Sophia Oku

Georgia Tech Institute of Technology

Practicum Project - Sandia National Labs

A Hybrid Model Approach for Predicting Jet Engine Degradation Using Advanced Machine Learning Techniques

Introduction

In high risk systems, the goal is to identify and address issues that could lead to systematic failure and loss of life. In most systems or processes involving the interaction of various complex parts and systems, it is difficult to manually monitor the lifespan of each component without risking human error, which could be costly depending on the components function. Prognostics and Health Management (PHM) is a systematic framework that uses computation-based programs to assess system degradation (Zio, 2021). PHM utilizes physical knowledge, detailed information about individual systems, and simulated data to diagnose and predict maintenance needs more reliably and efficiently than a physical assessment. The primary objective of predictive maintenance is to forecast the failure of a component before actual manifestation of true failure (Fu & Avdelidis, 2023). Another objective of PHM is to reduce the cost associated with scheduled inspections, while promoting more efficient and reliable operating conditions, especially in high risk environments like aviation and manufacturing (Kordestani et al., 2023). In most cases, implementing a PHM system involves intensive data collection through various non-disruptive automated tools, such as sensors. The tools are used to simulate the wear and tear under different environments and operational conditions. In the aviation industry, PHM systems play a crucial role in aircraft maintenance and operational safety.

Due to the complexity of an aircraft's overall system, PHM offers the most cost-effective and reliable method of predicting Remaining Useful Life (RUL) of various aircraft components. There are several methods of determining RUL, and their effectiveness depends largely on how the data is collected and utilized. These methods include a physics-based approach, a data-driven approach, and a hybrid approach (Kim et al., 2016). While the goal of each method is equally the same, their strengths and limitations differ. The physics-based approach relies on the understanding of fundamental physical principles, using mathematical models to describe degraded processes. It depends heavily on measurable factors such as material properties, stress levels, and operating conditions. However, a key limitation of the physics-based approach is that it requires in-depth understanding of the system's conditions in order to properly establish a relationship between the causative forces and their effects (Kim et al., 2016; Liao & Kottig, 2014). The limitation increases as the complexity of the system increases. To address this limitation, a data-driven approach uses historical data to identify patterns associated with

degradation events, enabling the prediction of future outcomes. However, like many data-modeled systems, its predictive power is restricted to the patterns it has been trained to identify. For example, if a model has been trained on a linear relationship between two variables, any change to the operation condition or variables that can affect this relation can render the linear model ineffective (Liao & Kottig, 2014). The hybrid approach addresses the limitation of both the physics-based and data-driven approach. It combines physical understanding of the system with data modeling to improve the accuracy of prognosis predictions (Liao & Kottig, 2014). Among these methods, the data-driven approach remained the most commonly used for determining RUL.

Remaining useful life (RUL) is the time span from the starting point to the end of the useful life of a monitored component (Si et al., 2010). It is the primary metric in prognostics. Other performance evaluation metrics include Prognosis Horizon (PH), Alpha-Lambda Performance, Relative Accuracy (RA), and Convergence Rate (Djeziri et al., 2020). The data-driven method has been widely used in numerous studies to predict RUL of aircraft turbofan jet engines. The engines operate by using a large fan at the front to bypass a significant portion of the air around the engine core. This design, compared to older turbojets, allows for more efficient thrust creation by moving a larger mass of air. Over the years, turbofan engines have evolved to require less starting power, while offering improved efficiency (Lungu & Tudosie, 2014). Turbo fan jet engines consist of a low-pressure and high-pressure compressor, a combustion chamber, a fan, turbine, inlet and other components, all of which are exposed to varying external conditions. Effective data collection from these components must track each component across all flight activities. This can be achieved through simulated data collection. The recorded data applied into statistical algorithms to predict RUL. The application of machine learning algorithms has grown from traditional machine learning models that require manual feature selection to more intense end-to-end deep learning methods. This is because in many cases deep learning algorithms such as Convolutional Neural Networks (CNNs), Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs) have significantly outperformed other traditional machine learning models. Heimes (2008) found that RNN models outperformed Multi-layer perceptron models, producing more accurate and less noisy predictions. However, more recent reviews of RNN models noted that short-term memory and training instability could limit the ability of the model to predict RUL accurately, especially when compared to other models such as Long Short-Term Memory

(LSTM) networks (Wang et al., 2020). LSTMs model uses a system of gates to learn patterns, enabling it to navigate short and long-term dependencies. Yuan et al. (2023) reported improved predictive accuracy with the addition of a multi-layer LSTM network compared to a traditional RNNs model. These are similar to the findings in Yousuf et al. (2022). CNNs have become a common choice for RUL prediction, due to the ability to extract features and use this information in training. To improve generalization and avoid overfitting, techniques such as dropout regularization and max-pooling layers are often integrated into CNN architectures (Ensarioğlu et al., 2023; X. Li et al., 2017). In some studies, these advanced algorithms are stacked to further improve the predictive ability of a Neural Network (NNs) performance. For example, Zhu et al. (2021) demonstrated that an LSTM model combined with a t-SNE and DBSCAN for dimensionality reduction outperformed standalone models. Multiple studies have shown that stacking multi-layers CNNs and LSTMs algorithms can produce lower error rates as compared to a 1D-CNN model by itself (Al-Dulaimi et al., 2019; Ensarioğlu et al., 2023; Hong et al., 2020).

