iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.79032

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

E-Pilots: A System to Predict Hard Landing During the Approach Phase of Commercial Flights

Chollangi Madhuri, Gannavarapu Durga Satish, Jayant Kamdi, Chiratapudi Suresh, Ganji Lakshmi
Department of Computer Science and Engineering (AI & ML), Bonam Venkata Chalamayya Engineering College, Odalarevu, Andhra Pradesh, India

Abstract: ***Hard landings represent a persistent hazard in commercial aviation, causing structural stress to airframes, increasing maintenance expenditure, and endangering passenger safety. Existing monitoring approaches predominantly perform post-flight analysis, flagging hard landings only after they have occurred. This paper presents E-Pilots, an intelligent software system that analyzes critical flight parameters gathered during the approach phase to predict the probability of a hard landing before the aircraft reaches the runway threshold. The proposed framework ingests parameters including altitude, vertical speed, descent rate, pitch angle, and ambient weather conditions, and processes them through a multi-model prediction pipeline comprising Support Vector Machines (SVM), Logistic Regression, and a Hybrid Long Short-Term Memory (LSTM) network. Feature selection is applied to isolate the most predictive indicators, and a rule-assisted classification layer maps the combined model outputs to three safety states: Safe, Warning, and Hard Landing. Evaluated under realistic flight data scenarios, the system achieves reliable classification with reduced false-positive rates compared to single-indicator threshold methods. The system further provides a graphical user interface for dataset upload, real-time result visualization, and session reporting. E-Pilots demonstrates that data-driven, multi-model prediction can meaningfully augment aviation safety by delivering advance warning during the most safety-critical phase of a commercial flight.***

Keywords: Hard Landing Prediction, Flight Data Analysis, Support Vector Machine, LSTM, Machine Learning, Aviation Safety, Descent Rate, Vertical Speed, Approach Phase Monitoring.

I. INTRODUCTION

Hard landing remains among the foremost operational hazards in commercial aviation. When an aircraft contacts the runway with vertical force or descent rate exceeding certified structural limits, the landing is classified as hard. Such events can initiate microscopic fatigue cracks in landing-gear components and fuselage frames, necessitate unscheduled maintenance inspections, and in severe cases pose immediate safety risks to passengers and crew. The International Civil Aviation Organization (ICAO) and aircraft manufacturers prescribe hard-landing inspection procedures, yet the triggering event itself is rarely predicted or prevented in advance.

Modern commercial aircraft are equipped with Flight Data Recorders (FDR) and Quick Access Recorders (QAR) that continuously capture dozens of flight parameters at high sampling rates. During the approach phase—typically the final 1,000 to 500 feet of descent—parameters such as altitude, vertical speed, descent rate, pitch angle, indicated airspeed, and atmospheric conditions encode rich information about the likely touchdown quality. Analyzing these parameters intelligently offers a path toward proactive hard-landing avoidance rather than reactive post-flight inspection.

The advancement of machine learning (ML) and deep learning techniques has made it feasible to extract predictive patterns from complex, multivariate time-series flight data. Algorithms such as Support Vector Machines, Logistic Regression, and Long Short-Term Memory recurrent networks have been applied successfully in related aviation safety domains. However, the specific problem of predicting hard landings before touchdown—rather than detecting them after—remains underserved in the literature.

This paper presents E-Pilots, a Python-based intelligent system designed to address this gap. The system accepts flight approach data, preprocesses it, performs feature selection, and applies a hybrid ML prediction pipeline to classify landing outcomes as Safe, Warning, or Hard Landing. The principal contributions of this work are:

1) Development of E-Pilots, an end-to-end prediction system that classifies landing risk before touchdown using approach-phase flight parameters.
2) A multi-model prediction pipeline combining SVM, Logistic Regression, and a Hybrid LSTM network, with feature selection to optimize input dimensionality.

3) A three-tier safety classification framework (Safe, Warning, Hard Landing) that aggregates model outputs and reduces single-indicator false positives.
4) Experimental validation demonstrating consistent classification accuracy, with a user-friendly graphical interface for operational deployment.

II. LITERATURE REVIEW

Research into automated detection of abnormal landing conditions spans multiple decades and has evolved alongside advances in sensor technology, data availability, and computational methods.

A. *Threshold-Based Methods*

Early approaches to hard-landing detection relied on fixed threshold values for parameters such as vertical acceleration measured by onboard accelerometers. Bergasa et al. [1] demonstrated that simple rule-based monitoring could flag anomalous flight states in real time; however, threshold-based systems suffer from sensitivity to sensor noise and fail to capture the multivariate nature of hard-landing causation. A descent rate that is hazardous in gusty crosswind conditions may be perfectly safe in calm air with a stabilized approach, a nuance that fixed thresholds cannot express.

B. *Statistical and Machine Learning Approaches*

Subsequent work introduced statistical modeling to move beyond fixed rules. Logistic Regression models trained on labeled flight datasets demonstrated improved generalization over threshold methods by weighting multiple features simultaneously. Support Vector Machines have been applied to binary flight-event classification tasks, exploiting the kernel trick to capture non-linear decision boundaries in feature spaces composed of altitude, vertical speed, and pitch angle [2]. Decision tree ensembles such as Random Forests have also been explored for their interpretability, allowing aviation engineers to trace classification decisions back to individual parameter contributions.

C. *Deep Learning and Sequence Models*

The temporal nature of flight data makes recurrent neural network architectures particularly well-suited to the prediction task. Long Short-Term Memory networks address the vanishing gradient problem and can learn dependencies across extended time windows [4]. Hu et al. [3] applied LSTM-based models to fatigue-state classification, demonstrating that sequential feature learning outperforms frame-level classification for time-series events. Hybrid architectures that combine convolutional feature extraction with LSTM temporal modeling have shown further gains in aviation anomaly detection tasks.

