FT2025
SWEDISH AEROSPACE AT THE CROSSROADS
The 12th Swedish Aerospace Technology Congress
FT2025 in Stockholm
October 14-15, 2025

Convolutional Neural Network for Detection and Quantification of Pilot-Induced Oscillations

Andre L. A. Paladini[1,2], Daniel Drewiacki[3], Raghu Chaitanya Munjulury[1,4], Petter Krus[1], and Jorge H. Bidinotto[2]

[1]*Department of Management and Engineering – Division of Fluid and Mechatronic Systems, Linköping University, Linköping, Sweden*
E-mail:, andre.paladini@liu.se , petter.krus@liu.se, raghu.munjulury@liu.se
[2]*Aeronautical Engineering Department, University of São Paulo, São Carlos, Brazil*
E-mail:, andrepaladini@usp.br, jhbidi@sc.usp.br
[3]*Embraer S.A., São José dos Campos, Brazil*
E-mail:, daniel.drewiacki@embraer.com.br
[4]*SAAB Aeronautics, Linköping, Sweden*

Abstract

Pilot-induced oscillations (PIO) are a long-standing challenge in aircraft handling, characterized by destabilizing oscillatory behaviour resulting from closed-loop interactions between the pilot and the aircraft. Traditionally, PIO identification and quantification rely on the subjective PIO Rating Scale (PIOR), used by pilots during handling qualities evaluations in aircraft testing and certification. This paper presents a data-driven approach to objectively identify PIO using flight data collected from a fixed-base simulator during a flight test campaign. The data were labelled according to the PIOR scale, preprocessed using the Wavelet Transform, and used to train a Convolutional Neural Network (CNN). This approach enables objective detection and quantification of PIO while maintaining alignment with the pilot-assessed evaluative framework. A k-fold cross-validation strategy yielded an average validation accuracy of 52.8%, with the best-performing model achieving 63.1%. By applying the soft voting ensemble technique, the validation accuracy increased to 84.6%. The overall accuracies are attributed to challenges in classifying low-to-intermediate PIO ratings (PIOR 2 and 3). Despite this limitation, the proposed model shows potential for further improvement and demonstrates promise as a complementary tool in handling qualities evaluations, providing a quantitative counterpart to traditional pilot assessments.

Keywords: Pilot-induced oscillations, Convolutional neural network, Wavelet Transform, Flight simulators

1 Introduction

Pilot-Induced Oscillations (PIO) are a long-recognized challenge in aircraft handling qualities, where undesired and often destabilizing oscillatory behaviour emerges from the closed-loop interaction between the pilot and the aircraft. These oscillations typically arise from a mismatch between pilot control behaviour and aircraft response characteristics, particularly under condition of high gain, time delay, and/or nonlinearities within the pilot-aircraft system.

According to a widely accepted framework established in industry and military research [1, 2], PIO is most fundamentally described as a "sustained or uncontrollable unintentional oscillation resulting from the pilot's efforts to control the aircraft". According to Mitchell and Klyde [3], these oscillations typically occur when the aircraft's response angle is approximately 180 degrees out of phase with the pilot's input. Importantly, a PIO can be convergent, constant-amplitude, or divergent, and may involve just a few cycles or persist throughout a task. Frequency alone does not define a PIO; events have been recorded across a broad spectrum, from low-frequency oscillations near the phugoid mode to higher-frequency phenomena like pitch bobble or roll ratchet.

The severity of a PIO varies widely. Mild forms may be small in amplitude and tolerable in some scenarios, while moderate PIOs disrupt mission performance and often require pilot compensation or corrective action. At the most extreme, severe PIOs render the aircraft uncontrollable during the task,

demanding immediate system modifications and halting further testing [3]. Crucially, PIO is most often symptomatic of deeper issues in aircraft design, operational envelope extension, or control law implementation, and not a result of poor piloting [4].

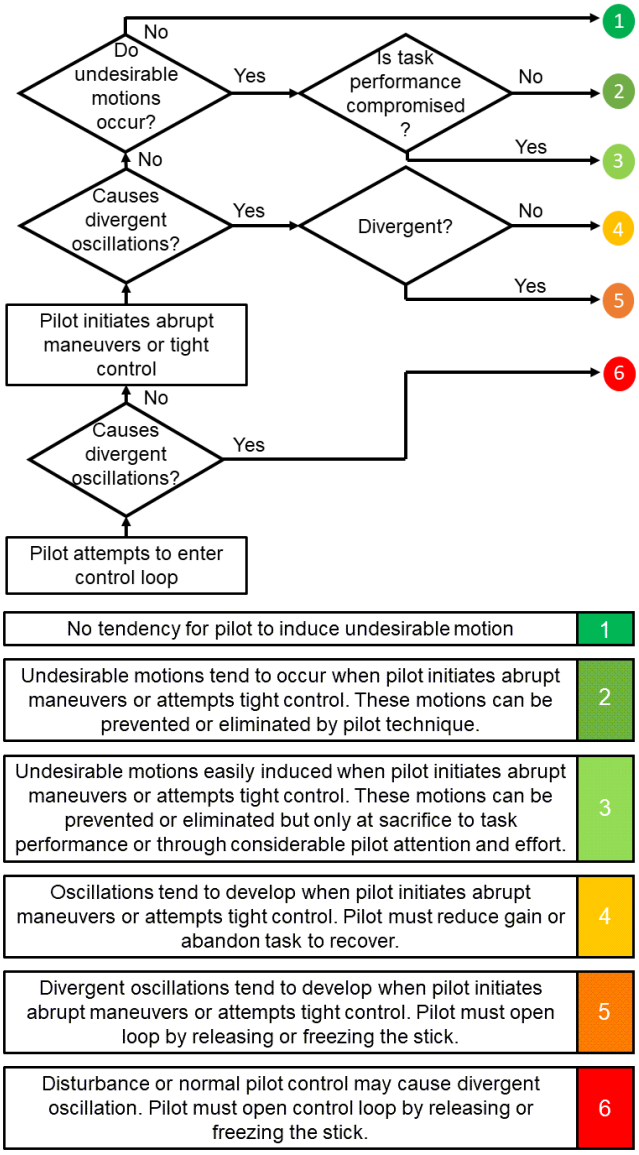

Figure 1: The PIO rating scale (PIOR) [5].