The value of double--stacking layers presents an effective approach for determining RUL with strong potential of reducing error rates as compared to single layers. Several factors must be considered when determining the RUL from sequential data. One important factor is to gather a base understanding of how the data was collected. This a benchmark for preprocessing and data engineering to enable any NNs model to learn the information embedded in the trends. This study uses the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset, retrieved from the NASA Ames Prognostics Center of Excellence (Saxena, 2008), to develop RUL predictions. According to the accompanying document, data simulated from 21 sensors with varying parameters such as pressure, temperature, and speed. This means that the data reflects time-based sensor readings from engines operating under different conditions, making it well-suited for sequential modeling. In X. Li et al. (2018), a piecewise linear degradation model was applied to generate RUL labels for the training data. This was done by assuming a constant RUL value during the early healthy phase, followed by a linear decline as the engine began to degrade. This setup mimics a real world scenario, where the engine unit works in the initial stages and starts to degrade over time. Another factor that was considered was the potential of noise and poor sensor readings to influence the RUL prediction. Ensarioğlu et al. (2023) demonstrated that passing the data through a low-pass filter can effectively mitigate the noise issue.

This study explores the use of a comprehensive multi-task classification and regression CNN-LSTM model to predict RUL, as compared to traditional models. The motivation behind this study is to determine if the “black box” issue should be a deciding criteria in selecting a model for RUL prognosis. The study explores evaluation metrics for classification and regression models for both traditional and advanced machine learning models. The primary focus is on the regression accuracy for RUL prediction, while classification performance is analyzed as a secondary outcome of the multi-task model.

Literature review

Developed using gradient-based learning, CNNs were designed to train multilayer networks to extract features directly from pixel data (Lecun et al., 1998). In review paper published by Sercu et al. (2015), CNNs have evolved from the basic LeNet-5 model with seven layers (Lecun et al., 1998), to more adaptive versions capable of supporting deeper architectures with up to 14 layers (10 convolutional and 4 fully connected layers). When compared to other models, the defining feature of CNNs models is the ability of the algorithm to utilize shared weights across spatial dimensions allowing for efficient pattern detection. Shared weights alongside spatial pooling makes CNNs a popular choice for dimensionality reductions with the ability to preserve key features. CNNs excel in capturing short-term temporal dependencies over a small window sequence and detecting patterns regardless of their position within the sequence. These properties make CNNs an ideal model when dealing with sequential data, making it a popular choice in studies using the C-MPASS dataset.

An example of such studies is what was done by Kim and Sohn (2020), who proposed a multi-task CNNs framework that simultaneously performs RUL estimation and health condition classification using the C-MAPSS dataset. The model used multivariate 1D convolution filters to capture both cross-sensor and temporal dependencies. The model used four layers with varying filter sizes and incorporated dropout regularization to address overfitting. Model training was done using the Adam Optimizer, similar to what was used in this study, alongside early stopping to improve convergence. Similar to the challenges faced in this study, the labeling of the health conditions is subjective. Regardless, the multi-task CNN structure performed successfully at

predicting RUL and system health state compared to single-task and RNN-based models (Kim & Sohn, 2020).

CNNs can also be optimized by integrating other NNs to further improve accuracy. Wang et al. (2025) reported significant improvements from combining Gated Recurrent Unit (GRU), which is a variation of RNNs and CNNs in predicting RUL's using the C-MAPSS dataset. In this dual-model approach,CNNs was used to extract spatial features, while the GRU captured temporal features. The main model architecture involved a Channel Attention Mechanism (CAM) that focused on the most informative channels and Self-Attention Mechanism (SAM) to capture the most relevant time steps. Similar to the study by Kim and Sohn (2020), a dropout regularization was used to minimize overfitting, with a learning rate of 0.001 and varying sliding window sizes. The CNN-GRU model outperformed other models examined in the study. However, while the proposed method did well overall, its RMSE on the C-MPASS dataset was slightly higher than BLSTM-CNN (Guo et al., 2024), indicating that in less complex conditions, simpler architectures may still perform competitively (Wang et al., 2025).

Li et al. (2018) used a Deep Convolutional Neural Network (DCNNs) using four stacked 1D convolution layers of uniform kernel sizes on the C-MAPSS dataset. To preserve the temporal sequence, pooling layers was used, along with a dropout applied to avoid overfitting. As in many similar studies, the Adam optimizer was used. The limitation of the DCNN model was the limited training data used in training the model, thereby reducing the ability of the model to perform any better even after optimization. In contrast, Li et al. (2021) combined Principal Component Analysis (PCA) with an LSTM model to control sensor dimensionality and improve temporal modeling. This hybrid PCA-LSTM model showed high performance using only the first three principal components, which preserved over 90% of data variance. However both models suffered from the limited availability of training data. Liu et al. (2023) addressed this challenge in his study by using transformer encoders and different CNN layers with varying kernel sizes.

