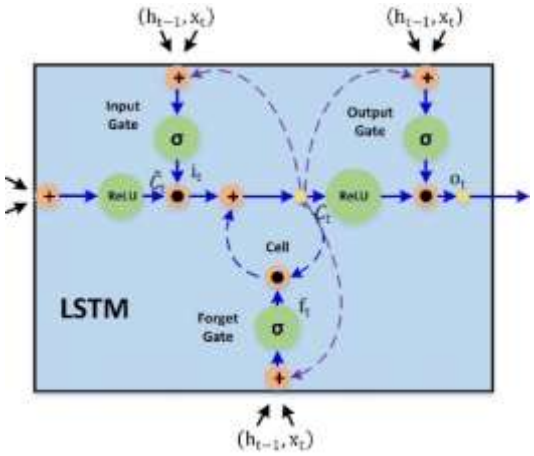

Figure 2. LSTM schematic diagram

Here, h_{t-1} is considered as the previous LSTM cell output, x_t is the present input, and σ activation function is named a hyperbolic tangent. Finally, the weight matrix and bias term are denoted as W_f and b_f, respectively. Furthermore, the function of the input gate i_t is to determine which information is being updated, where $\tilde{C}_t$ is instant condition and activation function ReLU is selected, these terms can be shown as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

$$\tilde{C}_t = ReLU\ (W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$

Here, the weight matrix of the input gate and offset item are denoted by W_i, b_i, respectively. On the other hand, W_C is denoted as a state weight matrix and $\tilde{C}_t$ is the long-term state. Further, the LSTM cell output h_t is achieved based on the updated state of the cell.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{6}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t * \text{Relu}\ (C_t) \tag{8}$$

Here, W_o and b_o are weight matrix and offset item of the output gate, respectively.

6. Attention Mechanism (AM)

When the human brain processes information, more Attention always concentrates on concerned information and neglects unimportant information. Therefore, the Attention Mechanism is presented to power the effect of significant details. Further, Different weights can be applied by AM to the inferred states of the LSTM by parameter learning and mapping weight. The AM model was created to mimic the human brain's attention allocation mechanism, which could measure the input and output relevance for allocating robust features. Thus, the Attention mechanism is used for LSTM to focus on features with greater importance on outputs, which helps to upsurge model performance and accuracy. Figure 3. shows the AM structure, where X_1, X_2, ……. X_t are LSTM input, and h_1, h_2, ……. h_t are the output of the LSTM hidden layer, and these are used as AM inputs to achieve attention weight distribution. Further, the weight shows the significance of the state parameter. The following expressions show the AM model calculation.

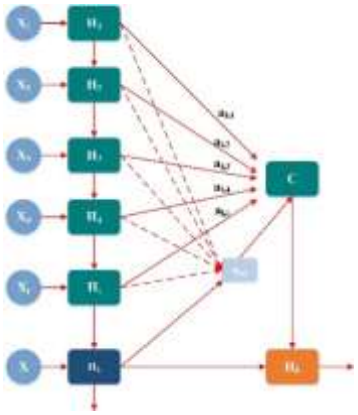

Figure 3. Structure of attention mechanism

$$e_i = utanh\ (wh_i + b) \tag{9}$$

$$\alpha_i = \frac{exp(e_i)}{\sum_i\ exp\ (e_i)} \tag{10}$$

$$C = \sum_i \alpha_i h_i \tag{11}$$

Here, e_i represents the attention probability distribution for h_i at $j - th$ moment. w and u are the weight coefficients, b shows the bias term, and the weighted feature is denoted as C.

7. The Proposed Attention Mechanism Based Lightweight CNN-LSTM Network

The Attention Mechanism-based lightweight CNN-LSTM network comprises an Input layer, CNN, LSTM, and AM models. CNN helps extract spatial features of the dataset, which are considered LSTM network inputs. On the other hand, the LSTM model helps to extract temporal features, which are inputs to the AM model. Finally, the AM model computes the weights and provides importance to significant weights. Figure 4. shows the model structure. In the input layer, we used correlation analysis to input characteristics. The convolutional kernels are set to 128 in the CNN model, and their length is set to 1, which helps the fitting degree improve the more the LSTM layer is concealed. In response, the training time will gradually increase; to balance this, we set the LSTM layer as 2. Further, 128 neurons are set for the first LSTM layer and 64 neurons in the second LSTM layer. Finally, the impact on output is emphasized by determining the feature weight and allocating the input vector for the AM layer. The complete parameter setting is given in Table 3.

Model Layers	**Parameter Settings**
ConvLayer1	Filter size=20, Kernel size = (10, 1), Stride = 1
Max-pooling + Dropout (0.25)	Pool-Layer = (2, 1), Stride = 2
ConvLayer2	Filter = 40, Kernel size = (5, 1), Stride = 1
Max-pooling + Dropout (0.25)	Pool-Layer = (2, 1), Stride = 2
ConvLayer3	Filter = 80, Kernel size = (3, 1), Stride = 1
Max-pooling + Dropout (0.25)	Pool-Layer = (2, 1), Stride = 2
LSTM-1	Hidden-size = 128
LSTM-2	Hidden-size = 64

