Comparative Analysis of LSTM Variants for Fault Detection and Classification in Aircraft Control Surfaces

Muhammad Fajar[1], Teuku Mohd Ichwanul Hakim[2], Adi Wirawan[3], Prasetyo Ardi Probo Suseno[4], Arifin Rasyadi Soemaryanto[5], and Ardanto Mohamad Pramutadi[6]

[1,2,3,4,5,6]*Research Center for Aeronautics Technology, National Research and Innovation Agency, Bogor, Jawa Barat, 16350, Indonesia*

muha110@brin.go.id
teuk003@brin.go.id
adiw002@brin.go.id
pras012@brin.go.id
arif037@brin.go.id
arda001@brin.go.id

ABSTRACT

Aircraft control surfaces play a critical role in ensuring safe and efficient flight. Faults in these surfaces could lead to catastrophic consequences. This paper investigates the application of Long Short-Term Memory (LSTM) networks for fault detection and classification in aircraft control surfaces. Four deep learning models: LSTM, Stacked-LSTM, Bi-LSTM, and Attention-based LSTM (ALSTM), were trained, validated, and tested to classify faults based on residual features. The methodology involved data generation, preprocessing, normalization, and training the models over 200 epochs. Evaluation metrics, including confusion matrices, precision, recall, and F1-scores, were used to assess model performance. Results show that Bi-LSTM achieved the highest accuracy (98.93%) and lowest loss (0.0264), significantly outperforming other models in fault detection, particularly for challenging fault types such as hard-over and lock-in-place. ALSTM followed closely, with notable performance improvements over standard and stacked LSTM models.

1. INTRODUCTION

The rise of automation in flight control has enabled pilots to better control their aircraft. The automation that comes in the form of a flight control augmentation system supports the pilot by easing the workload. Flight augmentation system helps the pilot by taking over the mundane constant adjustment to the aircraft attitude, air speed, altitude, and other tasks so the pilot could then focus on flying the aircraft. On the other hand, flight automation with autonomous flight capability could take over the flying task from the pilot. The pilot could then focus on monitoring the aircraft after giving the system inputs of navigation waypoints.

One of the most widely used applications of full autonomous flight systems today is in Unmanned Aerial Vehicle (UAV) or commonly referred to as drone. Emerging trends in UAV research (Mohsan et al., 2022; Telli et al., 2023) focused on improving endurance, expanding payload capabilities, and refining cooperative swarm behavior to execute tasks more efficiently. This evolution did not only expand the possibilities for UAV deployment but also highlights the importance of reliable and secure control systems, especially in environments where precision and safety are critical, such as autonomous delivery systems or military operations as has been shown (Telli et al., 2023). However, the growing reliance on UAV also emphasizes the need for reliable control systems, particularly in autonomous operations where failure in control surfaces, sensors, or actuators could lead to mission failure or severe safety risks.

To further the research on UAV technology, BRIN (previously LAPAN) had developed the LSA-02. LSA-02 is a UAV technology demonstrator based on the S15, a two-seater aircraft by Stemme AG, that would act as a research platform as depicted in Figure 1. It was developed to explore UAV technologies, including Flight Control Laws (FCL), sensor systems, and Flight Control Panel interfaces as defined by (Bahri, 2016). Operated with a safety pilot for testing, the aircraft's control system remains mechanically based, using push rods and cables. An electronic flight control system (EFCS) supplements but does not replace this setup. When activated, the EFCS utilizes mechanical linkages to control flight controls surfaces via electro-

Muhammad Fajar et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

https://doi.org/10.36001/IJPHM.2025.v16i1.4231

mechanical actuators based on FCL commands as defined by (Hakim, 2016).

Figure 1 Stemme S15 as basic aircraft for UAV Technology Demonstrator

In autonomous flight operation mode, faults in the control system, particularly in control surfaces or actuators, could lead to loss of stability and potentially catastrophic failures. Therefore, ensuring safety during flight with EFCS operations is crucial. To address this, fault tolerant control strategies would be employed to manage and mitigate faults while ensuring safety in the presence of faults or failure in the system. The fault may arise from various sources, such as sensor malfunctions, actuator failures, communication losses, or software errors. By implementing robust fault detection and isolation techniques, the impact of these faults could be minimized, thereby enhancing the overall reliability and safety of the system as shown (Puchalski et al., 2022; Puchalski & Giernacki, 2022; Rudin et al., 2015). Early fault detection enables the system to implement mitigation measures such as switching to manual control, utilizing redundant controls, or adjusting automatic control algorithms to avoid complete failure. The inclusion of fault detection would enable the pilot to identify a potential issue and could prepare them to take over the aircraft control from the EFCS.

Fault in the control system could be identified by means of comparison between the commanded value and the measured value. Such method was performed by Zolghadri et al. (2011) by comparing command channel and monitor channel signal and by Park et al. (2009) with similarity measure.

Various research from 2016 to 2022 indicating that 57% of actuator fault diagnosis in UAVs is based on model-based approaches, while 43% relies on data-based techniques as has been shown (Puchalski et al., 2022). A model-based approach performs fault diagnosis using models. Simani et al. (2003) used this approach to define a comprehensive methodology for actuator, component and sensor fault detection and isolation using output estimation.

Meanwhile, a data-based approach uses data to check if a fault has occurred in a system. This approach has been utilized for the stratospheric airship control system as shown (Hu et al., 2024). More recently, the method of residual measurement was commonly used to detect faults. The work of Boni et al. (2024) developed a data driven residual generation to detect jamming faults. Yu et al. (2024) developed a model-based and neural network residual generation for an early and robust oscillatory fault and Ossmann et al. (2023) incorporated multi-model fault detection also for the same case. Singh et al. (2023) developed a graph theoretic approach where the detection of fault-tree residual was done by Grubbs outlier detection. The isolation of the fault is done by adding power spectral density over fixed FFT windows under a parameter selected by particle swarm optimization (PSO). Venkataraman et al. (2019) performed a comparison of fault detection models which are based on linear time-invariant (LTI), observer-based with robust linear parameter varying (LPV), and multiple model adaptive estimation framework (MMAE). The result was that each method should be used according to the needs: the second method used when it is sufficient that the needs is just to detect and isolate faults. While the third method should be used when the reconfigurable flight control requires fault magnitude estimate and performance deterioration. Regarding incipient fault, Wang et al. (2023) developed a residual observation comprising of sliding mode observer SMO and Luenberger observer.

Another method to detect faults was also observed from the work of Reynoso-Meza et al. (2023) where a time stacking approach in a detection tree classifier was used to detect an oscillatory failure case. Data-driven fault detection based on Support Vector Machine (SVM) was developed by Grehan et al. (2023). Cieslak et al (2016) developed a fault detection and diagnosis method with differentiation schemes comparison between finite difference method combined with moving average filter, Levant sliding-mode differentiator, and uniform robust exact differentiator. Additionally, Pang et al. (2025) in their studies explored the impact of communication reliability metrics on system performance, which can be utilized to enhance fault detection methods by addressing communication loss and latency uncertainties.

Transformer-style and Gated Recurrent Unit (GRU)-based methods are quite common as a method to detect faults in aerospace systems. Giral et al. (2024), Su et al. (2024), Ahmad et al. (2024), and Li et al. (2023) have used transformer-based method to monitor faults detection and condition monitoring citing improved fault detection, diagnostic, and robustness compared to other methods. While GRU-based method was cited to have strong performance particularly when dealing with limited data or computational power constraints. Peng et al. (2022) used GRU-based approach to diagnose electromechanical actuator faults. Masalimov et al. (2022) and Ma et al. (2021) combined GRU and Convolutional Neural Network (CNN) to improve condition monitoring for UAV components with high accuracy, speed, and robustness. Compared to transformers,

GRU-based method offers simpler architecture in modeling sequential dependencies.

LSTM is another method in the detection of faults. Tao et al. (2023) developed a fault detection and isolation method based on SAE-BiLSTM model for electromechanical actuators. LSTM method (Lei et al., 2019) has been used to develop a fault diagnosis of wind turbine. The Stacked LSTM is an extension to the LSTM model that has multiple hidden LSTM layers where each layer contains multiple memory cells. Yu (2019) proposed a hierarchical algorithm for bearing fault diagnosis based on Stacked LSTM aiming to overcome shortcomings of shallow structures. A novel fault diagnosis method for chemical process based on Stacked LSTM has been explored by (Zhang et al., 2020). Their proposed method able to observe the temporal correlation in the sequential observation signals of the chemical process.

Recent research on Bidirectional LSTM (Bi-LSTM) for fault detection has shown promising results in various applications, particularly in diagnosing faults in rotating machinery and electrical motors. Bi-LSTM can process input sequences in both forward and backward directions, this allows it to capture comprehensive temporal patterns, leading to improved fault diagnosis accuracy compared to traditional LSTM models. The Bi-LSTM's ability to analyze sequences from both directions enabled it to better understand the temporal dependencies and patterns that are critical for accurate fault diagnosis as shown (Bharatheedasan et al., 2023).One study (Vanga et al., 2023) focused on the fault classification of three-phase induction motors using Bi-LSTM networks. The research demonstrated that Bi-LSTM could effectively classify different types of motor faults based on current and voltage signatures. Another study applied Bi-LSTM to the fault diagnosis and remaining useful life prediction of rolling bearings. By combining a convolutional neural network with Bi-LSTM, the research achieved significant improvements in detecting and predicting faults in rolling bearings.

The application of the hybrid CNN-LSTM attention-based model, combined with the use of quantile regression to capture uncertainties has been used to predict electrical machine failures as shown (Borré et al., 2023). A similar method by Sun et al (2024) is used for predicting faults in a marine diesel engine. While Qin (2017) introduce an integration of dual stage attention mechanism with a recurrent Neural Network (RNN) with the aim to improve time series forecasting task.

One study (Lee et al., 2024) proposed a GCN-based LSTM auto-encoder combined with a self-attention mechanism to diagnose faults in bearing systems. This model was designed to handle multivariate time series data from multiple sensors, enhancing the model's ability to capture both long-term and short-term dependencies in the data. The addition of a self-attention mechanism allowed the model to focus on the most critical features, leading to improved fault classification accuracy. Another study (Yoo et al., 2023) explored using a two-stage attention-based variational LSTM for fault detection in the electrolytic copper manufacturing process. This model also leveraged attention mechanisms to improve the identification of subtle faults by dynamically adjusting the model's focus based on the importance of different input features.

Long Short-Term Memory (LSTM) is an advanced Recurrent Neural Networks (RNN) that can learn long-term dependencies between time steps in time series data. Huang et al. (2021) has used broad long short-term memory (BLSTM) to construct an adaptive finite time control structure for a UAV, citing preferable time memory characteristic and high learning speed. LSTM has better accuracy and stability to compensate for error due to noises, as verified by simulation (Mao et al., 2021). The model based on the LSTM was identified to have high operation efficiency and could meet real time requirements for compensation.

LSTM can be applied to detect and classify faults in aircraft control surfaces, which are driven by actuators. Actuator faults classified as can be classified into lock-in-place, float, hard-over, and loss effectiveness as defined by (Alwi et al., 2011; Bošković & Mehra, 2003). Lock failure occurs when an actuator becomes jammed and immovable, leaving it stuck in one position. Float failure involves the control surface moving freely without generating any force or affecting the aircraft's control. One of the most critical types of failure is a runaway or hard-over condition, where the control surface moves at its maximum rate limit until it reaches its maximum limit position.

From the literature review, LSTM and its variant has the advantage of accuracy in time series modeling (Tao et al, 2023, Huang et al., 2021). It is also noted that LSTM is uncomplicated to model short term and long-term dependencies (Lei et al., 2019, Lee et al., 2024). In comparison to GRU, LSTM provided more expressive memory mechanism which makes it better suited for capturing intricate fault patterns especially in safety-critical aerospace applications (Cho et al., 2014). While Transformer models are powerful, their need for large training data and computational intensity poses limitations for embedded or real-time fault detection (Vaswani et al., 2017). Aircraft actuators operate for a long period of time and need to be monitored constantly due to their critical control and safety function. For this case, LSTM approach offers its proven ability to capture long-term dependencies in sequential data, which is crucial for identifying gradual fault progression (Hochreiter and Schmidhuber, 1997). LSTM offers a practical balance of performance, interpretability, and resource efficiency, making it a suitable and reliable choice for fault detection (Gao et al., 2021). Therefore, the authors would like to explore LSTM and its variants for the purpose of fault detection and classification.

The motivation and contributions of this paper are twofold. First, the study focuses on evaluating different LSTM-based sequence classifiers—namely, standard LSTM, stacked-LSTM, Bi-LSTM, and ALSTM—for the task of fault detection and classification in actuator systems. In this work, residual magnitude derived from command and sensor signal differences is used as the sole input feature, allowing a direct and isolated comparison of the sequence modeling capabilities of each LSTM variant. This approach shifts the emphasis from residual engineering to classifier architecture, highlighting the core novelty of the study: an in-depth comparison of LSTM variants in their ability to capture temporal patterns in residual signals for fault diagnosis. The objective is to determine which model most effectively identifies the presented fault types, which include hard-over fault, lock-in-place fault, float fault, and loss of effectiveness fault. Second, the study contributes to the broader literature by demonstrating the applicability and performance of advanced LSTM architectures in the context of aircraft actuator fault diagnosis, a domain where reliable real-time classification is critical for system safety and performance.

2. METHODOLOGY

The methodology in this research is divided into three main phases. First, the data preparation phase, which involves generating data, calculating residuals, merging and labelling the data, normalizing and scaling, followed by splitting the datasets for training, validation, and testing. The second phase focuses on fault detection and classification, which includes training, validating, and testing different models such as LSTM, Stacked-LSTM, Bi-LSTM, and ALSTM. The final phase consists of plotting and evaluating the results. A flowchart illustrating this methodology is shown in Figure 2.

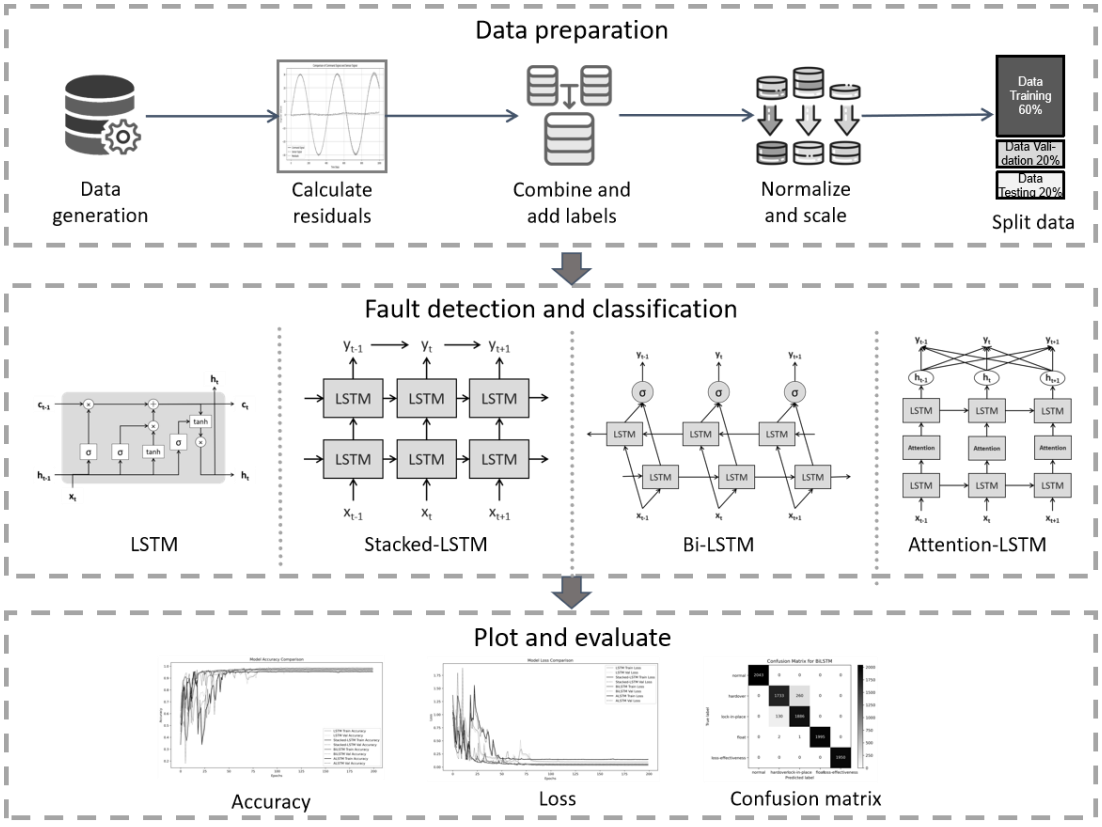

Figure 2 Flowchart of research methodology

2.1. Data Preparation

Datasets used in this research are synthetic time series data which are generated as command signal and sensor signal. A sinusoidal command signal in T time steps is generated and shifted for each sample of i as shown in Eq. 1 to capture every position of command signal within range. The signal was then scaled to represent the range of $\pm 30^{\circ}$ actuator deflection angle as in Eq. 2. The number of samples signal generated are ten thousand samples. Figure 3 depicts ten samples of command signal generated from sample 0 to sample 450 with interval 50 to show the shape of the generated signals.

$$x_i(t) = \sin\left(\frac{\pi t}{T} + i\frac{\pi}{T}\right) \quad (1)$$

$$c_i(t) = 30\big(x_i(t)\big) - 30 \quad (2)$$

$$s_i(t) = x_i(t - delay) + N(0, \sigma^2) \quad (3)$$

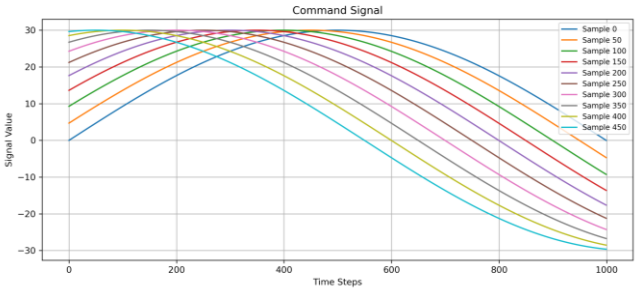

Figure 3 Sample of generated command signals

The sensor signal was created by adding a Gaussian noise of $N(0, \sigma^2)$. Due to the sensor response time, delayed response of the sensor was also added. The final sensor signal can be expressed in Eq. 3. The sensor signal would be modified to simulate various faults, as performed by Alwi et al. (2011) and Bošković et al. (2003), where they classified four fault types. The first is a hard-over fault, where the sensor signal abruptly jumps to either the minimum or maximum value $\pm 30^{\circ}$ from the fault start time. The second is a lock-in-place fault, where the sensor signal becomes fixed at a specific value when the fault occurs. The third is a float fault, in which random deviation values d_j, within the range of $\pm 30^{\circ}$, were added to the sensor signal starting from the fault initiation. Finally, there is the loss of effectiveness fault, where the sensor signal is scaled by a factor of i after the fault start point, simulating reduced sensor responsiveness. As a result, the sensor signal is altered according to Eq. 4 in the event of a fault. The visualization of faulty sensor signal presented in Figure 4. The time when faults occur are the same for every sample, in order to capture the variations of faulty signal. Figure 5 depicts the sensor signal generated for lock-in-place faults.

$$s_i(t) = \begin{cases} (s_i)_{\min} \text{ or } (s_i)_{\max}, & \text{for } t \geq t_{fault}, \quad \text{hard} - \text{over} \\ s_i(t_{fault}), & \text{for } t \geq t_{fault}, \quad \text{lock in place} \\ s_i(t) + d_j, & \text{for interval } j, \quad \text{float} \\ \varepsilon_i s_i(t), & \text{for } t \geq t_{fault}, \quad \text{loss effectiveness} \end{cases} \quad (4)$$

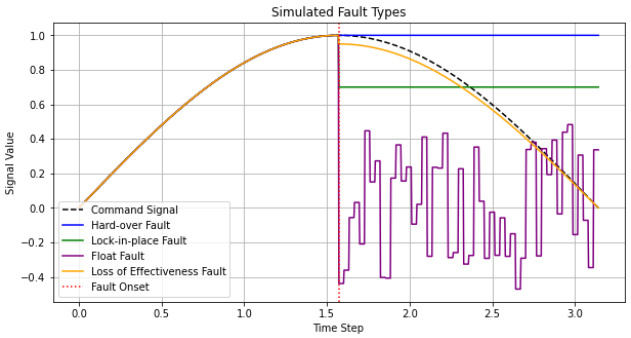

Figure 4 Visualization of faulty sensor signal

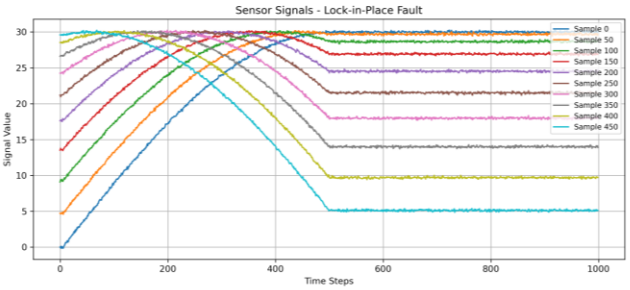

Figure 5 Sample of sensor signals of lock-in-place fault

Faults are detected using residual data which was gathered from the differences between command signal and sensor signal as performed by Ossmann (2023). The residuals data was then changed to an absolute value as in Eq. 5. Sample of residual signal of lock-in-place are illustrated in Figure 6. All residuals from nominal condition and fault condition were combined and then labeled according to the type of fault. The data were labeled as follows:

0 : normal condition

1 : hardover

2 : lock-in-place

3 : float

4 : loss effectiveness

Dataset was split into 60% data training, 20% for validation, and 20% for testing. The dataset is publicly shared in https://www.kaggle.com/datasets/muhammadfjr/simulated-actuator-fault.

$$residual(t) = |s_i(t) - c_i(t)| \quad (5)$$

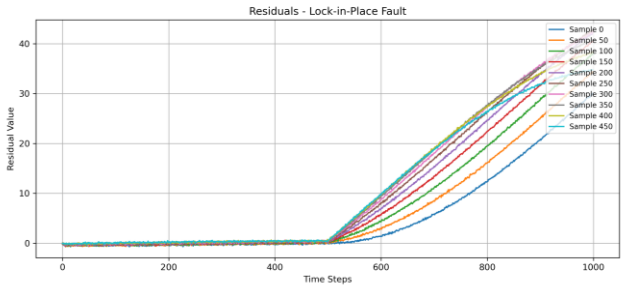

Figure 6 Sample of residual signals of lock-in-place

2.2. Fault Detection and Classification

This section outlines the fault detection and classification process using four different models: LSTM, Stacked-LSTM, Bi-LSTM, and ALSTM. Fault detection is performed by classifying residuals, which are computed as the difference between the command and sensor signals. These residuals quantify deviations caused by faults and serve as fixed input features across all model variants. The overall process is illustrated in Figure 7.

The model begins with an input layer that receives the time-series residual data. This is followed by the LSTM variant layer, which may implement one of the four architectures: LSTM, Stacked-LSTM, Bi-LSTM, or ALSTM. Each variant consists of one or more LSTM cells that process input data sequentially. These cells retain and update their internal memory at each time step, enabling the model to learn both short- and long-term dependencies essential for distinguishing among different fault types.

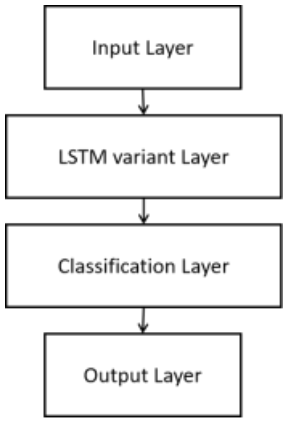

Figure 7 Fault detection and classification process

To improve generalization and mitigate overfitting, a dropout layer is applied after each LSTM cell across all variants. During training, dropuout randomly deactivates a subset of neurons by setting their values to zero, ensuring that the model does not overly rely on specific neurons. However, during inference, all neurons are used without dropout to maximize prediction accuracy.

The final layer is the classification layer, where the model assigns probabilities to each of the five fault classes using the softmax activation function. Softmax transforms raw output scores (logits) into a probability distribution, ensuring that each output value lies between 0 and 1 and that the sum of all class probabilities equals 1. For a 5-class classification problem, the softmax function is defined as shown in Eq. 6:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{5} e^{z_j}} \tag{6}$$

Where p_i represents the probability of class i, and z_i is the raw output score (logit) before applying softmax. The class with the highest probability is selected as the final prediction.

2.2.1. LSTM

LSTM networks, introduced by Hochreiter and Schmidhuber (1997) are a type of recurrent neural network (RNN) specifically designed to address the vanishing and exploding gradient problems that traditional RNNs face. LSTMs are particularly useful for time series data where long-term dependencies need to be captured, which is critical in fault detection as faults may not manifest immediately after they occur.

The LSTM cell, as shown in Figure 8, contains several key components that control the flow of information: the forget gate, input gate, and output gate. These gates regulate how much information is retained or discarded over time. In fault detection, this is especially important as the model must remember both the normal operating conditions and the points where faults begin to emerge in the time series data.

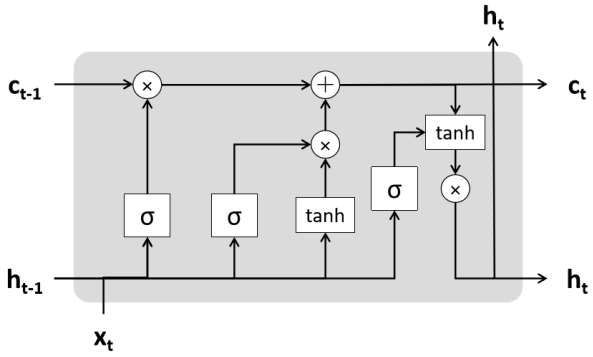

Figure 8 LSTM networks

The equations for LSTM cell operations include:

- Forget Gate (f_t): This gate determines what information to discard from the cell state. It is computed as in Eq. 7.

$$f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \tag{7}$$

where W_f is the weight matrix for the forget gate, b_f is the bias, h_{t-1} is the previous hidden state, and x_t is the

current input. The output of the forget gate is a value between 0 and 1, indicating how much of the past information to retain.

- Input Gate (i_t): This gate decides what new information to add to the cell state. It is computed as shown in Eq. 8:

$$i_t = \sigma(W_i[h_{t-1}, x_t]) + b_i \quad (8)$$

Additionally, candidate values ($\tilde{c}_t$) to be added to the cell state are generated as shown in Eq. 9:

$$\tilde{c}_t = \tanh(W_c[h_{t-1}, x_t]) + b_c \quad (9)$$

where W_i and W_c are weight matrices, and b_i and b_c are biases for the input gate and candidate values, respectively.

- Cell State Update (c_t): The cell state is updated by combining the old state modulated by the forget gate and the new candidate values modulated by the input gate, as shown in Eq. 10:

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t \quad (10)$$

- Output Gate (o_t): This gate determines which part of the cell state to output as the hidden state. It is computed as shown in Eq. 11:

$$o_t = \sigma(W_0[h_{t-1}, x_t]) + b_o \quad (11)$$

Finally, the hidden state (h_t) is calculated as shown in Eq. 12:

$$h_t = o_t \times \tanh(c_t) \quad (12)$$

2.2.2. Stacked-LSTM

Stacked LSTM refers to an architecture in which multiple LSTM layers are placed on top of one another. This deepens the network, enabling it to learn more intricate patterns in sequential data. By stacking multiple LSTM layers, the model can capture hierarchical temporal representations, where each layer abstracts different levels of features from the sequence. This structure allows the model to gain a more comprehensive understanding of the data, which is particularly useful when working with complex sequences.

In a standard single-layer LSTM, the output from the LSTM layer is usually passed directly to a dense (fully connected) layer or another type of processing layer. In contrast, the stacked LSTM architecture, as depicted in Figure 9, involves multiple LSTM layers. Here, the output of each LSTM layer is used as the input for the next layer in the stack. Each LSTM layer processes the input sequence sequentially, passing its hidden state to the following layer. This design enables the network to learn increasingly abstract and higher-level features as it moves through the layers. The additional LSTM layers increase the model's capacity to learn and represent complex patterns, which is especially advantageous when working with large, complicated datasets.

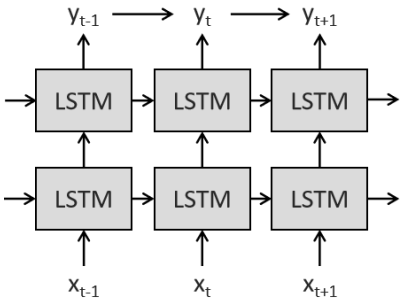

Figure 9 Stacked-LSTM

A typical stacked LSTM architecture begins with an input layer that provides the input sequence data to the first LSTM layer. The first LSTM layer processes this input and produces a hidden state sequence, which is then fed into the subsequent LSTM layers. Each layer in the stack generates a new hidden state sequence based on the input from the previous layer. After the final LSTM layer, the resulting output is typically passed to a dense layer or other types of layers to generate the final prediction.

2.2.3. Bidirectional-LSTM

Bidirectional LSTM (Bi-LSTM) was introduced by Schuster and Pawali (1997) in their paper, "Bidirectional Recurrent Neural Networks." Unlike standard LSTM networks, which process data sequentially in one direction (typically from the first to the last time step), Bi-LSTM processes data in both forward and backward directions. This enables the model to capture patterns from both past and future time steps simultaneously, providing a more holistic understanding of the sequence data. In certain tasks, predictions at a specific time step can benefit from information about both preceding and succeeding time steps, making Bi-LSTM particularly advantageous.

The architecture of a Bidirectional LSTM, as depicted in Figure 10, enhances traditional LSTM by employing two separate layers: first that processes the input sequence in the forward direction (from first to last) and second that processes it in reverse (from last to first). At each time step, the outputs of these forward and backward layers are combined, usually through concatenation, to form a final output that contains information from both temporal directions. This combined output is more robust and informative than using a single direction, making it highly useful in tasks that require comprehensive sequence modeling. After concatenation, the outputs are passed to a dense layer or other processing layers to generate the final prediction.

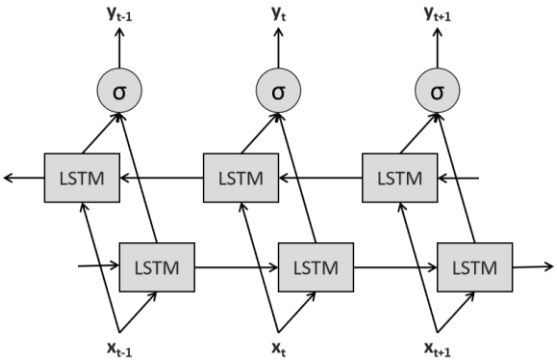

Figure 10 Bidirectional LSTM

2.2.4. Attention-LSTM

Attention-based LSTM (ALSTM) combines the long-term memory capabilities of LSTM with an attention mechanism that enables the model to focus on the most relevant parts of the input sequence. The attention mechanism, first introduced by (Bahdanau, 2014), computes relevance scores for each time step in the sequence, allowing the model to weigh certain time steps more heavily than others when making predictions.

In fault detection, the attention mechanism is particularly useful because not all-time steps contribute equally to identifying a fault. For instance, the time steps immediately before and after a fault occurs are often more informative than earlier or later steps. By using attention, the ALSTM model can focus on these critical time steps, improving its ability to detect faults.

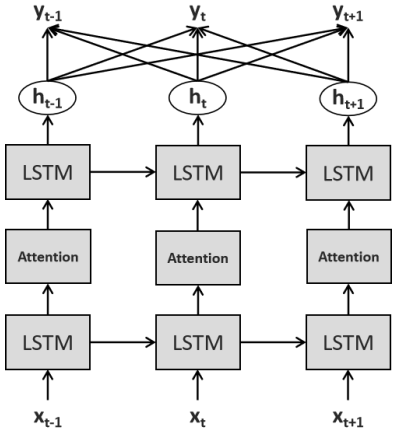

Figure 11 Attention-LSTM

As illustrated in Figure 11, the ALSTM architecture consists of an LSTM layer that processes the input sequence to generate hidden states, followed by an attention layer that computes a relevance score for each hidden state. These scores are used to compute a weighted sum of the hidden states, which emphasizes the most relevant time steps. The context vector generated by this process is then used to make the final prediction.

2.3. Model evaluation

All models were trained for 200 epochs with a batch size of 64, striking a balance between extensive learning and computational efficiency. Each model's layers consist of 50 units, ensuring consistent architecture across all approaches. Dropout layers with a dropout rate of 0.25 are included to mitigate overfitting by randomly deactivating a fraction of neurons during training. The Adam optimizer is employed for efficient and adaptive weight updates, enhancing convergence. Categorical cross-entropy loss is used as the objective function to quantify the discrepancy between predicted and true class labels. The output layer in each model utilizes the Softmax activation function, enabling multi-class classification across the five predefined fault categories.

2.3.1. Categorical Cross-Entropy Loss

The categorical cross-entropy loss is widely used for multi-class classification problems, where the target variable can belong to one of several classes. In this context, each class is assigned a one-hot encoded vector, where the true class is represented as 1, and all other classes are 0. The categorical cross-entropy loss for a given training example is written in Eq. 13, where C is the number of classes. y_i is the true label, which is 1 for the correct class and 0 for others (one-hot encoded).

$$Loss = -\sum_{i=1}^{C} y_i \log(p_i) \tag{13}$$

In the context of fault classification, as applied in this work, five classes are defined: normal, hard-over, lock-in-place, float, and loss-effectiveness. The model outputs a probability distribution for each class using Softmax, and categorical cross-entropy loss ensures that the highest probability is assigned to the correct class.

2.3.2. Accuracy Metric

The accuracy metric measures the proportion of correct predictions the model makes over the entire datasets. It's a very intuitive way to evaluate the performance of a classifier, though it might not always tell the full story, especially with imbalanced datasets. Accuracy is calculated as in Eq. 14. For a multi-class classification problem, accuracy simply counts how many times the predicted class matches the true class. If the predicted class matches the true class, that is counted as a correct prediction.

$$Accuracy = \frac{Correct\ Predictions}{All\ Predictions} \tag{14}$$

2.3.3. Confusion matrix

A confusion matrix is a fundamental tool in machine learning for evaluating the performance of classification models. It provides a detailed breakdown of the model's predictions compared to the actual outcomes, allowing for the computation of various performance metrics. The confusion matrix comprises four elements: true positive (TP) means the model correctly predicts the positive class. True negative (TN) means the model correctly predicts the negative class. False positive (FP) means the model incorrectly predicts the positive class. False negative (FN) means the model incorrectly predicts the negative class. The confusion matrix is depicted in Figure 12.

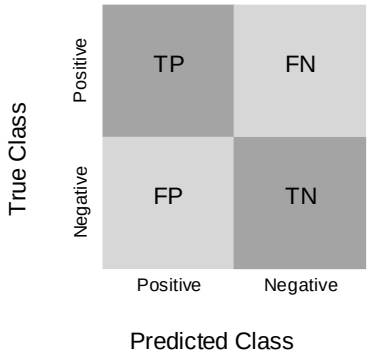

Figure 12 Confusion matrix

As mentioned earlier, the confusion matrix can be used to calculate various performance metrics such as Precision, Recall, and F1-Score. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations, as shown in Eq. 15. Recall is the ratio of correctly predicted positive observations to all actual positive observations, as shown in Eq. 16. Lastly, the F1-Score is the harmonic mean of Precision and Recall, as shown in Eq. 17.

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1 - Score = 2\frac{Precision.\,Recall}{Precision + Recall} \tag{17}$$

3. RESULTS AND DISCUSSION

The models were trained using a computer with the following specifications: a 13th Gen Intel(R) Core™ i7-13700 processor clocked at 2.10 GHz, 64.0 GB of RAM, and a 64-bit operating system. Four different LSTM variants were trained, and the corresponding training times are summarized in Table 1. The models were trained and validated for a total of 200 epochs.

For the basic LSTM model, the total training time was 20,537 seconds, with a mean epoch time of 103 seconds. The Stacked LSTM model required a total of 127,530 seconds to complete training, resulting in a mean epoch time of 638 seconds. The Bi-LSTM model, known for its capability to capture both past and future context, exhibited the longest training duration, requiring 237,628 seconds to finish training, with a mean epoch time of 1,188 seconds. This indicates that the Bi-LSTM model took more than ten times longer to train compared to the basic LSTM model, likely due to its doubled parameter size and complexity. Finally, the ALSTM model took 139,118 seconds to complete training, with a mean epoch time of 696 seconds.

Algorithm	Training time	
	Mean epoch (s)	Total (s)
LSTM	103	20,537
Stacked-LSTM	638	127,530
Bi-LSTM	1,188	237,628
ALSTM	696	139,118

Table 1. Summary of total training time and mean epoch time for different LSTM variants.

3.1. Model loss and accuracy

The model accuracy and loss comparisons for LSTM, stacked-LSTM, Bi-LSTM, and ALSTM models across 200 epochs are depicted in Figures 13 and 14. These figures illustrate the learning process and convergence behavior of each model. In Figure 13, which presents model accuracy, all models show significant improvement over the initial epochs, with varying levels of oscillation before stabilizing. The Bi-LSTM model achieves the highest overall accuracy for both training and validation data, followed closely by the ALSTM and stacked-LSTM models. The LSTM model demonstrates comparatively lower performance, showing less smooth learning behavior and slower convergence, particularly in the early stages of training.

In Figure 14, which presents the loss curves, the Bi-LSTM again shows the best performance, achieving the lowest loss for both training and validation data. The model converges more rapidly compared to the others, with minimal oscillation beyond the 50th epoch. The stacked LSTM and ALSTM models also perform well, with loss values that decrease steadily over time, although they experience occasional spikes, particularly in validation loss. The LSTM model, however, exhibits the highest loss among the four, suggesting that it struggles more with the generalization of the data and experiences greater difficulty in minimizing errors as training progresses.

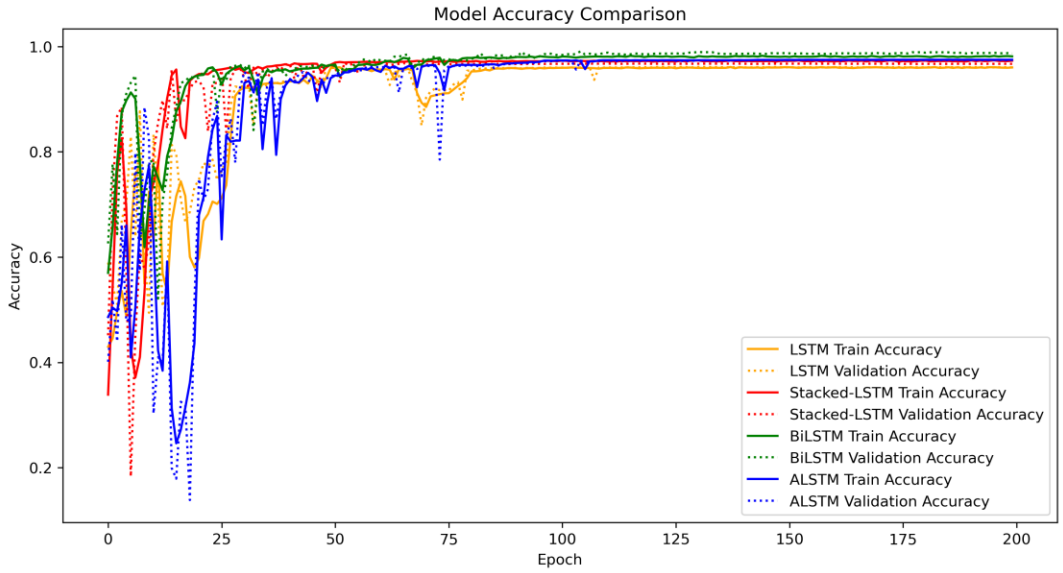

Figure 13 Model accuracy

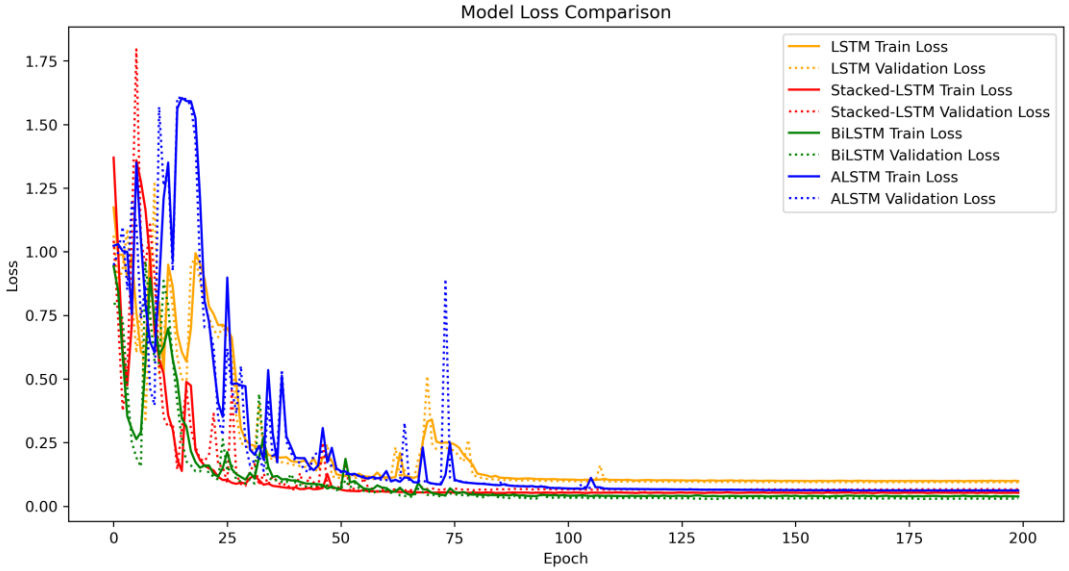

Figure 14 Model loss

The final model accuracy and loss values are summarized in Table 2. The Bi-LSTM model achieved the highest validation accuracy at 98.66% with a corresponding validation loss of 0.0304, outperforming the other models in both accuracy and loss metrics. The ALSTM model follows closely with a validation accuracy of 97.43% and a validation loss of 0.0621, indicating strong performance, particularly in fault detection and classification tasks. The stacked LSTM model also demonstrates good results, achieving a validation accuracy of 96.71% and a validation loss of 0.0665. The LSTM model, while functional, has the lowest performance metrics with a validation accuracy of 96.03% and a validation loss of 0.0940.

Algorithm	**Accuracy**		**Loss**	
	Train	**Valida -tion**	**Train**	**Valida -tion**
LSTM	0.9602	0.9603	0.0990	0.0940
Stacked-LSTM	0.9731	0.9671	0.0526	0.0665
Bi-LSTM	**0.9813**	**0.9866**	**0.0382**	**0.0304**
ALSTM	0.9749	0.9743	0.0619	0.0621

Table 2. Model accuracy and loss during training and validation

3.2. Test evaluation

To evaluate the performance of each algorithm, a confusion matrix was employed to assess the classification outcomes for fault detection. The experiments were conducted on a dataset containing 10,000 samples evenly distributed across four fault classes: normal, loss-effectiveness, hard-over, and float. Each class had approximately 2,000 samples. The test results for each model, including test loss and accuracy, are detailed below.

The results for the LSTM model are presented in Figure 15, where the LSTM achieved a test loss of 0.0841 and a test accuracy of 96.64%, correctly predicted 9,664 out of 10,000 test samples. The model achieved perfect accuracy for normal and loss-effectiveness conditions, with no false predictions. For hard-over faults, out of 1,993 test samples, 1,692 were correctly classified, while 301 were misclassified as lock-in-place faults. In the case of lock-in-place faults, 1,987 out of 2,016 test samples were correctly predicted, with 29 misclassified as hard-over. For float faults, 1,992 out of 1,998 samples were correctly identified, with 6 misclassifications, 1 as hard-over and 5 as lock-in-place.

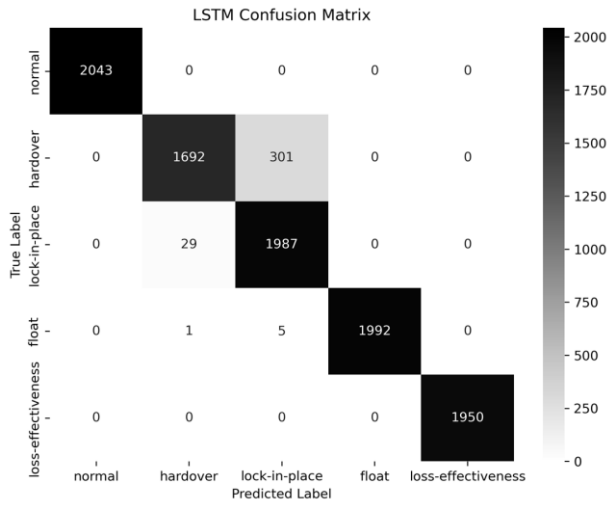

Figure 15 Confusion Matrix of LSTM

The results of the stacked-LSTM model are illustrated in Figure 16, showing an improvement over the standard LSTM model, with a test loss of 0.0549 and a test accuracy of 97.32%. The stacked-LSTM correctly predicted 9,732 out of 10,000 samples. This model achieved perfect accuracy in predicting normal, lock-in-place, and loss-effectiveness conditions. However, for hard-over faults, 1,737 out of 1,993 test samples were correctly identified, while 256 were misclassified as lock-in-place. In the float faults category, 1,986 out of 1,998 samples were correctly predicted, with 12 misclassified as hard-over.