These literature studies lay the foundation for the implementation of the 1D-CNN-LSTM multitask model used in this study. The model adopts a 1D convolutional structure with time-windowed inputs and dropout regularization to preserve temporal features while minimizing

overfitting similar to Li et al. (2018). This study used a multitask layer system similar to Hong et al. (2020), which used a combination of CNN, LSTM, and Bi-LSTM layers to improve RUL prediction. Like Hong et al. (2020), the multitask 1D-CNN-LSTM incorporates an early stopping mechanism and ReLU activations to control convergence and gradient flow. Overall, 1D-CNN-LSTM adopts various literature applications of NNs in manipulating sequential data achieving similar accuracy without the use of more complex attention-based or stacked bidirectional models(Hong et al., 2020; Li et al., 2018; Liu et al., 2023; Kim & Sohn, 2020).

Dataset Overview

The Nasa C-MPASS dataset typically used for RUL prognosis comprises four different datasets with varying operating settings and fault conditions of a dual-spool turbofan jet engine. The turbofan jet engine consists of various components such as fan, low-pressure and high-pressure compressors, low-pressure and high-pressure turbines, combustor and nozzle. To generate the dataset, 21 sensors were simulated across different locations in the turbofan jet engine, measuring temperature, speed, pressure and other factors as shown in Table 1. To bring the dataset to real-world flight conditions and scenarios, simulation noise was added during the simulation. For this study, the FD001 dataset was used to determine RUL.

Table 1. Description of Sensors and their accompanying symbols for the C-MAPSS database.

Sensor Number	Symbol	Description
1	T2	Total temperature at fan inlet
2	T24	Total temperature at LPC outlet
3	T30	Total temperature at HPC outlet
4	T50	Total temperature at LPT outlet
5	P2	Pressure at fan inlet
6	P15	Total pressure in bypass-duct
7	P30	Total pressure at HPC outlet
8	Nf	Physical fan speed

9	Nc	Physical core speed
10	epr	Engine pressure ratio
11	Ps30	Static pressure at HPC outlet
12	Phi	Ratio of fuel flow to Ps30
13	NRf	Corrected fan speed
14	NRc	Corrected core speed
15	BPR	Bypass ratio
16	farB	Burner fuel-air ratio
17	htBleed	Bleed enthalpy
18	Nf_dmd	Demanded fan speed
19	PCNfR_dmd	Demanded corrected fan speed
20	W31	HPT coolant bleed
21	W32	LPT coolant bleed

FD001 C-MAPSS Dataset

The FD001 dataset consists of simulated run-to-failure data for 100 turbofan engines using sensors under a single operating condition (three operating settings) and one fault mode (Table 2). Each engine is assumed to start from a healthy operating state, which is a prerequisite for dealing with RUL prognosis. The engines are then ran until failure is reached. The training dataset consists of a complete life cycle, from cycle 1 until failure for each engine. The test set consists of partial life cycles for another 100 engines, where the last cycle occurs at some point before failure. Some sensors provide visual information on engine degradation over the full cycle, while some sensors contain pure noise or provide no information. The aim of this study is to use the provided sensor information to predict RUL of an engine at a given time *t*. The classification part of this multitask model uses a binary classification to determine the engines health at a time *t* and also feeds into the regression model to enhance the RUL prediction. For the classification model, if the RUL ≤ 30, it is assigned “critical" (meaning that the engine is in the last 30 cycles until failure) and considered “normal” if RUL ≥ 30. There was no specific method

for determining this threshold, as existing literature used more than two categories. The choice for 30 as the threshold was chosen subjectively specifically for this study and, in practice, it could be tuned if maintenance lead times were provided or expert knowledge was available.

Table 2. FD001 NASA C-MAPSS Dataset

Dataset	**FD001**
Number of engines	100
Number of training samples	20631
Number of test samples	100
Number of the data column	26
Average life span (cycles)	206
Operating conditions	1
Fault conditions	1

Proposed Methodology

This section introduces the theoretical background behind the proposed models starting with the CNN model.

Convolutional Neural Network (CNNs)

CNNs is a deep learning neural network that extracts features from data with convolution structures. The one-dimensional Convolutional Neural Network (1D-CNNs) is used in this study to extract localized patterns from the C-MPASS multivariate time series data. By applying kernels across input sequences, CNNs can detect features that are indicative of system degradation. For a given input matrix $X \epsilon \mathbb{R}^{T \times F}$, where T is the number of time steps and F is the number of features (sensors), a convolution operation with kernel $k \epsilon \mathbb{R}^{h \times F}$, where bias b, and stride s is defined as:

$$y_i = \sum_{j=0}^{h-1} \sum_{f=0}^{F-1} X_{i+j,f} \cdot k_{j,f} + b$$

The result y_i is passed through a nonlinear activation function such as ReLU, yielding a feature map $m_i = ReLU_{(y_i)}$. These feature maps are subsequently downsampled via max pooling in order to reduce dimensionality model complexity. The CNNs does well handling short-term local dependencies among sensor data, making it an ideal model for identifying early signs of failure within fixed window sizes. The extracted features are flattened and passed through connected layers for both RUL regression and health state classification. Key challenges that have to be monitored for CNNs is the sensitivity to the kernel size selection and pooling strategy, as small kernels can overlook long-term trends, while aggressive pooling can disregard important degradation patterns. To avoid overfitting, dropout layers and early stopping based on validation loss are most used during implementation. The CNNs in this study are deliberately designed for a surface level performance to allow interpretability and computational efficiency.