Historically, PIOs have been identified by test pilots through flight tests, simulator evaluations, and aircraft certification procedures, primarily using the PIO Rating Scale (PIOR), as shown in Figure 1. Although the PIOR provides valuable insights, it depends heavily on subjective pilot assessments, introducing variability influenced by factors such as pilot experience, familiarity with the scale, and individual perception and interpretation of the oscillatory levels [5, 6]. This has led to the development of some objective, data-based identification methods, including the Real-time Oscillation Verifier (ROVER) [7] and the Inceptor Peak Power Phase (IPPP) [8] metrics. These techniques aim to detect the onset of PIO by analysing control input and aircraft response data. However, aligning these tools with pilot assessments remains a key challenge due to differing sensitivity and interpretive models. These metrics are yet incomplete and do not provide information about the PIO severity.

In recent years, machine learning has shown growing promise in handling quality research. For example, Xu et al. [9] and Mori and Suzuki [10] have used fuzzy logic and neural networks, respectively, to model pilot behaviour in closed-loop tracking tasks. Specifically in PIO identification, Bruschi et al.[11] proposed a deep neural network trained on handcrafted features derived from flight data and aligned with existing PIO definitions, such as maximum phase difference between pilot input and aircraft response, to classify events according to the PIOR scale.

Convolutional Neural Networks (CNNs) have demonstrated strong potential in analysing time-series data, particularly when the data is converted into "image-like" format (scalograms and spectrograms) using mathematical tools such as the Wavelet Transform or the Short-Time Fourier Transform. This approach has been successfully applied in various domains, including speech emotion recognition [12] and arrhythmia detection in electrocardiogram signals [13].

This paper presents a data-driven approach to PIO identification using flight data obtained from a flight simulator campaign and labelled according to the PIO rating scale. In this method, the flight data is transformed into scalograms via the Wavelet Transform and then processed by a Convolutional Neural Network, which outputs a classification aligned with the PIO rating scale. A key advantage of this approach is its ability to autonomously learn PIO-related features directly from the scalograms, without requiring the inclusion of predefined PIO definitions as part of the network input. The ultimate goal is to develop a tool that reduces dependence on subjective pilot input while maintaining consistency with the PIOR framework, thereby enhancing the accuracy and repeatability of PIO assessments in both development and certification contexts.

2 Data acquisition

To achieve the proposed objectives, flight data must be made available for use by the neural network algorithm responsible for identifying pilot-induced oscillations and quantifying their severity. To this end, a flight simulation campaign was designed to collect sufficient data, generating a dataset to support the neural network training and validation processes, as detailed in the work of Bruschi et al.[11].

Although flight simulator data lacks high fidelity in representing real flight conditions, this approach offers significant operational advantages. Notably, it enables the generation of large volumes of data at a lower cost. Another key benefit of simulations is the ability to easily modify flight conditions and aircraft stability characteristics. The use of flight simulators has become a standard practice in research on PIO and other handling qualities issues, attracting considerable interest from the aerospace industry [14, 15, 6].

$$\dot{x} = Ax + B\delta e \tag{1}$$

$$A = \begin{bmatrix} \frac{X_u}{m} & \frac{X_w}{m} & 0 & -g\cos(\theta_0) \\ \frac{Z_u}{m-Z_{\dot{w}}} & \frac{Z_w}{m-Z_{\dot{w}}} & \frac{Z_q - mu_0}{m-Z_{\dot{w}}} & \frac{-mg\sin(\theta_0)}{m-Z_{\dot{w}}} \\ \frac{1}{I_y}\left[M_u + \frac{M_{\dot{w}}Z_u}{m-Z_{\dot{w}}}\right] & \frac{1}{I_y}\left[M_w + \frac{M_{\dot{w}}Z_u}{m-Z_{\dot{w}}}\right] & \frac{1}{I_y}\left[M_q + \frac{M_{\dot{w}}(Z_q+mu_0)}{m-Z_{\dot{w}}}\right] & -\frac{M_{\dot{w}}mg\sin(\theta_0)}{I_y(m-Z_{\dot{w}})} \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2}$$

$$B = \begin{Bmatrix} \frac{X_{\delta e}}{m} \\ \frac{Z_{\delta e}}{m-Z_{\dot{w}}} \\ \frac{M_{\delta e}}{I_y} + \frac{M_{\dot{w}}Z_{\delta e}}{I_y(m-Z_{\dot{w}})} \\ 0 \end{Bmatrix} \tag{3}$$

$$x = \begin{Bmatrix} \Delta u \\ w \\ q \\ \Delta\theta \end{Bmatrix} \tag{4}$$

2.1 Aircraft model

The first step in developing the simulation environment for the campaign was to define the aircraft model. For this work, a linear longitudinal model of a FAR 25 commercial aircraft in cruise condition at 40,000 feet and Mach 0.8 was selected. The model, as described by Etkin and Reid [16], is presented in state-space form and is defined by Eqs. 1 to 4.

In this formulation, A, B, x, and δ_e represent the system dynamics matrix, input matrix, state vector, and control input vector, respectively. The state vector comprises the longitudinal velocity perturbation (Δu), vertical velocity (w), pitch rate (q), and pitch angle perturbation ($\Delta\theta$). The input vector consists solely of the elevator deflection. The elements of the A and B matrices correspond to the dimensional stability and control derivatives, respectively. The values of these derivatives, not presented in this article, are also provided by Etkin and Reid [16].

The $M_{\dot{w}}$ derivative (pitching moment with respect to $\dot{w}$) was varied arbitrarily to generate six different model configurations, as summarized in Table 1:

Table 1: Values of the $M_{\dot{w}}$ derivative.

Model configuration	$M_{\dot{w}}$ Value	Unit
1 (baseline)	$-1.12 \cdot 10^4$	N · m
2	$-5.78 \cdot 10^3$	N · m
3	$1.73 \cdot 10^3$	N · m
4	$6.19 \cdot 10^4$	N · m
5	$1.22 \cdot 10^5$	N · m
6	$1.74 \cdot 10^5$	N · m

These different model configurations were introduced to pilots within the simulation environment to create scenarios where the aircraft could more readily enter a PIO condition due to pilot input. From the perspective of dataset generation, it is essential that the collected samples represent a wide range of PIO intensities. Varying the model's PIO proneness is one effective way to achieve this diversity. Accordingly, model configuration 1 is the least prone to PIO, while model configuration 6 exhibits the highest susceptibility.

