Lifetime Prediction of Turbofan Engines

Rafael Zequeira Jiménez

Machine Learning Engineer Program

@FourthBrain

Berlin 2021

I. Motivation:

Turbofan engines are a critical component of aircraft. These are complex systems with many interconnected pieces that require high levels of reliability. The remaining engine useful life (RUL) is an important metric that has been under constant investigation[1][2][3][4]. Accurate predictions of RUL enable the implementation of intelligent maintenance strategies and costs reductions.

However, this is a challenging topic. The development of these data-driven predictions models requires datasets with run-to-failure trajectories. And such datasets are often unavailable in real applications because failures are rare in many safety-critical systems.

At present, most of the datasets available are synthetic, generated with simulators or developed in a lab environment by governmental and academic institutions[5]. While the availability of even such limited datasets is one of the most relevant contributions to the progress of predictive maintenance (prognostic) and health management (PHM) in the last decade, these datasets lack important factors of complexity that are present in real systems. Consequently, the developed data-driven prognostics algorithms are often not transferable to real applications.

The goal of this project is to develop state-of-the-art machine learning models to predict the remaining useful life of turbofan engines. Two datasets will be employed, i.e., the extensively used CMAPSS [5] dataset, and the N-CMAPSS dataset [6][7] that was recently developed to account for more realistic run-to-failure trajectories for a fleet of aircraft engines under real flight conditions.

The developed prognostic predictions models are relevant for the implementation of intelligent maintenance strategies, that would enable reducing costs, machine downtime, and the risk of potentially catastrophic consequences if the systems are not maintained in time.

II. Related Work:

Paper	Year	Comments	Model	Dataset	Results
[3]	2021	Transformer architecture applied to RUL prediction. Dual Aspect Self-Attention based on Transformer (DAST). DAST consists of two encoders working in parallel to simultaneously extract features of different sensors and time steps (i.e., sensor encoder and time step encoder). *Among the 21 sensors in the C-MAPSS dataset, sensors 1, 5, 6, 10, 16, 18 and 19 always have constant values during the run-to-failure experiments, meaning that data from these sensors cannot characterize the degradation process of the engine. Then, the authors removed these sensor data series and use the data of the remaining 14 sensors for RUL prediction.*	Model based on a Transformer architecture, using encoder and decoder blocks.	C-MAPSS PHM 2008	*RMSE = 11.43 (FD001)* *S = 203.15 (FD001)* *RMSE = 11.32 (FD003)* *S = 154.92 (FD003)* *S = 845 (PHM 2008)*
[1]	2021	Novel feature-attention mechanism applied to the input data to give more attention weights to more important features dynamically in the training process.	Combination of CNN and BGRU (a simpler version of BiLSTM)	C-MAPSS	*RMSE = 12.42 (FD001)*
[8]	2021	They used a multi-head mechanism in which each sensor output is processed on a fully independent head in a multi-head network- Each head is responsible for extracting meaningful features from the sensor data. Employed an attention mechanism on top of the multi-head implementation of different network architectures. They also tested a stand-alone attention model (SAN)	Tested different model architectures. Best results with a fully connected neural network (multi-head FNN).	C-MAPSS	*RMSE = 8.68 (FD001)* *S = 28.43 (FD001)* *RMSE = 9.69 (FD003)* *S = 77.01 (FD003)*
[2]	2019	Investigates the effect of unsupervised pre-training in RUL predictions utilizing a semi-supervised setup. They used a Genetic Algorithm (GA) approach to tune the diverse amount of hyper-parameters in the training procedure.	Restricted Boltzmann Machines as unsupervised algorithm. GA	C-MAPSS	*RMSE = 12.56 (FD001)*
[4]	2018	They used auto encoder as a feature extractor (as kind of a PCA replacement) to compress condition monitoring data. BiLSTM was used to capture features' bidirectional long-range dependencies. They provide a result comparison with other work.	Hybrid autoencoder-BiLSTM	C-MAPSS	*RMSE = 13.63* *Se-3 = 0.261*
[url]	2017	"Deep Learning Basics for Predictive Maintenance"[1]	Basic LSTM	C-MAPSS	*Acc: 0.98, F1:0.96*