Long Short-Term Memory networks (LSTMs)

Long Short-Term Memory networks (LSTMs), a variant of RNNs, are designed to model sequential dependencies by identifying temporal patterns of extended events in noisy input streams (Varsamopoulos et al., 2019). In the context of RUL prediction, LSTMs are ideal because engine degradation is often gradual and controlled by patterns spanning many cycles. Each LSTMs cell at time t receives an input x_t, a hidden state h_{t-1}, and a cell state c_{t-1}, and produces its new states through the following set of gating mechanisms:

$$f_t = \sigma\,(W_f x_t + U_f\, h_{t-1} + b_f \quad (forget\ gate)$$

$$i_t = \sigma\,(W_i x_i + U_i\, h_{t-1} + b_i \quad (input\ gate)$$

$$c_t = tanh\,(W_c x_t + U_c\, h_{t-1} + b_c \quad (candidate\ gate)$$

$$o_t = \sigma\,(W_o x_t + U_o\, h_{t-1} + b \quad (output\ gate)$$

Here, σ denotes the sigmoid function and W, U, b are learnable parameters. The input and output gates are used to determine which signals are going to move to the next node. The W is the recurrent connection between the previous hidden layer and the current hidden layer that allows the LSTMs cells to remember information from the past. The U is a weight matrix that allows the LSTMs to extract relevant features from each input by connecting the inputs to the hidden layer. c is the internal memory unit where the LSTMs proposes new information to be added to a cell

state based on current input weighted by U and the previous hidden state weighted by W. In all of this, the LSTMs approach is baked on learning and understanding features to make better predictions in future time steps. In this study, sequences of 30-time steps are fed into the LSTMs, which outputs a hidden state summarizing the decline in engine health over time cycles and then passed to a dense layer for RUL prediction. The limitation when working with LSTMs is that a lot of computational power is required compared to CNNs and it is prone to overfitting on small datasets. So, hyperparameter tuning is required, but is usually limited by computational requirements. In this study a CNN-LSTMs model is compared to a CNNs model.

Experimental Study

Exploratory Data Analysis (EDA)

Initial exploratory data analysis showed varying cycle lengths across all 100 engines. As shown in Figure 1, the distribution of cycle times ranged from ~130 to ~360 cycles, with the highest concentration around 200 cycles.

The exploratory data analysis showed interesting trends on the usefulness of some sensors. Some sensors showed distinct patterns indicative of engine lifecycle degradation. For example, sensor 14 increased as RUL decreased, suggesting that it captured a potential system wear (Figure 6). Similarly, sensors 17, 13, and 8 exhibited varying trends of engine degradation, with 8 showing somewhat moderate trends (Figure 5). This moderate trend was seen in sensors 2, 4, 3, 11 and 15. The different degradation patterns could be as result of simulation noise or the impact of the different control factors (temperature, pressure, etc.) that were being manipulated during the simulation. In contrast, sensor 1 exhibited constant flat trends suggesting that there might not be any research value to include these sensors in the model (Figure 2). This trend was similar to those seen in sensors 10, 16, 18 and 19. Aside from the flat patterns, 6 showed a very distinct variation that could not be tied to any reasonable explanation (Figure 3).

A heatmap was there used to visualize the pearson coefficients of all sensors as an additional exploration tool. Among all sensors, there were patterns of high correlation (sensors 2, 3, 4, 8, 11, and 15), low correlation, and no correlation (Figure 7). Sensors 12, 20 and 21 had similar

correlation strength to 7, which was similar to the RUL trend chart (Figure 4). These correlation relationships were noted and used as a guide in regularization and feature selection. Similar to the trend seen in the trend chart, sensors 1, 5, 6, 10, 16, 18 and 19 had no correlation strength and were completely removed from the dataset to avoid noise.