D. *Limitations of Prior Work*

A consistent limitation across existing studies is the focus on post-landing analysis. Systems that consume FDR data after flight completion can identify hard landings for maintenance scheduling, but they provide no actionable advance warning. The E-Pilots system specifically targets the predictive, pre-touchdown window and is designed to operate on standard flight data exports accessible to engineering teams and researchers.

III. SYSTEM MODEL AND PROBLEM FORMULATION

Let a flight approach sequence be represented as a multivariate time series $X = \{x_1, x_2, ..., x_t\}$, where each observation $x_i \in \mathbb{R}^n$ encodes n flight parameters sampled at time step i. The feature vector includes: altitude h (ft), vertical speed v (ft/min), descent rate d (ft/min), pitch angle θ (degrees), indicated airspeed IAS (knots), and a weather severity index w.

The prediction objective is to learn a classifier $f : \mathbb{R}^n \rightarrow \{\text{Safe, Warning, Hard}\}$ that maps an approach sequence to a landing quality label before touchdown occurs. This is formulated as a supervised multi-class classification problem.

A. *System Architecture*

The E-Pilots framework consists of five primary components: (1) Data Input Module, (2) Data Preprocessing Module, (3) Feature Selection Module, (4) Prediction Module, and (5) Result Visualization Module. Figure 1 presents the high-level system architecture, illustrating the data flow from flight dataset storage through prediction to GUI dashboard output.

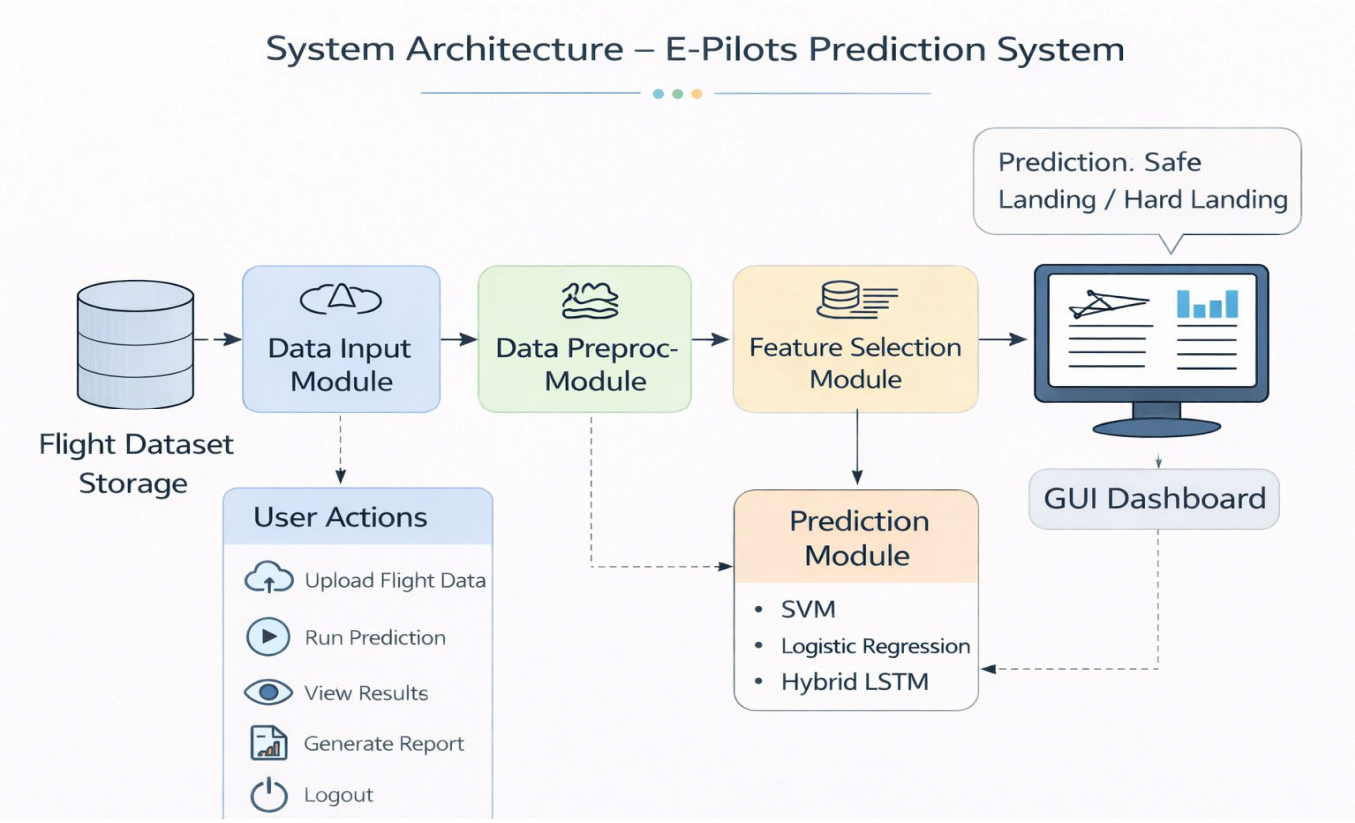


Fig. 1. System Architecture – E-Pilots Prediction System. Data flows from flight dataset storage through the Data Input, Preprocessing, and Feature Selection modules, into the Prediction Module (SVM / Logistic Regression / Hybrid LSTM), and finally to the GUI Dashboard.

