FSM Online Internship Report

Project Title: Remaining Usable Life Estimation(Bearing Dataset)
Domain: Machine Learning
FINAL REPORT

Submitted by:
Akanksha Singh
Thapar Institute Of Engineering & Technology
Patiala, Punjab - 147004

Under Mentorship of:
Devesh Tarasia

IITD-AIA Foundation for Smart Manufacturing

[1-June-2022 to 31-July-2022]
Remaining Usable Life Estimation (Bearing Dataset)

ABSTRACT

Components of rating machines, such as shafts, bearings, and gears are subject to performance degradation, which if left unattended could lead to failure or breakdown of the entire system. Analyzing condition monitoring data, implementing diagnostic techniques, and using machinery prognostic algorithms will bring about an accurate estimation of the remaining life and possible failure that may occur. Furthermore, the prediction of the bearing life bearing (RUL) plays a pivotal role in ensuring the safe operation of machinery and reducing maintenance loss. Many techniques for prognostics depend on estimating and then forecasting health indicators that reflect the overall health or performance of an asset. For vibration data, health indicators are typically calculated by combining various vibration measures along with derived features extracted from time, frequency, or time-frequency domain analysis. However, selecting or handcrafting good features is a labor-intensive task. Traditional prediction methods only consider the features of one domain or integrate the features of multiple domains into a one-dimensional sequence as the model input, which leads to some inaccuracy in prediction. On the other hand, deep learning models might be able to learn health indicators automatically from vibration data but require a large amount of training data, which is hard typically hard to obtain from real assets. This project implements the already proposed method using considering Continuous Wavelet Transform to extract time-frequency domain features from raw vibrational signals and then pass the extracted features through the CNN + LSTM model. The result obtained is then used to estimate health indicators which are then extrapolated to estimate the future health condition of the asset and its remaining useful life and is presented by deploying the model on a web browser.

Keywords: Continuous- Remaining Useful Life, Wavelet Transform, CNN + LSTM, Flask, Deployment, Python

Table of Content

1. INTRODUCTION

Rolling element bearings are significant mechanical components in rotating machinery, which is a common cause of operational failures. The precise RUL estimation of the bearings can obviously improve the reliability and operation safety of the rotating machinery and avoid accidents. Different from the related topics in PHM such as fault diagnosis and detection, remaining useful life (RUL) prediction aims to predict the remaining working time of bearings before failure occurrence. Due to some uncertain reasons like noise and varying working conditions, RUL prediction is always challengeable. Utilizing historical health data to collect data. Thanks to the quick development of artificial intelligence, data-driven methods have become increasingly popular. Although various types of monitoring data like temperature, pressure, and voltages can be collected from sensors, vibration signal can provide a straight forward expression of the bearing's working status and then is a good choice for RUL prediction. The essence of the data-driven methods is viewing the degradation process as a functional relationship between health states and monitoring data. In order to characterize the bearing's degradation behavior, machine learning algorithms and other intelligent techniques are introduced to model this function. Generally speaking, machine-learning-based. RUL prediction includes two strategies:1 (1) build a regression model for prognostics directly from the fault feature and the corresponding RUL value and (2) construct a health indicator (HI) and then predict RUL by analyzing the trend of HI. Well-constructed health indicators with good facilities can accurately inspect, check and keep track of mechanical element breakdown. The vibrational signal features should be thoroughly investigated before extracting data from them. Most data analysis can be performed in two ways - time-domain analysis and frequency-domain analysis, as well as derive vibrational signal features. However, time-frequency domain analysis is an appropriate method to examine mechanical bearing vibration signals because the vibration signals of the mechanical bearing are interconnected. Features derived from time-frequency domain analysis have properties of both time components and frequency components.

2. PROBLEM DEFINITION

To create and deploy a Machine Learning Model that predicts the Remaining Usable Life Estimation Of a bearing. The dataset was created as a part of the IEEE PHM 2012 challenge in which the data set was derived ed involved three different load conditions (by varying load force and rotating speed). Condition 1 had seven ball bearings operated at 1800 rpm with 4000 N radial load. Complete run-to-failure data for algorithm training were provided for two of the bearings, and truncated data for algorithm testing were provided for the other five bearings. Condition 2 featured seven ball bearings operated at 1650 rpm with 4200 N radial load. Of the seven bearings, complete run-to-failure data for training were provided for two, and data for testing were provided for five of the bearings. Condition 3 featured three bearings operated at 1500 rpm with a 5000 N radial load. Data from two of the bearings were provided for training and data from the other bearing was provided for testing. Data collected was segregated into the Learning and Test Dataset. For this project, the Learning Set includes operational data (time-series of horizontal and vertical measurements) from 6 different bearings, more exactly, 2 bearings for each of 3 groups exposed to different operating conditions. In each Learning Data Set, the unknown bearing was run for a variable time until failure. The lengths of the runs varied, with the minimum run length of 5,150 sec and the maximum length of 28,030 sec. The Test Set included operational data from bearing3 from the first operating conditions. In each Test Data Set, the unknown bearing was run for a variable time until the failure, but researchers have got only truncated time series.

a) Two accelerometers were mounted on the bearing housing to measure vibration in the vertical and horizontal directions.
b) Vibration measurements were performed every 10 sec at 25.6 kHz sampling rate and 0.1 s duration; hence, each observation contained 2560 points. (2,560 measurements per 0.1 sec).

c) The Learning and Test Sets also included temperature measurements (with a period 0.1 sec), but only for 10 bearings from all 17.