Figure 1. Distribution of cycle times across 100 engines

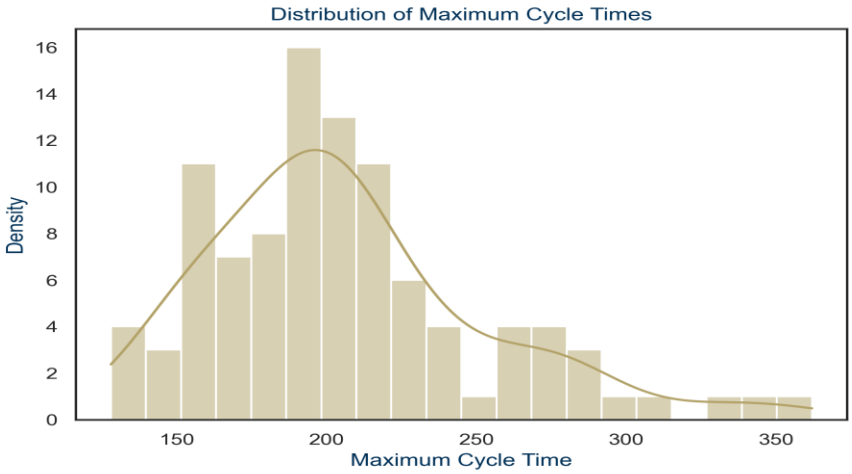
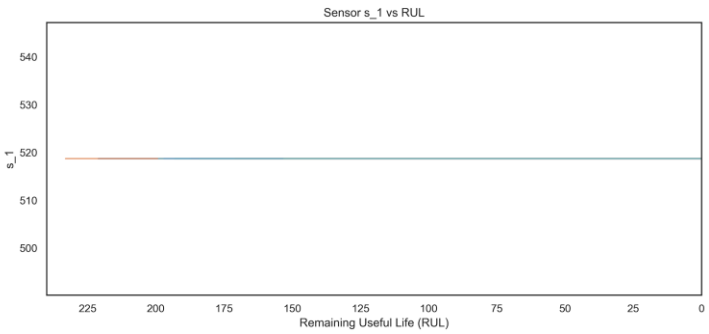
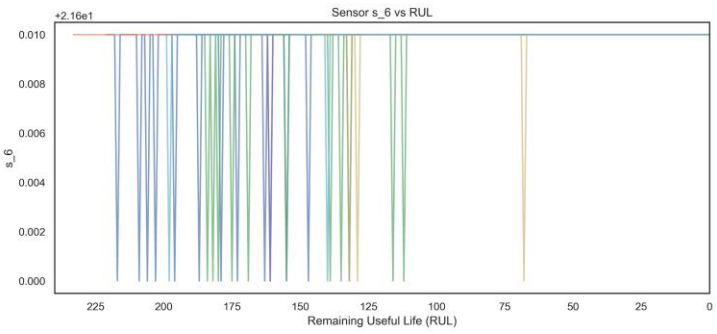
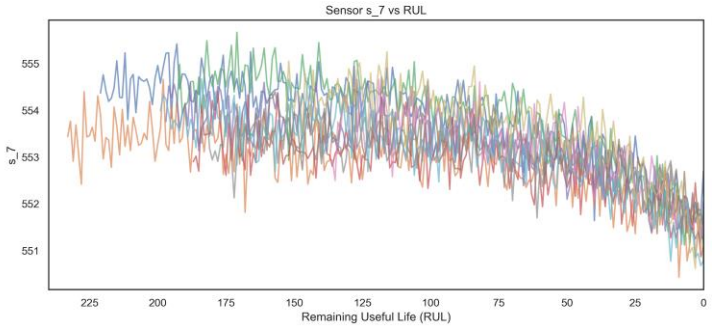
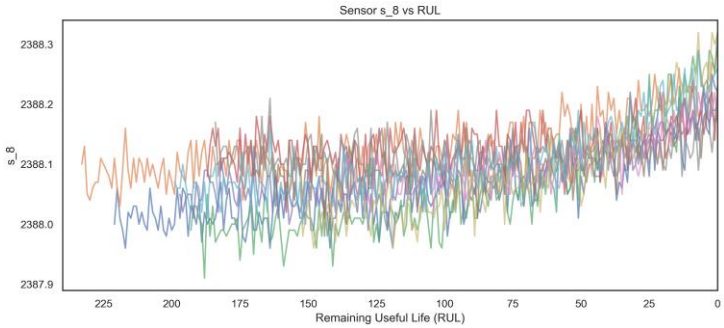


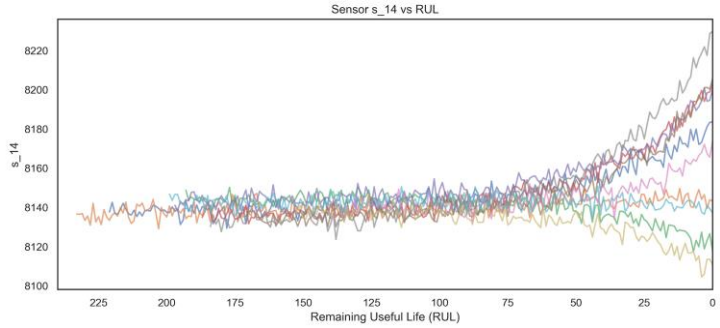
Figure 2 - 6 showing Sensor Degradation Patterns Using the Remaining Useful Life of 100 Jet engines