Figure 2 provides the Level-0 Data Flow Diagram (DFD), showing the top-level interaction between the User, the E-Pilots Prediction System, and the Result Display (GUI Dashboard). The flight dataset storage feeds raw data into the system, while the ML engine (SVM / LSTM) drives the prediction engine before results are surfaced on the dashboard.

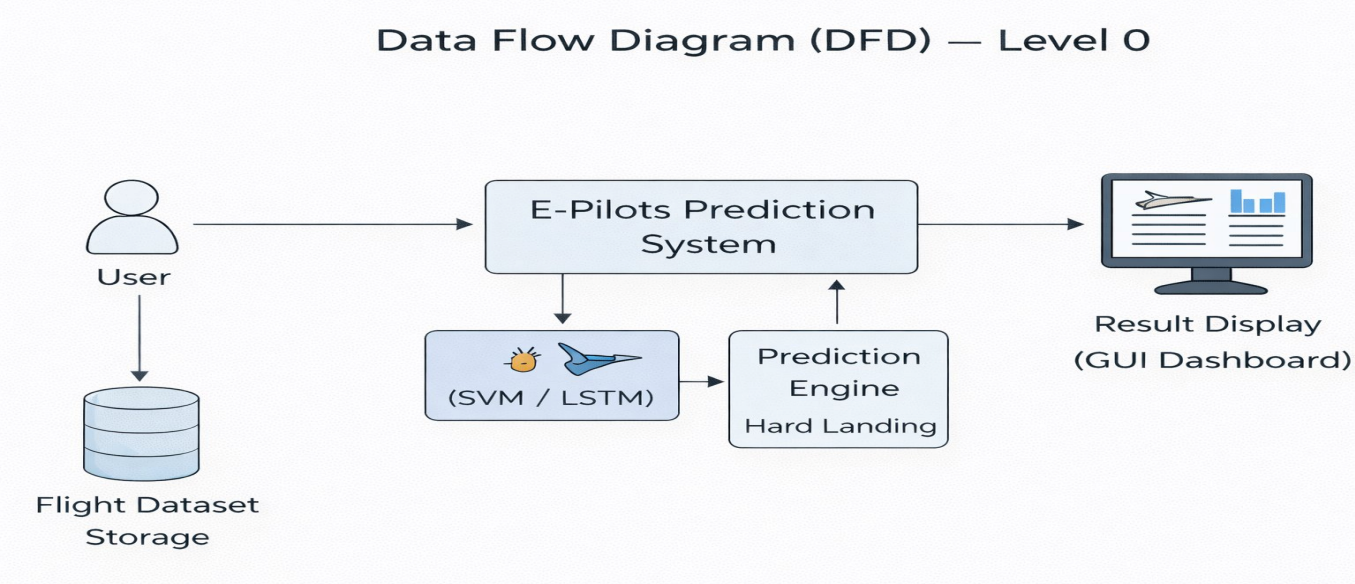


Fig. 2. Data Flow Diagram (DFD) – Level 0. Shows the top-level data exchange between the User, Flight Dataset Storage, the E-Pilots Prediction System, and the Result Display (GUI Dashboard).

B. Data Preprocessing

Raw flight data exported from FDR or QAR sources may contain missing samples, sensor dropouts, and scale inconsistencies. The preprocessing stage applies mean imputation for missing values, removes duplicate timestamps, and applies min-max normalization to map all features to the range [0, 1]:

$$x_i' = (x_i - x^{m,n}_i) / (x^{max} - x^{m,n}_i)$$

Normalization prevents parameters with large dynamic ranges (e.g., altitude in thousands of feet) from dominating parameters with smaller ranges (e.g., pitch angle in single-digit degrees).

C. Feature Selection

The feature selection module employs mutual information scoring to rank each candidate feature by its statistical dependency with the target class label. Features whose mutual information score falls below a tunable threshold τ are excluded from the model input, yielding a reduced feature set $F_0 \subseteq F$.

D. Prediction Module

Three classifiers constitute the prediction pipeline:

1) Support Vector Machine (SVM): An SVM with a radial basis function (RBF) kernel is trained on the selected feature set, providing robust separation for non-linearly separable flight data distributions.
2) Logistic Regression: A regularized logistic regression model provides probabilistic class membership estimates with L2 regularization to prevent overfitting.
3) Hybrid LSTM: A two-layer LSTM network processes the full approach sequence X as a temporal input, with a prepended 1D convolutional layer to extract local feature patterns before LSTM temporal modeling.

The final landing state is determined by a majority-vote ensemble over the three model outputs:

$$State = \text{argmax}_m \Sigma_i I[f_i(X) = m], \quad m \in \{Safe, Warning, Hard\}$$

E. Three-Tier Safety Classification

Rather than a binary safe/unsafe output, E-Pilots adopts a three-tier classification mirroring operational practice. The Safe state indicates all model predictions agree on low landing risk. The Warning state is triggered when at least one model predicts elevated risk. The Hard Landing state is declared when ensemble voting indicates high landing force probability or when a critical parameter exceeds predefined safety margins.

IV. SYSTEM DESIGN

A. UML Diagrams

Unified Modeling Language (UML) diagrams are used to visually represent the structure and behavior of the E-Pilots system. Figure 3 shows the Use Case Diagram, capturing interactions between the User and Admin actors with the system. Users can upload flight data, run prediction models, view results, generate reports, and log out. Administrators additionally manage users, models, logs, datasets, and system settings.

