Finding Early Signs of Structural Fatigue Using Vibration Signals

Capstone Analytics Practicum Final Report

MGT 6748

Diane Egret – degret3@gatech.edu
Sponsor: Sandia National Laboratories
Summer 2025

Abstract— Structural fatigue is a progressive failure mechanism that poses significant risks in critical systems such as aircraft engines and industrial turbines. Early detection of fatigue through predictive maintenance can reduce operational costs, prevent unexpected breakdowns, and improve overall safety. This project investigates the use of vibration signal data to predict structural degradation and estimate the Remaining Useful Life (RUL) of components using machine learning and deep learning techniques. Leveraging a subset of NASA's C-MAPSS dataset, we experimented with both traditional regression models and advanced sequence-based architectures, including Gated Recurrent Units (GRUs) and attention-based Long Short-Term Memory (LSTM) networks. Our findings show that while non-sequential models perform mildly, some deep learning models that capture temporal dependencies can improve predictive accuracy. The best results were obtained using an attention-enhanced LSTM, which also offers interpretability by identifying critical time steps in the degradation process. Additionally, a binary classification model was developed to flag components nearing failure, offering a potential secondary signal for real-time decision-making. This report documents the full modeling pipeline, highlights lessons learned, and offers recommendations for future work in this domain.

1 INTRODUCTION

Structural fatigue refers to the weakening and eventual failure of materials or mechanical components due to repeated loading and stress over time. It is a significant concern in industries such as aerospace, energy, and transportation, where unexpected equipment failure can have catastrophic consequences. As a result, predictive maintenance has emerged as a critical focus area for both engineering and data science. By anticipating failures before they occur, organizations can reduce maintenance costs, avoid downtime, and extend the lifespan of critical assets.

This practicum project explores how machine learning and deep learning can be used to detect early signs of structural fatigue using vibration signals. The primary goal was to develop a model that could accurately estimate the Remaining Useful Life (RUL) of machinery components using sequential sensor data. The project uses the FD001 subset of the NASA C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset. The approach involved experimenting with various modeling strategies, starting with traditional machine learning regressors and culminating in deep learning models capable of learning temporal dependencies. This report provides a comprehensive overview of the modeling journey, highlighting both successes and failures, and presents final recommendations based on performance metrics and interpretability.

2 LITERATURE REVIEW

Numerous studies have been conducted on data-driven RUL prediction using the C-MAPSS dataset. Saxena and Goebel (2008) introduced the dataset and demonstrated its use for benchmarking prognostics algorithms. Since then, it has become a standard in the predictive maintenance community.

Babu et al. (2016) proposed a deep convolutional neural network for RUL prediction and showed that deep learning models could outperform traditional techniques. Their work emphasized the importance of learning hierarchical features directly from raw sensor data.

Zheng et al. (2017) introduced a hybrid model combining Convolutional Neural Networks (CNNs) and LSTMs. The CNN component extracted spatial features from the data, while the LSTM captured temporal dependencies. Their model achieved state-of-the-art performance on several C-MAPSS subsets, reinforcing the effectiveness of hybrid architectures.

Recent advances have focused on attention mechanisms. Wu et al. (2021) demonstrated that self-attention could be used to enhance LSTM models by allowing them to assign varying importance to different time steps. Attention not only improved performance but also enhanced model interpretability.

Other notable contributions include the use of physics-informed neural networks and ensemble learning methods. Researchers have also explored transfer learning approaches to generalize models across different fault modes and operating conditions. These developments highlight the growing sophistication of data-driven prognostics and the importance of model interpretability in industrial settings.

This literature review motivated me to compare traditional machine learning techniques with deep learning approaches. In particular, it was my first exposure to LSTM models, and I was eager to experiment with them. Where this project differs from many existing studies is in its focus on detecting early signs of structural fatigue rather than solely predicting the point of total failure.

3 DATASET DESCRIPTION

The C-MAPSS dataset was developed by NASA to support the development of prognostics and health management systems. It simulates realistic run-to-failure scenarios for jet engines under different operating conditions and fault modes. The FD001 subset, selected for this project, includes 100 engines operating under one consistent condition and exhibiting one fault mode.