Table 3. Model Parameter Settings

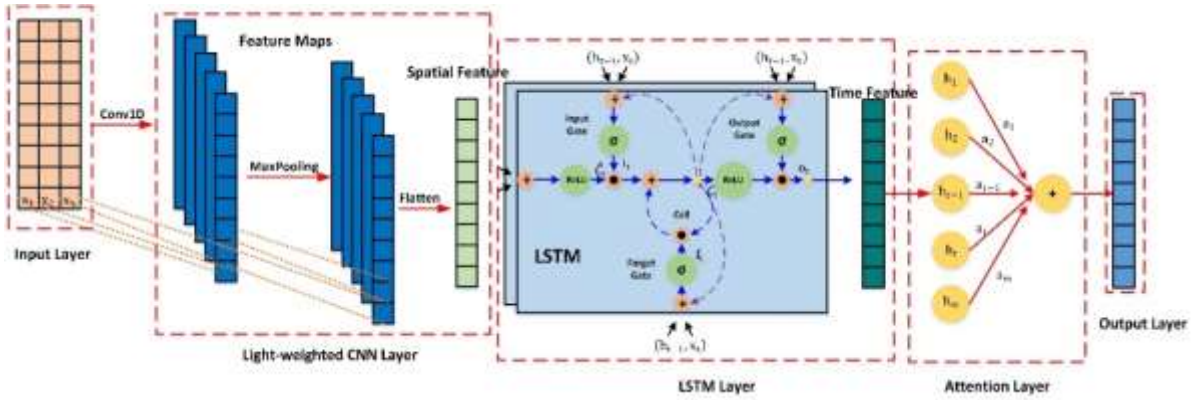

Figure.4 Complete architecture of lightweight CNN-LSTM based on AM model

The output layer is involved in the component's fault. Therefore, the excess is further examined in order to ascertain the UAV's operational condition. The Adam optimizer is applied to collectively update the model weights during model training, and this optimizer is pretty different compared to traditional optimizers such as stochastic gradient descent. For various factors, the independent adaptive learning rate is developed by computing the gradient's first and second-order moment estimation, which helps the algorithm enhance the performance. Further, the early stopping technique is used to overcome the model overfitting. Furthermore, the pseudocode of the LW-CNN-LSTM-based AM model algorithm is shown in Algorithm 1. First, a complete set of trainable parameters is initialized. Second, the train set $T = \{(x_i, y_i) \mid i = 1, \cdots, m\}$ is input to the lightweight CNN model. CNN layers convolute with datasets to produce deep features $X = [X_1, X_2, \cdots, X_i]$ and the Pooling layer is applied further. After, the output of lightweight CNN is fed into the LSTM model as input. Furthermore, LSTM can extract the feature from data at several continuous times to achieve the time series relationship among features, and each time step output is described as $H = [h_1, h_2, \cdots, h_i]$, which is input for the attention mechanism module to rearrange the feature weights. Finally, based on weighted features a_m, the prediction layer produces the predicted output y. The model will update the trainable weights based on the error between actual labels and the output results. The model will check whether the training time approaches the upper limit for the total number of epochs; if not, in this case, one more epoch is added, and the training process can be repeated. The optimized hyperparameter values are listed in Table 4. To verify the AM-based CNN-LSTM performance, wide-ranging experiments on real-time flight data were performed by comparing our introduced model with other state-of-the-art models such as machine learning, deep learning, and LSTM. The details of the dataset, complete experiment setup with results, and their corresponding study are given in the following sections.

Hyperparameters settings	**Value**
Batch size	50
Learning rate	0.001
Optimizer	Adam
dropout_rate	0.25
β_1	0.9
β_2	0.99
Loss function	Categorical cross-entropy
Feature scaling	[0,1]
Activation functions	ReLU (ConvLSTM) sigmoid (ConvLayer)

Table. 4 Model Final parameter values

Algorithm 1: Fault Detection in FW-UAV Based on Light-Weighted CNN-LSTM AM Model

Data: Training data of FW-UAV after EDA: $T = \{(x_i, y_i) \mid i = 1, \cdots, m\}$
Result: Fault Detection Results

- Fully Connected Layer (FC-Layer);
- Pooling layer (MaxPool);
- σ sigmoid function;
- ω convolution layer, s is size of convolution, $i : i + s - 1$ are the extracted features from i to $i + s - 1$ and b is bias term;
- W_o, W_f are weight matrices, b_o is the bias vector, previous hidden time state is h_{t-1}, current time is x_t, memory unit of current time is C_t;
- Predicted results of model and real labels are y and y', respectively.

Training:
Initialization: Initialize complete set of weights in the model;
for p *in* n **do**
  1. Spatial features are extracted Through LW-CNN:
  $X_i = FC\left\{\max\left\{\sigma\left(\omega \times x_{i:i+s-1} + b\right)\right\}\right\}$;
  2. LSTM incorporates the fault features of each time period into a unified sequential fault feature: $h_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) * \tanh\left(C_t\right)$;
  3. The attention mechanism (AM) assign the weights and the FC layer produces the final outcomes: $a_t = \sum \sigma\left(\tanh\left(H_t\right)\right) * h_t, y = FC\left(W_f \cdot a_t\right)$;
  4. Compute the cost value of the model $L = \sum_{i=1}^{N} y^{(i)} \log y'^{(i)} + \left(1 - y^{(i)}\right) \log\left(1 - y'^{(i)}\right)$, then update the model weights accordingly
end

Algorithm 1: Lightweight CNN-LSTM-based AM model algorithm

7.1. **Real-time Flight Dataset**

The publicly available dataset, which is taken from a fixed-wing UAV [56], is used. The flight test system specifications are given in Table 5, and Figure 5 shows the complete UAV system. This flight system has three main parts: the aircraft, the transmitter, and the ground control station. The flight used for testing was a fixed-wing UAV powered by a battery. The ground control system operated on the computer to communicate with and control the drone during autonomous flight through a radio modem called X-Bee. The control can be lost while inserting faults into the flight in autonomous mode, so the drone was manually controlled using the RC transmitter. For all experiments, the Paparazzi Autopilot device was utilized.

Figure 5. Flying test system for data collection [55]

Features	Value
Wing Extent	1.2 meters
Surface Area	0.28 meter^2
Weight	0.75 kilogram
Battery Backup	30 Wh
Flight Acceleration	12 meter/sec
Flight Duration	60 minutes
GPS Model	U-Blox M8N
Motor	T-Motor 2208/18 - Kv 1100
Autopilot	Paparazzi Chimera v1.0

Table 5. Feature Specifications

Numerous faults of effective control loss were inserted into the drone by setting the GCS signal and updating the onboard controller, which assisted in recording different flight logs for different days in July 2020. Flight datasets of July 12th, 13th, 21st, and 23rd were obtained and used in our experiments. Following the data acquisition's fault mode parameters, the flight data of the 12th and 13th of July possess two modes of status: Fault and Normal mode, with 30% efficiency for right-wing control (r_1 = 0.3); further, the flight data of the 21st and 23rd consist of nine faults, in which the remaining seven faults were produced by minimizing the efficiency of left-wing control by 0.9~0.3; therefore, we divide datasets into six groups. There are three groups for binary classification, including 12th and 13th July data and combined data of these two dates, and three groups for the multi-classification problem, in which 21st, 23^{rd,} and integrated data is created. We did not change the variable name as mentioned in [55], which is (a_x, a_y, a_z), (w_x, w_y, w_z), and (u_{cal1}, u_{cal2}) as linear acceleration, angular rates and commands, respectively. We used the presented correlation analysis on obtained flight data to generate training and testing datasets of six kinds of datasets, and then these data were used with different experiments to validate the proposed method's performance. For evaluation and analysis, the datasets of 2000 samples are assigned to a training set, allowing the model to be developed significantly. In addition, the datasets of 200 samples were saved for the test set, which will assist in monitoring the accuracy and the performance of the well-trained model for this scenario dates are 12th, 13^{th,} and combined (12th and 13th) with a wind speed of less than 2m/s, 8m/s and combined, respectively. Further, we used three more dates (21st, 23^{rd,} and combined) and divided the training set and test set into 500 and 50 samples, respectively. We performed all these experiments on ground station laptop computer with specifications: operating system Ubuntu 18.04 with Xeon W-2123 3.60GHz CPU, Tensorflow Keras framework is used on Google colab Pro+ version, and NVIDIA GTX 1080Ti GPU with CUDA.

7.2. Analysis of Experimental Results:

We analyze the introduced lightweight CNN-LSTM framework performance by detecting binary and multiclass faults with diverse sizes of training samples. Here, we denote the training sample size as N and set it to 45, 90, 180, 270, 360, 450, 900, 1800, 2700, and 3600 samples. Further, we randomly select samples from training data, including data for the 12th and 13th of July with combined wind speed. The training data is divided into 70% for training the lightweight CNN-LSTM framework, and the remaining 30% is taken as a test set to get the best performance of the model. On the other hand, the SVM model applied all the training samples, and this same idea was used for 21st and 23rd-day data with combined wind speed for multiclass fault detection. For the final result, the experiment was repeated thrice for each model, and the average detection accuracy was computed as the outcome.

Initially, we examined the performance of the lightweight CNN-LSTM model with diverse training samples for binary fault detection tasks. As we know, SVM is one of the best ML algorithms for binary classification tasks; when trained with N=45, SVM obtains an accuracy of 76.50%, which is 11.65% lower than our model, and this performance gap continues until we try all training samples. Further, we compared our lighted-weighted CNN-LSTM performance with other deep learning-based models and found out the performance of LSTM was worse. On the other hand, hybrid models like CNN-LSTM do better than CNN models; however, the performance of these models is not as good as lightweight CNN-LSTM based on AM, considering all experiments. Further, we examined our model with a sample size of 360, and our model got around 95.66% accuracy compared with CNN, LSTM, and CNN-LSTM-based models with an accuracy of 91.95%, 85.15% and 91.15%, respectively.

We also obtained results on multiclass fault classification problems while maintaining similar experiment settings. Model performances decrease by about 35% at all training samples compared to binary classification, showing that multiclass classification has a greater complexity level. While training the models with a small amount of data, such as 45 and 90 samples, the deep-learning models suffered from overfitting, and the results were not as good as the SVM and CNN-LSTM models. On the other hand, CNN can achieve an accuracy of 28.13% on the sample size of 45, compared to our model, which is 8.28% less. Further, when we tried to expand the training sample from 180 to 450, the deep learning models outclassed SVM, where our model obtained slightly lower accuracy than CNN-LSTM; both models have an accuracy of about 50%. We increased training samples to 3600, and CNN exhibited its power in managing massive datasets and obtaining better results. However, our model also outperformed CNNs and produced better outcomes on all sets of samples.

8. Results and Discussion:

As we know, model size plays a vital role in small drones due to limited space. Further, model size depends on computation complexity, which also consumes more battery, and drones have minimal battery backup; therefore, discussing and comparing our lightweight model with the original models is necessary. To broadly examine the significance of our introduced algorithm on different sizes of samples, we further divide our model into other child models and compare the results with similar models with similar sample sizes and data, including both binary-class fault detection and multiclass fault detection samples. For instance, we compare CNN models with our lightweight CNNs, and then, we compare the CNN-LSTM model with our lightweight CNN-LSTM. As shown in Figure 6, our lightweight CNN model outperformed the CNN-original model in all binary class fault detection aspects. If we compare each sample size separately, our lighted-weighted model has the upper hand on every sample in terms of accuracy with minimum model specifications. To extend our experiment, we compare similar models for multiclass fault detection, as shown in Figure 7. The models' 45, 90, 180, and 270 sample sizes have almost identical accuracy. Still, our model is smaller and has less computation complexity, which shows that our model can be implemented on drones with limited resources.

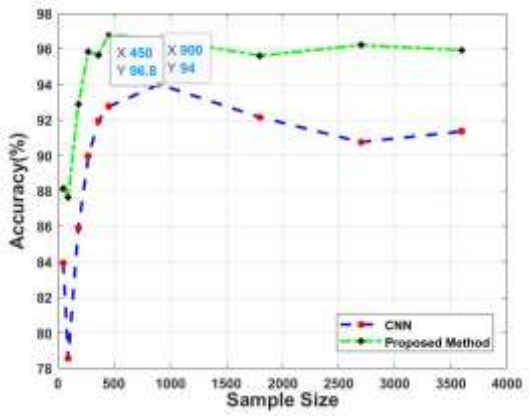

Figure 6. Resulting in graphs comparing the proposed method with the CNN-original model (binary class)

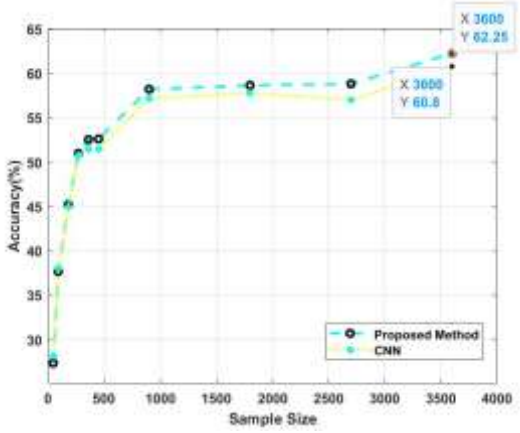

Figure. 7 Resulting graphs comparing our (LW-CNN) with the CNN-original model (multiclass)

Further, we compare the results of lightweight CNN-LSTM with CNN-LSTM on binary and multiclass fault detection data. Our lightweight model outperformed CNN-LSTM and obtained 94.7% accuracy on the sample size 2700, illustrated in Figure 8. Similarly, we checked our model accuracy on multiclass fault detection. We achieved 57.55% accuracy on the sample size 900, as shown in Figure. 9, which is more significant than the CSNL model on the same sample size. Overall, our lightweight model achieved greater accuracy than other state-of-the-art models, showing that lightweight models have the upper hand, contain a smaller size and less complexity, and can be implemented on limited-resourced drones. Table 6. displays the complete set of binary and multiclass fault detection results on all samples. Finally, this discussion did not highlight the single LSTM model; however, Li et al. [56] have experimented on different sizes of neurons (from 2 to 128) of the LSTM model, and it performed well on neuron size of 32 with the

accuracy of 87.45% and 37.96% on binary and multiclass fault detection, respectively. Figure. 10 shows the overall results of the LSTM model with different neuron sizes.

Binary class Fault Detection										
Model / Sample Size	**45**	**90**	**180**	**270**	**360**	**450**	**900**	**1800**	**2700**	**3600**
SVM model accuracy [55]	76.5%	82.95%	86.20%	88.50%	88.95%	90.9%	87.85%	89.3%	92.15%	94.55%
CNN model accuracy [38]	83.95%	78.55%	85.9%	89.95%	91.95%	92.75%	94%	92.15%	90.75%	91.35%
LSTM model accuracy [56]	68.35%	77.8%	82.65%	86.6%	85.15%	85.35%	88.9%	90.7%	89.7%	89.85%
CNN-LSTM model accuracy [50]	83%	87.6%	83.05%	90.2%	91.15%	91.05%	93.7%	91.8%	94.7%	90.35%
SHNN model accuracy [56]	87.45%	87.1%	91.65%	94.65%	96.45%	96.9%	95.8%	94.4%	**96.35%**	94.25%
Proposed Model accuracy	**88.15%**	**87.63%**	**92.9%**	**95.83%**	**95.66%**	**96.8%**	**96.56%**	**95.62%**	94.95%	**95.93%**
Multiclass Fault Detection										
Model / Sample Size	**45**	**90**	**180**	**270**	**360**	**450**	**900**	**1800**	**2700**	**3600**
SVM model accuracy [55]	34.27%	39.2%	43.87%	45.16%	45.73%	46.36%	49.2%	52.89%	55.29%	55.07%
CNN model accuracy [38]	28.13%	38.04%	44.93%	50.71%	51.47%	51.47%	57.16%	57.82%	**57.02%**	**60.8%**
LSTM model accuracy [56]	30.76%	38.53%	46.04%	46.53%	47.96%	49.79%	47.56%	47.16%	50.71%	51.78%
CNN-LSTM model accuracy [50]	30%	38.62%	52.22%	53.42%	55.07%	53.82%	56.67%	51.42%	56.04%	54.76%
SHNN model accuracy [56]	**35.16%**	39.69%	49.87%	50.71%	54%	54.18%	52.84%	**54.04%**	52.44%	51.73%
Proposed Model accuracy	34.89%	**40.12%**	**53.86%**	**54.52%**	**56.66%**	**54.8%**	**57.55%**	53.66%	56.84%	55.88%

Table 6. Comparison results of our model with other state-of-the-art methods on binary and multiclass fault detection

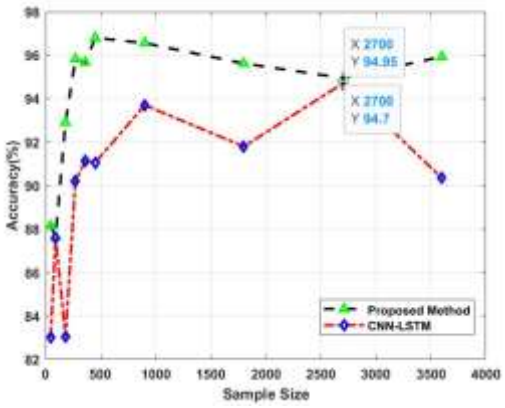

Figure 8. Resulting graphs comparing our (LW-CNN-LSTM) with the CNN-original-LSTM model (binary class)

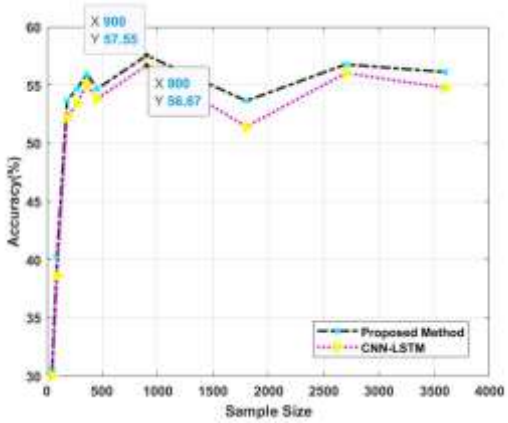

Figure 9. Resulting graphs comparing our (LW-CNN-LSTM) with the CNN-original-LSTM model (multiclass)

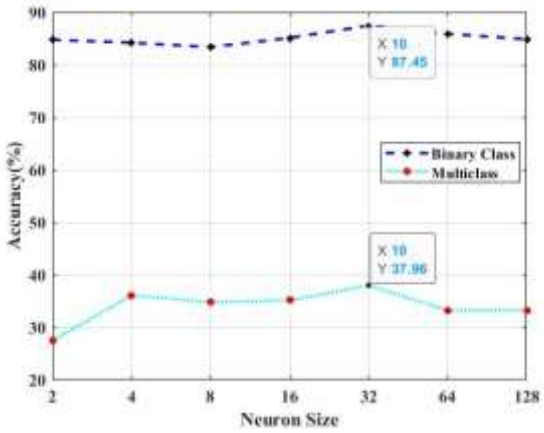

Figure 10. Hidden neuron in the LSTM layer

Finally, we compare our lightweight model with machine learning approaches such as SVM for binary class and multiclass fault detection; comparison results are shown in Figures. 11a and 11b; our model outperformed the machine learning model with a more significant margin. Afterward, we tried to compare our model with a time-series model such as LSTM, and the results shown in Figures. 12a and 12b, our model has a better approach than the LSTM model on binary and multiclass fault detection. Finally, the Siamese hybrid neural network (SHNN) is also compared with our state-of-the-art lightweight CNN-LSTM-based AM model, and the results are similar in a few samples; however, our model outperformed the SHNN model on large sample sizes such as 900, 1800, 2700, and 3600 as illustrated in Figures 13a and 13b. This shows that our model has better learning ability on large samples in binary and multiclass scenarios, consumes lower memory, and saves power consumption. Further, we also compared our proposed algorithm with other methods in the context of detection efficiency, time efficiency, resilience, and model complexity in Table 7, which shows that our proposed method outperformed other state-of-the-art methods in most of settings.

Reference and Year	Detection Efficiency	Time Efficient	Used Deep Learning	Resilient	Model Complexity
[57] - 2019	Good	✓	✓	×	High
[58] - 2020	Poor	×	✓	×	High
[59] - 2021	Acceptable	×	✓	×	High
[60] - 2022	Good	✓	✓	✓	High
Our Proposed	Good	✓	✓	✓	Low

Table 7. Comparison results of the proposed method with other state-of-the-art methods in different contexts

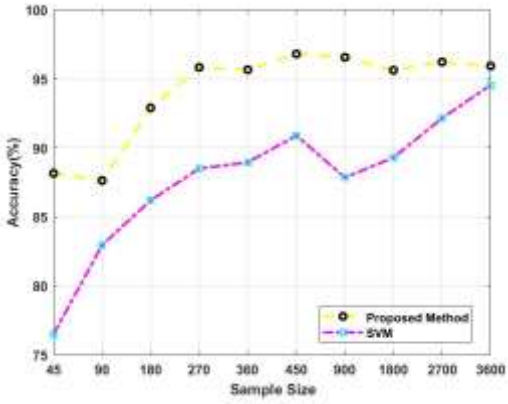

Figure. 11a Resulting graphs comparing our model with SVM (binary class)

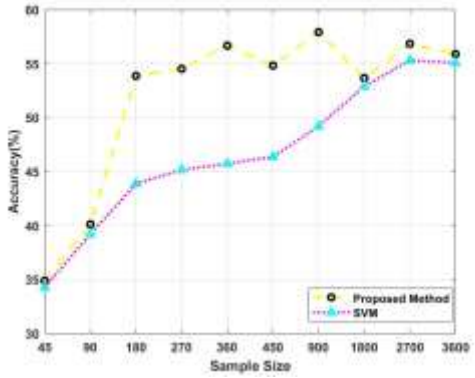

Figure. 11b Resulting graphs comparing our model with SVM (multiclass class)

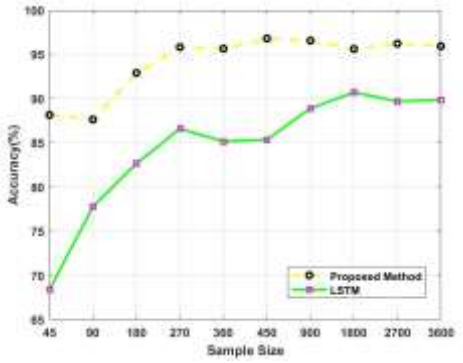

Figure. 12a Resulting graphs comparing our model with LSTM (binary class)

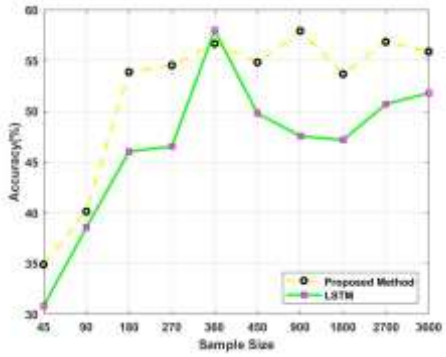

Figure. 12b Resulting graphs comparing our model with LSTM (multiclass)

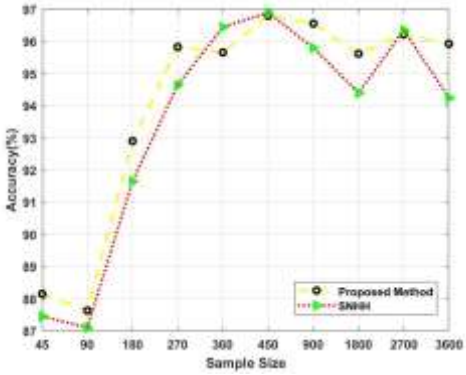

Figure. 13a Resulting graphs comparing our model with SHNN (binary class)