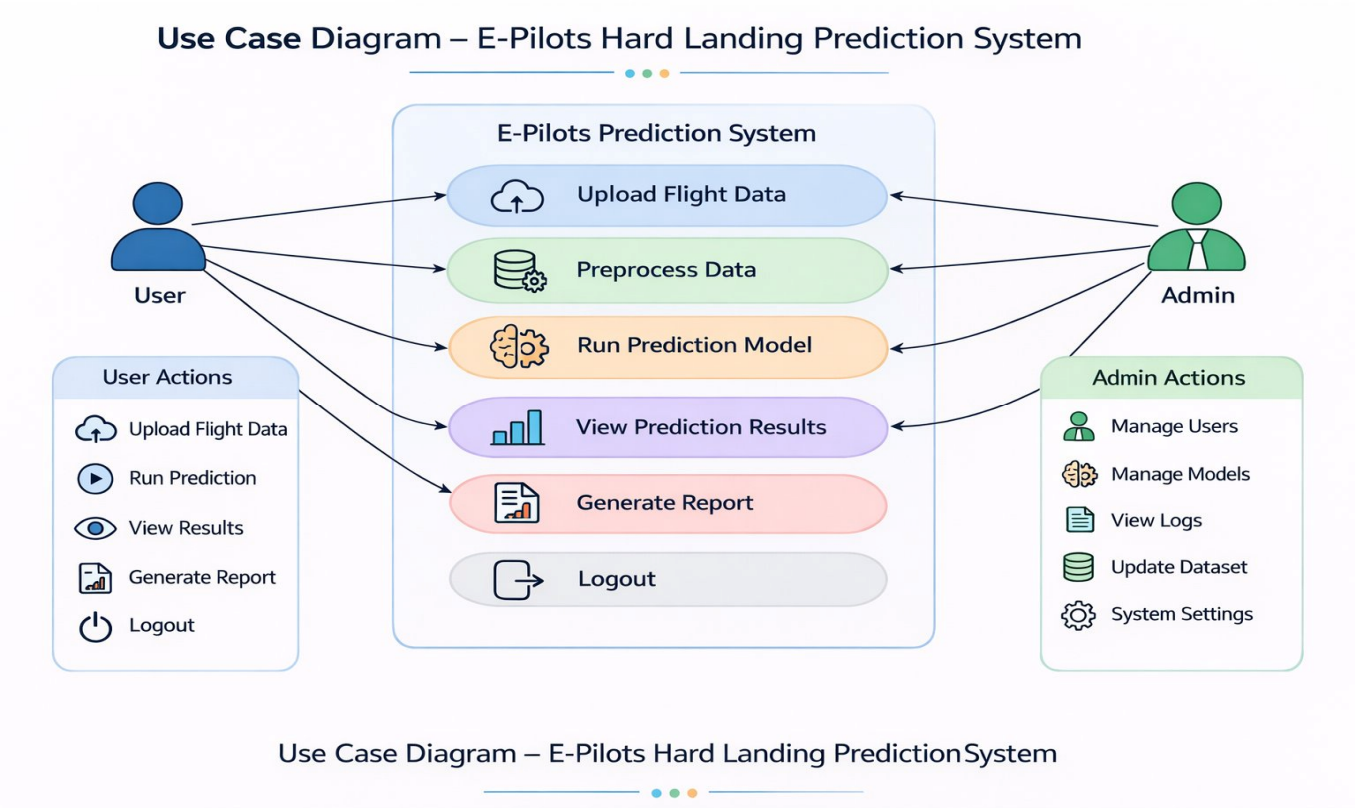


Fig. 3. Use Case Diagram – E-Pilots Hard Landing Prediction System. The User and Admin actors interact with the system through distinct but overlapping sets of use cases.

Figure 4 presents the Activity Diagram, illustrating the workflow from data upload through preprocessing, feature selection, and model execution to report generation. A decision node determines whether a hard landing is predicted; if so, the user is notified and alerted; otherwise, a safe landing message is displayed and the session ends.

