Remaining Useful Life Prediction of Aircraft Engines with Variable Length Input Sequences

Andreas Lövberg[1]

[1] *RISE Research Institutes of Sweden, Box 104, SE-431 22, Mölndal, Sweden*
andreas.lovberg@ri.se

ABSTRACT

Remaining useful life (RUL) is the expected remaining operating life of an asset until it can no longer perform its intended function. The 2021 PHM Society Data Challenge posed the problem of estimating the RUL of aircraft engines with various competing failure modes and underlying degradation trajectories. In this work, we describe the approach and solution to the challenge where we map from the multivariate time series sensor readings to the remaining useful life, measured in the remaining number of flight cycles until failure. The proposed solution utilizes a deep convolutional neural network that can take inputs of variable length. Furthermore, we preprocess the data according to a normalization procedure that help reveal the degradation trend that is obfuscated by the continuously varying flight conditions. The normalization procedure involves training a feedforward neural network on a non-degraded subset of the data that maps from the flight conditions to the sensor outputs. The difference between the expected sensor readings and the actual observations is then interpreted as the extent of deviation from normal, i.e., degradation. Finally, we describe the sampling techniques which is designed to reduce the number of non-informative samples fed to the neural network.

1. INTRODUCTION

In the context of Prognostics & Health Management (PHM) (Vachtsevanos, Lewis, Roemer, Hess, & Wu, 2006), prognostics refers to predicting the remaining useful life (RUL) of an asset or industrial system. Accurate RUL predictions enable predictive maintenance strategies as it can give advanced warning of incipient system failures. Such prognostics capabilities not only enable maintenance and reduce downtime, but it can also prevent catastrophic failure while maintaining safety and reliability. As the availability of condition monitoring data for engineering systems is increasing, the application of data-driven models to estimate time to failure is becoming more common.

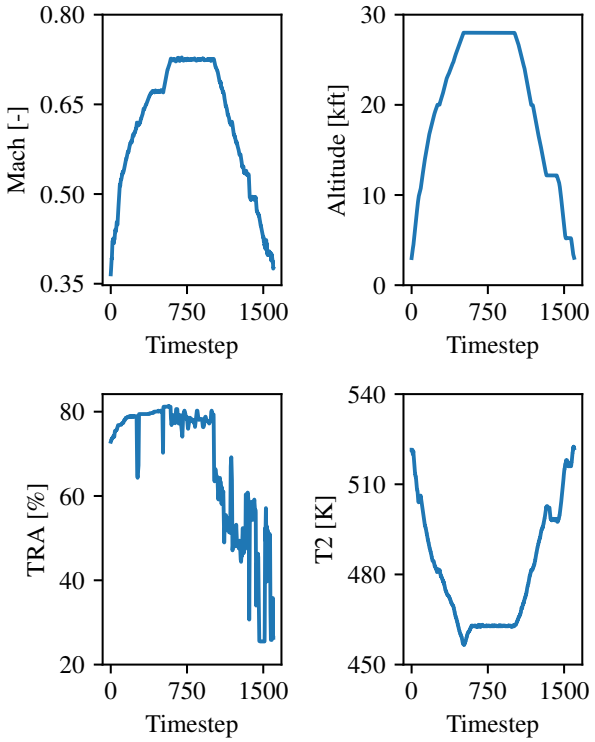


Figure 1. The four flight condition settings for a randomly selected cycle and engine.

The most common dataset used to benchmark prognostics algorithms is the CMAPSS dataset (Saxena & Goebel, 2008b) which contains run-to-failure trajectories of simulated aircraft turbofan engines. A first instance of this dataset was first introduced as the PHM08 Data Challenge (Saxena & Goebel, 2008a). Since the release of the datasets, many different strategies for RUL estimation of turbofans have been proposed. The most successful models on the usually involve deep learning based architectures. Architectures ap-

Andreas Lövberg et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

plied to this dataset are most commonly recurrent networks (Heimes, 2008) or convolutional networks (Li, Ding, & Sun, 2018). While the architecture and details of the models applied varies between different studies, the data preprocessing and data ingestion typically remain the same. A fixed size sliding window technique is usually applied to create the input samples. Using this approach, the window size is limited by shortest sequence length on which you want to be able to make predictions at test time. For sequences that are longer than the window size, information and context is discarded as it is outside the window length. An alternative way is to use an architecture that can handle sequences of variable length. Convolutional and recurrent networks are generally input length agnostic, hence using the fixed sliding window is not a requirement. For convolutional networks, a fixed length feature representation can be achieved by a global pooling operation before the final output layer. By taking the average of each feature over the time dimension, the output features will be of the same dimensions, regardless of the input sequence length. In this paper, we describe the development of a data-driven model to predict RUL of turbofan engines as part of the 2021 PHM Society Data Challenge.