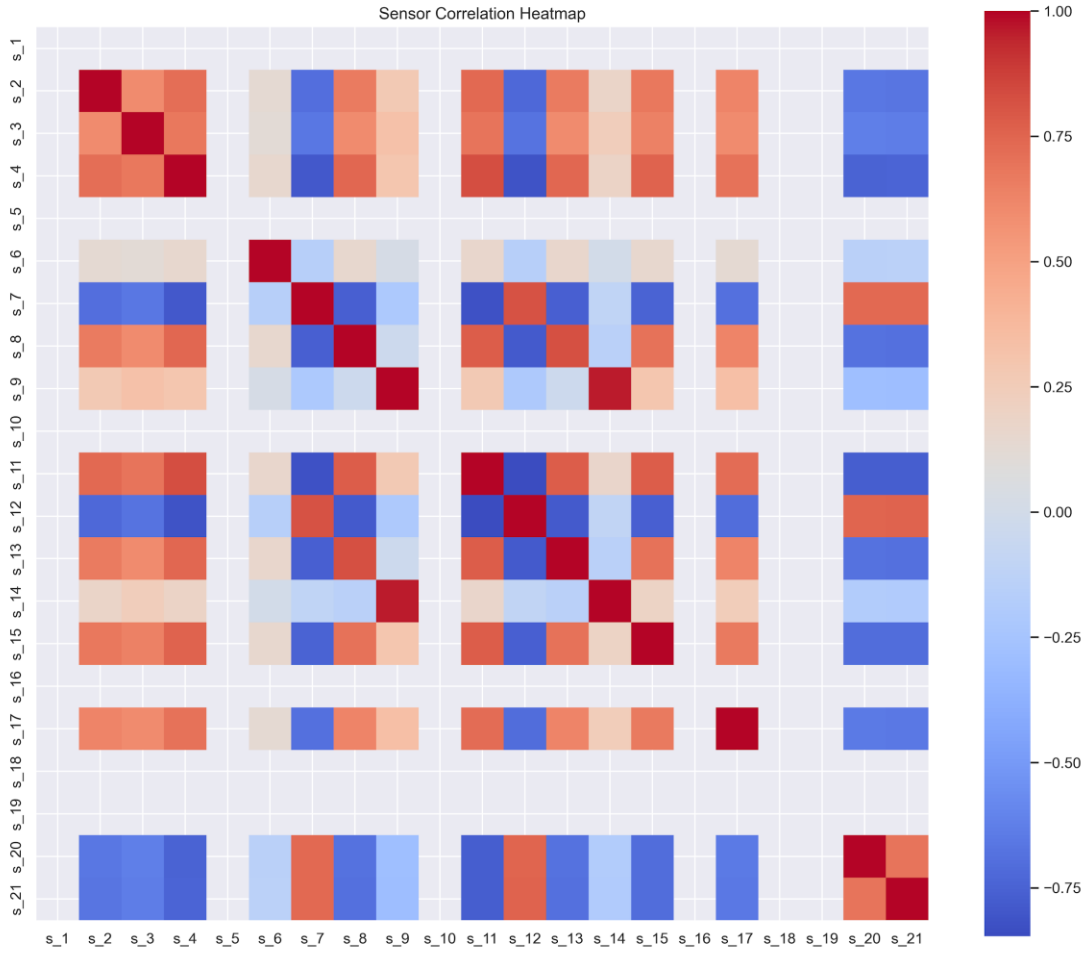




Data Preprocessing and Feature Engineering

Rolling Statistics: Rolling statistics was computed to ensure that the model captured all short term trends. If the rolling statistics wasn't done, the model would only see the actual RUL value without any trend information applied. This was done by calculating the rolling mean and rolling standard deviation over a window of 5. The reason behind choosing a window of 5 was due to lack of computational power as of the time the models were created. Wang et al. (2025) found that increasing the time window from 25 to 35 cycles significantly improved RUL prediction performance, after which too large a window hurt performance. Yuan et al. (2023) found that a window of 30 gave the lowest RMSE. This is an area for improvement for future modeling. As research has shown that there is a potential to lower the error rates by increasing the window size, there could be a possibility to use cross-validation to try different ranges of values to determine the optimum window size.

Figure 7. Correlation Plot of Sensor Signals Relationships based on Pearson Correlation



Normalization: It was important to normalize each feature to a common scale because, based on the EDA, the distribution of the starting RUL was not uniform, with some cycles having a higher starting RUL than others. This is a very important step for models, especially NNs because of the sensitivity to the scale of the data input. Each continuous feature (sensor readings and controlled features) was normalized to have zero mean and unit variance (z-score normalization), based on the statistics of the training set. This standardization is important because the sensors have different units and ranges, as seen in Table 1. Normalizing ensures that no single feature dominates due to its scale, which helps the model converge faster. The test set was normalized using the same mean and standard deviation from the training data to avoid data leakage.

Sliding Window Sequence: Because sequence-based models (e.g., CNN, LSTM), as well as baseline models in our study expect a fixed-size input, the data was using a sliding window

approach. A chosen window length of 20 cycles was used based on literature review and to avoid losing early degradation signals. The goal here was to find a suitable middle group that captures the temporal dependencies in the data. For each engine, sequences were generated from cycles 1 - 20, with the RUL at the 20 being the target. This was to ensure that each generated input sequence was consistent in length and label in order to correspond to the last time step. Each generated sequence was a 2D array fed into the CNN and CNN-LSTM model.

Training/Validation Split: The primary concern during the dataset split was to avoid leakage. In a previous model run, data leakage was a significant issue, resulting in the initial CNN model falling to meet up to expectations. To avoid data leakage, the dataset was carefully partitioned into training and validation sets based on engine numbers, ensuring that the same engine did not appear multiple times in a particular split subset. 20% of the data was held out for validation with the remaining 80% used for training. The main test set containing 100 unseen engine runs was used to evaluate the model's performance.

Evaluation Metrics: In this study, the method of evaluation varied depending on the type of task. For the classification models, F1 score, and recall were used as evaluation metrics as defined in the equation below. Due to the labeling method used for classification, the dataset was imbalanced; therefore, accuracy would not have been an appropriate metric for assessing model performance, especially for the neural networks applied in this study. For example, a naive classifier will achieve high accuracy by predicting the majority class in an imbalanced dataset. The use of the F1 score and recall will force the tuner to find hyperparameters that improve the recall of failure events and not just success alone. For the regression model, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), common methods for assessing regression models, were used.

$$Recall_i = \frac{True\ positive_{\ i}}{True\ Positive_i + False\ Negative_i}$$

$$F1\ Score_i = \frac{2 \times Precision_i \times Recall_i}{Precision_i \times Recall_i}$$