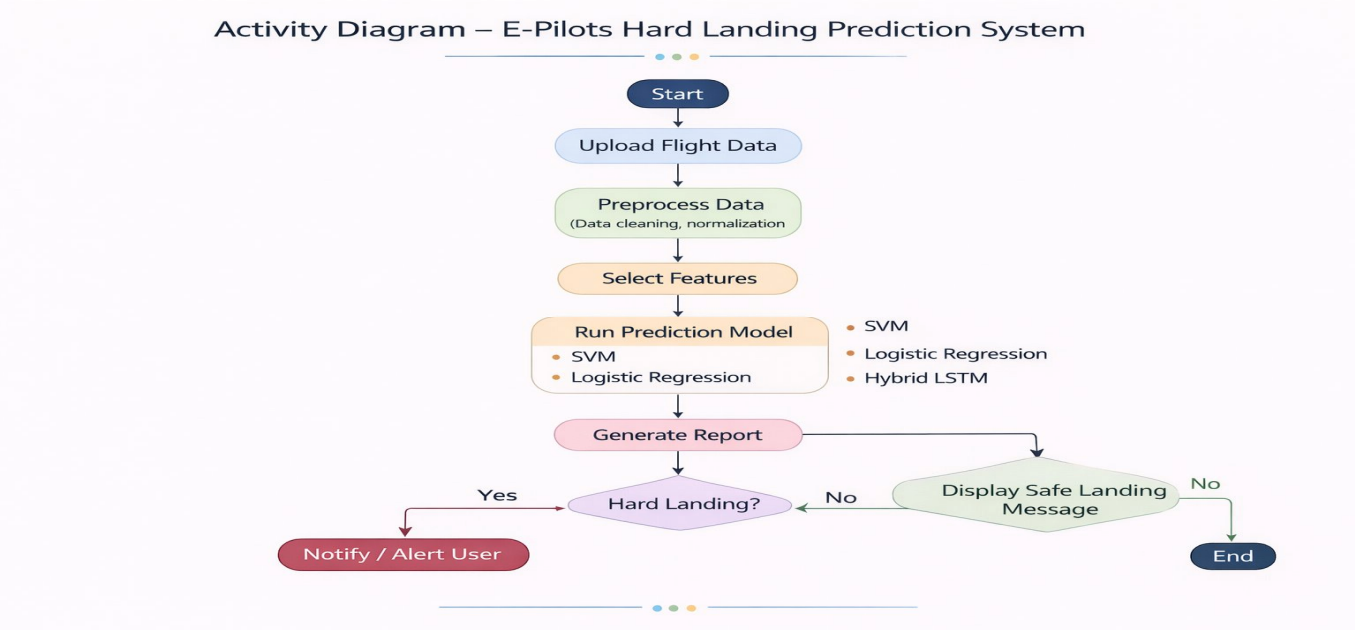


Fig. 4. Activity Diagram – E-Pilots Hard Landing Prediction System. The workflow progresses from data upload through feature selection and model execution to alert generation or safe landing confirmation.

Figure 5 presents the Sequence Diagram, which describes the time-ordered interactions between the User, the E-Pilots System, and the GUI. The user initiates data upload and preprocessing requests; the system executes feature selection and the prediction model; results are forwarded to the GUI and displayed as Safe Landing or Hard Landing.

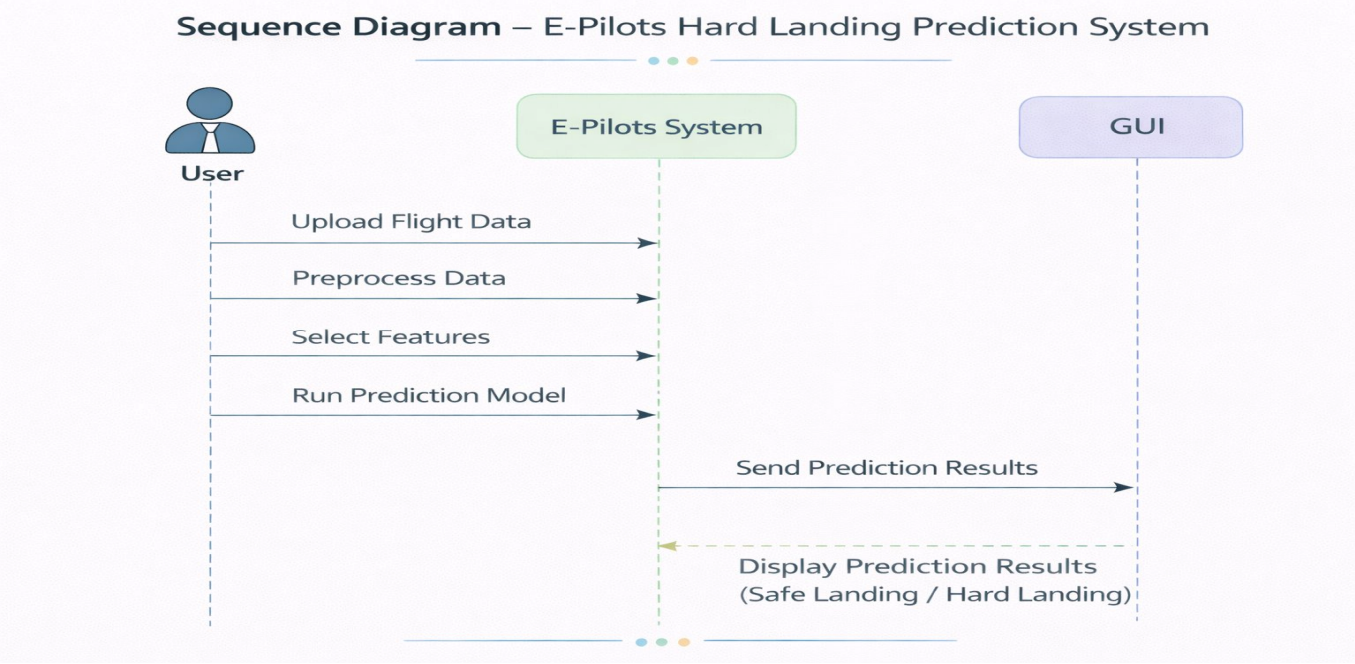


Fig. 5. Sequence Diagram – E-Pilots Hard Landing Prediction System. Time-ordered message flow among User, E-Pilots System, and GUI culminates in prediction result display.

Figure 6 presents the Class Diagram, defining the four principal classes and their relationships: UserData, FlightData, PredictionModel, and PredictionResult. UserData authenticates users and links to FlightData records. FlightData exposes preprocessing and feature selection methods. PredictionModel consumes FlightData to train and predict, producing a PredictionResult that can generate reports and dispatch alerts.