For model training only acceleration vibrational data was used. The reason for the same will be explained in the upcoming study.

3. EXISTING SOLUTION

Babuet al. [1] proposed a CNN model for RUL prediction, which has two convolutional layers and two pooling layers to extract the characteristics of the original signal, and combined it with a multilayer perceptron (MLP) to achieve bearing RUL prediction. Zhu et al. [2] used continuous wavelet transform to obtain the time–frequency images of bearing vibration signals and then input the images into CNN to achieve the bearing RUL prediction. Li et al. [3] performed a short-time Fourier transform on multiple time signals of bearings to obtain time–frequency images and then used time–frequency images to train CNN models; the correlation between neighboring signals was used to obtain good prediction results. Yang et al Liang Gao and others [4] introduced a data-driven method that uses traditional neural networks (CNN) to identify defects and a transformation technique that transforms signals into 2D images. CNN is by default an expert model for correctly extracting features from unprepared data. Feature extraction is quite important in data-driven methods and regression problems because it is helpful in understanding more insights about data and it also highly affects the end result, but feature extraction techniques are complex processes. So to avoid complexity in the work, the researchers used the CNN because it has the ability to extract the feature directly from the input image. The authors used 3 different data sets to evaluate the neural network model: (1) self-priming centrifugal pump fault diagnosis data set, (2) axial piston hydraulic pump fault diagnosis data set, (3) motor. Bearing Data Set. The suggested approach achieves high efficiency over all 3 data sets. In [5], the authors tackled an important in-depth study on RUL prediction for machines. In their study, many important issues such as data acquisitions, predictive models, health index (HI), and health stages (HS) were discussed.

4. PROPOSED DEVELOPMENT

This project focuses on implementing the already researched bearing's Remaining Useful Life (RUL) prediction data-driven method [6], combining the CNN and LSTM model, which considers both time-domain features & time-frequency utilized for training the model. Wherein the time-frequency features are extracted by using Continuous Wavelet Transform. Finally, the model is locally deployed using Flask for presenting the final result. Here the frontend of the final screen is created using the NicePage app later on which which changes were made as per requirement.

5. FUNCTIONAL IMPLEMENTATION

5.1. DATA STORAGE

Processing of each file every time when data is required is time-consuming as well as a hectic process. So, to make easier our work easier, all datafiles in each dataset are merged together. At the end, we get 6 Bearing Dataset files.

```
file merge successful Bearing2_2  csv file created
Bearing2_2 .pkz file created
file merge successful Bearing1_1  csv file created
Bearing1_1 .pkz file created
file merge successful Bearing3_1  csv file created
Bearing3_1 .pkz file created
file merge successful Bearing1_2  csv file created
Bearing1_2 .pkz file created
file merge successful Bearing2_1  csv file created
Bearing2_1 .pkz file created
file merge successful Bearing3_2  csv file created
Bearing3_2 .pkz file created
```

Fig1. Snippet of 6 Bearing folders after merging files

```
Bearing1_1 files with not 600 datapoints 1
Bearing3_1 files with not 600 datapoints 2
Bearing1_2 files with not 600 datapoints 1
Bearing2_1 files with not 600 datapoints 1
```

Fig2. Snippet of Bearing with not 600 datapoints of temperature dataset

HOW is our Dataset a Time-Series?

Time-series data is a set of observations i.e. data usually collected at discrete and equally spaced time intervals. Here, in our collected dataset, vibration signals are collected every 10 seconds which is a different time and equal time intervals. Hence our data is time-series data.

WHY 'PICKLE' FOR STORAGE INSTEAD OF CSV

Pickle is 80 times faster alternative. It's also 2.5 times lighter and offers functionality every data scientist must know. Storing data in the cloud can cost you a pretty penny. Naturally, you'll want to stay away from the most widely known storage format – CSV – and pick something a little lighter. Pickling isn't limited to datasets only.

Why not use Temperature Dataset

- Data in temperature measurement files are either semicolons or commas or dot-separated. This created problem in combining the files into one CSV file and then converting them to a pickle file. [But later on, the issue was resolved]
- The number of data points in a few bearing datasets was different Fig3
- The data points were not sufficient enough to train the model. It was observed that in Bearing2_2 and Bearing3_2 there are 0 temp files.
- All the graph have shows almost constant temperature change at the end except Bearing1_1