[1] https://github.com/Azure/lstms_for_predictive_maintenance

III. Datasets:

There are two publicly available datasets at the "NASA's Prognostics Data Repository":

- CMAPSS dataset that contains 4 sub-datasets (i.e., FD001, FD002, FD003, and FD004). This dataset was published in 2008 [5], and it has been used extensible on the literature to benchmark run to failure and predictive maintenance systems.
  However, there is a fidelity gap in this dataset as the simulated degradation trajectories lack important factors of complexity that are present in real aircraft engines.
  Each of the sub-datasets is divided into a train and a test file.

  The following table presents the total number time events including the train and test files.

Sub-dataset Name	No. Units	No. of time events
FD001	100	33730
FD002	100	67468
FD003	100	41319
FD004	100	78689

To fulfill the unrealistic characteristics of the CMAPSS dataset, the N-CMAPSS dataset was created:

- N-CMAPSS was developed in 2020/2021 [7][6]. This dataset contains 8 sub-datasets and is relatively new. Therefore, to date, not many studies have been done with this dataset. Overall, this dataset brings higher fidelity to the degradation and the operating conditions represented in the old dataset which could improve the usability and the transferability of the developed data-driven models to real-world applications.

  The following table presents different characteristics of the N-CMAPSS dataset. "Flight Classes" and "Failure Modes" corresponds to the operating conditions, and "Size" represents the total number of time events.

Name	# Units	Flight Classes	Failure Modes	Fan		LPC		HPC		HPT		LPT		Size
				E	F	E	F	E	F	E	F	E	F	
DS01	10	1, 2, 3	1							✓				7.6 M
DS02	9	1, 2, 3	2							✓		✓	✓	6.5 M
DS03	15	1, 2, 3	1							✓		✓	✓	9.8 M
DS04	10	2, 3	1	✓	✓									10.0 M
DS05	10	1, 2, 3	1					✓	✓					6.9 M
DS06	10	1, 2, 3	1			✓	✓	✓	✓					6.8 M
DS07	10	1, 2, 3	1									✓	✓	7.2 M
DS08	54	1, 2, 3	1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	35.6 M

In this work I used the DS03 dataset as is the second one with the highest number of engine units, i.e., 15.

IV. Method and Results:

To accomplish this project, I first used the CMAPSS dataset to test different models as a baseline benchmark. Then, I used one of the sub-dataset from N-CMAPSS to train the model that performs best on the CMAPSS dataset.

Features

The CMAPSS dataset had three operational settings and 23 sensor readings, which account for 26 input features. The last two sensor reading feature columns were removed as it contains just NaN values. Then, a total of 24 input features were used in this work. The target variable "remaining useful life" (RUL) was computed by decreasing linearly the maximum cycle value for each engine unit.

The input features for the prognostic problem (i.e., RUL estimation) in the N-CMAPSS dataset were four scenario descriptor (*W*) (see Table 4 below), a set of 14 sensor measurements (*Xs*) (see Table 5 below), and 14 virtual sensor readings (*Xv*) (see Table 6 below). Then, a total of 32 input features were used for RUL estimation with the N-CMAPSS dataset.

Table 4. Scenario descriptors (i.e., flight data)—w.

#	Symbol	Description	Units
1	alt	Altitude	ft
2	Mach	Flight Mach number	-
3	TRA	Throttle–resolver angle	%
4	T2	Total temperature at fan inlet	°R

Table 5. Measurements—x_s.

#	Symbol	Description	Units
1	Wf	Fuel flow	pps
2	Nf	Physical fan speed	rpm
3	Nc	Physical core speed	rpm
4	T24	Total temperature at LPC outlet	°R
5	T30	Total temperature at HPC outlet	°R
6	T48	Total temperature at HPT outlet	°R
7	T50	Total temperature at LPT outlet	°R
8	P15	Total pressure in bypass-duct	psia
9	P2	Total pressure at fan inlet	psia
10	P21	Total pressure at fan outlet	psia
11	P24	Total pressure at LPC outlet	psia
12	Ps30	Static pressure at HPC outlet	psia
13	P40	Total pressure at burner outlet	psia
14	P50	Total pressure at LPT outlet	psia

Table 6. Virtual sensors—x_v.