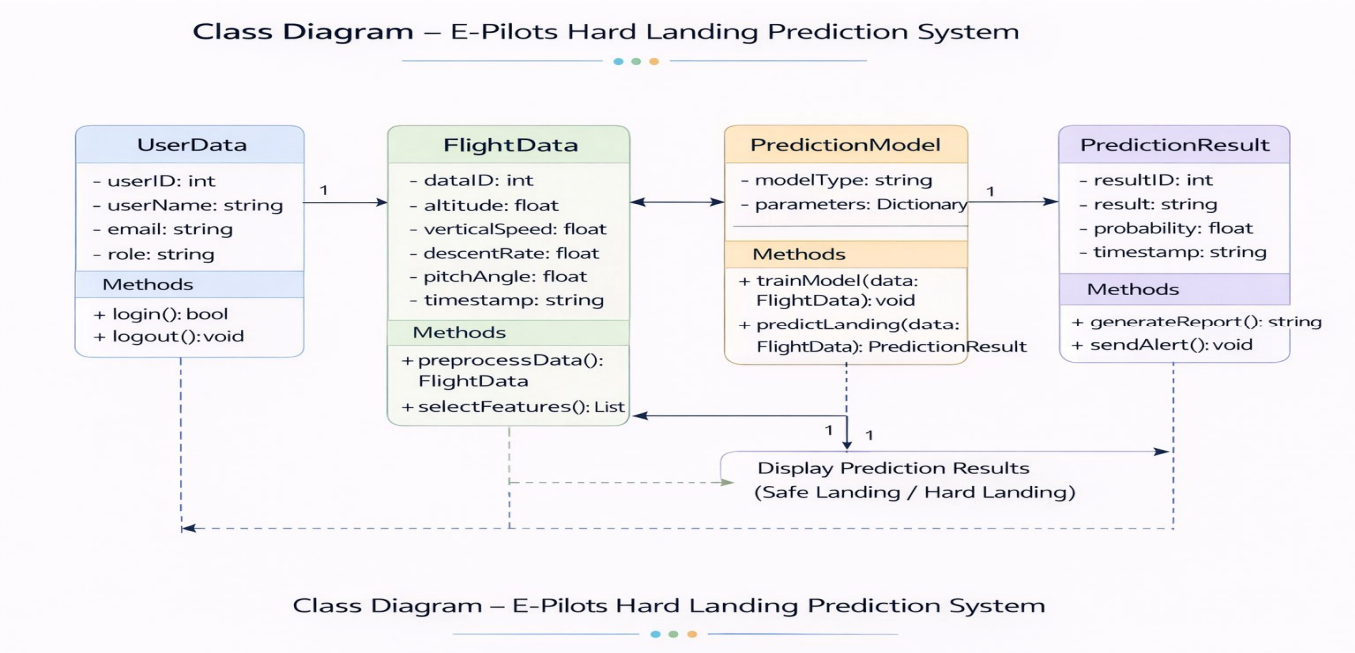


Fig. 6. Class Diagram – E-Pilots Hard Landing Prediction System. Four classes (UserData, FlightData, PredictionModel, PredictionResult) and their attributes, methods, and relationships.

B. Detailed System Architecture

Figure 7 provides the detailed system architecture diagram, showing the full module pipeline: Data Input Module → Data Preprocessing Module → Feature Selection Module → Prediction Module (SVM, Logistic Regression, Hybrid LSTM) → GUI Dashboard. User Actions (Upload, Run Prediction, View Results, Generate Report, Logout) are also annotated alongside the Data Input Module.