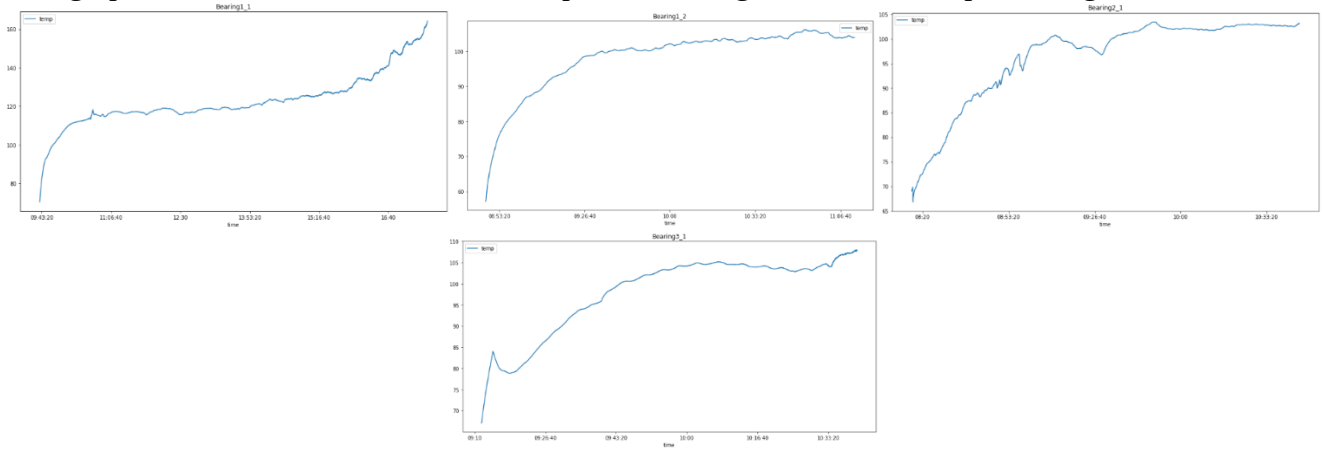


Fig3. Time-Series Temperature Signal Plot of Bearings

Data Pre-Processing:

In each bearing dataset, there are 6 columns ['hour', 'minute', 'second', 'microsecond', 'horiz accel', 'vert accel']
Since our dataset is time-series so time column was created by merging first four column and then the time column was converted to DateTime format. Now there are 3 columns ['time', 'horiz accel', 'vert accel']

5.2. DATA EXPLORATION

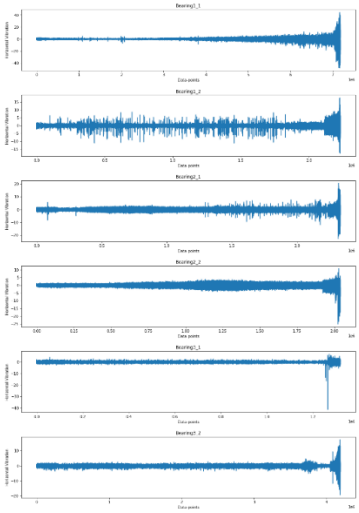


Fig4. Time-Series Horizontal Vibration Plot of Bearings

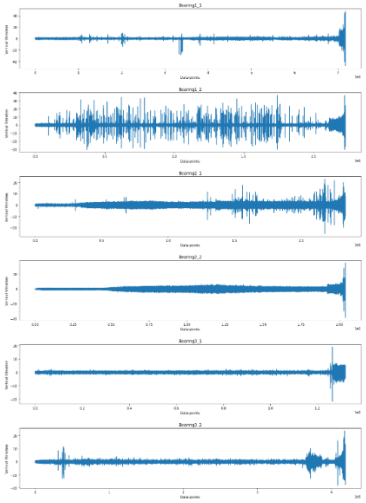


Fig5. Time-Series Horizontal Vibration Plot of Bearing

From the plot in Figure4, Bearing1_1 it is clear, that until the last 1/3 of the full time-series the trendability of the vibration behavior is absent. Similarly in the remaining plots. Given this "non-trendability" in the data, it is impossible to easily observe any trends in the data seeking for possible correlation to the degradation in the bearing. The reason for the loss of trendability is the abnormal conditions of bearing working – large radial force applied on the bearing (4000 N in the first conditions, 4200 N in the 2nd operating conditions & 5000 N in the third operating conditions). Due to such abnormal conditions, the bearings' time lives were only 1 to 7 hours. We can also observe that the measured data is very noisy, it is impossible to use them directly. Therefore, the first task is to perform de-noising. There are 2 different ways to perform de-noising: Trendability dependent de-noising & Trendability independent de-noising.Most of the data-driven RUL predicted methods are oriented for Trendability Statistics. Unfortunately, for the non-trendable and non-periodical statistics other wide-known prognostics models are not applicable.

Getting no good result from the time-series plot of raw data, next the goal is to squeeze the long sequence of large data points to extract some meaningful features out of it. So using this series data we get features such as Max Value, Min. Value, Standard Deviation, RMS Value, Kurtosis Value, Skewness Value, Crest Factor & Form Factor. So these 9 time domain features will get extract out of this (no. of rows in each bearing) datapoints

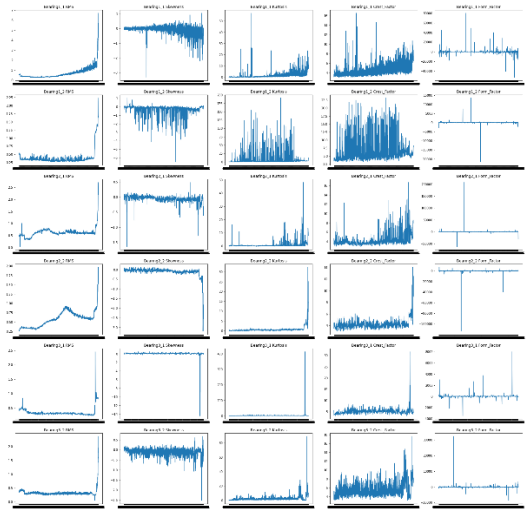


Fig6. Plots of Time Domain Features of Horizontal Acceleration Data

Figure6 Shows the values and trends of 'RMS','Skewness','Kurtosis','Crest_Factor','Form_Factor' for 6 bearing instances under the operating conditions of 1800 rpm & 4000N, 1650 rpm & 4200 N, 1500 rpm & 5000 N (2 each). They were calculated for every time cycle. However, it is apparent that not a single parameter demonstrates consistent patterns across all 6 instances as bearing faults progress. Thus, it is difficult to derive the RUL estimation model based on these statistical parameters.

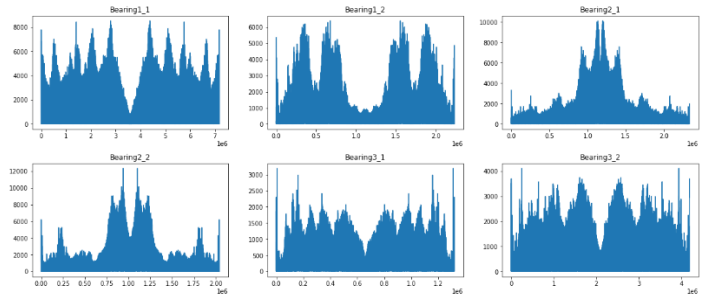


Fig7. Plots of Fast Fourier Transform of Horizontal Acceleration Data

IMPLEMENTATION:

- The entire exploratory analysis is performed in a Collaboratory platform using python programming
- For easy data access & faster data retrieval the CSV files were converted to pickle format
- The Data Files in each dataset are very combined and stored in a pickle format. As a result, we get 6 total datasets for preprocessing. Total number of rows in each data set = no. of files in each dataset * 2560
- Date time columns were combined into time column and converted to datetime format
- Plots of Raw Data
- Time-Domain Features were calculated and graphs were plotted

5.3. FEATURE ENGINEERING & CLEANING

DATA PREPROCESSING:

Data Filtering is the process of choosing a smaller part of your dataset and using that subset for viewing or analysis. Filtering is generally (but not always) temporary (= lasting only a short while, short-term, non permanent, provisional) – the complete dataset is kept, but only part of it is used for the calculation)
Data Ordering or Data Sorting is the process of arranging data into meaningful order so that we can analyze it more effectively

In data exploration part, we concluded that due to the loss of trend in the vibrational data, it was impossible to perform de-noising (smoothing) & consequently – RUL prediction directly for vibration parameters. So, instead of directly using measured vibration values, we proposed to use accumulated values. In current statistics, the vibration values are measured by means of an accelerometer as acceleration, in unit "g".
Physically the power at the moment t (both for platform and balls) is proportional to the acceleration, and current (instantaneous) degradation is proportional to the power.

There are two possibilities to handle measured Horizontal and Vertical Vibration parameters:

- Use an Integral value and perform accumulation for the integral value of the vibration – instead of the measured value of the Horizontal and Vertical Vibrations.
- Use one of the wide-known Machine Learning methods for RUL prediction (*for example, SVR – Support Vector Regression, RVM – Relevance Vector Machines, etc.)* for obtained multi-parameter data – after performing an accumulation separately for the Horizontal and Vertical Vibrations.

Generally speaking, the second approach can provide better accuracy, but it simultaneously leads to over-fitting due to the small number of training bearings in each of the operating conditions groups (two bearings per group)..

Features were extracted from vibration signals of the bearings in the time and frequency domains.
The time domain featured included root mean square (RMS), peak, crest factor, and kurtosis of the vibration signals. The frequency domain features included the magnitude at bearing defect frequencies.
In a preliminary analysis of the vibration signals of the six training bearings, we found that neither the time domain features nor the frequency domain feature exhibited a consistent trend of bearing degradation.
Time Domain analysis and frequency domain analysis; there are two such tools that can provide invaluable signal insight if properly used.A time-domain graph can show how a signal changes with time, whereas a frequency-domain graph will show how much of the signal lies within each given frequency band over a range of frequencies.

METHODOLOGY TRIED ADOPTING FOR FEATURE EXTRACTION

Finding Start Prediction Time (SPT)
In data-driven methods for bearing prognostics, there are several unresolved issues, such as deciding the time to start prediction (TSP), handling random anomalies in the measured data or features extracted from it, and determining the failure threshold. The proposed approach selects the best regression model to approximate the degradation behavior of a bearing based on the evolving trend in its health indicator. The approach has 2 distinct phases, TSP detection, and RUL estimation. The TSP indicates the onset of bearing degradation, and until it is detected the proposed algorithm does not estimate the RUL of the bearing. But due to its complicated algorithm and execution it is not adopted [1].

METHODOLOGY ADOPTED FOR FEATURE EXTRACTION

Data-Driven approaches calculate the Health Indicator (HI) from sensor data like vibration signals & with machine learning methods and deep learning methods. HI differentiates a healthy operation & from a faulty operation of a machine. The features of the signals should be thoroughly investigated before extracting data from them. Most data analysis can be performed in two ways – time domain analysis & frequency-domain analysis. In time-zone analysis, vibration signals can be examined & determined based on time. Frequency domain analysis monitors vibration signals based on the frequency. With time-field analysis, the variation of vibration signals over time is visualized. Features derived from time-domain analysis perform well and are accurate for stationary signals. Although these features respond quickly to random variations within vibration signals. Features of frequency domain analysis can correctly interpret vibration signals because they determine more insight about signals than time-zone analysis features. Characteristics of frequency-domain analysis can assess and separate frequency parts, which helps to understand more insights than time-domain analysis. Mechanical bearing vibration signals are generally variable and at the same time these signals have undetermined faults in high noise environments. Therefore time-frequency domain analysis is an appropriate method to examine mechanical bearing vibration signals because the vibration signals of the mechanical bearing are interconnected. Features derived from time-frequency domain analysis have properties of both time components & frequency components.

Here we extract the feature from Bearings Vibration Data by applying Continuous Wavelet Transform (CWT). Continuous Wavelet Transform is a transformation technique that transforms the signals into 2D Images (time-frequency features). Processing time is less with Wavelet transform for extracting Time-Frequency Features as compared to other methods such as Empirical Mode Decomposition. After converting the signal Data Normalization technique is applied to normalize the coefficients of 2D CWT data. To achieve Data Normalization we performed the Data Re-scaling method a.k.a min-max normalization to re-scale or

adjust all 2D data to a particular range (say 0 to 1). Signal Processing visualizes, evaluates, & controls vibration signals.
Here we extracted 2D features i.e. time-frequency domain features because time-frequency features contains more information about the vibration signals, in regression problem like Remaining Useful Life Prediction, more data means fast & easy analysis.

CWT (Continuous Wavelet Transform):

- The purpose of CWT is to generate a time-frequency depiction of a vibration signal which delivers an accurate positioning of both time & frequency of the signal.
- It is also used for computing the vibration signal varying features.
- The signals are represented as wavelets in Wavelet Transform.
- The CWT transforms the signals into wavelets
- Wavelets are wave-shaped vibrations of magnitude which initiates at zero, progresses & later reduces again to zero
- The wavelets produced from CWT provide well-defined and comprehensible interpretation of time & frequency components which helps in understanding the bearing deterioration process clearly.
- The CWT contains several in-built CWT wavelets like morlet wavelet, gaussian derivative wavelet, frequency b-spline Wavelet, Mexica hat wavelet, Shannon wavelet, complex morlet wavelet, etc. which can be accessed using python programming.
- And among them, we used morlet wavelet python implementation i.e. CWT Morlet PyWavelet as it shows better outcomes for regression problems & compared to other wavelet types & Fourier transform.
- Theoretically, the Morlet wavelet works well for time-frequency vibration signals.
- In the model, the CWT Morlet PyWavelet has been applied to reform the recorded 1D vibration signals of PROGNOSTIA Bearing Learning Dataset into 2D CWT image-like features or time-frequency domain image features.
- It is quite rough to interpret the signals in only one dimension (time), so converted the signals from 1D to 2D.
- 2D signals i.e. 2D CWT image features carry the information about the signals from both time & frequency domains which helps in visualizing the damage process of bearings perfectly.

IMPLEMENTATION

- The entire feature engineering & cleaning is performed in a Collaboratory platform using python programming
- After the data exploration and data merging the 6 pkz files are read where Continuous Wavelet Transform with Morlet Wavelet on 1D vibration signals was performed on 1D vibration signals which returned the 2D CWT feature images
- Normalization was performed on the obtained feature images to a specific range, say 0 to 1
- Then signal processing was applied to extract the time-frequency features which helps in differentiating the healthy operation and the faulty operation of mechanical bearing
- Since or dataset is a run-to-failure dataset, at starting the functionality of bearing is in good condition & eventually fails while progressing.

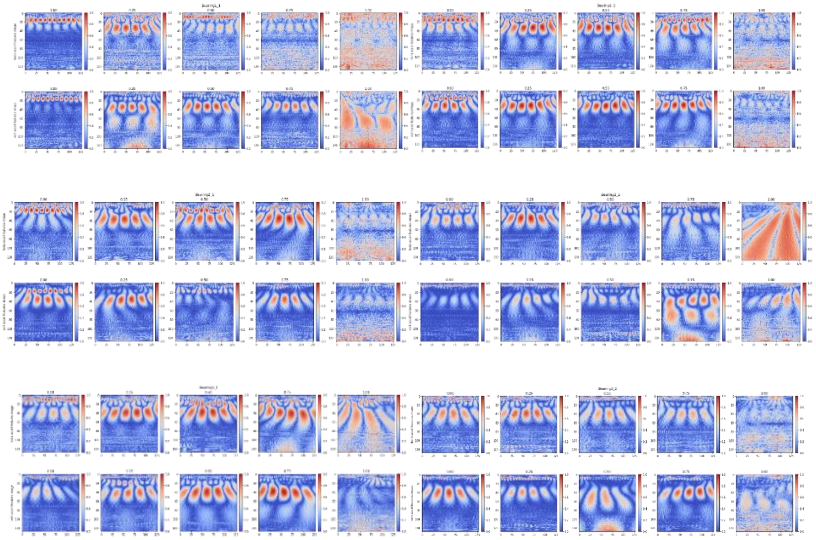


Fig8. 1D to 2D Continuous Wavelet Transform

Miscellaneous Observations in During Feature Extraction and research papers:

- The root mean square (RMS) is the most appropriate indicator of bearing health because it shows the trends of different health statuses of the bearing: in the normal stage, the RMS remains stable; in the initial degradation stage, the RMS starts linear growth; in the severe degradation stage, the RMS starts nonlinear growth, but there are some outliers caused by random noises in the RMS.
- The spurious fluctuation in RMS has a great impact on the RUL prediction performance.
- By reviewing the already done research work, we infer that RUL is a regression problem. And hence RUL prediction mechanical element (mechanical bearing) can be calculated using that mechanical bearing's previous performance data.

5.4. TESTING & COMPARISION

The already processed data is divided into Training and validation data in ration 9:1. First, we preprocess the train data & using that preprocessed train data, draw out important features, & implemented algorithms to train the model to predict. Using validation data, the model validation can be performed. During data preprocessing, we divide the learning dataset into train data + validation data. Validation data is the set of data that is separated from learning dataset to validate or check the trained model.
10% of Learning data is kept for validation & remaining 90% is used for training the model. Validation data is used to validate the model whether the model is working correctly or not on the data which the model is not trained.
One of the major reasons we need validation data is to make sure that our model is not over-fitted to the data in the learning dataset.

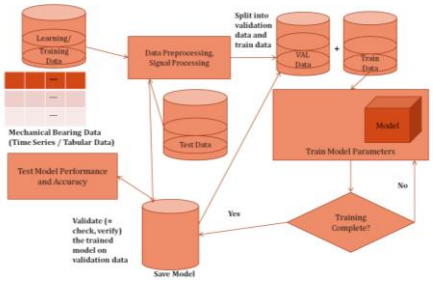


Fig9. Modules Flow Diagram

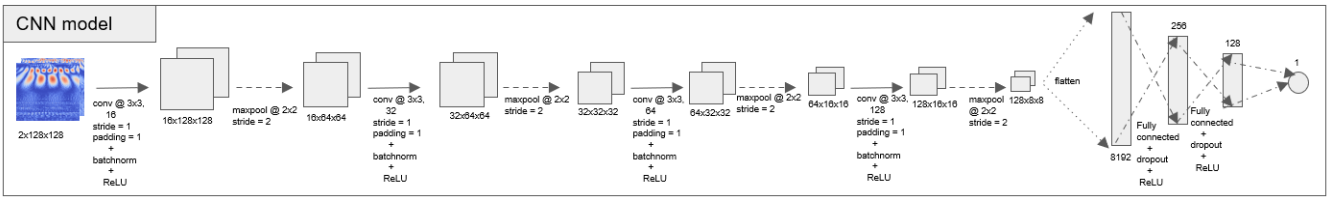


Fig10. CNN Model

CNN + LSTM Model

Input data is still represented as 2D CWT images i.e. time-frequency CWT feature images, but a window of pre-define frames es (say 5) is used as a sequence. First passed through a CNN architecture for encoding and then encoded feature vector sequence is passed through LSTM architecture, hence CNN + LSTM architecture.

- Splitting the learning data to train and validation (90:10 ration). Train data to train the model & validation data to validate the model. After splitting both data were stored in separate pickle files.
- Imported torch Dataset & DataLoader for easy accessing of data. Torch provides many tools to makes the data loading into the model easy. DataLoader helps in easy feeding the data into the model, and also makes the code more readable.
- Initially we calculated the fault probability with the pure CNN model
- To train the model, we switched from CPU device to GPU device in collaboratory in order to reduce the training time. We utilized GPU for computation with CUDA tensor.
- During training, first we calculated the loss function to measure the difference between the actual expected value and the model predicted value. The loss function evaluates how well our algorithm performs on our dataset.
- If our model predictions are totally off i.e. wrong, our loss function will output a higher number. If our predictions are pretty good, our loss function will output a lower number.
- loss function will tell us whether our model predicting correctly or not
- Since RUL Prediction is a regression problem, MSE (Mean Squared Error) Loss is used. MSE Loss is estimated as MSE loss is the average of squared differences between actual expected values and predicted values.
- Loss helps us to understand how much the predicted value differs from the actual value. We then use this loss to train our network model such that it performs better.
- We applied Adam (Adaptive Moment Estimation) optimizer or optimization algorithm for reducing the loss and providing the most accurate results possible. Implemented a learning rate scheduler to improve the optimization process.

- The learning rate controls how much to change the model in response to the estimated error each time the model weights are updated.
- Epochs are the batches of data samples used to train the artificial neural network. Calculated the loss on train data and validation data for 30 random epochs.
- The blue dots represent predicted fault probability values of train data, the red dots represent predicted fault probability values of validation data, and the black line represent expected failure probability values. Expected values are the ideal values.
- In Fig 4 you can observe that the distance between the val loss curve & train loss curve decreases from model 1 to model 3. This is because in model1 is the curve after first training the model and model 3 is curve after training the model 3rd time. Model3 is overfitted as train curve is a straight line.

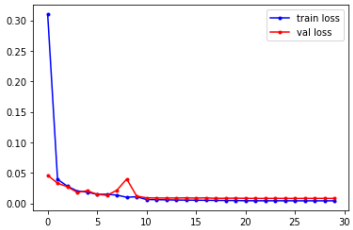


Fig11. CNN Model (Val Loss & Train Loss Curve)

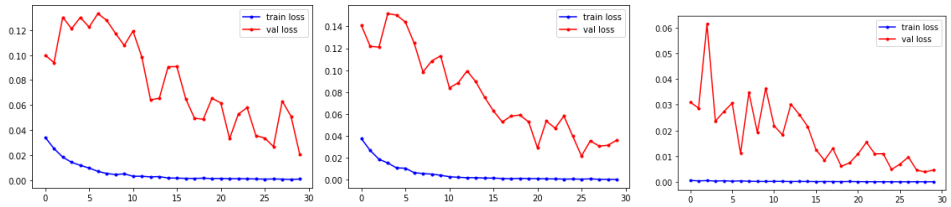


Fig12. CNN + LSTM Model (Val Loss & Train Loss Curve) (Mode11, Model2, Model3)

IMPLEMENTATION:

- The entire exploratory analysis is performed in a Collaboratory platform using python programming
- For easy data acSplitting the learning data to train and validation (90:10 ration). Train data to train the model & validation data to validate the model. After splitting both data were stored in separate pickle files.
- Imported torch Dataset & DataLoader for easy accessing of data. Torch provides many tools to makes the data loading into the model easy. DataLoader helps in easy feeding the data into the model, and also makes the code more readable.
- Initially we calculated the fault probability with the pure CNN model
- To train the model, we switched from CPU device to GPU device in colaboratory in order to reduce the
  training time. We utilized GPU for computation with CUDA tensor.
- During training, first we calculated the loss function to measure the difference between the actual expected value and the model predicted value. The loss function evaluates how well our algorithm performs on our dataset.
- If our model predictions are totally off i.e. wrong, our loss function will output a higher number. If our predictions are pretty good, our loss function will output a lower number.
- Loss function will tell us whether our model predicting correctly or not. Since RUL Prediction is a regression problem, MSE (Mean Squared Error) Loss is used. MSE Loss is estimated as MSE loss

is the average of squared differences between actual expected values and predicted values. Loss helps us to understand how much the predicted value differs from the actual value. We then use this loss to train our network model such that it performs better.

- We applied Adam (Adaptive Moment Estimation) optimizer or optimization algorithm for reducing the loss and to provide the accurate results possible. Implemented learning rate scheduler to improve the optimization process. The learning rate controls how much to change the model in response to the estimated error each time the model weights are updated.
- Epochs are the batches of data samples used to train the artificial neural network. Calculated the loss on train data and validation data for 30 random epochs

MODEL TEST RESULT OF BEARING1_3

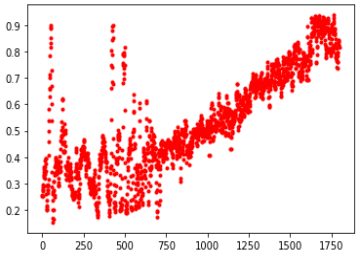


Fig 13. Scatter Plot

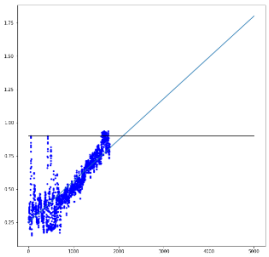


Fig 14. Result 1

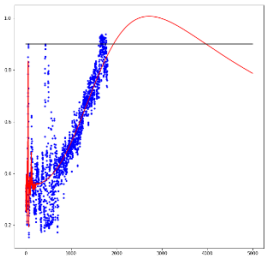


Fig 15. Result 2

MODEL RESULT CALCULATION

Pass the test data till the current time index through the architecture and store the predicted fault probability values

Then fit the predicted fault probability values and corresponding time indices through a regressor to extrapolate and find the time index when the fault probability exceeds a predefined threshold to consider the occurrence of failure. Here we used Linear regression and Guassian Process Regression (GPR) provided by the sklearn python toolbox and set the fault probability threshold to be 0.9

RUL is estimated from the estimated failure occurrence time and current time as follows,
RUL = predicted fault time - current time

CONCLUSION

From the result obtained the accuracy was good for the training and validation dataset but on test dataset, it showed an up-down curve. So this model is not accurate and more changes in parameters needs to do to get a better prediction.

6. FINAL DELIVERABLE

Finally, the final model is locally deployed using Flask and now the user can input sets of files and then press on the predict button and the model will present you with graphs, showing the result. Also, you can download the predicted results list and graphs.

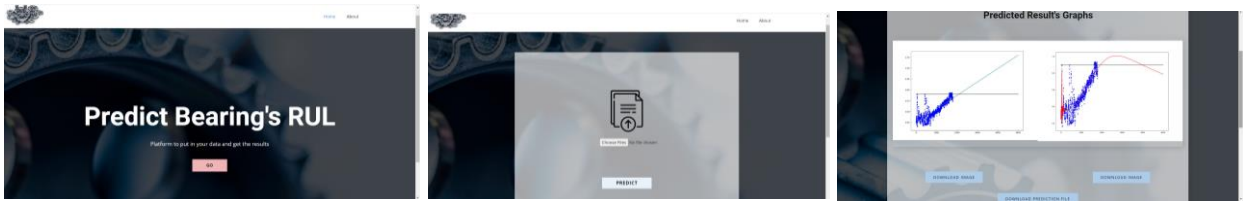
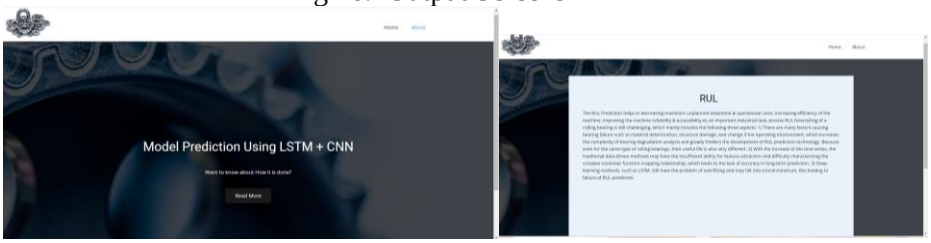


Fig 16. Output Screens



7. INNOVATION IN IMPLEMENTATION

The training model was taken from [7] research paper. And changes in learning parameters like loss function, batches, optimizer, filters; etc. But not enough improvement in prediction accuracy was found. Hence it was concluded that more research needs to be done in CNN + LSTM model and even in feature extraction methods so as to get good prediction accuracy. The frontend design in this project is created using the NamePage app. Later changes were made in Html and CSS files as per required for wrapping it with Flask .py file. The text editor used in the development phase is Sublime and the Git Bash was used to locally deploy the model as well as to type the command lines. Efforts were made to deploy the model globally using different hosting sites such as Heroku, digital ocean, and Azure. But due to memory limit exceeding 1 GB in Heroku, Memory Error while installing torch library (due to less RAM) in digital ocean, and .net file not accessible in case of Azure the model was unsuccessful in getting globally deployed. Further efforts were made to deploy the model by connecting these sites with GitHub but due to uploading a file limit of 100MB in GitHub again, the attempt was unsuccessful. Later GitHub desktop was used for uploading the 100 MB exceeding file but again no positive response was received. In the end, an attempt was made to decrease the size of the file using Large Storage File (LFS) but that was also unsuccessful. The last option left is using the docker container. It is still in progress.

8. SCALABILITY TO SOLVE INDUSTRIAL PROBLEMS

Prediction Latency is the amount of time taken to handle a particular request or the elapsed time necessary to make a prediction. On the local server the latency is around 3 minutes. Because of the complex CNN + LSTM model latency is quite high. Using less complex model can solve the latency.

REFERENCES

1. Babu, G.S.; Zhao, P.; Li, X.L. Deep convolutional neural network based regression approach for estimation of remaining useful life. In Proceedings of the 21st International Conference on Database Systems for Advanced Applications, Dallas, TX, USA, 16–19 April 2016.
2. Zhu, J.; Chen, N.; Peng, W. Estimation of bearing remaining useful life based on multiscale convolutional neural network. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3208–3216.
3. Li, X.; Zhang, W.; Ding, Q. Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. *Reliab. Eng. Syst. Saf.* **2019**, *182*, 208–218.
4. Liang Gao, Long Wen, Xinyu Li, and Yuyan Zhang, "A New Convolutional Neural Network Based Data-Driven Fault Diagnosis Method", *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5990-5998, 2017
5. Lei, Y.; Li, N.; Guo, L.; Li, N.; Yan, T.; Lin, J. Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mech. Syst. Signal Process.* **2018**, *104*, 799–834.
6. Wang, X.; Qiao, D.; Han, K.;Chen, X.; He, Z. Research on Predicting Remain Useful Life of Rolling Bearing Based on Parallel Deep Residual Network. *Appl. Sci.* **2022**, *12*, 4299.
7. Bhumireddy Naga Sai Abhiji, Levaku Tharun Sai Reddy, and S. Sharanya "Remaining Useful Life (RUL) Prediction of Mechanical Bearings Using Convolution Neural Network (CNN) and Long Short Term Memory (LSTM)" vol. 27, Issue 5, 2021 pp. 592- 607