The dataset is composed of three main files:

- *train_FD001.txt*: Contains run-to-failure sensor readings for each engine. Each engine begins operation in a healthy state and degrades over time until failure.
- *test_FD001.txt*: Contains truncated sensor readings for a separate set of engines. The engines in the test set are still operational at the end of the recordings.
- *RUL_FD001.txt*: Provides the true remaining useful life for each engine in the test set, to be used for model evaluation.

Each row in the dataset represents one time cycle of an engine and includes the engine unit number, time cycle, three operational settings, and readings from 21 sensors. These sensors capture various aspects of engine performance such as temperature, pressure, and vibration. The complete list of sensors and their description is available in the Appendix. A diagram of a turbofan engine is also attached in the Appendix to help the reader visualize what parts are being monitored.

The prediction task is to estimate the RUL of each engine based on its sensor history. In the training data, the RUL is calculated as the number of cycles remaining before failure. In the test data, the task is to extrapolate future degradation and predict how many cycles remain until failure.

4 DATA PREPROCESSING

To ensure effective model training, preprocessing was required. The first step was calculating the RUL for each data point in the training set. This was done by finding the maximum cycle number for each engine and subtracting the current cycle. For the test set, true RUL values were obtained directly from the provided RUL file.

Sensor readings were then normalized using min-max scaling to ensure that all features contributed equally to the model. Several sensors were found to provide little useful information, either showing little variance or duplicating information from other sensors and were removed from the dataset.

The data was then restructured into fixed-size sequences to be used as input for recurrent neural networks. For each engine, overlapping windows of a fixed length (e.g., 30-time steps) were created, with the label corresponding to the RUL at the final time step in the window. This reshaping allowed the deep learning models to learn from the temporal evolution of sensor signals.

An exploratory binary classification label was also created by setting a threshold on RUL (e.g., 30 cycles). Data points with RUL below this threshold were labeled as 'critical,' allowing for experimentation with classification models that predict imminent failure rather than exact RUL.

5 EXPLORATORY DATA ANALYSIS

Exploratory data analysis (EDA) was conducted to gain insights into the structure and variability of the data. Sensor readings were plotted over time for individual engines to observe how values changed as engines approached failure (Figure 1). Note that Tableau Desktop was utilized here for ease of compiling numerous figures. Future plots will all be generated with Python. We can observe below that some sensors showed clear signs of degradation, while others remained stable throughout the engine’s life.

