PrognosAI: AI-Driven Predictive Maintenance System Using Time-Series Sensor Data

Prepared by: Shivani Sharma

Dataset: NASA Turbofan Jet Engine (CMAPSS)

Abstract

PrognosAI is an advanced AI-based maintenance prediction framework designed to determine the Remaining Useful Life (RUL) of turbofan engines through the analysis of time-series sensor information. The system utilizes deep learning algorithms, intelligent alert mechanisms, and interactive dashboards to deliver clear, data-driven insights. This approach helps improve maintenance scheduling, reduce unexpected failures, and increase overall operational efficiency.

PROJECT OVERVIEW AND GOALS

Predictive maintenance focuses on anticipating potential equipment malfunctions before they happen, helping to reduce downtime and enhance efficiency. This project applies AI-based techniques to improve engine health monitoring and optimize maintenance strategies.

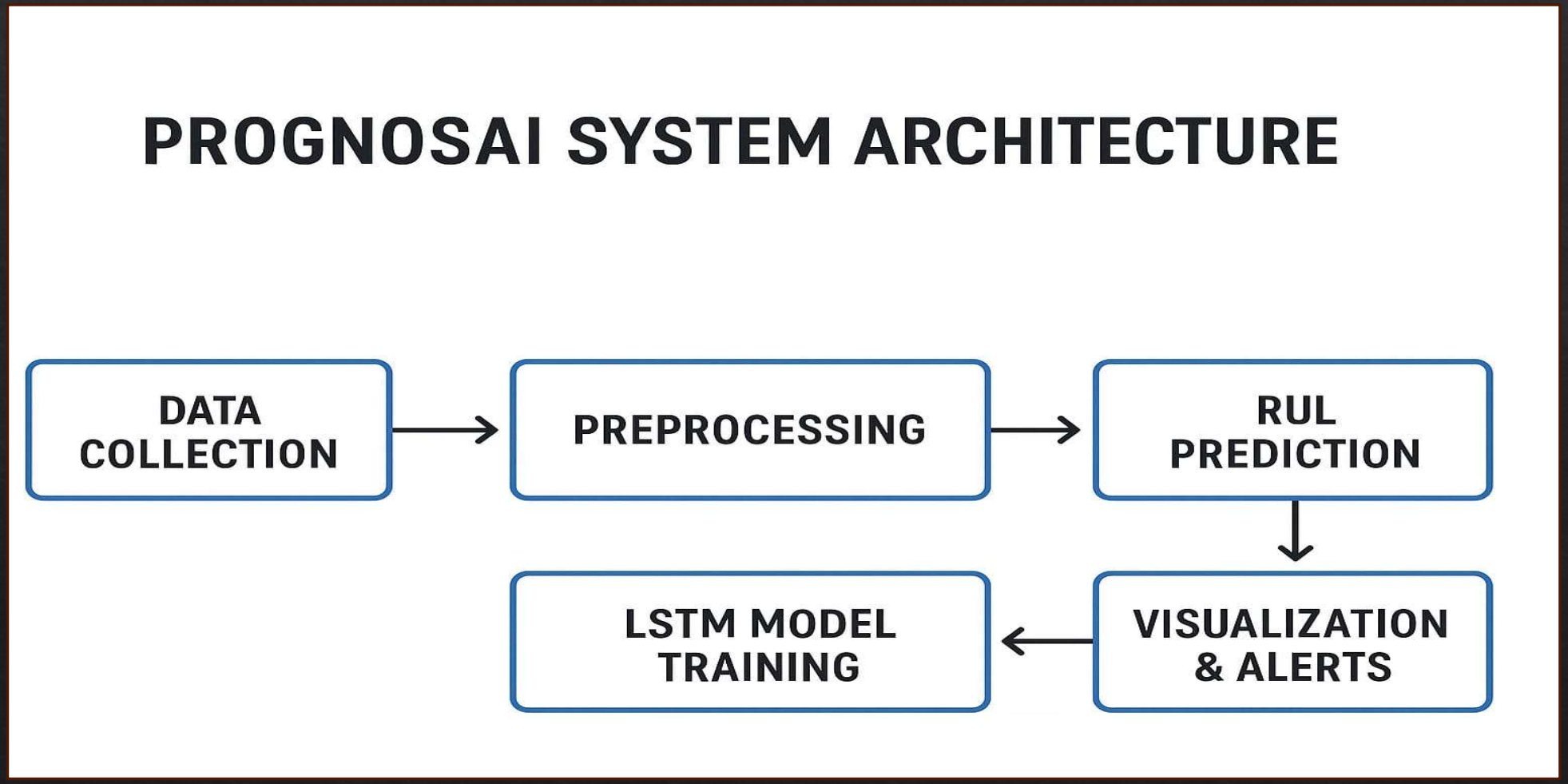
Goals:

- Build an LSTM model to accurately estimate Remaining Useful Life (RUL).
- Implement an intelligent alert mechanism for timely maintenance.
- Create a live visualization dashboard to display predictive insights effectively.