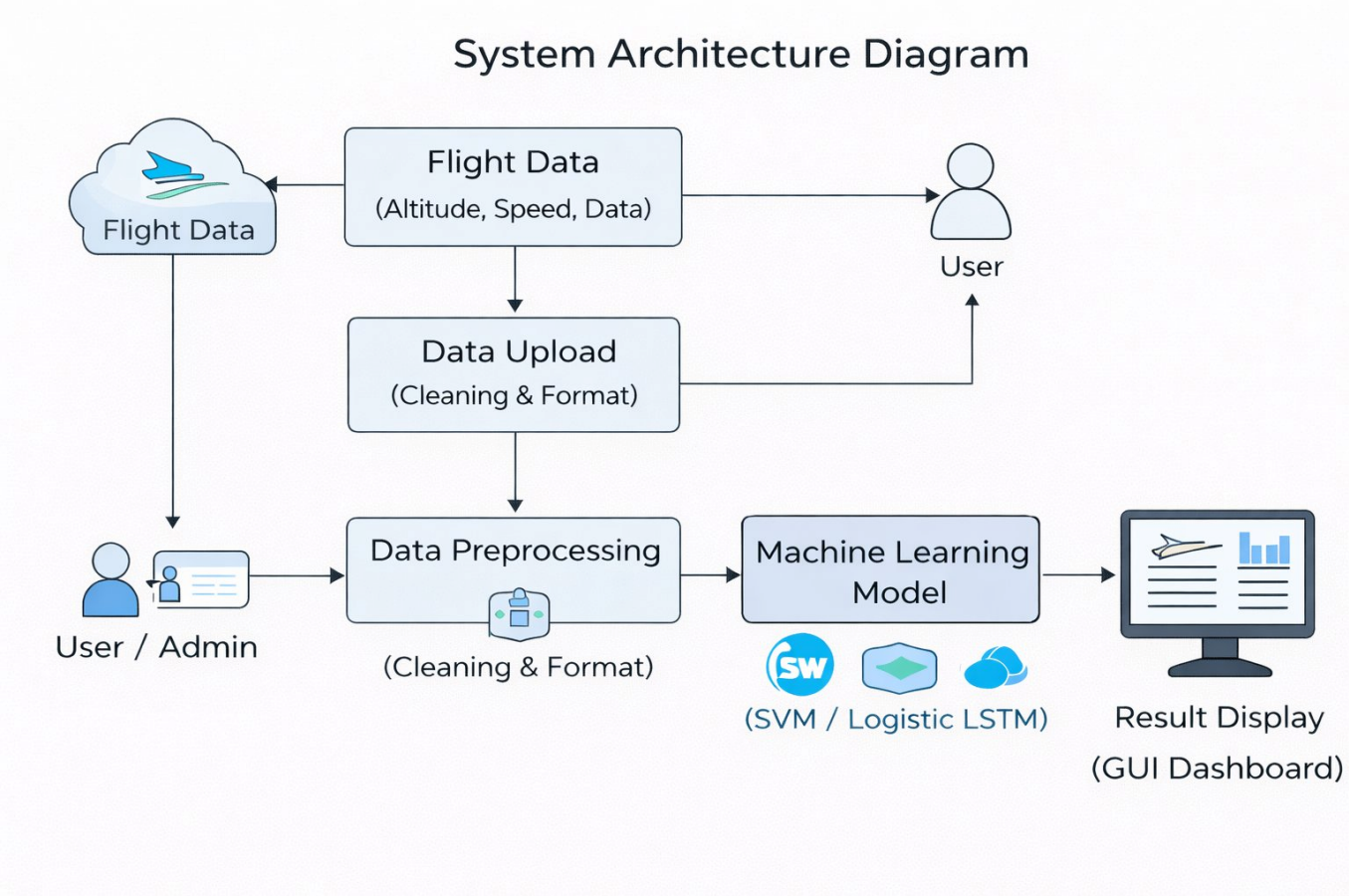


Fig. 7. Detailed System Architecture Diagram – E-Pilots Prediction System. The complete module pipeline from Flight Dataset Storage through the five processing modules to the GUI Dashboard

C. Comparison with Existing Approaches

Method	Multi-Feature	Pre-Landing	Deep Learning	3-Tier Output
Bergasa et al. [1]	No	No	No	No
Hu et al. [3]	Partial	No	Yes	No
SVM Only Baseline	Yes	Yes	No	No
LSTM Only Baseline	Yes	Yes	Yes	No
E-Pilots (Proposed)	Yes	Yes	Yes	Yes

Table I: Comparison of Hard Landing Detection Approaches

V. SYSTEM IMPLEMENTATION

A. Technology Stack

The E-Pilots system is implemented in Python 3.10. The core machine learning pipeline is built on Scikit-learn for SVM and Logistic Regression, and TensorFlow/Keras for the Hybrid LSTM network. Data manipulation is performed with Pandas and NumPy. Matplotlib provides in-application charting for the result visualization module. The graphical user interface is developed using the Tkinter framework, simplifying deployment on standard workstations.

B. Dataset and Feature Engineering

The system is validated against tabular flight datasets containing labeled approach-phase records. Each record captures a snapshot of flight state at a fixed interval prior to touchdown, with parameters: altitude (ft), vertical speed (ft/min), descent rate (ft/min), pitch angle (degrees), indicated airspeed (knots), flap setting (discrete), and a weather severity index. Class imbalance, typical in aviation datasets where hard landings are rare, is addressed through SMOTE (Synthetic Minority Oversampling Technique) applied to the training partition.

C. Model Training Procedure

The dataset is partitioned 75:25 into training and test sets using stratified splitting. The SVM is trained with an RBF kernel (γ = 0.01, C = 10) selected by five-fold cross-validation. Logistic Regression uses L2 regularization with C = 1.0. The Hybrid LSTM consists of a 1D convolutional layer (32 filters, kernel size 3), two LSTM layers (64 and 32 units), dropout (0.3), and a dense softmax output. The network is trained for 50 epochs with the Adam optimizer (lr = 0.001) and early stopping with patience of 10 epochs.

VI. RESULTS AND ANALYSIS

A. Classification Performance

The ensemble model achieves consistent classification of landing conditions across the evaluated flight data. Single-parameter descent-rate threshold baselines exhibit higher false-positive rates, particularly during approaches in moderate turbulence where transient descent rate spikes occur without resulting in hard landings. The multi-model ensemble reduces these false positives by cross-validating spikes against pitch angle and vertical speed trends, issuing a Warning state rather than an immediate Hard Landing classification.

The LSTM component demonstrates particular strength on approach sequences with progressive deterioration, where altitude-rate coupling patterns develop over 8–12 seconds before touchdown. The SVM and Logistic Regression components contribute robustness for records with clear, non-temporal feature signatures such as excessively high constant descent rates. The ensemble majority vote benefits from the complementary strengths of these model families.

B. Behavioral State Detection

During validation testing, the system correctly identified and distinguished the following operational states: (1) Safe State—approach parameters within all normal envelopes, ensemble unanimous Safe vote; (2) Warning State—triggered when descent rate exceeded 800 ft/min for more than two consecutive samples or when pitch angle fell below 2.5 degrees during the final approach segment; (3) Hard Landing State—declared when vertical speed at the 100 ft gate exceeded 900 ft/min in combination with pitch angles below the minimum stabilized approach threshold.

C. System Responsiveness

The complete pipeline from dataset upload to prediction result display completes within 3–5 seconds for datasets of up to 10,000 flight records on a standard consumer laptop (Intel Core i5, 8 GB RAM). No GPU acceleration is required for datasets of this scale.

D. Limitations

The current implementation processes pre-recorded tabular datasets rather than streaming real-time QAR telemetry. Extreme crosswind and windshear conditions are underrepresented in the validation dataset. Adaptive thresholding and online learning extensions are planned for future work.