2. PROBLEM FORMULATION

The 2021 PHM Society Data Challenge data consists of a subset of the N-CMAPSS dataset (Arias Chao, Kulkarni, Goebel, & Fink, 2021). In total, 90 synthetic run-to-failure trajectories of turbofan engines, generated by the CMAPSS model (Frederick, DeCastro, & Litt, 2007), are provided. There is a total off seven different failure modes within the dataset. Each engine has an unknown initial health state. As opposed to the PHM08 Data Challenge, N-CMAPSS has flight conditions from real flights, recorded on a commercial jet, as inputs to the model. Figure 1 shows the four different flight settings for a randomly selected cycle in a randomly selected engine. These operating conditions also relate to the degradation conditions where its history determine the onset of abnormal degradation. Each engine has 14 sensor measurements, X_s, recorded for the full trajectory until failure. The flight conditions, W, consisting of the 4 variables in Figure 1, are also provided. Additionally, 4 auxiliary variables are made available: unit number, flight cycle number, flight class and health state (h_s).

The objective of the competition is to use the 90 trajectories in the training set to predict the RUL of 38 engines in a test set. In the test set, the engine trajectories are truncated at random points in time. In other words, the engines in the test set are only partially degraded. The metric of the competition evaluation is an aggregation of the root mean squared error and a scoring function:

$$\text{score} \;=\; \text{RMSE} \cdot 0.5 \;+\; s \cdot 0.5 \tag{1}$$

$$\text{RMSE} \;=\; \sqrt{\frac{1}{n}\sum_{i=1}^{n}\Delta_i^2} \tag{2}$$

$$s \;=\; \frac{1}{n}\sum_{i=1}^{n}\exp(\alpha|\Delta_i|) \;-\; 1 \tag{3}$$

Where n is the number of engines to score, Δ_i is the difference between the true and predicted RUL for the i-th engine, and α is $\frac{1}{13}$ if RUL is under-estimated and $\frac{1}{10}$ if it is over-estimated. Hence, the scoring function is asymmetric and penalizes over-estimations of the RUL.

3. METHODOLOGY

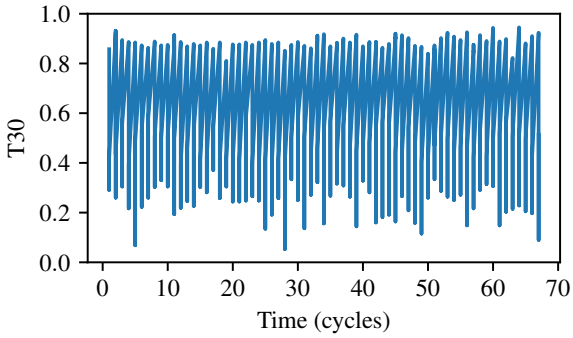
In this section we describe the methodology from the preprocessing step to the training procedure.

3.1. Preprocessing

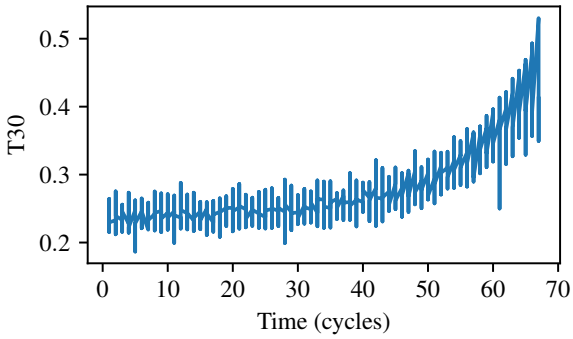
Due to the time constraints of the competition, we first reduce the size of the dataset to improve the iteration speed. This is done in two ways. First, we change the sensor readings from double-precision floating-point format to half-precision floating-point format, effectively reducing the storage and memory footprint by 75%. Second, upon inspecting the sensor measurements, we reduce the sampling frequency from 1Hz to 0.1Hz by decimation, further reducing the size. A side effect of reducing the frequency is that we achieve a longer receptive field for the same number of layers in the neural network. The downside is that, for both data reduction steps, information is lost. The motivation behind it is that the long-range trend is more important than the local perturbations within individual flight cycles as the degradation is not abrupt enough to be lost in the decimation step.