Experimental results and discussion

Classification Performance

All classification models performed reasonably well with major differences all depending on the model structure and data preprocessing. Among the baseline models, Logistic Regression had a high precision of 0.91 and recall of 0.91 for class 1 (critical state), resulting in a high F1-score of 0.91. The trend of the same number for the recall, precision and F1 score suggests high overfitting. The Random Forest Classifier also had a similar performance, with slightly better precision for the critical class. However, several limitations restrict the validity and interpretability of these results. The baseline models were trained and evaluated on the training set itself. This is a practice that introduces high risk of overfitting leading to over inflated performance metrics. Also, these models operated on a flattened input, meaning they only observed a single snapshot of sensor readings at each time step. This is a flawed approach as it failed to capture the temporal degradation trends, which is key in predicting or classifying time-series data. Finally, these models are memoryless, so their predictions do not improve even as more sensor readings accumulate over time.

In contrast, the 1D-CNN-LSTM multi-task model demonstrated more realistic performance of a time-series data although the evaluation metrics were lower as compared to the base model. Although the accuracy reached 98%, which may seem impressive, it is not an ideal metric in this context due to the class imbalance and the low prevalence of the critical class. Instead, the precision and recall for the critical class are better measurement metrics. The precision for the critical class was 0.75, indicating that when the model predicted an engine was in a critical state, it was correct 75% of the time. The recall was 0.60, meaning it successfully identified 60% of all truly critical states, while missing 40% (false negatives). The resulting F1-score was 0.70, a reflection of the model's ability to handle critical cases. The recall was quite poor but impressive for a base 1D-CNN-LSTM model without intense optimization. It shows that there is room for improvement. For example, Kim and Sohn (2020) used a three-state classification model to build a multi-task CNN. The states were encoded using ordinal logic which helps the model understand progressive degradation as opposed to the two-state classification used in this model. Their choice of Exponential Linear Unit (ELU) as opposed to ReLU could have played a

difference as ELU helps to address the vanishing gradients issues. Importantly, Kim and Sohn (2020), optimized their learning decay rate, which was not done in this study.

Table 3. Multi-task 1D-CNN-LSTM Classification Results

	Precision	Recall	F1-Score
Not Critical (RUL $>$ 30)	0.99	0.99	0.99
Critical (RUL $\leq$ 30)	0.75	0.60	0.70

Regression Performance (RUL Prediction)

The table below lists the MAE for each model. The random forest (RF) model performed better than the multi-task CNN, with an MAE of 11. As mentioned regarding the classification results, the base models were evaluated on the training set, while the multi-task CNN was evaluated on the test set. In most cases, performance evaluation on the training set always yields better results than on the test set, which suggests that the RF would likely have performed worse than the multi-task CNN if evaluated on the test set. In general, it was expected that the base models would yield higher errors due to their inability to capture temporal changes, so no further evaluation on the test set was pursued. The Decision Tree (DT) had the poorest performance with an MAE of 14, indicating it performed poorly at capturing any of the degradation trends or patterns. SVR performed slightly better than DT, with an MAE of 12, likely thanks to its nonlinear kernel. The RF uses bootstrap sampling that allows the regressor to capture nonlinear relationships and achieved the best result among the non-neural models, with an MAE of 11. This indicates that an ensemble model might be able to learn some of the complex relationships in the temporal dataset. However, consistent with findings in the literature (e.g., Li et al., 2018), RF's typically struggle to extrapolate trends beyond the scope of the training data. As a result, the model tends to underestimate RUL for longer cycles and overestimate for shorter time periods.

On the other hand, the multitask 1D-CNN-LSTM model had an MAE of 12 and an RMSE of 19 on the test set. Although this is better than most base models used in this study, the performance lags behind most of the models reviewed in past literature. Ensarioğlu et al. (2023) achieved an RMSE of 16.1 by integrating a 1D-CNN-LSTM model with change-point detection-based labeling and first-order difference features, which improved the model's sensitivity to degradation patterns over time compared to the 1D-CNN-LSTM implemented in this study. This showed that feature engineering should be a priority in ensuring improved accuracy in future considerations. Another important limitation is the lack of dimensionality reduction techniques. Although the MAE was not reported, Hong et al. (2020) reported improved prediction abilities from implementing SHAP-based dimensionality reduction. This strategy helps to reduce noise from interfering with the RUL predictions. Other forms of dimensionality reduction that could be explored are PCA reduction. Li et al. (2021) recorded improvements from using PCA analysis to improve the LSTM temporal interpretations.

Table 4. Mean Absolute Error for all Regression Models

1D-CNN-LSTM	12
Linear Regression	14
Support Vector Regression	12
Decision Tree	14
Random Forest	11

Conclusion

This study compared the use of traditional regression and classification models to deep learning techniques for predicting RUL and classifying the health of jet engines using the CMAPSS dataset. While ensemble models like RF performed slightly well (MAE = 11.03), they lacked the ability to model temporal deterioration, which is key in prognosis. Although the multi-task CNN-LSTM model didn't perform as well as most referenced models, if the limitations are addressed,

the model offers a computationally inexpensive method of predicting RUL compared to other deep learning models being proposed by other studies. The model could benefit from feature engineering, tuning, and dimensionality reduction as a starting point.