Dataset Description

- Source: The dataset used in this project is the NASA CMAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset.
- Attributes: It contains readings from 21 engine sensors along with 3 operational condition variables.
- Instances: Data is collected from multiple turbofan engine units, categorized into subsets FD001 to FD004.
- Target Variable: The primary prediction goal is the Remaining Useful Life (RUL) of each engine.

System Architecture



Milestone 1 – Data Preparation

- Imported the **NASA CMAPSS dataset** for model training.
- Computed the **Remaining Useful Life (RUL)** for each engine at every cycle.
- Performed **data normalization** using the **MinMaxScaler** technique to bring values within a common range.
- Created **time-series window sequences** to prepare structured input data suitable for the LSTM model.
- Utilized Python libraries including **pandas**, **NumPy**, and **scikit-learn (preprocessing module)** for data handling and scaling.

Milestone 2 – Model Development

- Designed an **LSTM-based neural network** using the **Keras Sequential framework**.
- The network structure included **LSTM (128 units)** followed by **Dropout** layers and **Dense layers** with sizes **64, 32, and 1** for final output prediction.
- Adopted the **Adam optimizer** with **Mean Squared Error (MSE)** as the loss function to ensure stable training.
- Implemented **5-Fold Cross-Validation** to validate model performance and reduce overfitting.
- Stored the **trained model** along with the **data scalers** for future deployment and inference.
- Employed essential libraries like **TensorFlow**, **Keras**, and **Scikit-learn** for model creation and evaluation.

Milestone 3 – Evaluation and Alert System

- The model achieved an **R^2 score above 0.95**, indicating excellent predictive accuracy, along with **very low RMSE and MAE** values.

- Developed a **dynamic alert mechanism** based on RUL percentage levels:

  Critical: Remaining Useful Life ≤ 20%
  Warning: 20% < RUL ≤ 50%
  Normal: RUL > 50%

- Created **automated alert summaries** and **interactive visualization reports** to display engine health conditions.

- Utilized libraries such as **NumPy**, **Pandas**, **Scikit-learn (metrics)**, **Matplotlib**, and **Plotly** for data analysis and visualization.

Milestone 4 – Interactive Visualization Dashboard

- Built an **interactive web dashboard** using **Streamlit** for real-time visualization and user interaction.
- Enabled users to **upload test data files** in **CSV or TXT format** for evaluation.
- Displayed **engine-specific RUL predictions** along with **corresponding alert levels** in an intuitive layout.
- Integrated options to **download prediction results** and **adjust alert thresholds** dynamically.
- Utilized essential libraries including **Streamlit**, **Pandas**, **NumPy**, **Plotly**, **Joblib**, and **TensorFlow Keras** for backend processing and visualization.

PrognosAI Engine Health Dashboard

Avg RUL	Critical Engines	Total Files
9.8	47	4

Engine Life Summary Table

Engine_ID	Elapsed_Days	Remaining_Days	Total_Days	Remaining_Years	Alert
88	89	142	150	0.0271	CRITICAL
89	114	145	130	0.0271	CRITICAL
90	133	131	248	0.0672	CRITICAL
91	139	133	234	0.0272	CRITICAL
92	97	143	212	0.0715	CRITICAL
93	136	197	206	0.0251	CRITICAL
94	124	144	135	0.0171	CRITICAL
95	97	110	121	0.0327	CRITICAL
96	121	121	143	0.0781	CRITICAL