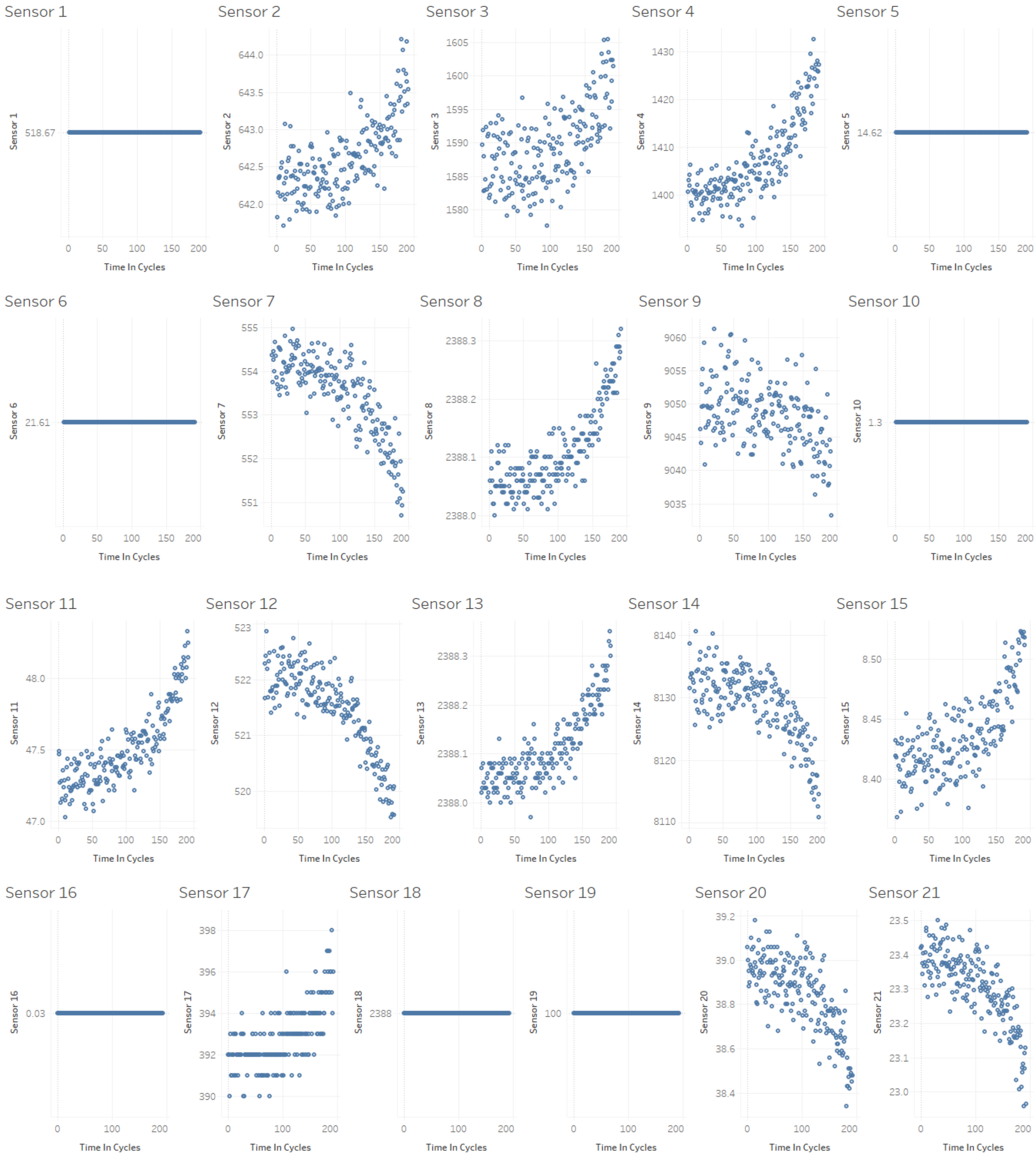

Figure 1—Sensor trends over time for a sample engine (Unit number: 1)

A correlation matrix, Figure 2, was constructed to identify high degrees of linear correlation among all 21 sensor readings. Features with strong linear dependencies (>95% correlated) were removed to reduce redundancy.