3.2. Normalization procedure

The sensor signals are first normalized with respect to the flight conditions, W. We apply a feedforward neural network that map from the flight conditions to the sensor outputs. The input consists of 5 features: the flight condition variables flight mach number (mach), altitude (alt), throttle resolve angle (TRA) and total fan inlet temperature (T2), as well as the flight class (a categorical feature describing the flight length) and a positional variable of time elapsed within the flight cycle (ranging from 0 to 1 within each cycle). A subset of the data, where the health state variable, h_s, indicates a non-degraded engine, is used as training data. By using healthy state data, we create a model that can output the expected sensor readings given some flight condition. The difference between the model estimates and the actual sensor readings can then be interpreted as the degree of degradation



(a)



(b)

Figure 2. (a) T30 sensor values for a randomly selected engine. The degradation trend is obfuscated by the flight conditions. (b) shows the same sensor signal after transformation and normalization with respect to flight conditions. The degradation trend is now observable.

or deviation from expected normal. The sensor data at each timestep t can then be calculated as

$$\hat{X}_t = X_t - f(W_t) \tag{4}$$

where the function f is the neural network with four layers and ReLU activations. The input layer of the network f has 5 units, the hidden layers have 128 units, and the output layer has 14 units that corresponds to the 14 sensor signals. We use the Adam optimizer with decoupled weight decay (Loshchilov & Hutter, 2019). A learning rate of 5e-4 is used and the weight decay is set to 1e-3. The model is trained for 100 epochs. After the transformation step, $\hat{X}$ is normalized in the [0,1] range through min/max-normalization.

Figure 2a shows the T30 sensor values for a randomly selected engine. Due to the continuously varying operating conditions, no clear degradation signal can be observed. Figure 2b shows the same sensor signal after the transformation step described above. A clear degradation trend can now be observed.

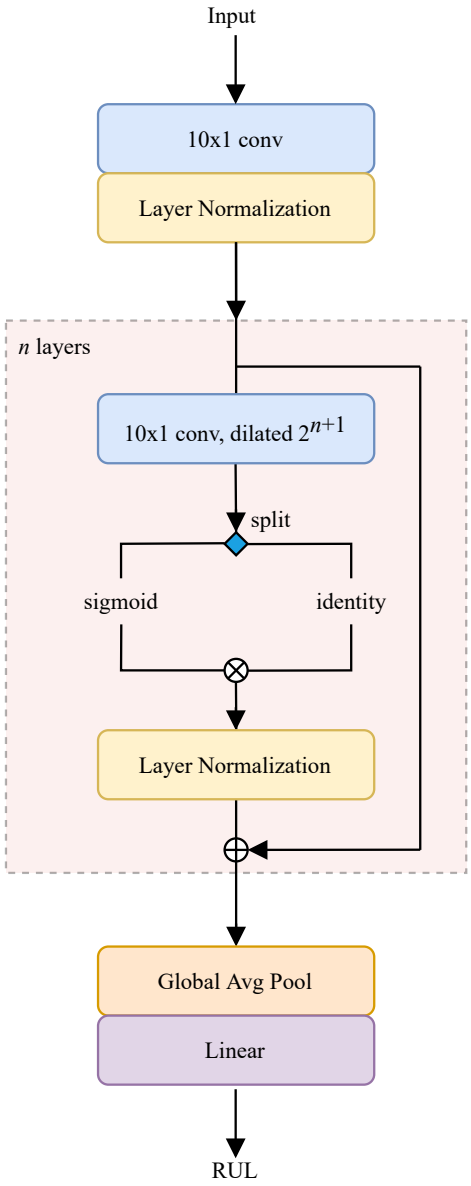


Figure 3. Diagram of the remaining useful life prediction model. It has 10 layers where each layer consists of a dilated convolution, a gated linear activation unit, and layer normalization. A skip connection is also added, indicated by the arrow bypassing the above operations.

3.3. Remaining useful life prediction model

To map from the condition monitoring sensor signals to the remaining useful life, we apply a deep neural network consisting of dilated convolutions with gated linear activations and residual skip connections. The purpose of the dilation factor is to increase the effective receptive field of the network without increasing the number of parameters. A dilated convolution is a convolution where the receptive field of a kernel is extended by skipping input values with a certain step size.

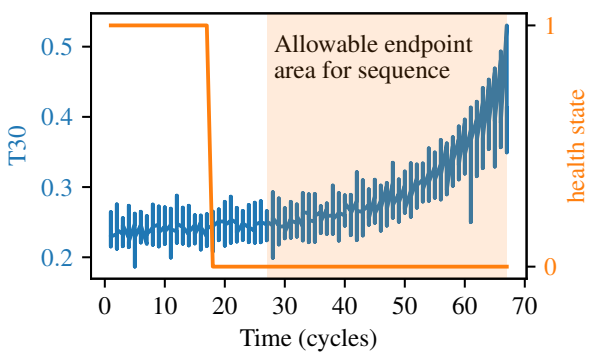


Figure 4. Illustration of the allowable sampling area for sequences. The endpoint of the sequence must be in the shaded area. The shaded area begins when the 20% of the total time in abnormal operation has elapsed. This is applied across all 14 sensors.

In other words, we artificially increase the span of the kernel without increasing the number of parameters. The combination of the decimation of the data with the dilated convolutions allow the model to operate on a much coarser scale than if we were to use standard convolutions on non-decimated data. By stacking the dilated convolutions, we can achieve a very large receptive field in just a few layers. Since we also apply skip connections, the receptive field can be considered flexible. An overview of the model architecture is shown in Figure 3.

The convolutional layers have 8 filters and a kernel size of 10. In the penultimate layer, global average pooling is applied, where the features are averaged over the time dimension. This way, the input to the final linear layer is fixed in size, regardless of the input sequence length. The dilation rate for layer n is 2^n. We stack 10 layers resulting in a dilation rate sequence of 1,2,4...,1024. For each layer, we apply layer normalization (Ba, Kiros, & Hinton, 2016). The activation applied is the gated linear unit (GLU) where the input features are split in to two feature maps. One of the feature maps is then passed through a sigmoid function and then multiplied with the other half. Each layer also includes a residual (skip) connection such that

$$x_{i+1} = x_i + g_i(x_i, \theta_i) \tag{5}$$

where x_i is the feature map at the i-th layer, g_i is the convolutional operation and θ_i is the parameters of the kernel.

For training we use the Adam optimizer with decoupled weight decay (Loshchilov & Hutter, 2019) and a learning rate of 5e-4. The weight decay is set to 1e-3. The model is trained for 11000 gradient steps, where Stochastic Weight Averaging (SWA) (Izmailov, Podoprikhin, Garipov, Vetrov, & Wilson, 2018) is applied for the last 1000 steps. The filter size, kernel size and weight decay hyperparameters is chosen based on two factors. First, the initial value is set based on priors obtained from previous experiments on the CMAPSS dataset. Second, the parameters are adjusted based on the validation score (we keep 6 engines out of the training data for validation). In addition to the 14 sensor values, we add 3 categorical variables to the input tensor. Cycle number, flight category and health state are concatenated to the input via entity embeddings (Guo & Berkhahn, 2016). The final submission to the competition is an average of six model training runs, where all 90 trajectories as used as training inputs.

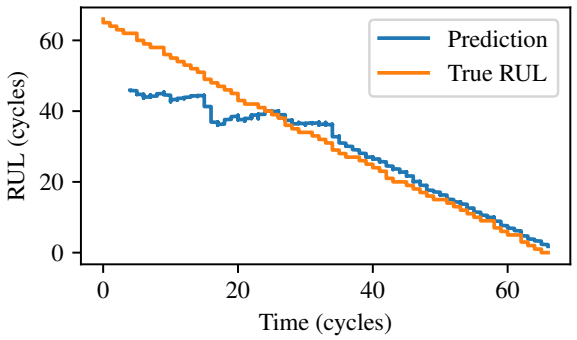
3.4. Sequence Sampling

As described in the N-CMAPSS dataset description document (Arias Chao et al., 2021), degradation first occur linearly and then transitions to abnormal degradation. The transition happens slowly and the health decreases throughout the lifetime of the engine. We speculate that models which can operate on a coarse scale is more likely to discover the true underlying degradation trajectory than one that operates on a fine-grained scale. Intuitively, a shorter sequence of the condition monitoring signals yields a more uncertain estimate of the true degradation state due to noise and perturbations. Conversely, a model operating on a coarse scale may be disadvantaged when the degradation and failure occur rapidly, or when a transient event has a large impact on the trajectory.

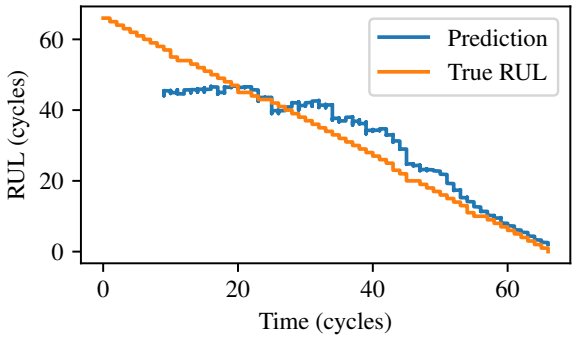
As the degradation of the turbofan engines progresses, the stronger this signal becomes in the sensor data. Early in the turbofan engine lifetime, in its healthy state, there is no clear degradation observable. Sequences that do not contain any degradation signal then act as noise in the training of the prognostics model. To alleviate this issue, we propose a sampling procedure where any training sequence fed to the model must contain some degree of degradation signal. The health state variable given in the dataset, h_s, indicates whether engine is operating normally ($h_s = 1$) or abnormally ($h_s = 0$). We make use of this label by making sure that the endpoint of our sampled sequence must be in the abnormal degradation regime. More specifically, we define this as the point in time when 20% of the total abnormal operation period has elapsed. This is arbitrarily chosen.

While sequences can still contain long segments of non-degraded signal, the end of the sequence must include degradation information. This is illustrated in Figure 4, where the allowable end-point RUL label area is marked. The figure shows a single sensor (T30) but it is applied across all 14 input sensors.

Input to the neural network typically consist of sequences of fixed length where a sliding window is applied across the input signals to create the training samples. The downside of such an approach is that the window size must be the length of the shortest sequence on which you want to make predictions



(a)



(b)

Figure 5. RUL predictions for two engines in the validation set. Predictions are conservative early in the engine lifetime.

at test time. In other words, you cannot consider a longer time horizon than the chosen window length even when the input you want to make predictions on is significantly longer than that.

Training loop: Here we describe the training procedure. For each gradient step, a sequence length in the range $[L_l, L_h]$ is randomly chosen. L_l is defined by the shortest sequence length in the test set, while L_h is defined by the shortest trajectory in the train set. Given the sequence length, we randomly sample a sequence from N number of engines in the allowable range, given by the change point location ($h_s = 0$). The N number of sequences is then concatenated to form the batch dimension. The resulting input tensor has the shape [N, C, L], where C is the number of input channels (i.e., number of sensors in the condition monitoring data) and L is the sequence length. Each sequence has a RUL label associated with it, forming the target variable Y of shape [N,1]. N is set to 20.

4. RESULTS AND DISCUSSION

The top 5 entries in the competition is shown in Table 1. Figure 5 shows the RUL predictions for two engines in the validation set. As expected, the prediction comes closer to the true value as we move closer to the failure point. As we have chosen a minimum sequence length L_l, the first prediction point will occur after L_l timesteps (this corresponds to 8000 timesteps), as the model outputs a single estimate of the RUL for the whole sequence. Predictions early in the engine lifetime tend to be conservative. This is intentional as we want to avoid the large penalty score given by late predictions in the scoring function (Equation 3). The reason for the conservative predictions is twofold. First, the sampling technique described in section 3.4 limits the number of samples with a high RUL, making the model less likely to make such predictions. Second, we apply weight decay to all the convolutional layers, as well as the final linear output layer, helping suppress large RUL values from being output.

Table 1. Top 5 entries in the 2021 PHM Society Data Challenge

Rank	Team	Score
1	**IJoinedTooLate**	**3.006**
2	YellowJackets	3.327
3	DatrikUS	3.651
4	SHRMer	3.689
5	XJTUPHM	4.017

In the PHM08 Data Challenge, this problem was handled by introducing a piece-wise linear RUL label where the number of cycles to failure is truncated at some value representing the shift to abnormal degradation (Heimes, 2008). A similar strategy can also be applied here, but in contrast to the PHM08 Data Challenge, we have access to the change points for each individual engine. If access was not given to the health state label, a similar label can be constructed via an autoencoder. As the normalization scheme presented in section 3.2 reveals the degradation trend, compressing the multivariate sensor signals to a single dimension can give an overall degradation trend. The substitute for the health state label can then inferred by inspecting the derivative of the latent space signal, where the knee-point indicating the transition to abnormal degradation may be observed. This makes the proposed techniques applicable to scenarios where a health state label might not be readily available.

There are many ways in which the approach presented here can be improved. First, the transition point that indicated the allowable sampling space is somewhat arbitrarily chosen and could be improved by being treated as a tunable hyperparameter. Second, it is likely the case that a longer training time would improve the results. Due to the time constraints of the competition, the training was stopped before the validation loss had reached a minimum or plateaued. Third, results may improve if the sampling frequency is not decreased by such a large factor. Future work outside the time constraints of the competition setting will determine whether results improve if the data is not reduced. Lastly, a longer allowable sequence

length may also reduce the error for longer sequences in the test set. In our case, we have picked a maximum input length sequence of 30k timesteps (L_h) due to this being the shortest trajectory in the training set. Since we do not sample from all the engines in each gradient step, it is possible to adapt the maximum sequence length allowed based on the lengths of the engine trajectories being sampled. This should increase the accuracy of the predictions on trajectories in the test set that is longer than 30k timesteps. It should be noted that the time steps here refer to the decimated dataset. In the original format, this corresponds to 300k timesteps. Furthermore, given the large weight decay and the relatively small number of filters, the model is probably underfit. Combining this with the fact that the procedure presented here severely decimates the data, we believe that there is significant room for improvement in the score.

5. CONCLUSIONS

This paper has presented the winning solution to the 2021 PHM Society Data Challenge. The solution is based on the following main concepts:

- Transformation and normalization with respect to flight conditions.
- Dilated convolutions with a large receptive field.
- Variable length input sequences.

As the degradation process is slowly developing over time, working on a coarser scale over many flight cycles is more important than working on a local scale within individual cycles. To enable this, we propose an architecture that can make predictions on input sequences of variable length. Furthermore, we preprocess the multivariate time series sensor readings by calculating the distance from expected normal for each of the 14 sensors. This reveals the degradation trend in the data that is otherwise obfuscated by the flight settings.

ACKNOWLEDGMENT

This work received funding by the ECSEL Joint Undertaking (JU) under grant agreement No 876659. Support for the project by the Swedish Governmental Agency for Innovation Systems (Vinnova) under contract 2020-00991, is gratefully acknowledged.

REFERENCES

Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2021). Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, *6*(1). Retrieved from https://www.mdpi.com/2306-5729/6/1/5 doi: 10.3390/data6010005

Ba, J., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *ArXiv*, *abs/1607.06450*.

Frederick, D., DeCastro, J., & Litt, J. (2007, 01). User's guide for the commercial modular aero-propulsion system simulation (c-mapss). *NASA Technical Manuscript*, *2007–215026*.

Guo, C., & Berkhahn, F. (2016). Entity embeddings of categorical variables. *ArXiv*, *abs/1604.06737*.

Heimes, F. O. (2008). Recurrent neural networks for remaining useful life estimation. In *2008 international conference on prognostics and health management* (p. 1-6). doi: 10.1109/PHM.2008.4711422

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., & Wilson, A. (2018). Averaging weights leads to wider optima and better generalization. In R. Silva, A. Globerson, & A. Globerson (Eds.), *34th conference on uncertainty in artificial intelligence 2018, uai 2018* (pp. 876–885). Association For Uncertainty in Artificial Intelligence (AUAI).

Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering System Safety*, *172*, 1-11. Retrieved from https://www.sciencedirect.com/science/article/pii/S0951832017307779 doi: https://doi.org/10.1016/j.ress.2017.11.021

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International conference on learning representations.* Retrieved from https://openreview.net/forum?id=Bkg6RiCqY7

Saxena, A., & Goebel, K. (2008a). *Phm08 challenge data set.* http://ti.arc.nasa.gov/project/prognostic-data-repository. NASA Ames Research Center, Moffett Field, CA.

Saxena, A., & Goebel, K. (2008b). *Turbofan engine degradation simulation data set.* http://ti.arc.nasa.gov/project/prognostic-data-repository. NASA Ames Research Center, Moffett Field, CA.

Vachtsevanos, G. J., Lewis, F. L., Roemer, M. J., Hess, A. J., & Wu, B. (2006). Intelligent fault diagnosis and prognosis for engineering systems..