#	Symbol	Description	Units
1	T40	Total temp. at burner outlet	°R
2	P30	Total pressure at HPC outlet	psia
3	P45	Total pressure at HPT outlet	psia
4	W21	Fan flow	pps
5	W22	Flow out of LPC	lbm/s
6	W25	Flow into HPC	lbm/s
7	W31	HPT coolant bleed	lbm/s
8	W32	HPT coolant bleed	lbm/s
9	W48	Flow out of HPT	lbm/s
10	W50	Flow out of LPT	lbm/s
11	SmFan	Fan stall margin	–
12	SmLPC	LPC stall margin	–
13	SmHPC	HPC stall margin	–
14	phi	Ratio of fuel flow to Ps30	pps/psi

Data Preprocessing

First, the input and target variables were scaled using the MinMaxScaler[2] from the "scikit-learn" python package. Then, the test files in the CMAPSS dataset were divided into validation and test sets. Specifically, out of the 100 engine units, units number 1 to 69 were assigned to the validation set and the rest to the test dataset.

Whereas for the N-CMAPSS dataset, out of the 15 engine units, units number 1 to 9 were assigned to the training set, 10 to 12 assigned to the validation set, and the remaining three to the test set.

Model and Results

The initial baseline model architecture contained: two LSTM layers with 100 units each, and one dropout layer after each LSTM with probability 0.2, and a fully connected linear layer as an output with a ReLu activation function.

The following table presents the employed baseline model architecture:

Layer Type	No. Layers	No. Nodes
LSTM	2	100
Dropout (0.2)	1	--
FC	1	100

[2] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

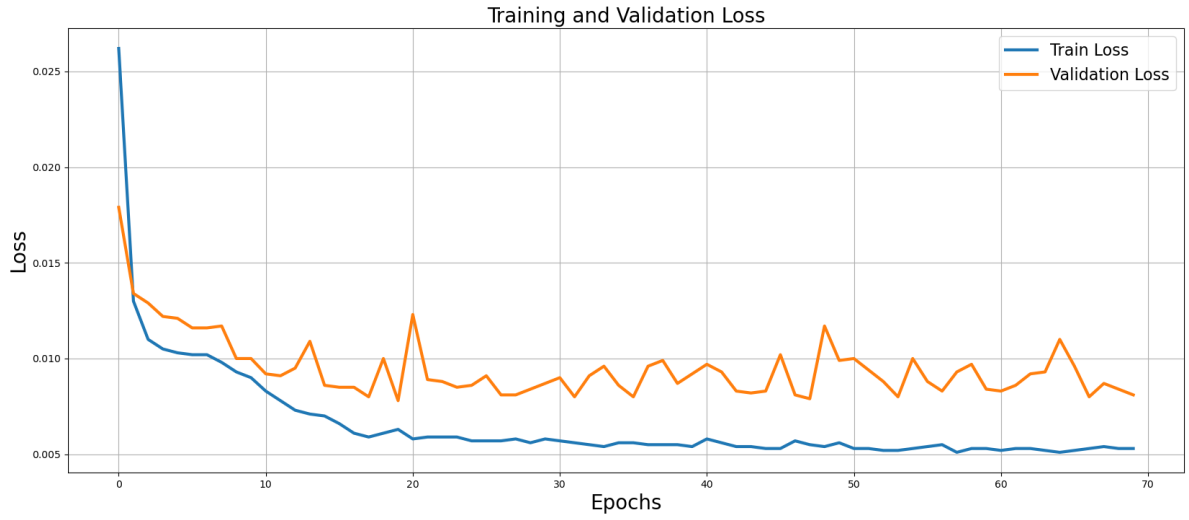
The selection of a proper window sequence is very important for time series prediction problems. I used 40 as a sequence length as authors of [3] achieved better results with this length after testing multiple windows lengths (i.e., 30, 40, 50, 60, and 70).