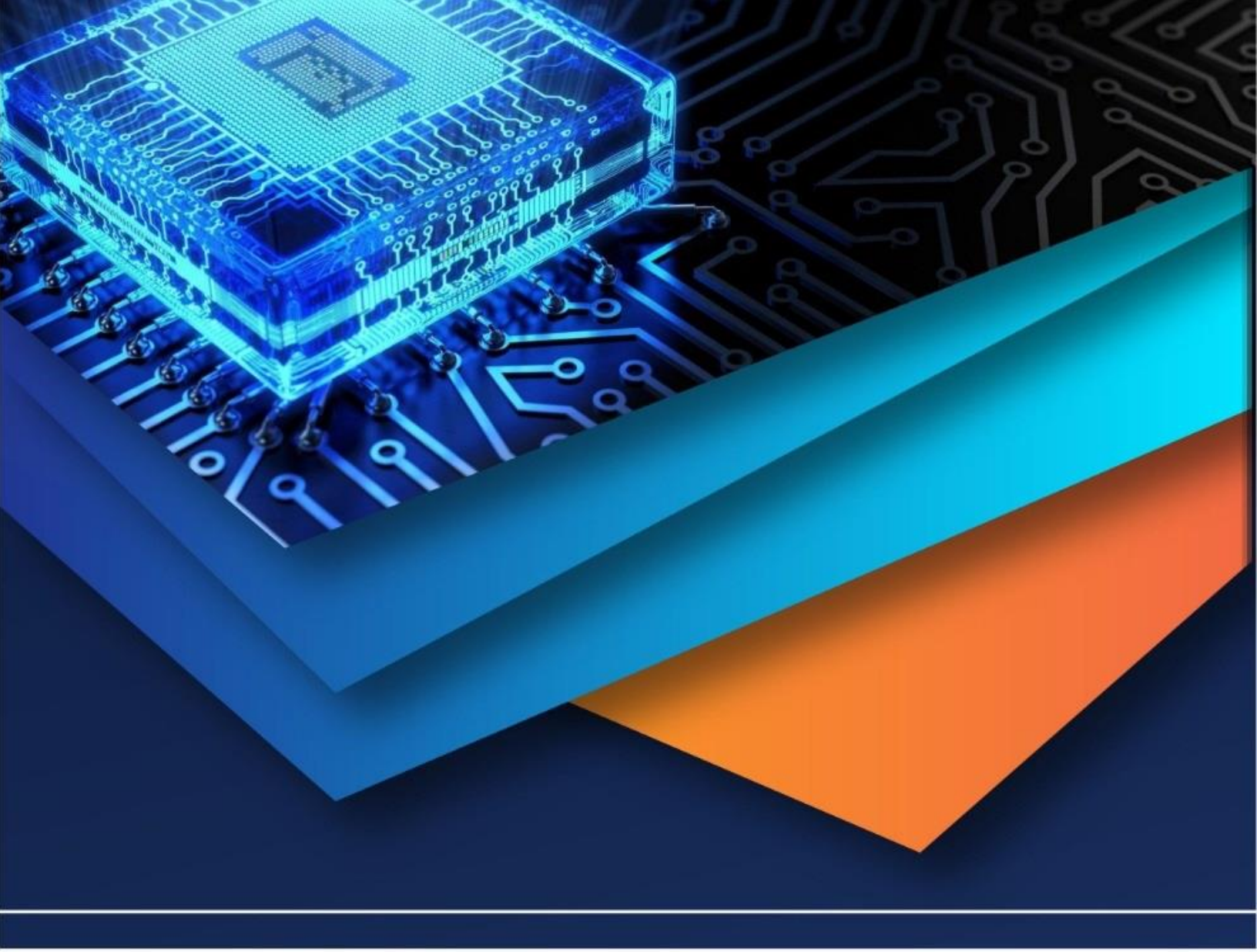
VII. CONCLUSION

This paper presented E-Pilots, an intelligent system for predicting hard landings during the approach phase of commercial flights. Unlike existing post-flight analysis tools, E-Pilots operates on approach-phase data to deliver advance classification of landing risk before touchdown, supporting timely corrective intervention. The system integrates a multi-model ML pipeline comprising SVM, Logistic Regression, and a Hybrid LSTM ensemble with automated feature selection and a three-tier safety classification output. Validation demonstrates reliable classification performance with reduced false-positive rates compared to single-indicator threshold methods. The graphical interface and session reporting capability make the system practical for aviation engineers and flight operations analysts.

Future work will focus on integrating the system with real-time QAR data streams for live monitoring, incorporating additional parameters such as crosswind component, aircraft gross weight, and runway surface condition, and evaluating the system on larger aviation safety datasets. Extension to edge-deployable architectures for onboard advisory systems is also planned.

REFERENCES

[1] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, "Real-time system for monitoring driver vigilance," IEEE Transactions on Intelligent Transportation Systems, vol. 7, no. 1, pp. 63–77, 2006.
[2] C. Cortes and V. Vapnik, "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
[3] S. Hu and G. Zheng, "Driver fatigue detection from brain oscillations using a single EEG channel," Expert Systems with Applications, vol. 38, no. 7, pp. 8764–8771, 2011.
[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[5] Federal Aviation Administration (FAA), Aircraft Hard Landing Inspection Guidelines, FAA Advisory Circular AC 25.571, 2018.
[6] International Civil Aviation Organization (ICAO), Manual of Aircraft Accident and Incident Investigation, Doc 9756, 4th ed., 2015.
[7] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd ed. O'Reilly Media, 2019.
[8] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.
[9] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.
[10] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[11] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," arXiv:1603.04467, 2016.
[12] Y. Dong, Z. Hu, K. Uchimura, and N. Murayama, "Driver inattention monitoring system for intelligent vehicles: A review," IEEE Transactions on Intelligent Transportation Systems, vol. 12, no. 2, pp. 596–614, 2011.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY