Remaining Useful Life cycle Prediction
PHM08 Challenge - Nasa Challenge Data
Siddharth Agarwal

Problem Description

In this problem we are given sensor time series data for 21 sensors originating from 100 engines with their total lifecycle (TUL). And we are asked to predict the RUL (remaining useful life of new set of engine). Each engine is considered to be of same type, with different degree of initial wear and manufacturing variations which is unknown.

DATASET:

1. Historical data for 100 engines with TUL values. Attributes represent Unit number, time, in cycles, operation settings, and sensor measurements.
2. Testing data for 7 engines for which we are required to predict the Remaining Useful Lifecycle

It is required that we build a generalized model to predict Remaining Useful Life cycle which is independent of particular engine.

Approach

This section has been divided into following different subsections:

Why it is not a time series forecasting problem?

Since in this problem we want to predict Remaining Useful life cycle of each engine we can break the problem by first predicting Total Life Cycle of each engine and then subtract the current cycle to find the Remaining Useful Life Cycle. Since total useful life cycle is one number for each each engine, therefore we are not predicting (or rather forecasting) any future time series from the current time series. Instead, we can think of this problem as regression problem where data happens to be time series.

Why it is not an anomaly detection problem?

It is very tempting to proceed with this problem in terms of anomaly detection framework since the setup of the problem is mostly similar to the anomaly detection problem. Data is a time series data with condition of any hardware asset worsening with time. Data from the hardware is captured by sensors. Now, the difference between current problem and any anomaly detection problem is that in anomaly detection problem we don’t know if hardware asset is performing normal or anomalous. We have to differentiate between anomalous data and normal data. In current problem we already know that all the input data provided to us is anomalous. In that anomalous data we need to predict total time for which that hardware will survive.

Why it is a prognosis problem?

Prognosis is a subfield in preventive and predictive maintenance in IOT which can make better predictions about “when” something going to happen. This helps in better planning, inventory management, and maintenance. This is exactly what we want to predict in this problem given our setup.

Different approaches:

1. **Regression based approach:** In this we can just take the output variables for each cycle samples of time series with their TUL values, and use all the cycles of all the engines together of the training data to predict the output TUL value for each cycle sample of the new engine. **Drawbacks:** This approach ignores the temporal behaviour of the data. It takes in the data points as if they are generated from some time independent distribution.

2. **Clustering - Classification based approach:** In this engines are grouped together in a cluster based on the similarity between sensor values and the output TUL is provided predicted based the cluster in which the engine lies. **Drawbacks:** Again this approach ignores temporal behaviour of the data and it predicts output only an approximate value of the output TUL.

3. **Data Compression:** Since during data collection phase data from multiple sensors are collected irrespective of which data is useful and not redundant in predicting the output TUL. It is now therefore necessary to reduce the dimensions of the data to remove the redundancy and also to retain the maximum variance in the data. Many data compression techniques are there:
    a. Principal Component Analysis: It uses eigenvalue decomposition to find orthogonal components with maximum variance, along with the original data is projected. It does not takes into account time dependence between samples.
    b. Other dimensionality reduction techniques such ICA, RP, LDA,etc: ICA is used to separate components in time series speech data.
    c. **Generative approaches:** Many generative approaches are used to find the latent variables (or encoding, or embeddings) in smaller dimensions which represents information about the data. Many generative models can be used especially in neural network based eg. Generative Adversarial Networks, Auto encoders, etc. **Recurrent Neural Network (RNN) based auto encoder** is especially valid for time series data since each LSTM block in RNN has memory that is it takes into account previous state.

4. **Data Compression + Regression:** Both data compression and regression can be used together to first reduce the dimensions of data and then use the reduced data to make predictions using regression. In this challenge, I have used two types of this approach :
    a. PCA + Regression: this doesn’t take into account temporal nature
    b. RNN auto encoders + Regression : this takes into account the temporal nature of data

5. **Domain knowledge based approaches:** Many researches have been conducted RNN autoencoders for data compression and then they calculate health indexes representing how good an engine is in terms of health. To calculate health indexes latent values of normal instances and latent value of current instances are compared. Then domain knowledge base methods are used to calculate RUL from health indexes, which represents function approximation techniques [1][2][3].

Approach Used:

In this submission I have used two approaches:

1. PCA + Direct Support Vector Regression
2. RNN based auto encoder and Support Vector Regression + Walk forward prediction to predict RUL as early as possible

Data Preparation and Cleaning:

Data is scaled for zero means and unit variance representing Gaussian distribution.

PCA + SVR:

In this approach I have first used Principal Component Analysis to first reduce the dimensions of the dataset from 26 dimensions to 14 dimensions. Number of dimensions are chosen such that 99% variance is retained among the components.

Then I have used grid search over the C and lambda parameters in Support Vector Regressor which represents the regularization term and degree of importance given to misclassification.

Optimal parameters obtained were C = 50, gamma = 1, kernel = ‘rbf’ (fixed)
3 fold cross validation R2 score obtained was 0.99

After this trained SVR is used predict the TUL and in turn RUL for each new engine.

Predicted RUL for engine 101: 180
Predicted RUL for engine 102: 159
Predicted RUL for engine 103: 84
Predicted RUL for engine 104: 102
Predicted RUL for engine 124: 26
Predicted RUL for engine 125: 162
Predicted RUL for engine 134: 0

TUL for an engine is taken as mean TUL for all instances of that engine

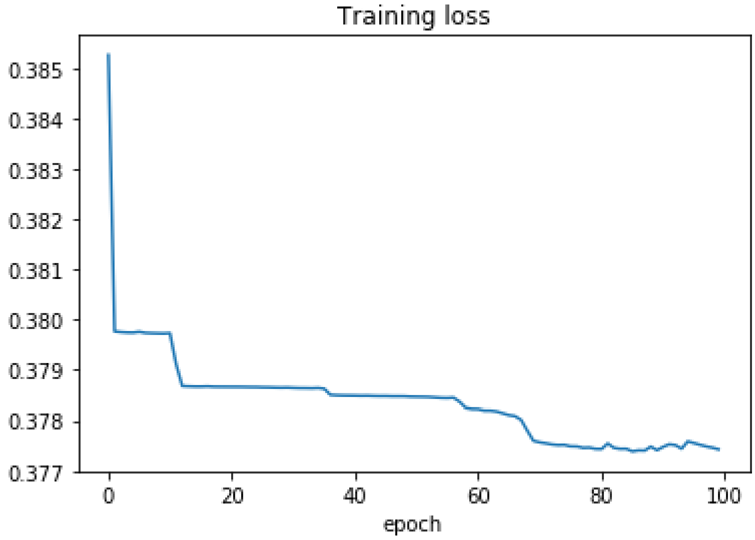
RNN based auto encoder and Support Vector Regression + Walk forward prediction:

In this I have first trained an RNN based auto encoder to minimize mean squared error between output and input time series using ‘Adam’ optimizer. I have used keras library for this.

Encoder of the RNN has 4 layers, last layer being dense layer and first three layers are LSTM layers which converts hidden state to latent variable (using ‘Rectified linear unit’ activation function). Output size of dense layer is 5, while output size of each other three layer is 6.
I am taking batch size as 4. Because I am training all the engines together I convert number of cycles in each training engine such that they are divisible by 4.

Similar corresponding decoder layer is used with a dense layer and 3 LSTM layer. Output of the decoder is the generated time series and parameters are tuned such that error between generated and original time series is minimized. Number of epochs = 100, and optimal MSE = 0.3774

Plot of error vs epoch is shown below



Now the output of the dense layer is taken as embeddings and I have taken mean embedding for each batch. This one embedding for each batch is taken as a data point for final support vector regressor which predicts TUL in a walk forward fashion.

Again grid search is performed on this classifier which gives us the best parameters for SVR as C = 100, gamma = 0.1, kernel = ‘rbf’ (fixed)

Best Cross validation R2 score was 0.156

In the end difference of mean TUL and total current cycle is taken as predicted the RUL for each engine.

Results:

RUL for engine 101: 179
RUL for engine 102: 152
RUL for engine 103: 69
RUL for engine 104: 95
RUL for engine 124: 32
RUL for engine 125: 158
RUL for engine 134: 0

Observations and improvements:

1. RNN encoder is performing better than PCA since it is incorporating temporal nature of the data.
2. SVR R2 score in the second approach is not very encouraging showing more scope for improvements.
3. Number of epochs for training could be increased for better embeddings.
4. Other regression and generative methods can be used.

References:

1. Gugulothu, Narendhar, Vishnu TV, Pankaj Malhotra, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. "Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks." *arXiv preprint arXiv:1709.01073* (2017).
2. Heimes, Felix O. "Recurrent neural networks for remaining useful life estimation." In *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, pp. 1-6. IEEE, 2008.
3. Malhotra, Pankaj, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. "LSTM-based encoder-decoder for multi-sensor anomaly detection." *arXiv preprint arXiv:1607.00148* (2016).