The model was trained for 50 Epochs, 0.001 as learning rate, 1024 as batch size, Adam optimizer, and MSELoss as loss function. The total number of trainable parameters was: 131,301. All implementation are based on the PyTorch deep learning framework[3]. Training, validation and test was conducted in the dedicated NVIDIA GPU RTX 3000 with 6Gb of memory.

The following table presents the Root-Mean-Squared-Error (RMSE) for the validation and test dataset.

Sub-dataset Name	Validation (RMSE)	Test (RMSE)
FD001	32.5	39.39
FD002	39.37	37.2
FD003	47.43	57.2
FD004	71.75	55.62

The following figure shows the training and validation loss for the FD001 sub-dataset for this baseline model:



In the following, I tested bidirectional LSTM as I discovered in the literature that it usually performs better than simple LSTM for this type of problem [1][4]. And in doing so, I increased the model complexity. So, the new model under test had:

- Three fully connected bidirectional LSTM layers with 100 units each, and a dropout layer after each Bi-LSTM with probability 0.5, and a fully connected linear layer as an output with ReLu activation function.

[3] https://pytorch.org/

The following table presents the new Bi-LSTM model architecture:

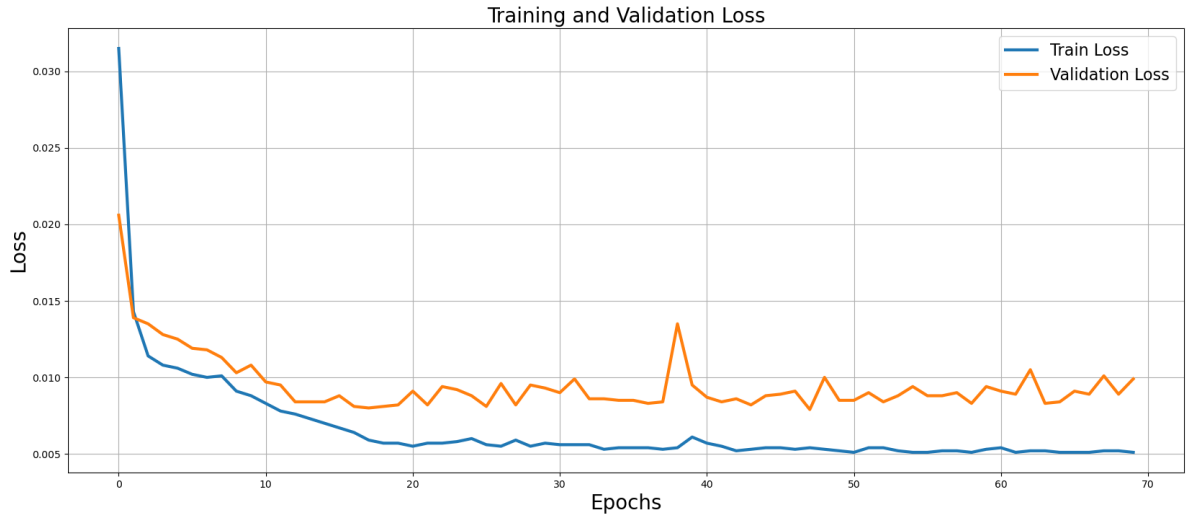
Layer Type	No. Layers	No. Nodes
Bi-LSTM	3	100
Dropout (0.5)	2	100
FC	1	200

- The resulting model had 584,201 trainable parameters. The following table present the RMSE achieved in the validation and test set:

Sub-dataset Name	Validation (RMSE)	Test (RMSE)
FD001	32.87	37.7
FD002	42.1	39.6
FD003	49.3	55.34
FD004	71.5	59.5

Overall these results suggest that the new Bi-LSTM model is able to generalize better than the previous baseline model. The new RMSE scores for the test set had decreased except of FD002 and FD004.

The following figure shows the training and validation loss for the FD001 dataset of the Bi-LSTM model.