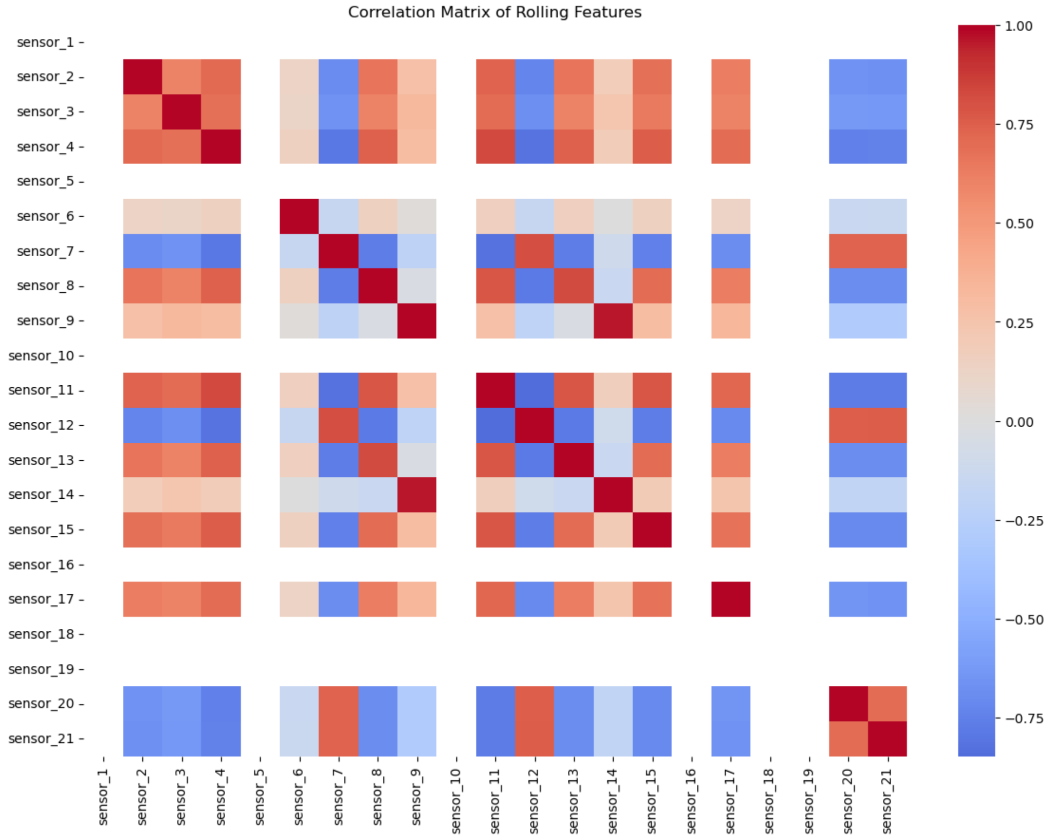

Figure 2 — Correlation matrix of normalized sensor data

Principal Component Analysis (PCA) was used to visualize the underlying structure of the dataset and to examine whether engines could be distinguished based on dominant signal patterns (Figure 3).

The figure below plots the first two principal components of the normalized sensor data. Each point represents a cycle, colored by engine unit. This illustrates the variance captured by PCA and highlights how different engine behaviors cluster in the feature space.

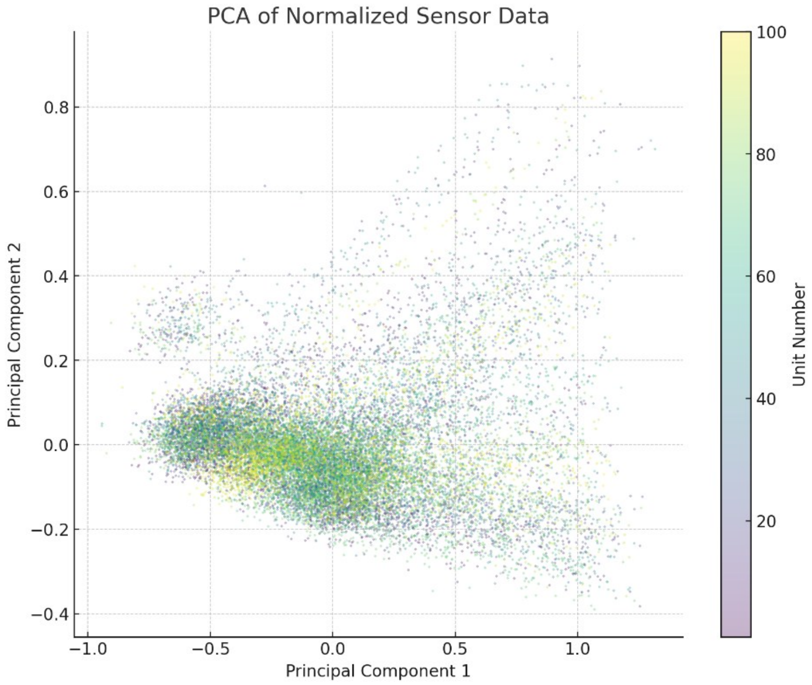

Figure 3—Principal component analysis of normalized sensor data

These visual tools informed decisions on feature selection and confirmed the need for temporal modeling.

The exploratory analysis did show that degradation was not uniform across all engines. Different engines showed distinct degradation trajectories, even under the same operating condition. This reinforced the importance of capturing sequence-based patterns rather than relying on static models.

6 MODELING APPROACHES

6.1 Baseline Modeling Attempts

Initial modeling efforts focused on traditional machine learning algorithms, beginning with a Random Forest Regressor. This model was trained on a flattened version of the data, where a fixed number of recent sensor readings were used as input features. As can be observed in Table 1 below, the initial Random Forest model had a mild performance.