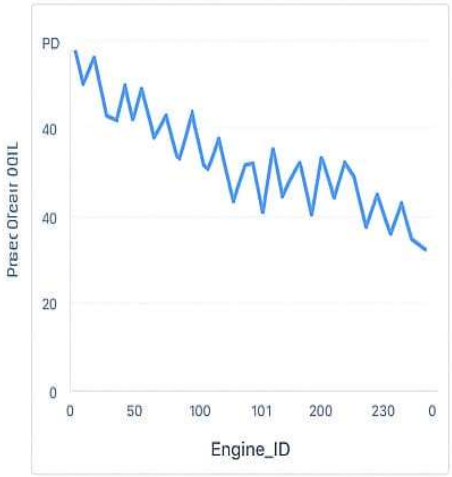
Remaining Useful Life

Select Test File

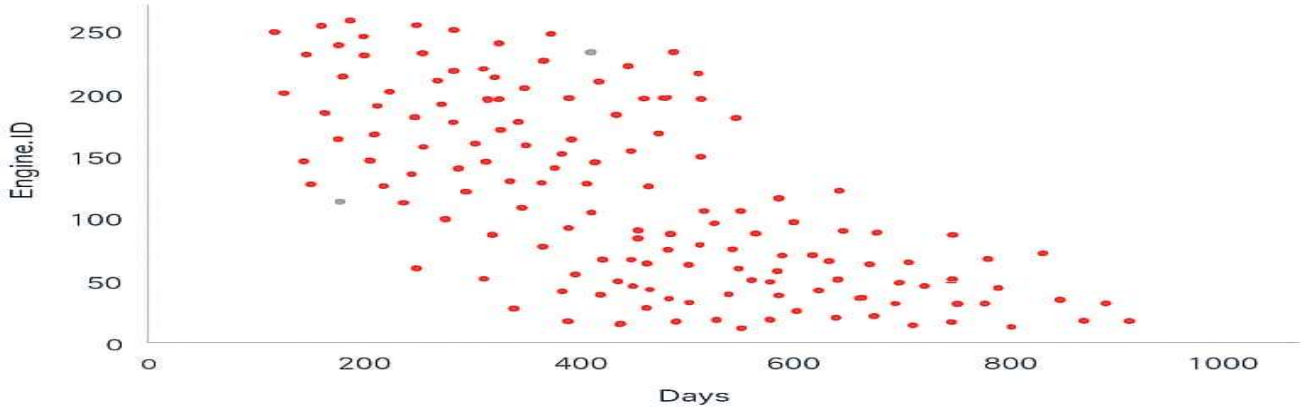
All Files

Process

Test File Settings



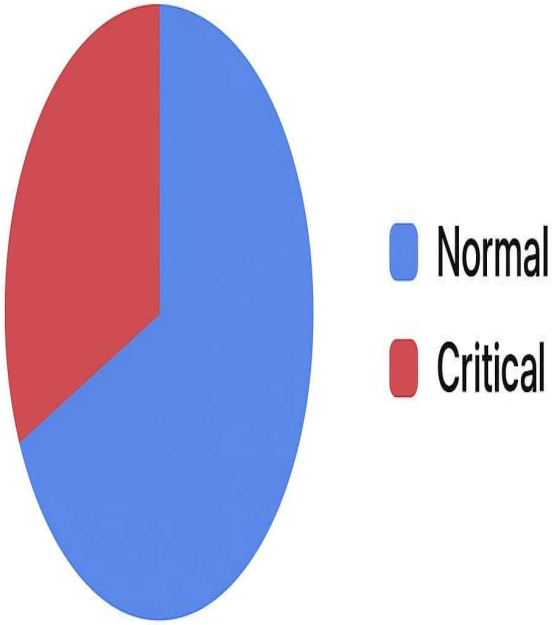
Engine Life Span Overview



RUL Comparison

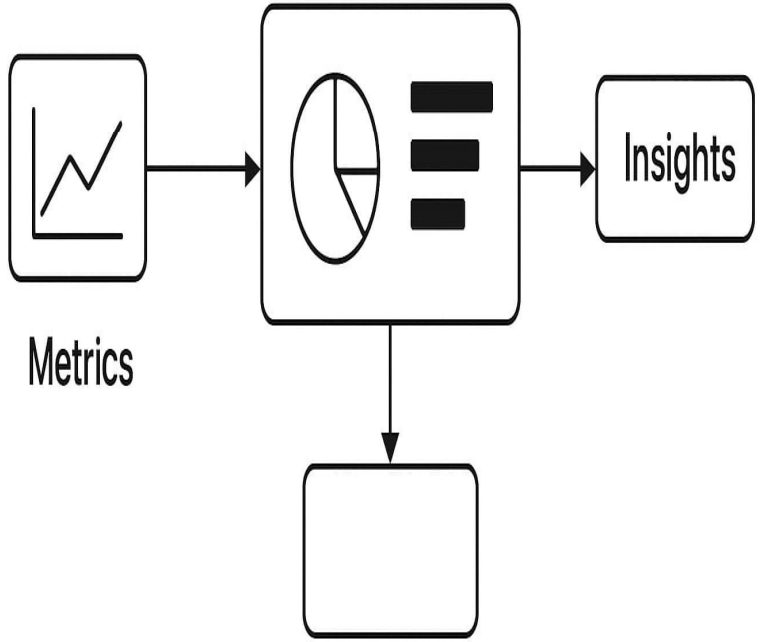
PrognosAI

Alert Distribution



PrognosAI

Dashboard Visualization



MODEL EVALUATION AND PERFORMANCE

Evaluation Metrics:

- **R^2 Score:** Above 0.95 (high model accuracy)
- **RMSE:** Significantly low, indicating minimal prediction deviation
- **MAE:** Low, confirming consistent prediction reliability

Key Outcomes:

- The **LSTM model** demonstrated **strong and stable performance** during testing.
- Produced **highly precise RUL estimations** with minimal error rates.
- The **alert mechanism** effectively supported **timely maintenance decisions** based on prediction levels.

RESOURCES USED

- **Dataset:** NASA CMAPSS (Commercial Modular Aero-Propulsion System Simulation)
- **Programming Language:** Python 3.10
- **Development Tools:** Jupyter Notebook, Visual Studio Code, Streamlit.
- **Libraries & Frameworks:** TensorFlow, Keras, Pandas, NumPy, Scikit-learn, Matplotlib, Plotly, Joblib, ReportLab.
- **Documentation & Guides:**
  - TensorFlow & Keras Official Docs
  - Scikit-learn API Reference
  - Streamlit and Plotly User Guides
  - NASA Dataset Description Portal

Thank You