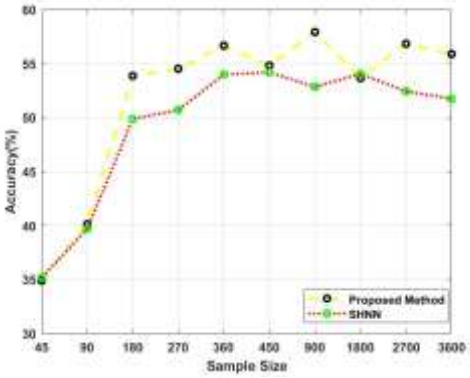

Figure. 13b Resulting graphs comparing our model with SHNN (multiclass)

9. Conclusion:

Building on prior work, the CNN-LSTM-AM model, which is designed to classify and detect faults in FW-UAVs on real-flight datasets, is introduced in this article. The lightweight CNN model is adopted to obtain spatial features buried in real-flight datasets. During training, the convolutional layers of lightweight CNN can achieve remarkably differentiated features, and later layers perform classification tasks. The LSTM is best at fusion time characteristics features at the status of different parts. Further, AM is built for LSTM to focus on the characteristics that highly influence output variables to further enhance the algorithm's accuracy. Considerable experiments on real-flight data for small-sized FW-UAV, including binary and multiclass fault detection, authenticate that the lightweight CNN-LSTM-AM can achieve high accuracy with low power consumption and consumes less memory. The algorithm illustrates better results than other state-of-the-art machine learning, deep learning, and LSTM-based approaches, which highlights encouraging FW-UAV data-effective results. In the future, the model can be further enhanced to achieve more accurate and stable detections. In the meantime, transfer learning and data augmentation techniques are considered to be embedded with the model, which can be used with other types of UAVs to improve the model's performance. Further, time-frequency features can be achieved via Wavelet transform, which can guide shorter identification time and help increase accuracy further.

Acknowledgments: This work is funded by the International Cooperation Foundation of Jilin Province (20210402074GH). This work is also supported by the 111 Project of China (D21009, D17017).

Decelerations

Competing Interests
The authors have no competing interests to declare relevant to this article's content.

Authors contribution statement
Aakash Kumar conceptualized the research, collected the data, and performed data analysis. Shifeng Wang assisted with formal analysis and investigation and also served as the corresponding author. Ali Muhammad Shaikh and Hazrat Bilal reviewed and edited the draft. Shifeng Wang and Shigeng Song supervised the entire research instructed during various phases of the study and provided meaningful guidelines. Aakash Kumar and Lu Bo wrote the first draft of the manuscript, and all authors commented on previous versions. All authors read and approved the final manuscript.

Data availability and access

The datasets generated during and analyzed during the current study are available from the corresponding author upon reasonable request.

Compliance with Ethical Standards
I certify that this article has no actual or potential conflict of interest. This research does not involve human participants and animals and does not require informed consent.

References:

1. Zhao Y, Ma J, Li X, Zhang J (2018) Saliency Detection and Deep Learning-Based Wildfire Identification in UAV Imagery. Sensors 18:712. https://doi.org/10.3390/s18030712

2. Sibanyoni SV, Ramotsoela DT, Silva BJ, Hancke GP (2019) A 2-D Acoustic Source Localization System for Drones in Search and Rescue Missions. IEEE Sensors J 19:332–341. https://doi.org/10.1109/JSEN.2018.2875864

3. Li M, Zhen L, Wang S, et al (2018) Unmanned aerial vehicle scheduling problem for traffic monitoring. Computers & Industrial Engineering 122:15–23. https://doi.org/10.1016/j.cie.2018.05.039

4. James AT, Kumar G, Khan AQ, Asjad M (2022) Maintenance 4.0: implementation challenges and its analysis. International Journal of Quality & Reliability Management 40:1706–1728. https://doi.org/10.1108/IJQRM-04-2021-0097

5. Cortés-Leal A, Cárdenas C, Del-Valle-Soto C (2022) Maintenance 5.0: Towards a Worker-in-the-Loop Framework for Resilient Smart Manufacturing. Applied Sciences 12:11330. https://doi.org/10.3390/app122211330

6. Fourlas GK, Karras GC (2021) A Survey on Fault Diagnosis Methods for UAVs. In: 2021 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, Athens, Greece, pp 394–403

7. Guo D, Zhong M, Zhou D (2018) Multisensor Data-Fusion-Based Approach to Airspeed Measurement Fault Detection for Unmanned Aerial Vehicles. IEEE Trans Instrum Meas 67:317–327. https://doi.org/10.1109/TIM.2017.2735663

8. Zuo L, Yao L, Kang Y (2020) UIO Based Sensor Fault Diagnosis and Compensation for Quadrotor UAV. In: 2020 Chinese Control And Decision Conference (CCDC). IEEE, Hefei, China, pp 4052–4057

9. Gao J, Zhang Q, Chen J (2020) EKF-Based Actuator Fault Detection and Diagnosis Method for Tilt-Rotor Unmanned Aerial Vehicles. Mathematical Problems in Engineering 2020:1–12. https://doi.org/10.1155/2020/8019017

10. 5G_SDR-Assisted_Cognitive_Communication_in_UAV_Swarms_Architecture_and_Applications.pdf. In: Google Docs. https://drive.google.com/file/d/1PF__isoP0Y_p7BTm8oBI2T3TEoG5nsW3/view?usp=embed_facebook. Accessed 8 Feb 2024

11. Akhunzada A, Zeadally S, Islam SU (2021) Power and Performance Efficient SDN-Enabled Fog Architecture. IT Prof 23:24–30. https://doi.org/10.1109/MITP.2021.3085840

12. Ali Z, Henna S, Akhunzada A, et al (2019) Performance Evaluation of LoRaWAN for Green Internet of Things. IEEE Access 7:164102–164112. https://doi.org/10.1109/ACCESS.2019.2943720

13. Miao J, Wang J, Wang D, Miao Q (2021) Experimental investigation on electro-hydraulic actuator fault diagnosis with multi-channel residuals. Measurement 180:109544. https://doi.org/10.1016/j.measurement.2021.109544

14. Wang N, Ren J, Luo Y, et al (2021) UAV Actuator Fault Detection using Maximal Information Coefficient and 1-D Convolutional Neural Network. In: 2021 Global Reliability and Prognostics and Health Management (PHM-Nanjing). IEEE, Nanjing, China, pp 1–6

15. Li C, Li S, Zhang A, et al (2021) Meta-learning for few-shot bearing fault diagnosis under complex working conditions. Neurocomputing 439:197–211. https://doi.org/10.1016/j.neucom.2021.01.099

16. Bhatia AK, Jiang J, Kumar A, et al (2021) Adaptive preview control with deck motion compensation for autonomous carrier landing of an aircraft. Int J Adapt Control Signal Process 35:769–785. https://doi.org/10.1002/acs.3228

17. Perez E, Kiela D, Cho K (2021) True Few-Shot Learning with Language Models

18. Bilal H, Yin B, Kumar A, et al (2023) Jerk-bounded trajectory planning for rotary flexible joint manipulator: an experimental approach. Soft Comput 27:4029–4039. https://doi.org/10.1007/s00500-023-07923-5

19. Kumar A, Shaikh AM, Li Y, et al (2021) Pruning filters with L1-norm and capped L1-norm for CNN compression. Appl Intell 51:1152–1160. https://doi.org/10.1007/s10489-020-01894-y

20. Ding SX (2013) Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms and Tools. Springer London, London

21. Gangsar P, Tiwari R (2020) Signal based condition monitoring techniques for fault detection and diagnosis of induction motors: A state-of-the-art review. Mechanical Systems and Signal Processing 144:106908. https://doi.org/10.1016/j.ymssp.2020.106908

22. Hoang D-T, Kang H-J (2019) A survey on Deep Learning based bearing fault diagnosis. Neurocomputing 335:327–335. https://doi.org/10.1016/j.neucom.2018.06.078

23. Chakrapani G, Sugumaran V (2023) Transfer learning based fault diagnosis of automobile dry clutch system. Engineering Applications of Artificial Intelligence 117:105522. https://doi.org/10.1016/j.engappai.2022.105522

24. Wu TY, Lei KW (2019) Prediction of surface roughness in milling process using vibration signal analysis and artificial neural network. Int J Adv Manuf Technol 102:305–314. https://doi.org/10.1007/s00170-018-3176-2

25. Boztaş S, Özbudak F, Tekin E (2022) New Correlations of m-sequences over the finite field \mathbbF_4 compatible with a new bijection to \mathbbZ_4. In: 2022 10th International Workshop on Signal Design and Its Applications in Communications (IWSDA). pp 1–5

26. Thomas JB, Chaudhari SG, K. V. S, Verma NK (2023) CNN-Based Transformer Model for Fault Detection in Power System Networks. IEEE Transactions on Instrumentation and Measurement 72:1–10. https://doi.org/10.1109/TIM.2023.3238059

27. Wang T, Zhang L, Wang X (2023) Fault Detection for Motor Drive Control System of Industrial Robots Using CNN-LSTM-based Observers. Trans Electr Mach Syst 7:144–152. https://doi.org/10.30941/CESTEMS.2023.00014

28. Freddi A, Longhi S, Monteriù A (2012) A Diagnostic Thau Observer for a Class of Unmanned Vehicles. J Intell Robot Syst 67:61–73. https://doi.org/10.1007/s10846-012-9650-4

29. Ortiz-Torres G, Castillo P, Sorcia-Vazquez FDJ, et al (2020) Fault Estimation and Fault Tolerant Control Strategies Applied to VTOL Aerial Vehicles With Soft and Aggressive Actuator Faults. IEEE Access 8:10649–10661. https://doi.org/10.1109/ACCESS.2019.2963693

30. Baldini A, Felicetti R, Freddi A, et al (2022) Hexarotor Fault Tolerant Control Using a Bank of Disturbance Observers. In: 2022 International Conference on Unmanned Aircraft Systems (ICUAS). pp 608–616

31. Ghalamchi B, Jia Z, Mueller MW (2020) Real-Time Vibration-Based Propeller Fault Diagnosis for Multicopters. IEEE/ASME Trans Mechatron 25:395–405. https://doi.org/10.1109/TMECH.2019.2947250

32. Ai S, Song J, Cai G, Zhao K (2022) Active Fault-Tolerant Control for Quadrotor UAV against Sensor Fault Diagnosed by the Auto Sequential Random Forest. Aerospace 9:518. https://doi.org/10.3390/aerospace9090518

33. Liang S, Zhang S, Huang Y, et al (2022) Data-driven fault diagnosis of FW-UAVs with consideration of multiple operation conditions. ISA Transactions 126:472–485. https://doi.org/10.1016/j.isatra.2021.07.043

34. Park J, Jung Y, Kim J-H (2022) Multiclass Classification Fault Diagnosis of Multirotor UAVs Utilizing a Deep Neural Network. Int J Control Autom Syst 20:1316–1326. https://doi.org/10.1007/s12555-021-0729-1

35. Kang W, Xiao J, Xue J (2022) Complex system health condition estimation using tree-structured simple recurrent unit networks. Complex Intell Syst 8:5203–5221. https://doi.org/10.1007/s40747-022-00732-7

36. Liu G, Wei P, Chen K, et al (2022) Polymer gear contact fatigue reliability evaluation with small data set based on machine learning. Journal of Computational Design and Engineering 9:583–597. https://doi.org/10.1093/jcde/qwac020

37. Chen X-M, Wu C-X, Wu Y, et al (2019) Design and Analysis for Early Warning of Rotor UAV Based on Data-Driven DBN. Electronics 8:1350. https://doi.org/10.3390/electronics8111350

38. Guo D, Zhong M, Ji H, et al (2018) A hybrid feature model and deep learning based fault diagnosis for unmanned aerial vehicle sensors. Neurocomputing 319:155–163. https://doi.org/10.1016/j.neucom.2018.08.046

39. Cui A, Zhang Y, Zhang P, et al (2020) Intelligent Health Management of Fixed-Wing UAVs: A Deep-Learning-based Approach

40. Ahmad MW, Akram MU, Ahmad R, et al (2022) Intelligent framework for automated failure prediction, detection, and classification of mission critical autonomous flights. ISA Transactions 129:355–371. https://doi.org/10.1016/j.isatra.2022.01.014

41. Liu W, Chen Z, Zheng M (2020) An Audio-Based Fault Diagnosis Method for Quadrotors Using Convolutional Neural Network and Transfer Learning

42. Zhang Y, Zhao H, Ma J, et al (2021) A Deep Neural Network-Based Fault Detection Scheme for Aircraft IMU Sensors. International Journal of Aerospace Engineering 2021:e3936826. https://doi.org/10.1155/2021/3936826

43. Miao J, Wang J, Wang D, Miao Q (2021) Experimental investigation on electro-hydraulic actuator fault diagnosis with multi-channel residuals. Measurement 180:109544. https://doi.org/10.1016/j.measurement.2021.109544

44. Wang X, Roy S, Farì S, Baldi S (2020) The problem of reliable design of vector-field path following in the presence of uncertain course dynamics ∗∗This work was partly supported by the Fundamental Research Funds for the Central Universities grant no. 4007019109 (RECONSTRUCT), and by the special guiding funds “double first-class” grant no. 4007019201. S. Farì was with TU Delft and is now with German Aerospace Center. (corresponding author: S. Baldi). IFAC-PapersOnLine 53:9399–9404. https://doi.org/10.1016/j.ifacol.2020.12.2409

45. Li Z, Zhang Y, Ai J, et al (2023) A Lightweight and Explainable Data-Driven Scheme for Fault Detection of Aerospace Sensors. IEEE Transactions on Aerospace and Electronic Systems 59:8392–8410. https://doi.org/10.1109/TAES.2023.3303855

46. Jiang T, Zong M, Ma Y, et al (2023) MobileACNet: ACNet-Based Lightweight Model for Image Classification. In: Yan WQ, Nguyen M, Stommel M (eds) Image and Vision Computing. Springer Nature Switzerland, Cham, pp 361–372

47. Ghose P, Biswas M, Gaur L (2023) BrainSegNeT: A Lightweight Brain Tumor Segmentation Model Based on U-Net and Progressive Neuron Expansion. In: Liu F, Zhang Y, Kuai H, et al (eds) Brain Informatics. Springer Nature Switzerland, Cham, pp 249–260

48. Cai G, Lum K-Y, Chen BM, Lee TH (2010) A brief overview on miniature fixed-wing unmanned aerial vehicles. In: IEEE ICCA 2010. pp 285–290

49. Ducard G (2015) Actuator Fault Detection in UAVs. In: Valavanis KP, Vachtsevanos GJ (eds) Handbook of Unmanned Aerial Vehicles. Springer Netherlands, Dordrecht, pp 1071–1122

50. Fu J, Sun C, Yu Z, Liu L (2019) A hybrid CNN-LSTM model based actuator fault diagnosis for six-rotor UAVs. In: 2019 Chinese Control And Decision Conference (CCDC). pp 410–414

51. Kumar A (2022) Structure Level Pruning of Efficient Convolutional Neural Networks with Sparse Group LASSO. IJMLC 12:. https://doi.org/10.18178/ijmlc.2022.12.5.1111

52. Li Y, Wang L, Peng S, et al (2019) Using Feature Entropy to Guide Filter Pruning for Efficient Convolutional Networks. In: Tetko IV, Kůrková V, Karpov P, Theis F (eds) Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning. Springer International Publishing, Cham, pp 263–274

53. Kumar A, Yin B, Shaikh AM, et al (2022) CorrNet: pearson correlation based pruning for efficient convolutional neural networks. Int J Mach Learn & Cyber 13:3773–3783. https://doi.org/10.1007/s13042-022-01624-5

54. Zhao Z, Chen W, Wu X, et al (2017) LSTM network: a deep learning approach for short-term traffic forecast. IET Intelligent Transport Systems 11:68–75. https://doi.org/10.1049/iet-its.2016.0208

55. Bronz M, Baskaya E, Delahaye D, Puechmore S (2020) Real-time Fault Detection on Small Fixed-Wing UAVs using Machine Learning. In: 2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC). pp 1–10

56. Li C, Li S, Zhang A, et al (2022) A Siamese hybrid neural network framework for few-shot fault diagnosis of fixed-wing unmanned aerial vehicles. Journal of Computational Design and Engineering 9:1511–1524. https://doi.org/10.1093/jcde/qwac070

57. Lindemann B, Fesenmayr F, Jazdi N, Weyrich M (2019) Anomaly detection in discrete manufacturing using self-learning approaches. Procedia CIRP 79:313–318. https://doi.org/10.1016/j.procir.2019.02.073

58. Titouna C, Nait-Abdesselam F, Moungla H (2020) An Online Anomaly Detection Approach For Unmanned Aerial Vehicles. 2020 International Wireless Communications and Mobile Computing (IWCMC) 469–474. https://doi.org/10.1109/IWCMC48107.2020.9148073

59. Park KH, Park E, Kim HK (2021) Unsupervised Fault Detection on Unmanned Aerial Vehicles: Encoding and Thresholding Approach. Sensors 21:2208. https://doi.org/10.3390/s21062208

60. Tlili F, Ayed S, Chaari L, Ouni B (2022) Artificial Intelligence Based Approach for Fault and Anomaly Detection Within UAVs. In: Barolli L, Hussain F, Enokido T (eds) Advanced Information Networking and Applications. Springer International Publishing, Cham, pp 297–308