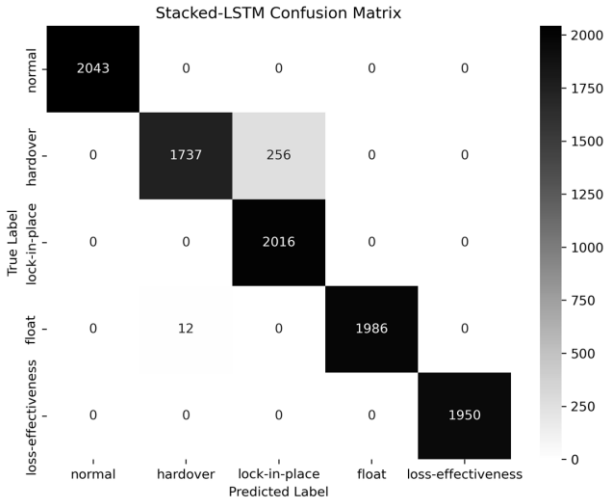

Figure 16 Confusion Matrix of Stacked-LSTM

In Figure 17, the results of the Bi-LSTM model demonstrate superior performance compared to the standard LSTM and stacked-LSTM models. The Bi-LSTM achieved a test loss of 0.0264 and a test accuracy of 98.93%, accurately predicting 9,893 out of 10,000 test samples. It achieved perfect accuracy in predicting normal, float, and loss-effectiveness conditions. For hard-over faults, 1,887 out of 1,993 test samples were correctly classified, with 106 misclassified as lock-in-place. In the case of lock-in-place faults, only 1 sample out of 2,016 was incorrectly classified as hard-over. These results highlight the Bi-LSTM's superior predictive accuracy over the LSTM and stacked-LSTM models.

The ALSTM model's performance, presented in Figure 18, indicates 9,759 correct predictions out of 10,000 samples, with a test loss of 0.0613 and a test accuracy of 97.59%. The ALSTM performing better than both LSTM and stacked-LSTM models but slightly underperforming compared to the Bi-LSTM. The ALSTM achieved perfect accuracy for normal, lock-in-place, and loss-effectiveness conditions. For hard-over faults, 1,754 out of 1,993 test samples were correctly identified, with 239 misclassified as lock-in-place. In the float faults category, 1,996 out of 1,998 samples were correctly predicted, with only 2 samples misclassified as hard-over.

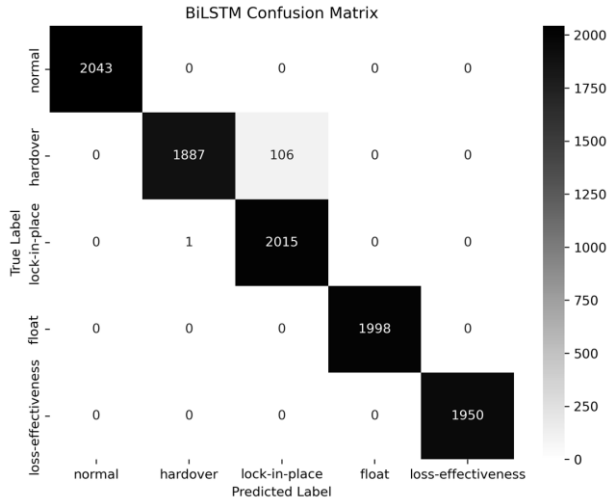

Figure 17 Confusion Matrix of Bidirectional LSTM

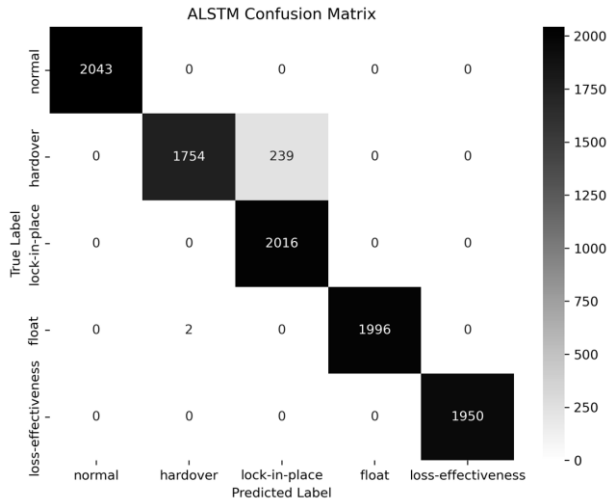

Figure 18 Confusion Matrix of Attention-LSTM

From the results discussed above, it is evident that all models, including LSTM, stacked-LSTM, Bi-LSTM, and ALSTM, are capable of accurately predicting faults and distinguishing between normal and fault conditions. Each model successfully captured most test samples across the five classes: normal, lock-in-place, hard-over, float, and loss-effectiveness. However, a recurring challenge across all models was the misclassification of hard-over faults. Specifically, hard-over faults were frequently misclassified as lock-in-place, indicating that the models struggled to distinguish between these two fault types. This misclassification pattern suggests that the feature characteristics of hard-over and lock-in-place faults may share some similarities in the dataset, causing confusion for the models.

Despite this challenge, all models demonstrated strong performance in correctly predicting the other fault categories, particularly lock-in-place, float, and loss-effectiveness conditions. The Bi-LSTM model showed the highest accuracy, with significantly fewer misclassifications compared to the other models, especially in the hard-over and float fault categories. This highlights the Bi-LSTM's ability to capture more nuanced temporal dependencies in the data, allowing it to differentiate between fault types with greater precision.

To further evaluate the models, performance metrics such as precision, recall, and F1-score were calculated based on the correct (TP, TN) and incorrect predictions (FP, FN) for each fault type: normal (N), hard-over (H), lock-in-place (LP), float (F), and loss-effectiveness (LE). The results of these metrics are shown in Table 3, and they are visually represented in the bar charts (Figures 19, 20, and 21). The precision values, depicted in Figure 19, show that all models perform well in classifying normal, float, and loss-effectiveness faults, achieving perfect scores. However, the lock-in-place categories present a challenge, with precision dropping in these categories. The Bi-LSTM model exhibits the highest precision across all fault types, outperforming the other models, especially in the lock-in-place category.

Model	**Precision**					**Recall**					**F1-score**				
	N	**H**	**LP**	**F**	**LE**	**N**	**H**	**LP**	**F**	**LE**	**N**	**H**	**LP**	**F**	**LE**
LSTM	1.00	0.98	0.87	1.00	1.00	1.00	0.85	0.99	**1.00**	1.00	1.00	0.91	0.92	1.00	1.00
Stacked-LSTM	1.00	0.99	0.89	1.00	1.00	1.00	0.87	**1.00**	0.99	1.00	1.00	0.93	0.94	1.00	1.00
Bi-LSTM	1.00	**1.00**	**0.95**	1.00	1.00	1.00	**0.95**	**1.00**	**1.00**	1.00	1.00	**0.97**	**0.97**	1.00	1.00
ALSTM	1.00	**1.00**	0.89	1.00	1.00	1.00	0.88	**1.00**	**1.00**	1.00	1.00	0.94	0.94	1.00	1.00

Table 3. Summary of model precision, recall, and F1-score for each model.

The recall values similarly indicate high performance across all models for the normal, float, and loss-effectiveness fault types as shown in Figure 20. The recall for hard-over faults demonstrates a notable difference between models, with the Bi-LSTM model again outperforming the others. The F1-scores as presented in Figure 21, reflect a balance between

precision and recall, confirming that the Bi-LSTM model provides the best overall performance across all fault types. In contrast, the LSTM and stacked-LSTM models exhibit weaker performance, particularly for hard-over and lock-in-place faults, where their F1-scores are significantly lower.

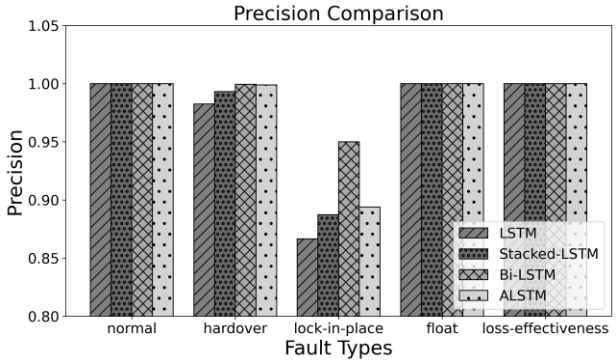

Figure 19 Precision comparison across models

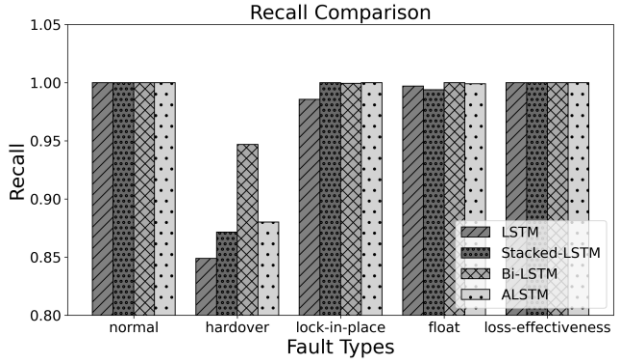

Figure 20 Recall comparison across models

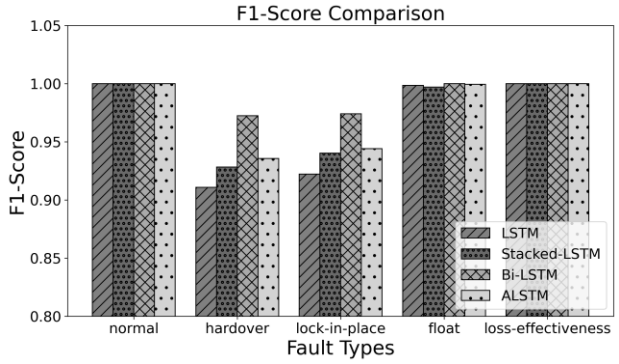

Figure 21 F1-score comparison across models

4. CONCLUSION

This study demonstrated the effectiveness of LSTM-based models in detecting and classifying actuator faults in time-series format. Among the tested models, the Bi-LSTM model demonstrated highest performance, achieving a test accuracy of 98.93% with minimal misclassification across various fault types. Its superior ability to capture temporal dependencies allowed for precise differentiation between challenging faults, such as hard-over and lock-in-place. However, Bi-LSTM also exhibited the longest computation time, which should be considered for real-time deployment. The ALSTM model also performed well, with the accuracy of 97.59% with slightly higher test loss, followed by Stacked-LSTM and the standard LSTM model. Overall, all models exhibit strong fault detection performance, particularly for normal, float, and loss-of-effectiveness conditions, though distinguishing between hard-over and lock-in-place faults remain a recurring challenge.

REFERENCES

Ahmad, M.W., Akram, M.U., Mohsan, M.M., Saghar, K., Ahmad, R., Butt, W.H. (2024). Transformer-based sensor failure prediction and classification framework for UAVs. *Expert Systems with Applications*. Volume 248, 15 August 2024. https://doi.org/10.1016/j.eswa.2024.123415

Alwi, H., Edwards, C., & Tan, C. P. (2011). Fault detection and fault-tolerant control using sliding modes. Springer. https://doi.org/10.1007/978-0-85729-650-4

Bahdanau, D. (2014). Neural machine translation by jointly learning to align and translate. *arXiv Preprint arXiv:1409.0473*.

Bharatheedasan, K., Maity, T., Kumaraswamidhas, L. A., & Durairaj, M. (2023). An intelligent of fault diagnosis and predicting remaining useful life of rolling bearings based on convolutional neural network with bidirectional LSTM. *Sādhanā*, *48*(3), 131. https://doi.org/10.1007/s12046-023-02169-1

Boni, P., Mazzoleni, M., Previdi, F. (2024). Robust data-driven design of a jamming detection filter for airborne electromechanical actuators. *European Journal of Control*. *75*. 100926. https://doi.org/10.1016/j.ejcon.2023.100926

Borré, A., Seman, L. O., Camponogara, E., Stefenon, S. F., Mariani, V. C., & Coelho, L. dos S. (2023). Machine Fault Detection Using a Hybrid CNN-LSTM Attention-Based Model. *Sensors*, *23*(9). https://doi.org/10.3390/s23094512

Bošković, J. D., & Mehra, R. K. (2003). Failure Detection, Identification and Reconfiguration in Flight Control. In F. Caccavale & L. Villani (Eds.), *Fault Diagnosis and Fault Tolerance for Mechatronic Systems:Recent Advances* (pp. 129–167). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45737-2_5

Cieslak, J., Zolghadri, A., Goupil, P., Dayre, R. (2016). A Comparative Study of Three Differentiation Schemes for the Detection of Runaway Faults in Aircraft Control Surfaces. *IFAC-PapersOnLine*. 49(17). 70-75. https://doi.org/10.1016/j.ifacol.2016.09.013

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *Association for Computational Linguistics*. Doi: 10.3115/v1/D14-1179. https://aclanthology.org/D14-1179/

Gao, Y., Kim, C. H., & Kim, J.-M. (2021). A Novel Hybrid Deep Learning Method for Fault Diagnosis of Rotating Machinery Based on Extended WDCNN and Long Short-Term Memory. *Sensors*, 21(19), 6614. https://doi.org/10.3390/s21196614

Grehan, J., Ignatyev, D., Zolotas, A. (2023). Fault Detection in Aircraft Flight Control Actuators Using Support Vector Machines. *Machines*, 11, 211. https://doi.org/10.3390/machines11020211

Giral, F., Gómez, I., Vinuesa, R., Clainche, S.L. (2024). Transformer-Based Fault-Tolerant Control for Fixed-Wing UAVs Using Knowledge Distillation and In-Context Adaptation. *arXiv Preprint arXiv:2411.02975*.

He, C., Yasenjiang, J., Lv, L., Xu, L., & Lan, Z. (2024). Gearbox Fault Diagnosis Based on MSCNN-LSTM-CBAM-SE. *Sensors*, *24*(14). https://doi.org/10.3390/s24144682

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hu, J., Zhu, M., Zheng, Z., & Chen, T. (2024). A data-based fault detection scheme for the stratospheric airship control system. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *238*(5), 803–819. https://doi.org/10.1177/09596518231209542

Huang, S., Rong, L., Chang, X., Wang, Z., Yuan, Z., Wei, C. (2021). BLSTM-Based Adaptive Finite-Time Output-Constrained Control for a Class of AUSs with Dynamic Disturbances and Actuator Faults. *Mathematical Problems in Engineering*. https://doi.org/10.1155/2021/2221495

Keshun, Y., Puzhou, W., & Yingkui, G. (2024). Toward Efficient and Interpretative Rolling Bearing Fault Diagnosis via Quadratic Neural Network With Bi-LSTM. *IEEE Internet of Things Journal*, *11*(13), 23002–23019. https://doi.org/10.1109/JIOT.2024.3377731

Lee, D., Choo, H., & Jeong, J. (2024). GCN-Based LSTM Autoencoder with Self-Attention for Bearing Fault Diagnosis. *Sensors*, *24*(15). https://doi.org/10.3390/s24154855

Lei, J., Liu, C., & Jiang, D. (2019). Fault diagnosis of wind turbine based on Long Short-term memory networks. *Renewable Energy*, *133*, 422–432. https://doi.org/10.1016/j.renene.2018.10.031

Li, Y., Liu, B., Jia, Z., Liu, Z. (2023). Transformer-based fault diagnosis method for fixed-wing UAV elevator. *Maintainability and Safety (ICRMS)*. Urumuqi, China. 1018-1023. doi: 10.1109/ICRMS59672.2023.00178.

Ma, L., Guo, J., Wang, S., Wang, J. (2021). Multi-Source Sensor Fault Diagnosis Method Based on Improved CNN-GRU Network [J]. *Transactions of Beijing Institute of Technology*, 2021, 41(12): 1245-1252. DOI: 10.15918/j.tbit1001-0645.2020.183

Masalimov, K., Muslimov, T., Munasypov, R. (2022). Real-Time Monitoring of Parameters and Diagnostics of the Technical Condition of Small Unmanned Aerial Vehicle's (UAV) Units Based on Deep BiGRU-CNN Models. *Drones*, 6(11), 368. https://doi.org/10.3390/drones6110368

Mao, N., Xu, J., Li, J., He, H. (2021). A LSTM-RNN-Based Fiber Optic Gyroscope Drift Compensation. *Mathematical Problems in Engineering*. *1-10*. https://doi.org/10.1155/2021/1636001

Mohsan, S. A. H., Khan, M. A., Noor, F., Ullah, I., & Alsharif, M. H. (2022). Towards the Unmanned Aerial Vehicles (UAVs): A Comprehensive Review. *Drones*, *6*(6). https://doi.org/10.3390/drones6060147

Ossmann, D., Weiser, C., Biertumpfel, F., Pfifer, H., Cieslak, J. (2023). Flight Control Reconfiguration in Oscillatory Failure Case Scenarios. *IFAC-PapersOnLine*. *56*(2). 396-401. https://doi.org/10.1016/j.ifacol.2023.10.1600

Ossmann, D., & van der Linden, F. L. J. (2015). Advanced sensor fault detection and isolation for electro-mechanical flight actuators. *2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 1–8. https://doi.org/10.1109/AHS.2015.7231162

Pang, Y., Kendall, A. P., & Clarke, J. P. (2025). The Reliability of Remotely Piloted Aircraft System Performance under Communication Loss and Latency Uncertainties. arXiv preprint arXiv:2501.07743. https://doi.org/10.48550/arXiv.2501.07743

Park, W., Lee, S., Song, J. (2009). Fault detection and isolation of DURUMI-II using similarity measure. *Journal of Mechanical Science and Technology*. 23(2). 302-310. https://doi.org/10.1007/s12206-009-0107-z

Peng, Z., Sun, Z., Chen, J., Ping, Z., Dong, K., Li, J., Fu, Y., Zio, E. (2022). A Fault Diagnosis Approach for Electromechanical Actuators with Simulating Model under Small Experimental Data Sample Condition. *Actuators*, 11(3), 66. https://doi.org/10.3390/act11030066

Puchalski, R., Bondyra, A., Giernacki, W., & Zhang, Y. (2022). Actuator fault detection and isolation system for multirotor unmanned aerial vehicles. *2022 26th International Conference on Methods and Models in Automation and Robotics (MMAR)*, 364–369. https://doi.org/10.1109/MMAR55195.2022.9874283

Puchalski, R., & Giernacki, W. (2022). UAV Fault Detection Methods, State-of-the-Art. *Drones*, *6*(11). https://doi.org/10.3390/drones6110330

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., & Cottrell, G. W. (2017). A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. *Proceedings of the Twenty-Sixth International Joint Conference on*

Artificial Intelligence, IJCAI-17, 2627–2633. https://doi.org/10.24963/ijcai.2017/366

Reynoso-Meza, G., Ribeiro, V. H. A., Filho, M. F., de Abreu, B. Z. (2023). Time Stacking Decision Tree for Oscillatory Failure Case Detection in a Flight Control System. *IFAC-PapersOnLine*. *56*(2). 354-359. https://doi.org/10.1016/j.ifacol.2023.10.1593

Rudin, K., Mosimann, L., Ducard, G. J. J., & Siegwart, R. Y. (2015). Robust Actuator Fault-tolerant Control using DK-Iteration: Theory and Application to UAS. *IFAC-PapersOnLine*, *48*(21), 392–397. https://doi.org/10.1016/j.ifacol.2015.09.558

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, *45*(11), 2673–2681. https://doi.org/10.1109/78.650093

Simani, S., Fantuzzi, C., & Patton, R. J. (2003). Model-based fault diagnosis in dynamic systems using identification techniques. *International Journal of Robust and Nonlinear Control*, XV, 282. https://doi.org/10.1007/978-1-4471-3829-7

Singh, Y., Bagri, K., Jayakumar, A., Rizzoni, G. (2023). Fault Diagnostics for Oscillatory Failure Case in Aircraft Elevator Servos. *IFAC-PapersOnLine*. 56(2). 408-415. https://doi.org/10.1016/j.ifacol.2023.10.1602

Su, Z., Qi, B., Wang, B., Liu, D. (2024). A Transformer-Based Condition Monitoring Method for UAV. *Lecture Notes in Electrical Engineering*, vol 1173. Springer, Singapore. https://doi.org/10.1007/978-981-97-1087-4_45

Sun, J., Ren, H., Duan, Y., Yang, X., Wang, D., & Tang, H. (2024). Fusion of Multi-Layer Attention Mechanisms and CNN-LSTM for Fault Prediction in Marine Diesel Engines. *Journal of Marine Science and Engineering*, *12*(6). https://doi.org/10.3390/jmse12060990

Tao, X., Guo, Y., Zhao, W., Zhou, Q., Xu, K. (2023). Fault detection and isolation of electromechanical actuator based on SAE-BiLSTM. *Journal of Physics: Conference Series*. 2472(1). 012031. https://doi.org/10.1088/1742-6596/2472/1/012031

Telli, K., Kraa, O., Himeur, Y., Ouamane, A., Boumehraz, M., Atalla, S., & Mansoor, W. (2023). A Comprehensive Review of Recent Research Trends on Unmanned Aerial Vehicles (UAVs). *Systems*, *11*(8). https://doi.org/10.3390/systems11080400

Vanga, J., Ranimekhala, D. P., Jonnala, S., Jamalapuram, J., Gutta, B., Gampa, S. R., & Alluri, A. (2023). Fault classification of three phase induction motors using Bi-LSTM networks. *Journal of Electrical Systems and Information Technology*, *10*(1), 28. https://doi.org/10.1186/s43067-023-00098-x

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., Polosukhin, I. (2017). Attention Is All You Need. 10.48550/arXiv.1706.03762.

Venkataraman, R., Bauer, P., Sailer, P., Vanek, B. (2019). Comparison of Fault Detection and Isolation Methods for A Small Unmanned Aircraft. *Control Engineering Practice*. 84. 365-376. https://doi.org/10.1016/j.conengprac.2018.12.002

Wang, L., Zhao, W., Liu, Z., Dang, Q., Zou, X., Wang, K. (2023). Incipient Fault Detection and Reconstruction Using an Adaptive Sliding-Mode Observer for the Actuators of Fixed-Wing Aircraft. *Aerospace*, 10, 422. https://doi.org/10.3390/aerospace10050422

Yoo, J., Song, S., Chang, K., & Baek, J.-G. (2023). Fault detection and diagnosis using two-stage attention-based variational LSTM in electrolytic copper manufacturing process. *The International Journal of Advanced Manufacturing Technology*, *129*(3), 1269–1288. https://doi.org/10.1007/s00170-023-12356-3

Yu, J., Chang, J., Chen, W., Cieslak, J., Ossmann, D. (2024). *IFAC-PapersOnLine*. *58*(4). 115-120. https://doi.org/10.1016/j.ifacol.2024.07.203

Yu, L., Qu, J., Gao, F., & Tian, Y. (2019). A Novel Hierarchical Algorithm for Bearing Fault Diagnosis Based on Stacked LSTM. *Shock and Vibration*, *2019*(1), 2756284. https://doi.org/10.1155/2019/2756284

Zhang, Q., Zhang, J., Zou, J., & Fan, S. (2020). A Novel Fault Diagnosis Method based on Stacked LSTM. *IFAC-PapersOnLine*, *53*(2), 790–795. https://doi.org/10.1016/j.ifacol.2020.12.832

Zolghadri, A., Gheorghe, A., Cieslak, J., Henry, D., Goupil, P., Dayre, R., Le Berre, H. (2011). A Model-based Solution of Robust and Early Detection of Control Surface Runaways. *SAE International Journal of Aerospace*. 4(2). 2011-01-2803. https://doi.org/10.4271/2011-01-2803

BIOGRAPHIES

Muhammad Fajar holds a Master of Engineering degree in Aeronautics and Astronautics from Bandung Institute of Technology, Indonesia in 2017. He also received his B.Eng. in Informatics Engineering from IT Telkom (Telkom University), Indonesia, in 2007. He is currently a research engineer at the Research Center of Aeronautics Technology, National Research and Innovation Agency, Indonesia. He has experience in wind tunnel testing and towing tank testing for seaplanes. His research interests are machine learning related to engineering applications and automatic flight control.

Teuku Mohd Ichwanul Hakim is a associate researcher at BRIN, specializing in aerodynamics, flight mechanics, and flight simulation, with 18 years of experience in manned and unmanned aircraft development. Holding a Master's degree in Aerospace Engineering from ITB, his work includes leading the Flight Mechanics Research Group and advancing UAV technologies like the LSA-02 and the amphibious

N219A aircraft. His expertise spans hardware-in-the-loop simulations, flight control systems, and flight simulation.

Adi Wirawan is an associate researcher at the Research Center for Aeronautics Technology, part of Indonesia's National Research and Innovation Agency (BRIN). He specializes in avionics, flight control panels, and the integration of cutting-edge flight instruments, bridging the gap between pilots and advanced flight systems to ensure safer and more efficient aircraft operations. Holding a master's degree in aerospace engineering from the prestigious Bandung Institute of Technology (ITB), earned in 2018, he is passionate about developing innovative solutions for modern aviation challenges. His research covers a broad range of topics, including flight displays, human-machine interfaces, and the design and testing of flight control instruments.

Prasetyo Ardi Probo Suseno is an associate researcher at the Research Center for Aeronautics Technology, National Research and Innovation Agency (BRIN). He received a Master of Engineering degree in Aeronautics and Astronautics from ITB in 2017. His research primarily concerns unmanned aircraft, including aircraft design, structural strength, aerodynamics, and flight mechanics. He also has experience in dynamic systems modelling and simulation.

Arifin Rasyadi Soemaryanto holds a Bachelor of Engineering degree in Aeronautics and Astronautics from Bandung Institute of Technology (ITB), Indonesia in 2011. Currently, he is a researcher at the Research Center of Aeronautics Technology, National Research and Innovation Agency (BRIN), Indonesia. He has experience in aerodynamic design of aerial vehicles and some of the subject multi-discipline, such as hydrodynamics. At the moment, his research interests are on design optimization of aerial vehicles.

Ardanto Mohamad Pramutadi is a junior researcher at the Research Center for Aeronautics Technology, National Research and Innovation Agency with experience in Unmanned Aerial Vehicle design and flight control system testing, hardware design and integration. The author received his bachelor's degree and master's degree from Institut Teknologi Bandung (ITB) in 2011 and 2024 respectively. The author's research focused on air vehicle design, systems engineering, and optimization. The author is currently focusing on research of requirements elicitation with Natural Language Processing and the development of design optimization framework.