Table 1 — Performance metrics: Initial Random Forest model

RMSE	MAE	R^2
46.32	35.00	0.3831

RMSE (Root Mean Squared Error) penalizes larger errors, and considering RUL has values up to 150, then 46.32 is reasonable. An MAE (Mean Absolute Error) of 35 entails that, on average, predictions differed by 35 cycles. Finally, an R^2 of 0.3831 means that the model explains ~38% of the variance in the target variable. I some circumstances that can seem low, but in our case, where the data is simulated to be realistic, 38% of explained variance is quite good.

To improve performance, Principal Component Analysis (PCA) was applied to reduce the dimensionality of the input features before training the Random Forest. While this helped reduce training time and overfitting, the accuracy remained moderate, and slightly lower than the full model (Table 2).

Table 2 — Performance metrics: Reduced (PCA) Random Forest model

RMSE	MAE	R^2
47.64	35.96	0.3474

Finally, a Random Forest with hyperparameter tuning was executed with a grid search. Once again, the accuracy remained mild, and the process was computationally intensive.

Table 3 — Performance metrics: Tuned Random Forest model

RMSE	MAE	R^2
46.55	34.97	0.3770

These early results suggested that traditional machine learning models were somewhat satisfactory for this task but show room for improvement. Temporal dependencies that characterize degradation processes may not be adequately captured.

These baseline experiments served an important purpose by illustrating the strengths and limitations of non-sequential modeling in predictive maintenance. They helped justify the transition to deep learning architectures that are explicitly designed to handle time-series data.

6.2 Deep Learning Models

Recognizing the need for temporal modeling, the project shifted focus to recurrent neural networks. Two architectures were explored in depth: Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks.

Hyperparameter optimization was conducted using Optuna, an automatic hyperparameter tuning framework. A search space was defined for parameters including learning rate, hidden layer size, batch size, dropout rate, and the number of training epochs. The best-performing configuration was determined based on validation loss.

Hyperparameter tuning can help achieving optimal performance. Without tuning, models are prone to overfitting or underfitting, depending on the initial configuration.

The GRU model was designed to capture long-term dependencies in the sensor sequences. It took three-dimensional input data, representing sequences of sensor readings for each engine. The model obtained the results displayed below in Table 4.

Table 4 — Performance metrics: GRU model

RMSE	MAE	R^2
50.41	36.88	0.27

To attempt further enhancing performance, an attention-based LSTM model was developed. The attention mechanism allowed the model to dynamically focus on the most informative time steps within each sequence. This model demonstrated better generalization and produced lower RMSE and MAE values than the GRU, and a higher R^2 (Table 5).

Table 5 — Performance metrics: Attention-LSTM model

RMSE	MAE	R^2
47.51	34.62	0.40

The inclusion of attention also improved the interpretability of the model, enabling the identification of critical time steps that contributed most to the RUL prediction.

The following plots offer visualization and comparison of the training and results of both deep-learning models (Figures 4 and 5).

Figure 4 shows the Mean Squared Error (MSE) loss curves during training for both the GRU and Attention-based models across 20 epochs. Both models demonstrate rapid initial convergence, with the Attention model achieving slightly lower loss in the later epochs.

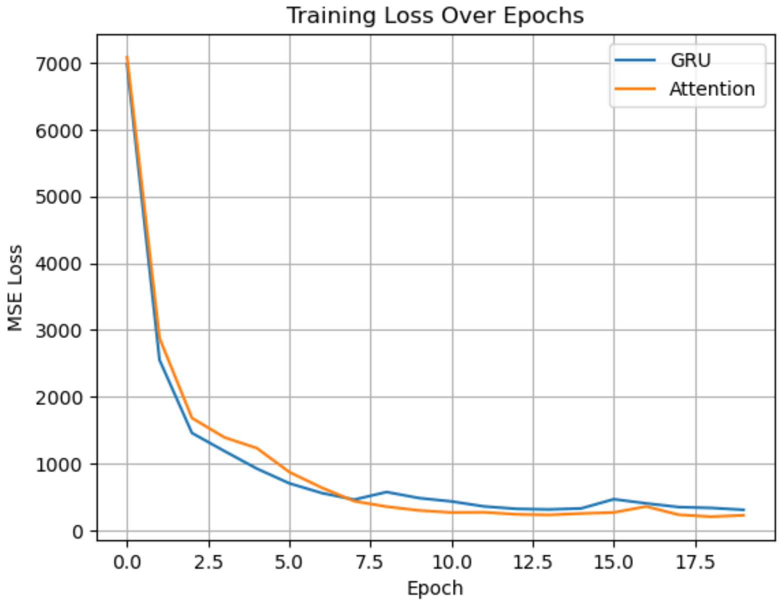

Figure 4 — Training loss over epochs for GRU and Attention models

Figure 5 compares predicted RUL values from the GRU and Attention-based models against the ground truth RUL for the first 200 test samples. The true RUL is shown as a dashed line for reference. While both models follow the general degradation trend, the Attention model shows smoother transitions and reduced noise, suggesting better generalization in early-stage prediction compared to the GRU baseline.

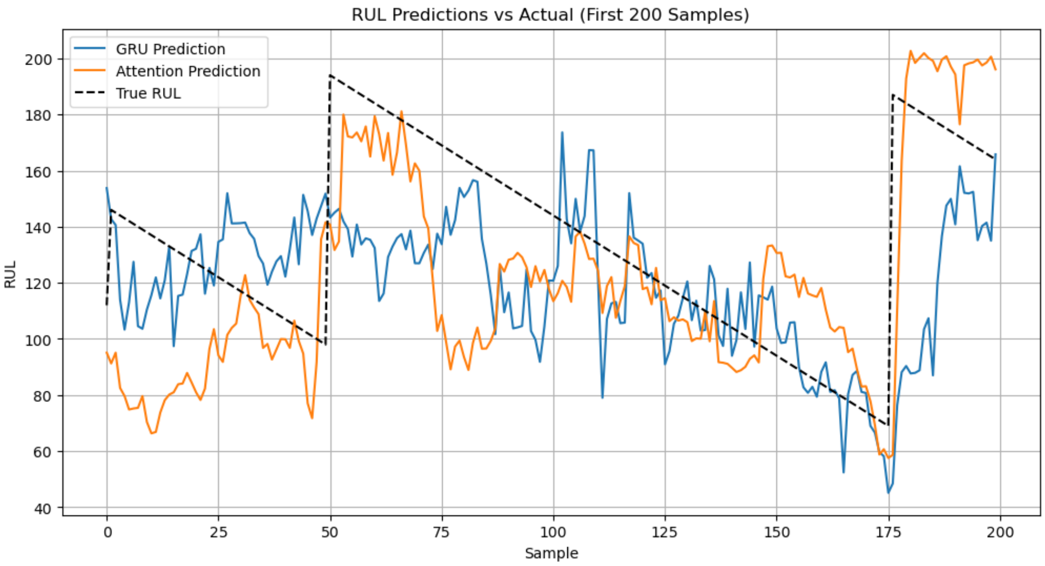

Figure 5 — RUL predictions vs. true values

6.3 Classification Exploration

In addition to regression, the project explored a binary classification task that involved predicting whether an engine was in a 'critical' state based on its RUL. A critical state was defined as having fewer than 30 cycles remaining before failure. Less than 30 cycles wouldn't correspond to imminent failure but rather an increase wariness and maintenance checks. A Random Forest Classifier was trained to perform this task, and the data was resampled with SMOTE (Synthetic Minority Oversampling Technique) to counter a heavy class imbalance, as 97% of datapoints represent 'non-critical' engines. As seen in Table 6 and Figure 6 below, this model achieving nearly perfect precision and recall on the validation dataset and full cycle.

Table 6 — Classification report (Validation set)

Class	Precision	Recall	F1-Score	Support
'Non-Critical'	0.999741	0.988482	0.994080	3907.00
'Critical'	0.988608	0.999744	0.994145	3906.00
Accuracy			0.994112	7813.00
Macro Avg	0.994174	0.994113	0.994112	7813.00
Weighted Avg	0.994175	0.994112	0.994112	7813.00

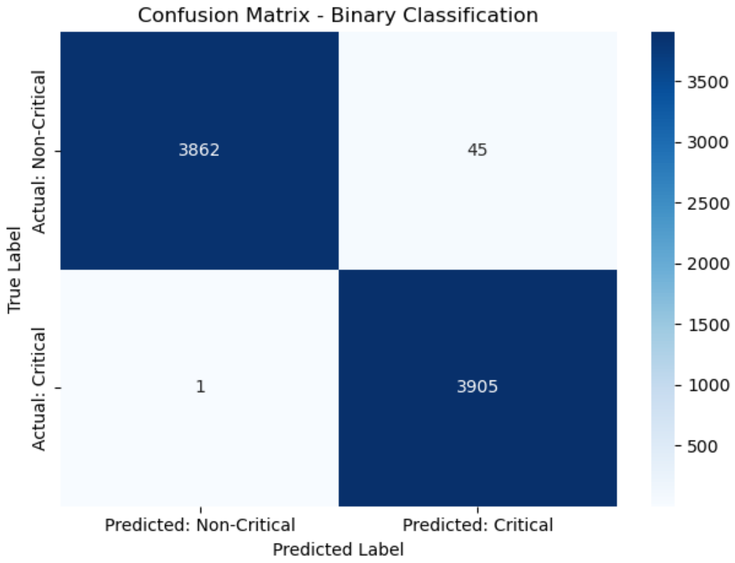

Figure 6 — Confusion Matrix (Validation set, full cycle)

However, after evaluating the model on the last cycle and the test data, which was not resampled, the results are far less impressive. Specifically, engines in a critical state were mislabeled close to half the time (Table 7 and Figure 7).

Table 7 — Classification report (Test set)

Class	**Precision**	**Recall**	**F1-Score**	**Support**
'Non-Critical'	0.967391	0.956989	0.962162	93.00
'Critical'	0.500000	0.571429	0.533333	7.00
Accuracy			0.930000	100.00
Macro Avg	0.733696	0.764209	0.747748	100.00
Weighted Avg	0.934674	0.930000	0.932144	100.00

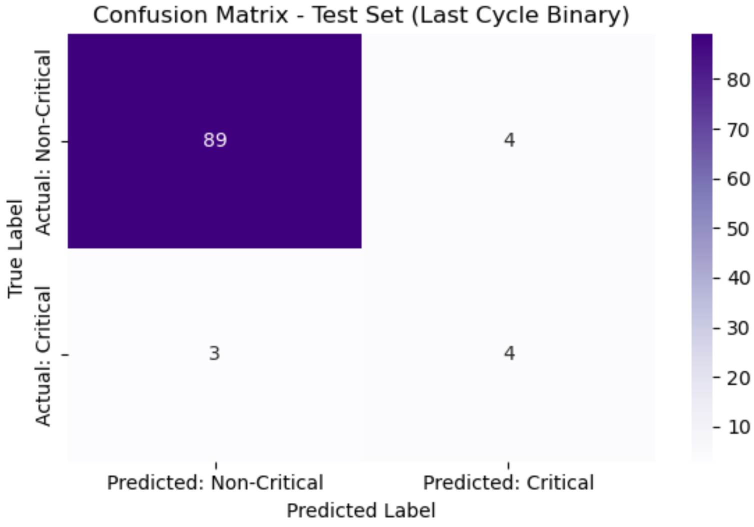

Figure 7—Confusion Matrix (Test set, last cycle)

While the classification model performed well, its practical utility was limited. Predicting whether an engine is near failure provides only coarse information and lacks the granularity required for long-term maintenance planning. However, such a model could be useful in real-time systems where immediate risk flagging is needed. It could serve as a supplementary tool to a regression-based RUL prediction model.

7 CONCLUSION AND RECOMMENDATIONS

This project explored the use of machine learning and deep learning to predict structural fatigue using vibration signals. Traditional models such as Random Forests were found to produce moderate performance metrics, likely due to their inability to model temporal dependencies. Recurrent neural networks produced mixed results. GRU performed similarly to some of the Random Forest models, while Attention-LSTM demonstrated improvements in predictive accuracy.

The best-performing model overall was an attention-based LSTM, which achieved the lowest error metrics and offered interpretability through its attention weights. Hyperparameter tuning showed to improve model performance, highlighting the value of systematic optimization.

In addition to regression, a binary classification model was developed to identify critical states. While this approach had limited standalone utility, it could complement RUL prediction models in practical applications.