The decision to vary this specific derivative, rather than any other, is based on its effect on the short-period mode: it alters the damping ratio without significantly affecting the natural frequency, as illustrated in Figure 2. Consequently, by modifying this derivative across configurations 1 to 6, the model progressively approaches dynamic instability in longitudinal motion. As a result, the pilot must apply stabilizing control inputs to counteract the increasing tendency toward instability. If the pilot is unsuccessful, a pilot-induced oscillation (PIO) may develop. It is also worth noting that the opposite can occur: a pilot may inadvertently destabilize a configuration that is not inherently prone to PIO, depending on how their piloting behaviour is tuned during a demanding control task [17].

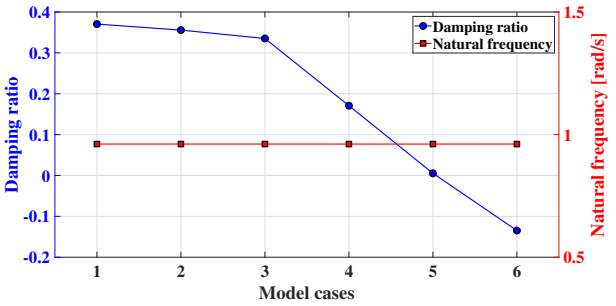

Figure 2: Dynamic parameters related to the short-period mode as function of $M_{\dot{w}}$.

2.2 The simulation environment

The second step in the data acquisition procedure was to establish a simulation environment that allowed a pilot to interact with the aircraft model described in the previous section.

To this end, a fixed-base flight simulator was developed, consisting of a display screen showing an artificial horizon and a joystick serving as the interface between the pilot and the model, as illustrated in Figure 3.

Figure 3: Simulation environment setup, featuring the artificial horizon display (top), joystick control interface (middle), and a pilot performing the simulation task (bottom) [11].

During the flight simulation, a target pitch angle is displayed to the pilot as a moving red line on the artificial horizon screen. The pilot must track this reference by applying appropriate stick deflections. This functionality is provided by the SynTask software, developed at the University of São Paulo and registered under the number BR512022002377-6 [18]. The development of SynTask is detailed in [19]. Implemented in MATLAB®, the software allows for modifications to both the pitch tracking task type and the aircraft dynamics with minimal changes to the code.

The joystick used in the simulation environment is a Thrustmaster HOTAS Warthog, positioned in a central configuration for right-hand operation. The pilot provides command inputs, denoted as η, through the joystick. These inputs are then converted into elevator deflections (δ_e), which serve as the input to the aircraft model. This conversion is performed by a second-order model representing the elevator actuation system, as defined in Eq. 5.

$$G_{actuator}(s) = \frac{\delta e}{\eta}(s) = \frac{-\omega_a^2}{s^2 + 2\zeta_a\omega_a + \omega_a^2} \tag{5}$$

In this actuator model, ω_a denotes the actuator's natural frequency, set to 8 rad/s, and ζ_a represents the actuator's damping ratio, with a value of 0.707.

2.3 Experimental procedure

The final step in the data acquisition procedure was to conduct the simulated flight tests using the aircraft model and the simulation environment described earlier. Figure 4 presents a block diagram summarizing a single flight simulator run.

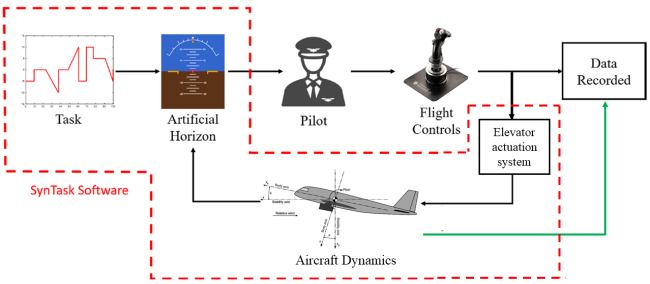

Figure 4: Experiment architecture [11].

For this flight simulator campaign, ten pilots with extensive experience and formal training in flight testing within the Brazilian aeronautical industry, as well as familiarity with the PIO rating scale, were invited to participate. The test procedures were approved by the Research Ethics Committee under process number CAAE 54,768,421.7.0000.5504.

Two different pitch angle tracking tasks were employed, based on flight test procedures for PIO commonly used in the industry [20]. The first procedure, known as the discrete synthetic task, involves a series of ramp and step commands that pilots are required to track using flight controls. To prevent pilot adaptation and ensure unpredictability, four variations of this task were created and randomly presented to the pilots. The second task, referred to as the pitch capture task, consists of smaller manoeuvrers involving long-duration step captures. Similarly, four variations of this task were developed and randomly presented to the pilots. Both types of tasks are illustrated in Figure 5, along with the pilot's corresponding attempts to track the demanded pitch angles.

The flight test campaign thus comprised a total of 480 simulation runs, resulting from ten pilots performing two types of pitch tracking tasks (each with four different variations) across six aircraft model configurations.

The flight data for each run was recorded in a *.mat* file containing time-series data of the pitch tracking task, aircraft pitch rate and angle (states 3 and 4 of Eq. 4), and the stick deflections inputs, with each run lasting approximately 50

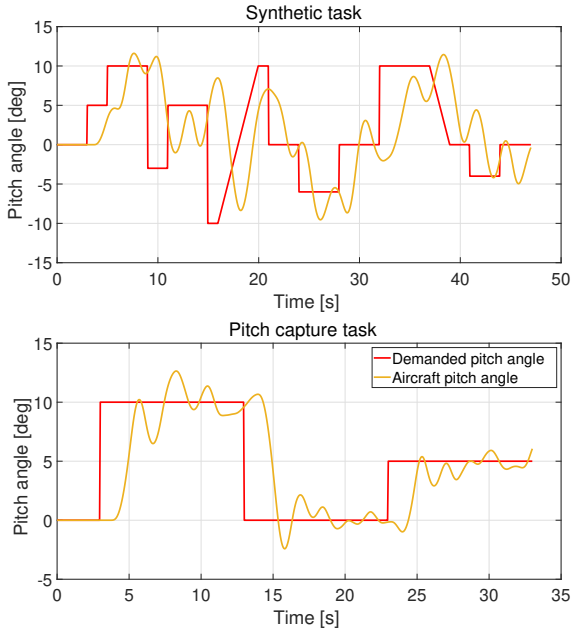

Figure 5: Types of pitch angle tracking tasks, both performed by Pilot 1 under the baseline model case.

seconds. All flight runs were conducted with a sampling frequency of 20 Hz. At the conclusion of each run, the pilot provided a PIO assessment using the PIO rating scale, indicating both the presence or absence of PIO and the intensity of the oscillations as perceived.

