FSM Online Internship Completion Report
on

Remaining Usable Life Estimation (NASA Turbine Dataset)

In

Machine Learning

Submitted by

Arun Kumar D
PSG College of Technology
Coimbatore, Tamil Nadu

Under Mentorship of
Mr. Devesh Tarasia

Smart Manufacturing Simplified

IITD-AIA Foundation for Smart Manufacturing

[1-June-2021 to 31-July-2021]

Remaining Usable Life Estimation using LSTM (NASA Turbine Dataset)

Abstract

Artificial intelligence has become an integral part of human life in the growing years. To be more precise the growth of technologies like machine learning and deep learning have proven inevitable in this era of high-end technologies. A turbofan engine is a type of jet engine responsible for producing thrust with the help of bypass air and jet core efflux. The turbofan engine plays important role in maintaining fuel efficiency. The role of the turbofan engine is highly necessary for the safe status of the aircraft. So, we can conclude that it is essential to avoid any kind of failure of the turbofan engine as the aircraft itself relies on the safe working of the engine. To prevent failure, we can fit the concept of artificial intelligence to be precise with the concept of machine learning. With the ingenious concept of preventive maintenance, we can prevent failure. We have used LSTM (Deep Learning) to predict the failure and deployed the model with the help of flask and implemented it with a help of a local hosting system. This technique will prove useful in avoiding failure in the turbofan engine.

Keywords: Remaining Usable Life Estimation, NASA Turbine Dataset, Preventive Maintenance, LSTM, Kalman filter, flask deployment, Deep Learning, Machine Learning

Table of Content

Introduction

1.1 Machine Learning:

Machine Learning is an important and ingenious concept of artificial intelligence that helps in predicting the outcome based on the current inputs passed on to them. The need for machine learning has risen highly in the growing era of high-end technologies. Machine Learning reduces the work and resources of human beings with its predictions by high-end computations. The algorithms of this software technology are very creative in predicting results only with the help of those variables that are essential for the prediction of the targeted value. Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns, and make decisions with minimal human intervention.

1.2 Deep Learning:

Deep learning is a subset of machine learning that is essentially a three- or more-layered neural net. These neural networks attempt to simulate the behavior of the human brain, albeit far from matching its ability, allowing it to "learn" from large amounts of data. While a single-layer neural network can still make approximations, additional hidden layers can help to optimize and refine for accuracy. Much artificial intelligence (AI) applications and services rely on deep learning to improve automation by performing analytical and physical tasks without the need for human intervention.

1.3 Preventive Maintenance:

Preventive maintenance (or preventative maintenance) is maintenance performed on physical assets on a regular and routine basis to reduce the likelihood of equipment failure and unplanned machine downtime, which can be extremely costly for maintenance teams and facility managers. This kind of approach is highly useful when it comes to any kind of machine maintenance. This type of maintenance helps us understand how the machine falls into failure and the prevention is made even easier and simpler. A preventive maintenance strategy is a commonly used approach that falls between reactive maintenance (or run-to-failure) and predictive maintenance. Preventive maintenance is critical because it establishes the foundation for effective facility management. Preventive maintenance keeps your equipment and assets running efficiently, ensures your employees' safety, and helps you avoid large and costly repairs down the road. Overall, well-functioning preventive maintenance keeps operational interruptions to a minimum.

1.4 NASA turbofan dataset:

The NASA turbofan dataset is a set of data obtained from the C-MAPSS software. C-MAPSS stands for **'Commercial Modular Aero-Propulsion System Simulation'** and it is a tool for the simulation of realistic large commercial turbofan engine data. Data sets consist of multiple multivariate time series data. Each data set is further divided into training and test subsets. The dataset consists of 21 sensor data, 3 operating conditions, the appropriate engine number, and the cycle number. With help of these data, it is easy to predict and fix an appropriate model to predict the failure.

2. Problem Definition

From the given NASA turbofan dataset, we can infer all the sensor and operational conditions dependencies of the turbofan. The main objective here is to set up a machine learning model which predicts the failure of the jet engine before its failure. The objective of the problem is to predict the number of remaining operational cycles before failure in the test set, i.e., the number of operational cycles after the last cycle that the engine will continue to operate. The main important thing is not about fitting a model for this problem but fitting a model that has higher accuracy and precession while predicting the number of remaining operational cycles before failure.