The project concludes that early detection of structural fatigue is feasible using data-driven models, provided that temporal dynamics are accounted for. However, it is a complex task, with performance results that are can be underwhelming. Future work could explore hybrid models, physics-informed learning, and consider other operating conditions.

8 PERSONAL REFLECTION

Overall, I am satisfied with the work I produced through this project, though I found myself running short on time as I tried to expand my analysis. In the field of data science, it's common for early-career professionals to encounter what is often referred to as “analysis paralysis”, which is the tendency to over-analyze in pursuit of the most accurate model possible. I experienced this firsthand. My focus on building the “best” model led to prolonged experimentation and prevented me from fully exploring some techniques I had initially intended to include.

For example, in my midterm progress report, I expressed interest in implementing SHAP (SHapley Additive exPlanations) to better explain my model's predictions. Despite making some progress, I encountered technical challenges and ultimately ran out of time before I could integrate it meaningfully into my analysis.

Finally, I would like to thank the organization that sponsored my practicum, Sandia National Laboratories, as well as my advisor, Steven Smith. Steven's guidance and feedback were instrumental in shaping the direction and depth of this project.

9 REFERENCES

- Babu, G. S., Zhao, P., & Li, X. (2016). "Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life". International Conference on Database Systems for Advanced Applications.
- Heimes, F. O. (2008). "Recurrent Neural Networks for Remaining Useful Life Estimation". IEEE Aerospace Conference.
- Saxena, A., & Goebel, K. (2008). "Turbofan Engine Degradation Simulation Data Set". NASA Ames Prognostics Data Repository.
- Wu, J., Tang, Y., & Han, Z. (2021). "Remaining Useful Life Prediction Using Attention-Based LSTM Network". IEEE Access.
- Zheng, S., Ristovski, K., Farid, M., & Ryan, M. (2017). "Long Short-Term Memory Network for Remaining Useful Life Estimation". Prognostics and Health Management Conference.

10 APPENDICES

Table 8 — Description of sensor parameters

Sensor #	Symbol	Description	Units
1	T2	Total temperature at fan inlet	°R
2	T24	Total temperature at LPC outlet	°R
3	T30	Total temperature at HPC outlet	°R
4	T50	Total temperature at LPT outlet	°R
5	P2	Pressure at fan inlet	psia
6	P15	Total pressure in bypass-duct	psia
7	P30	Total pressure at HPC outlet	psia
8	Nf	Physical fan speed	rpm
9	Nc	Physical core speed	rpm
10	epr	Engine pressure ratio (P50/P2)	—
11	Ps30	Static pressure at HPC outlet	psia
12	phi	Ratio of fuel flow to Ps30	pps/psi
13	NRf	Corrected fan speed	rpm
14	NRc	Corrected core speed	rpm
15	BPR	Bypass Ratio	—
16	farB	Burner fuel-air ratio	—
17	htBleed	Bleed Enthalpy	—
18	Nf_dmd	Demanded fan speed	rpm
19	PCNfR_dmd	Demanded corrected fan speed	rpm
20	W31	HPT coolant bleed	lbm/s
21	W32	LPT coolant bleed	lbm/s

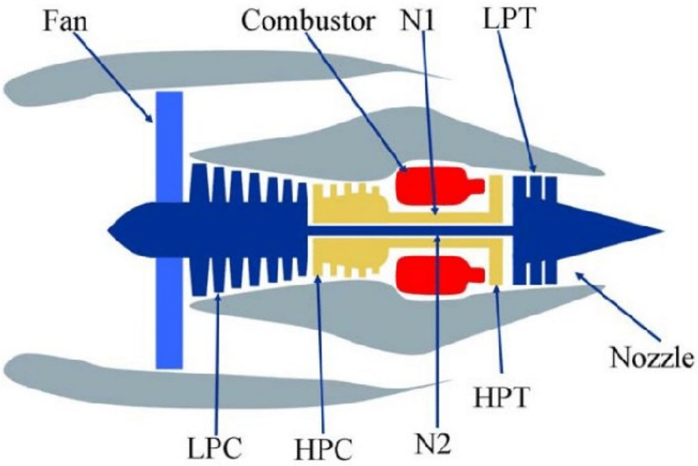

Figure 8—Structure of turbofan engine in C-MAPSS simulation