From a machine learning perspective, the raw dataset consists of 480 data samples (the *.mat* files), each labelled with the corresponding PIO rating (six classification classes). Table 2 presents the number of flight runs that received each specific PIO rating for every model case. As shown in this table, the resulting dataset is not perfectly balanced, with the proportion of samples for each PIO rating ranging from 11.7% to 21.4% of the total, whereas an ideal distribution would allocate 16.7% to each grade.

Table 2: Distribution of PIO ratings in the dataset.

Model	PIO rating scale					
case	1	2	3	4	5	6
1	29	32	16	3		
2	19	29	27	5		
3	8	32	34	6		
4		10	17	36	17	
5			9	26	35	10
6				4	18	58
Total	56	103	103	80	70	68
%	11.7	21.4	21.4	16.7	14.6	14.2

Given the structure and generation process of the dataset, there is potential to apply data augmentation techniques while preserving the original labels, in order to improve class balance. This could enhance the ability of the PIO classification algorithm to generalize more effectively. The data augmentation procedures will be described in the next section.

3 Flight data pre-processing

Between data acquisition and its application in a machine learning algorithm for classification purposes, a pre-processing stage is required. This section describes how the data was prepared to be fed into a convolutional neural network designed to classify PIO events.

Among the four time series collected during each flight run, only the pitch rate and stick deflections were selected for use in the PIO classification algorithm as inputs. These parameters were chosen because they are readily measurable in real flight conditions and exhibit zero mean, which facilitates the detection of oscillatory patterns. This rationale is consistent with other quantitative identification methods, such as the Real-time Oscillation Verifier (ROVER) [7, 6] and the Inceptor Peak Power Phase (IPPP) [8, 21], which also rely on these parameters for similar reasons.

The pitch angle time-series presents the challenge of a non-zero mean, requiring a reference signal to compute the tracking error and detect oscillatory behaviour. While the reference signal (i.e., the pitch tracking task) is available in this test campaign, such information is rarely accessible in real operational scenarios. Therefore, a robust PIO classification/detection method should not depend on a predefined reference pattern as part of its input. Instead, it should focus solely on identifying the oscillatory characteristics of the pilot-aircraft interaction.

The pitch tracking task time-series and the variation in model proneness to PIO were solely intended to generate data across the full range of the PIO rating scale by inducing pilot-induced oscillations at different severity levels. These variables were not used as input parameters for the classification algorithm, thereby eliminating dependence on data generated exclusively under controlled flight test conditions.

The pilot's input, whether in the form of stick deflection or force (as used in other methods [7, 8]), and the aircraft's response, represented by angular rate, constitute the most fundamental relationship in the pilot-aircraft interaction and form the basis of the PIO phenomenon. Any additional information, such as the phase difference between the pilot's input and the aircraft's response, can be interpreted as a consequence of these two signals and lies within the scope of specific PIO definitions. An identification method built upon a particular set of PIO definitions will only detect and quantify what that set describes as PIO. Slight variations in these definitions can lead to different identification outcomes.

For the reasons discussed above, the proposed machine learning-based identification method relies solely on these two input signals, without incorporating any additional parameters derived from existing PIO definitions. The intent is for the algorithm to autonomously identify and quantify PIO behaviour by learning patterns in the data and correlating them with the PIO rating scale. This approach contrasts with the work of Bruschi et al. [11], who performed machine learning-based PIO identification using input parameters already associated with established PIO definitions found in the literature.

This approach also differs from pilot assessments using the PIO rating scale in terms of underlying definitions. Although the scale is based on a structured set of definitions, the rating process is inherently subjective and varies according to each pilot's perception and interpretation of both the scale and the PIO phenomenon [6, 5]. Therefore, the algorithm is also expected to learn an approximate or "average" mapping function that captures the common patterns underlying the human-assigned PIO ratings.

3.1 The Wavelet Transform

One way to enrich the information contained within time-series data, without relying on predefined PIO characteristics, is through the application of the Wavelet Transform. This mathematical tool is a multi-scale time-frequency analysis technique particularly suited for non-stationary signals, which are common in piloting scenarios, including those involving PIO. Similar to the Fourier Transform, the Wavelet Transform reveals the frequency content of a signal, though at the cost of reduced frequency resolution. However, it offers a significant advantage by also providing the specific time intervals at which those frequencies occur, making it highly valuable for capturing transient events in dynamic flight data.

Over the past few decades, the Wavelet Transform has been applied to PIO analysis in a range of contexts: from basic applications, such as expanding flight data signals into the frequency domain while preserving time-localized transient information, to more advanced uses, including the estimation of time-varying frequency responses [22, 23] and the development of criteria such as the Inceptor Peak Power Phase (IPPP) [8, 21].

The Wavelet Transform is defined as the convolutional integral between the analysed signal $x(t)$ and the complex conjugate of the wavelet function $\Psi(t)$, as shown in Eq. 6.

$$\mathscr{W}(\tau, f) = \sqrt{f} \int_{\infty}^{\infty} x(t)\, \Psi^*(f(t-\tau))dt \tag{6}$$

The parameter τ is used to translate the wavelet function along the time axis during the convolution, while f scales the wavelet function, effectively adjusting its resolution in the frequency domain. These parameters correspond to the frequency and time axes in the output of the transform.

In this work, the wavelet function used is the Morlet wavelet, defined in Eq. 7, where $u = f(t-\tau)$, β controls the temporal width of the wavelet function, and c determines the number of oscillation cycles within the wavelet.

$$\Psi(u) = \frac{1}{\sqrt{\pi\beta}} e^{\frac{-u^2}{\beta}} \left[cos\left(\frac{2\pi}{c}u\right) + j\, sin\left(\frac{2\pi}{c}u\right) \right] \tag{7}$$

The Wavelet Transform was therefore applied to both the stick deflection and pitch rate time-series, expanding the original one-dimensional representation, amplitude over time, into a two-dimensional representation of amplitude over both time and frequency. Figure 6 illustrates the original time-series signals (in green and purple) alongside their corresponding scalograms, which represent the absolute value of the complex amplitude resulting from the Wavelet Transform. The pitch task and pitch angle signals are included in the top plot solely to illustrate the data generation process; however, they were not used as inputs to the machine learning algorithm, as discussed.

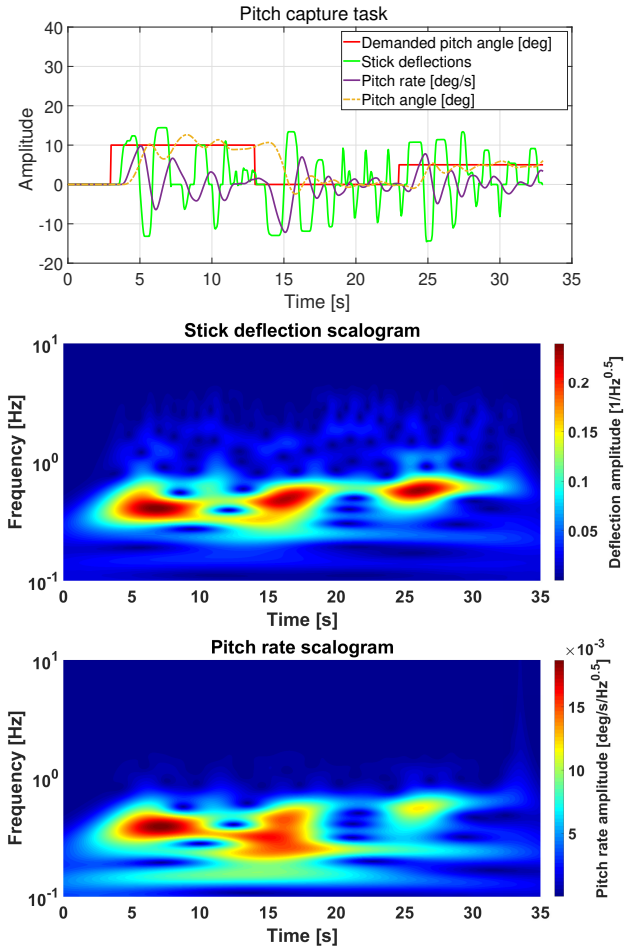

Figure 6: Stick deflections and pitch time-series (top), stick deflection scalogram (middle), and pitch rate scalogram (bottom), related to a pitch capture task.

The scalograms, obtained using MATLAB®, show that the pilot applied stick deflections across a varying range of frequencies during the task, and that the aircraft response exhibited excitation at some of those same frequencies. The colour scale represents the magnitude of the Wavelet Transform, corresponding to the square root of the power spectral density associated with each signal. This highlights the enhancement of the signal's informational content, as the relationship between pilot input and aircraft response can now be observed simultaneously in both the time and frequency domains.

From a machine learning perspective, the raw time-series data were transformed into two-dimensional tensors using the Wavelet Transform. These tensors can also be interpreted as grey-scale images (despite being displayed in colour in Figure 6), where the time and frequency axes define the pixel locations, and the square root of the power spectral density

determines the pixel intensity.

The dataset of 480 labelled samples described in Section 2, originally containing four raw time-series per sample, was thus transformed to contain two scalograms per sample, while preserving the same number of labelled instances. Therefore, the stick deflection and pitch rate scalograms form a two-channel input to the convolutional neural network algorithm. After the transformation, each scalogram had a resolution of 512×941 pixels, corresponding to the frequency and time axes, respectively. All scalograms were saved as matrices in *.csv* files.

3.2 Dataset augmentation

Before using the dataset in the PIO classification model training process, the *.csv* scalogram files and their corresponding PIO rating labels were consolidated into a single *.h5* file (Hierarchical Data Format) for use in a Python environment. This format offers the advantage of packaging the entire dataset into one file, allowing for easier manipulation and application within the algorithm.

As mentioned in Section 2, the dataset is imbalanced, with a disproportionate distribution of labels across samples (see Table 2). This imbalance can impair the model's ability to generalize effectively [24]. To mitigate this issue, the dataset was augmented using an oversampling strategy, which involves duplicating samples from under represented classes to increase their presence in the dataset. Specifically, classes 1, 4, 5, and 6 were expanded to 95 samples each, while classes 2 and 3 remained unchanged with 103 samples each. This approach constitutes a partial balancing process, intentionally preserving some degree of label imbalance to better reflect the real-world conditions of the flight simulator campaign.

Simple duplication, however, does not fully address the generalization problem. While oversampling helps balance the dataset, repeatedly exposing the model to identical data can lead to overfitting. To address this, data augmentation techniques were applied to all samples. Specifically, Gaussian noise was introduced, and a combination of time and frequency masking was employed by randomly zeroing out columns and rows of the scalograms, respectively. These augmentations ensured that duplicated samples varied slightly from the originals, enhancing diversity in the training data and promoting better generalization [25].

It is important to note that common augmentation techniques such as image flipping along the x-axis or y-axis and image rotations are not suitable for this particular scalogram dataset. Flipping along the frequency axis, for instance, would distort the fundamental structure of the signal, as low-frequency oscillations differ significantly in nature from high-frequency ones, meaning a pilot would not assign the same PIO rating to both. On the other hand, flipping along the time axis would not pose a problem if only one channel were used. However, since the scalograms represent both stick deflections and pitch rate, two channels with a causal relationship, reversing the time axis would violate this causality, which is critical to accurately capturing the PIO phenomenon.

Due to the inherent uncertainty in the pilot-assigned PIO ratings during flight tests, some samples are likely mislabelled. To account for this labelling noise, a label smoothing technique was incorporated into the training process. In this approach, a 90% confidence level was assigned to the original label, reflecting the assumption that each sample has a 90% probability of being correctly labelled. This value was selected based on findings in the literature related to PIO rating assignments in longitudinal motion scenarios [6].

4 PIO classification model

The proposed machine learning model for detecting and classifying Pilot-Induced Oscillations (PIO), based on the PIO rating scale severity metric, comprises two main components: a convolutional neural network (CNN) and a multilayer perceptron (MLP), as illustrated in the model schematic in Figure 7. The CNN is responsible for identifying patterns in the 2-channel scalogram inputs that are indicative of PIO behaviour. These extracted patterns are then passed to the MLP, which classifies them according to the corresponding PIO severity rating [26]. The model was implemented in Python using the PyTorch library.

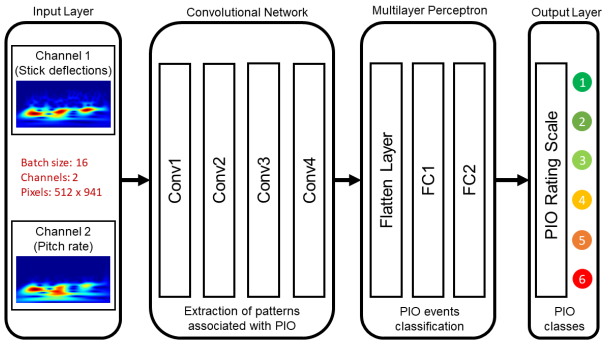

Figure 7: PIO classification model schematics.