Results on the N-CMAPSS Dataset:

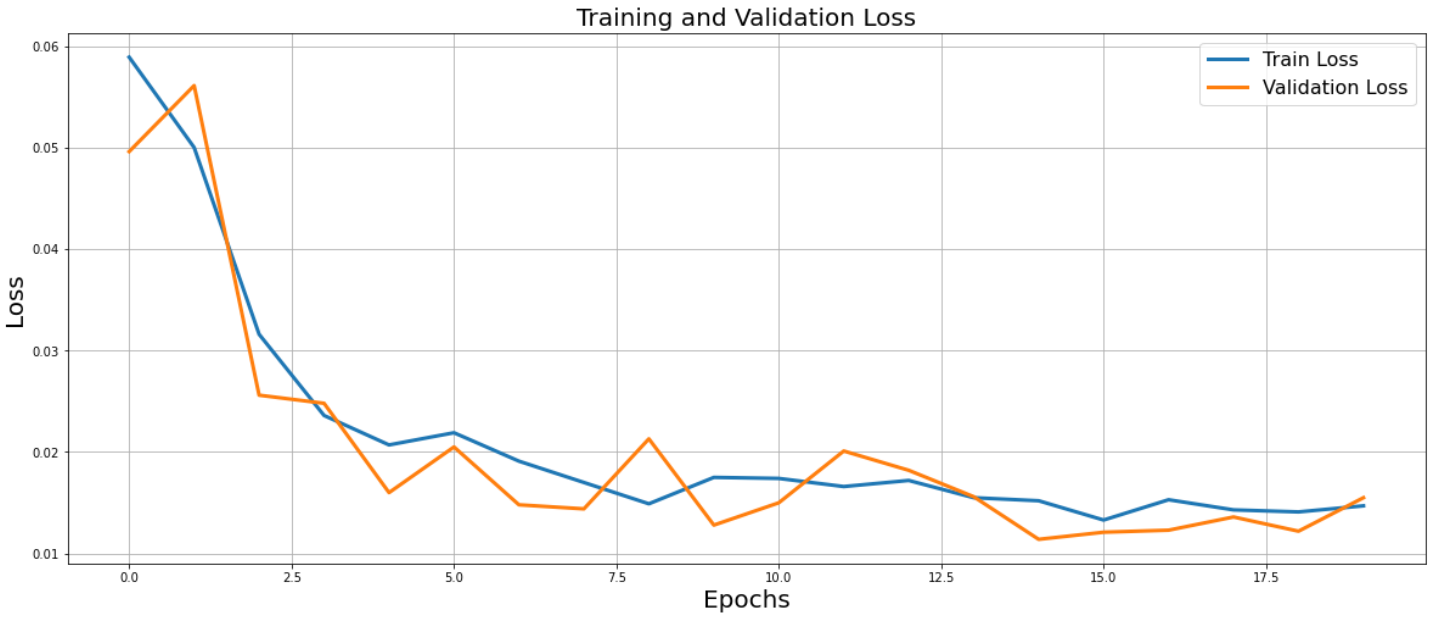
Furthermore, I tested the new Bi-LSTM model architecture on one of the large N-CMAPSS dataset. Specifically, I selected the dataset DS03 as it was the second one with the highest number of turbofan units, i.e., 15 in total. This dataset had a total of 9.8 million time events.

As previously noted, I split the dataset into 9 units for training, and three units for validation and test, respectively.

I trained this model in AWS, using Amazon SageMaker with a “ml.p2.8xlarge” instance. This instance had 8 GPUs and 32 CPUs cores and enough memory to train the model with such a large dataset. I used all GPUs and CPUs cores through data parallelization.

Batch size was set to 64512, 70 epochs and the remaining hyper parameters were kept constant. The overall training time was 53 minutes and 26 seconds. The Achieved RMSE on the test dataset was 8.67.

The figure below shows the training and validation loss:



An RMSE of 8.67 is a very good result. State-of-the-art benchmark on the CMAPSS dataset is 8.68 on the FD001, and the other best scores in the literature are around 11, 12 and 13 RMSE. Of course CMAPSS is not the same as the new and large N-CMAPSS dataset, but at least we can consider it as a reference. I was not able to find any new studies done with the new N-CMAPSS dataset.

V. Challenges:

At the beginning I scaled only the input features and the model was not working. It seems that the model was training but then it predicted always the same number. Then, after multiple headaches I discovered that in this case I had to scale also the target variable.

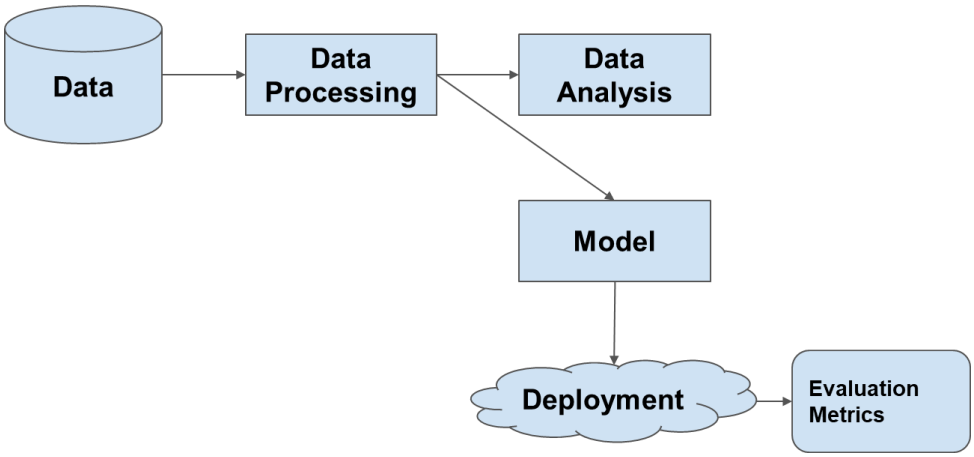
Another challenge was with the second dataset and memory requirements. For example, after splitting the DS03 sub-dataset from N-CMAPSS, the training set was 1.1Gb, but after creating the sequences it occupied more than 50Gb on RAM. Therefore, I had to rely completely on AWS. A solution could be to change the part of my implementation and save on disk the sequences and load to RAM the needed sequences on demand.

VI. System:

The implemented system has the following components:

- Data
- Data Processing
- Data Analysis
- Model
- Deployment
- Evaluation Metrics

The following diagram shows a broad overview of the connection between the different system components:



VII. Ethics:

I don't see mayor ethical concerns in this project as both dataset was synthetically generated and it doesn't contain any user data. Instead it contains sensor readings.

VIII. Future Work:

As a future work I would test the Bi-LSTM model on the other 7 datasets of the new large N-CMAPSS dataset. And I would also try different network architectures based on transformers and attention mechanism, that it has been found in the literature to perform very good on the old CMAPSS dataset.

Bibliography

[1] H. Liu, Z. Liu, W. Jia, and X. Lin, "Remaining Useful Life Prediction Using a Novel Feature-Attention-Based End-to-End Approach," *IEEE Trans. Ind. Informatics*, vol. 17, no. 2, pp. 1197–1207, 2021, doi: 10.1109/TII.2020.2983760.

[2] A. Listou Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliab. Eng. Syst. Saf.*, vol. 183, pp. 240–251, 2019, doi: https://doi.org/10.1016/j.ress.2018.11.027.

[3] Z. Zhang, W. Song, and Q. Li, "Dual Aspect Self-Attention based on Transformer for Remaining Useful Life Prediction." 2021.

[4] Y. Song, G. Shi, L. Chen, X. Huang, and T. Xia, "Remaining Useful Life Prediction of Turbofan Engine Using Hybrid Model Based on Autoencoder and Bidirectional Long Short-Term Memory," *J. Shanghai Jiaotong Univ.*, vol. 23, no. 1, pp. 85–94, 2018, doi: 10.1007/s12204-018-2027-5.

[5] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*, 2008, pp. 1–9, doi: 10.1109/PHM.2008.4711414.

[6] M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft Engine Run-to-Failure Dataset under Real Flight Conditions for Prognostics and Diagnostics," *Data*, vol. 6, no. 1, p. 5, 2021.

[7] M. Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft Engine Run-to-Failure Dataset under real flight conditions," 2020.

[8] A. Ayodeji, W. Wang, J. Su, J. Yuan, and X. Liu, "An empirical evaluation of attention-based multi-head models for improved turbofan engine remaining useful life prediction." 2021.