References

Al-Dulaimi, A., Zabihi, S., Asif, A., & Mohammadi, A. (2019). A multimodal and hybrid deep neural network model for Remaining Useful Life estimation. *Computers in Industry*, *108*, 186–196. https://doi.org/10.1016/j.compind.2019.02.004

Djeziri, M. A., Benmoussa, S., & Zio, E. (2020). Review on health indices extraction and trend modeling for remaining useful life estimation. In *Springer eBooks* (pp. 183–223). https://doi.org/10.1007/978-3-030-42726-9_8

Ensarioğlu, K., İnkaya, T., & Emel, E. (2023). Remaining Useful Life Estimation of Turbofan Engines with Deep Learning Using Change-Point Detection Based Labeling and Feature Engineering. *Applied Sciences*, *13*(21), 11893. https://doi.org/10.3390/app132111893

Fu, S., & Avdelidis, N. P. (2023). Prognostic and Health Management of Critical Aircraft Systems and Components: An Overview. *Sensors*, *23*(19), 8124. https://doi.org/10.3390/s23198124

Heimes, F. O. (2008, October 1). Recurrent neural networks for remaining useful life estimation. In *IEEE Explore*. https://doi.org/10.1109/phm.2008.4711422

Hong, C. W., Lee, C., Lee, K., Ko, M., Kim, D. E., & Hur, K. (2020). Remaining Useful Life Prognosis for Turbofan Engine Using Explainable Deep Neural Networks with Dimensionality Reduction. *Sensors*, *20*(22), 6626. https://doi.org/10.3390/s20226626

Kim, N.-H., An, D., & Choi, J.-H. (2016). Prognostics and health management of engineering systems. In *Springer eBooks*. https://doi.org/10.1007/978-3-319-44742-1

Kim, T. S., & Sohn, S. Y. (2020). Multitask learning for health condition identification and remaining useful life prediction: deep convolutional neural network approach. *Journal of*

Intelligent Manufacturing, *32*(8), 2169–2179. https://doi.org/10.1007/s10845-020-01630-w

Kordestani, M., Orchard, M. E., Khorasani, K., & Saif, M. (2023). An overview of the state of the art in aircraft prognostic and health management strategies. *IEEE Transactions on Instrumentation and Measurement*, *72*, 1–15. https://doi.org/10.1109/tim.2023.3236342

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Li, H., Li, Y., Wang, Z., & Li, Z. (2021). Remaining useful life prediction of Aero-Engine based on PCA-LSTM. In *2021 7th International Conference on Condition Monitoring of Machinery in Non-Stationary Operations* (pp. 63–66). https://doi.org/10.1109/cmmno53328.2021.9467643

Li, X., Ding, Q., & Sun, J. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, *172*, 1–11. https://doi.org/10.1016/j.ress.2017.11.021

Liao, L., & Kottig, F. (2014). Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. *IEEE Transactions on Reliability*, *63*(1), 191–207. https://doi.org/10.1109/tr.2014.2299152

Liu, Q., Zhang, Z., Guo, P., Wang, Y., & Liang, J. (2023). Enhancing aircraft engine remaining useful life prediction via multiscale deep transfer learning with limited data. *Journal of Computational Design and Engineering*, *11*(1), 343–355. https://doi.org/10.1093/jcde/qwae018

Saxena, A. (2008). *Turbofan Engine Degradation Simulation Data Set.* NASA. Retrieved January 6, 2025, from https://data.nasa.gov/Aerospace/CMAPSS-Jet-Engine-Simulated-Data/ff5v-kuh6/about_data

Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008, October 1). Damage propagation modeling for aircraft engine run-to-failure simulation. In *IEEE XPLORE*. 2008 International Conference on Prognostics and Health Management, Denver, Colorado, United States of America. https://doi.org/10.1109/phm.2008.4711414

Si, X., Wang, W., Hu, C., & Zhou, D. (2010). Remaining useful life estimation – A review on the statistical data driven approaches. *European Journal of Operational Research*, *213*(1), 1–14. https://doi.org/10.1016/j.ejor.2010.11.018

Wang, F., Liu, A., Qu, C., Xiong, R., & Chen, L. (2025). A Deep-Learning method for remaining useful life prediction of power machinery via Dual-Attention Mechanism. *Sensors*, *25*(2), 497. https://doi.org/10.3390/s25020497

Wang, Y., Addepalli, S., & Zhao, Y. (2020). Recurrent Neural Networks and its variants in Remaining Useful Life prediction. *IFAC-PapersOnLine*, *53*(3), 137–142. https://doi.org/10.1016/j.ifacol.2020.11.022

Yousuf, S., Khan, S. A., & Khursheed, S. (2022). Remaining useful life (RUL) regression using Long–Short Term Memory (LSTM) networks. *Microelectronics Reliability*, *139*, 114772. https://doi.org/10.1016/j.microrel.2022.114772

Yuan, W., Li, X., Gu, H., Zhang, F., & Miao, F. (2023). Engine remaining useful life prediction based on PSO optimized multi-layer long short-term memory and multi-source information fusion. *Measurement and Control*, *57*(5), 638–649. https://doi.org/10.1177/00202940231214868

Zio, E. (2021). Prognostics and Health Management (PHM): Where are we and where do we (need to) go in theory and practice. *Reliability Engineering & System Safety, 218,* 108119. https://doi.org/10.1016/j.ress.2021.108119