The convolutional neural network consists of four blocks (Conv1 to Conv4), each containing a convolutional layer followed by batch normalization, an activation function, dropout, and pooling operations. The multilayer perceptron (FC1 and FC2), in contrast, comprises two linear layers with activation functions and dropout. The full architecture and layer specifications are detailed in Table 3.

In Table 3, the two-dimensional convolutional layer parameters k, s, and p denote the kernel size, stride, and zero padding, respectively. The Rectified Linear Unit (ReLU) was selected as the activation function due to its widespread use and effectiveness in modern convolutional networks. Batch normalization and dropout layers were incorporated to enhance the model's generalization capability and reduce the risk of overfitting. The value that accompanies the dropout layers is the probability of neurons being deactivated during the training process.

The number of channels increases progressively as the scalograms pass through the network, enhancing its ability to extract meaningful features while compensating for the spatial

Table 3: Detailed architecture of the PIO classification model.

Layer	Type	Output Shape	Parameters
Conv1	Conv2D	$2 \times 256 \times 470$	$k = 4$, $s = 2$, $p = 1$
	BatchNorm + ReLU	$2 \times 256 \times 470$	–
	MaxPool2D	$2 \times 128 \times 235$	$k = 2$
Conv2	Conv2D	$3 \times 128 \times 235$	$k = 3$, $s = 1$, $p = 1$
	BatchNorm + ReLU + Dropout(0.2)	$3 \times 128 \times 235$	–
	MaxPool2D	$3 \times 64 \times 117$	$k = 2$
Conv3	Conv2D	$4 \times 64 \times 117$	$k = 3$, $s = 1$, $p = 1$
	BatchNorm + ReLU	$4 \times 64 \times 117$	–
	MaxPool2D	$4 \times 32 \times 58$	$k = 2$
Conv4	Conv2D	$2 \times 32 \times 58$	$k = 3$, $s = 1$, $p = 1$
	BatchNorm + ReLU	$2 \times 32 \times 58$	–
	MaxPool2D	$2 \times 16 \times 29$	$k = 2$
Flatten	–	928	$2 \times 16 \times 29$
FC1	Linear + BatchNorm + ReLU + Dropout(0.5)	64	-
FC2	Linear	6	-

resolution loss introduced by pooling operations. However, in the final convolutional layer, the number of channels is reduced back to its original value. This reduction helps limit the number of parameters in the multilayer perceptron, which receives the flattened output of the convolutional network.

5 Model training and validation

Model training and validation were performed using the stratified k-fold cross-validation procedure [25]. In this approach, the dataset is randomly shuffled and then partitioned into k equally sized subsets, or folds, ensuring that each fold maintains approximately the same class distribution. In each iteration, one fold is held out as the validation set, while the remaining $k-1$ folds are used to train the model. This process is repeated k times, with each fold used once for validation. As a result, k distinct model versions are trained, each based on a different training-validation split, providing a more robust estimate of the model's generalization performance. A value of $k = 9$ was chosen for the training-validation procedure of the PIO identification model.

For each fold iteration, the model was trained for 60 epochs using a batch size of 16 samples. The loss function employed was a modified version of the cross-entropy loss, implemented in PyTorch, which incorporated the label smoothing technique described in the previous section. Optimization was performed using the AdamW algorithm, with a learning rate of 0.001 and a weight decay parameter of 0.001. Weight decay is used as a regularization method to improve the model's generalization capability by penalizing large weight values. The highest accuracies achieved on the training and validation subsets for each model version are presented in Table 4.

Accuracy is a metric that evaluates the overall ability of the model to predict the correct PIO ratings by comparing its outputs with the corresponding ground truth labels. As shown in Table 4, the training accuracy consistently exceeds 93% across most folds, significantly outperforming the corresponding validation accuracy. This discrepancy suggests that

Table 4: Best training and validation accuracies for each model version.

Fold	Training Accuracy	Validation Accuracy
1	51.7%	39.4%
2	98.1%	63.1%
3	98.5%	61.5%
4	98.5%	58.5%
5	93.7%	50.8%
6	99.8%	63.1%
7	98.8%	50.8%
8	99.0%	60.0%
9	25.3%	27.7%

the model exhibits a certain degree of overfitting, despite the measures implemented to mitigate it. Nevertheless, folds 2, 3, 6, and 8 achieved validation accuracies above 60%, which can be considered a positive outcome given the limited size of the dataset, the use of data augmentation, and the inherent uncertainty in the pilot-assigned labels. On average, the model achieved a validation accuracy of 52.8%, with a standard deviation of 12.2%.

While accuracy provides an overall measure of model performance, it does not offer insight into how the model distributes its predictions across individual classes. To address this, a confusion matrix for the best-performing model, corresponding to Fold 6, is presented in Figure 8. This model was selected as the best version based on its highest training and validation accuracies among all folds.

The confusion matrix illustrates how the Fold 6 validation set, consisting of 65 samples, was classified by the model. The values along the main diagonal represent the number of correctly classified samples for each class. Values within a given row indicate samples that truly belong to the class associated with that row but were misclassified as other classes (these are considered false negatives). Conversely, values within a column represent false positives, i.e., samples that actually

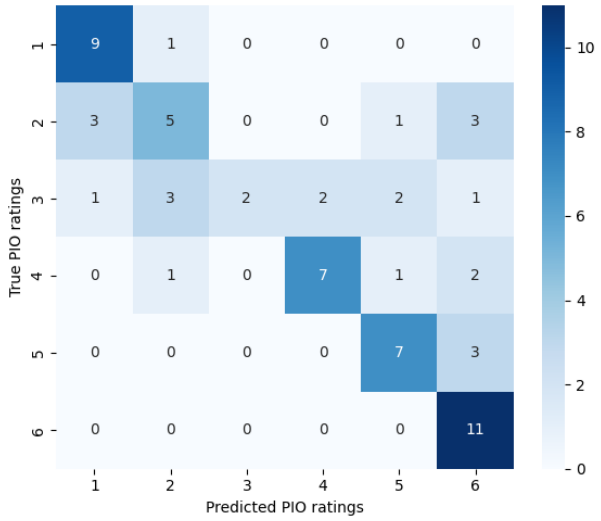

Figure 8: Confusion matrix of the validation set for the best-performing model (Fold 6).

belong to other classes but were incorrectly predicted as the class associated with that column. It can be noted that PIO ratings 2 and 3 were particularly challenging for the model to classify correctly, as they exhibited a high number of false positives. This misclassification contributed significantly to the reduction in overall model accuracy.

To provide a more detailed understanding of the model's classification performance, the Precision, Recall, and F1-score metrics were calculated, as shown in Table 5. Precision measures the proportion of true positives out of the total predicted positives (i.e., true positives plus false positives), while recall measures the proportion of true positives out of all actual positives (i.e., true positives plus false negatives). The F1-score represents the harmonic mean of precision and recall, offering a balanced metric that accounts for both false positives and false negatives [25].

Table 5: Precision, Recall, and F1-score metrics for the best-performing model (Fold 6).

PIOR	Precision	Recall	F1-score
1	69.2%	90.0%	78.2%
2	50.0%	41.7%	45.4%
3	100.0%	18.2%	30.8%
4	77.8%	63.6%	70.0%
5	63.6%	70.0%	66.7%
6	55.0%	100.0%	71.0%

As expected, the F1-scores for PIOR 2 and 3 are significantly lower than those of the other classes, which in turn negatively impacts the overall performance metrics. Furthermore, analysis of the Precision, Recall, and the confusion matrix reveals that classes 4, 5, and 6 exhibit a pattern of mutual misclassification, as indicated by their recall values. This grouping is consistent with the nature of high-intensity PIO events described by the PIO rating scale. The presence of false negatives among these classes may be attributed to the inherent subjectivity involved in the pilot's rating process, particularly when distinguishing between closely related severity levels of PIO behaviour. At the opposite end of the spectrum, PIOR 1 achieved the highest F1-score, indicating that the algorithm was most effective at identifying samples free from PIO events.

Since the k-fold technique was applied using nine versions of the training and validation datasets, nine different models were obtained, each with varying performance. A common practice to address this is to use an ensemble technique to combine their results. In this work, the ensemble model was obtained using the soft voting method [27], where the final prediction for a given class is made by averaging the predicted probabilities across all models.

The ensemble model achieved a validation accuracy of 84.6%, outperforming the individual models. Table 6 reports the Precision, Recall, and F1-score for the ensemble. While Precision and Recall show some variation due to the new classifications introduced by the model, the F1-score highlights the overall improvement, most notably in PIOR 2 and PIOR 3. Despite the considerable improvement, the model's ability to classify these two classes remains unsatisfactory, as their F1-scores are still below 80%.

Table 6: Precision, Recall, and F1-score metrics for the ensemble model.

PIOR	Precision	Recall	F1-score
1	100.0%	90.0%	94.7%
2	87.5%	58.3%	70.0%
3	61.1%	100.0%	75.9%
4	88.9%	72.7%	80.0%
5	100.0%	90.0%	94.7%
6	91.7%	100.0%	95.7%

The confusion matrix in Figure 9 illustrates the classification results of the validation set by the ensemble model. As shown, the number of false negatives (row elements) decreased substantially, particularly for PIOR 2 and PIOR 3. In contrast, the reduction in false positives (column elements) was more modest.

The individual and ensemble models' inability to accurately classify samples from classes 2 and 3 may be attributed to issues in the dataset generation process, where the collected data may lack sufficient distinction to allow for reliable class separation. Additionally, limitations in the model architecture and data pre-processing pipeline could have contributed to the observed misclassifications, suggesting that further refinement in these areas may be necessary. Another important factor to consider is the inherent subjectivity of the PIO rating scale. The similarity between classes 2 and 3 may pose a challenge even for the pilots to assign a rating, particularly in simulation environments lacking motion feedback, which can reduce the pilot's ability to perceive subtle differences in PIO severity [28]. Nevertheless, further research is required to investigate the underlying causes of the classification challenges observed in this study.

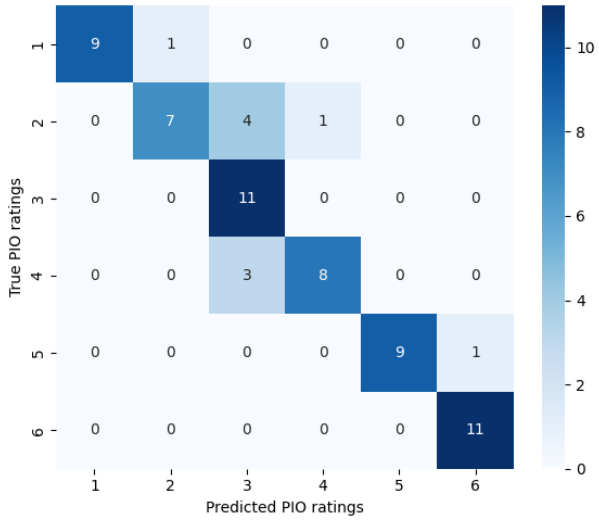

Figure 9: Confusion matrix of the validation set for the ensemble model.

6 Conclusion

This paper presents a data-driven approach to the PIO identification problem using machine learning techniques. Flight data were collected during a fixed-base flight simulator campaign and labelled according to the PIO rating scale by the participating test pilots. Stick deflection and pitch rate signals were selected as the primary inputs to characterize the PIO phenomenon. These signals were transformed into scalograms using the Wavelet Transform and combined into a two-channel input for a convolutional neural network. The network was designed to extract relevant features related to PIO behaviour and to output a severity level aligned with the PIO rating scale. The k-fold cross-validation methodology was applied during model training, resulting in an average validation accuracy of 52.8%, with the best-performing model achieving an accuracy of 63.1%. The soft voting ensemble method was subsequently applied, yielding improved classifications with a validation accuracy of 84.6%.

It can be concluded that the model, developed without incorporating any predefined knowledge of PIO definitions, either in its architecture or during data pre-processing, was able to effectively classify the extreme classes at both ends of the PIO rating scale spectrum in terms of severity. However, it demonstrated difficulty in distinguishing between PIOR 2 and 3, which correspond to low to intermediate levels of PIO intensity. This limitation significantly impacted the individual and ensemble models' overall classification accuracy and can be attributed to several factors, including the method used to construct the dataset, its limited size, the model architecture, and the selection of hyperparameters. Additionally, it may also reflect the inherent difficulty pilots face in accurately assigning PIOR 2 and 3 ratings, due to the subtle perceptual differences involved in distinguishing between these intermediate levels of PIO severity.

This work contributes to the field of handling qualities, particularly in the area of PIO identification, by introducing a data-driven alternative to the traditional use of the PIO rating scale. Although the developed model requires further refinement to improve its accuracy, it shows promising potential as a complementary tool in handling qualities evaluations, offering a quantitative counterpart to pilot assessments, while operating within the same evaluative framework used by human pilots.

Acknowledgements

The authors would like to acknowledge the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Brazil, and the Swedish-Brazilian Innovation Center (CISB) for their financial support through the scholarship MCTI/CNPq CISB SAAB AB 37/2023 200932/2024-1.

References

[1] Department of Defence Interface Standard (MIL-STD-1797A). *Flying qualities of piloted aircraft*. 2004.

[2] FAA Advisory Circular. 25-7d Flight test guide for certification of transport category airplanes. *Federal Aviation Administration*, 2018.

[3] David G Mitchell and David H Klyde. Identifying a pilot-induced oscillation signature: New techniques applied to old problems. *Journal of Guidance, Control, and Dynamics*, 31(1):215–224, 2008.

[4] David H Klyde, Duane T McRuer, and Thomas T Myers. Unified pilot-induced oscillation theory, volume i: Pio analysis with linear and nonlinear effective vehicle characteristics, including rate limiting. *Flight Dynamics Directorate, Wright Laboratory, Air Force Materiel Command*, 1995.

[5] John Hodgkinson. *Aircraft Handling Qualities*. AIAA Education Series, Reston, VA, 1st edition, 1999.

[6] André LA Paladini, Daniel Drewiacki, and Jorge H Bidinotto. Aeroelastic effects in pio occurrences: a dual approach on flight simulator tests. *Aerospace Science and Technology*, page 109337, 2024.

[7] David Mitchell, Alfredo Arencibia, and Susana Munoz. Real-time detection of pilot-induced oscillations. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, page 4700, 2004.

[8] David H Klyde, Philip C Schulze, Ricardo S Mello, and David Mitchell. Assessment of a scalogram-based pio metric with flight test data. In *AIAA Atmospheric Flight Mechanics Conference*, page 1641, 2017.

[9] Shuting Xu, Wenqian Tan, QU Xiangju, and Cheng Zhang. Prediction of nonlinear pilot-induced oscillation using an intelligent human pilot model. *Chinese Journal of Aeronautics*, 32(12):2592–2611, 2019.

[10] Ryota Mori and Shinji Suzuki. Analysis of pilot landing control in crosswind using neural networks. In *2009 IEEE Aerospace conference*, pages 1–10. IEEE, 2009.

[11] Adriano Ghigiarelli Bruschi, Daniel Drewiacki, and Jorge Henrique Bidinotto. Artificial neural networks for pio events classification comparing different data collection procedures. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 46(8):496, 2024.

[12] Abdul Malik Badshah, Jamil Ahmad, Nasir Rahim, and Sung Wook Baik. Speech emotion recognition from spectrograms with deep convolutional neural network. In *2017 international conference on platform technology and service (PlatCon)*, pages 1–5. IEEE, 2017.

[13] Jingshan Huang, Binqiang Chen, Bin Yao, and Wangpeng He. ECG arrhythmia classification using stft-based spectrogram and convolutional neural network. *IEEE access*, 7:92871–92880, 2019.

[14] Jana Schwithal, Jan-Philipp Buch, Carsten Seehof, Marco A Alves Júnior, Daniel Drewiacki, and Fernando J O Moreira. Simulating flexible aircraft in a full motion simulator. In *AIAA AVIATION 2023 Forum*, page 3318, San Diego, CA, 2023.

[15] Fernando J O Moreira, Daniel Drewiacki, Marco Antonio O Alves Junior, Jan-Philipp Buch, Jana Schwithal, and Carsten Seehof. Flight simulator result comparing three aircraft configurations: Quasi-static, flexible and extended flexibility. In *AIAA SCITECH 2023 Forum*, page 1368, National Harbor, MD, 2023.

[16] Bernard Etkin and Lloyd Duff Reid. *Dynamics of flight: stability and control*. John Wiley & Sons, 1995.

[17] David G Mitchell and David H Klyde. Defining pilot gain. *Journal of Guidance, Control, and Dynamics*, 43(1):85–95, 2019.

[18] Jorge. H. Bidinotto and Gabriel D. A. Neves. Universidade de São Paulo, São Carlos, SP, Brazil, 2022. Patent Application for a SynTask – Software de Tarefas Sintéticas Aeronáuticas No BR512022002377–6, filed 06 sep.

[19] Gabriel D. A. Neves, Adriano. G. Bruschi, and Jorge. H. Bidinotto. Development of a tool for pilot induced oscillations (PIO) testing in flight simulator. In *26th International Congress of Mechanical Engineering (COBEM)*, Brazil (virtual conference), 2021.

[20] David Mitchell and David Klyde. Recommended practices for exposing pilot-induced oscillations or tendencies in the development process. In *USAF developmental test and evaluation summit*, page 6810. 2004.

[21] Daniel Drewiacki, Fernando José d Moreira, Frederico A Ribeiro, and Jana Schwithal. Flexible aircraft evaluation in a full-motion simulator: An approach for handling qualities analysis. In *AIAA SCITECH 2025 Forum*, page 1440, 2025.

[22] David Klyde, Peter Thompson, Edward Bachelder, and Theodore Rosenthal. Evaluation of wavelet-based techniques for detecting loss of control. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, page 4702, 2004.

[23] Peter Thompson, David Klyde, Edward Bachelder, Theodore Rosenthal, and Martin Brenner. Development of wavelet-based techniques for detecting loss of control. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, page 5064, 2004.

[24] Zhan ao Huang, Yongsheng Sang, Yanan Sun, and Jiancheng Lv. A neural network learning algorithm for highly imbalanced data classification. *Information Sciences*, 612:496–513, 2022.

[25] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[27] Stamatis Karlos, Georgios Kostopoulos, and Sotiris Kotsiantis. A soft-voting ensemble based co-training scheme using static selection for binary classification problems. *Algorithms*, 13(1):26, 2020.

[28] Edward Bachelder, Eileen Bjorkman, and Bimal Aponso. PIO and handling qualities prediction using the USAFTPS Bjorkman PIO data set. In *Vertical Flight Society's 80th Annual Forum & Technology Display*, Montreal, Canada, 2024.