The prediction of the number of remaining operational cycles is not just necessary but also the deployment of the machine learning or deep learning model in the production is the most essential one. The deployment can either be a local or cloud-based hosting system. The main reason to deploy one's model is that the user at any production can understand the work easily. So, therefore, fitting the model and deploying the model in any hosting service is very to be considered the prime objectives of the problem.

3. Existing Solution

The existing solution for this problem of predicting the number of remaining operational cycles was referenced from various technical papers and websites which are listed under references. The solutions include

- Implementing linear regression model for this NASA turbofan dataset. The linear regression training metrics were good but when considering the test metrics, the accuracy was not that good.
- The logical regression model for the dataset was fitted. This existing method had less accuracy as the dataset was not normalized and that led the accuracy metrics to go very low.
- One of the existing solutions that were better suited was the implementation of the ARIMA - Autoregressive integrated moving average this time series model was fitted properly, and the accuracy metric score was good, but the model was not deployed.
- Another fair fitted model was the CNN model, which fit well with the dataset, but the model didn’t yield good results when it was fitted to the other dataset of the NASA turbofan dataset.

From the above existing solutions, the various things that could be inferred such as the box plot to drop the not dependent on values/columns. These solutions were analyzed and observed. One thing that could be brought up from the existing solutions is that no solution had a production deployment of the NASA turbofan dataset. So, deploying this dataset fitted model onto a production hosting service would be an innovative thing to develop.

4. Proposed Development

The proposed idea of the problem statement is as follows:

- First of all, reducing the noise in the dataset is of high priority as the accuracy score metrics go low because of this noisy data. Noisy data has highly prevailed in those datasets which have lots of all rows and columns of the data, to be precise the dataset with more dependent variable parameters is prone to noise. To avoid this both the train and test data are given onto the "Kalman filter". Kalman filter reduces the noise in the data.

- Then we normalize the dataset as each, and every dependent variable is scaled with different scaling metrics. The process the scaling is highly essential because when normalizing all the datasets are scaled to one single metric which leads to an increase in the accuracy metrics. So, we have used a MinMax scaler to normalize the data.

- Now to the important part of the problem, the model selected for the NASA turbine dataset, we have chosen the LSTM model- Long Short-Term memory. First of all, LSTM plays well when it comes to time series datasets. The accuracy of the model increases after many executions. The increase in the accuracy of this model is mainly behind the reason that the model easily learns and unlearns information based on the appropriate instances which leads to the better fitting of the model.

- To finally conclude the model is to be locally hosted i.e., locally deployed with the help flask- a python framework highly helpful when it comes to deployment of the model onto production. The deployment is locally executed in such a way that when the user inputs the necessary data variables in each column which are back routed as forms which then predicts the Remaining Usable Life (RUL) of the NASA turbofan dataset.

The said proposed system mentioned above is decided after confiding in with various references papers, websites, and GitHub repositories containing similar codes for the problem statement and also with similar problems. This method of technique acquired has high scalability of implementation on the production sites. The proposed system is designed in such a way that it accommodates high accuracy results by filtering noise, normalization of the dataset, and also fitting the model that easily learns and unlearns based on the information at the particular instances. The proposed idea could be separated into phases as follows:

- Exploratory data analysis
  - Box plot – dependencies of the data
- Data pre-processing
  - Kalman filter – reducing the noise
  - Normalization - scaling
- Training the LSTM model
  - Training and training metrics of the LSTM model
- Testing the LSTM model
  - Testing and testing metrics of the LSTM model
- Deployment of the model
  - Deployment of the model locally with flask
  - Setting up the local hosting environment with the HTML and CSS

5. Functional Implementation

5.1 Exploratory data analysis:

Exploratory data analysis is the process of understanding the dataset. This stage is very essential for any kind of model implementation. The need for this stage stems from the programmer's ability to fit the model. Understanding the dataset leads to a fair fitting of the model. So, while considering the dataset there aren't any missing data i.e., no null values present in the data. The dataset set consists of engine numbers as mentioned below:

- FD001- 100 engine numbers
- FD002- 260 engine numbers
- FD003 – 100 engine numbers
- FD004 – 255 engine numbers

Some of the sensors were observed to be remaining insignificant for finding the remaining usable life of the NASA turbofan engine. These sensors can be not considered in the training and testing of the said model. The relation of the sensors towards the remaining usable life is observed and analyzed using

- Box plot
- Correlation – Heat Map

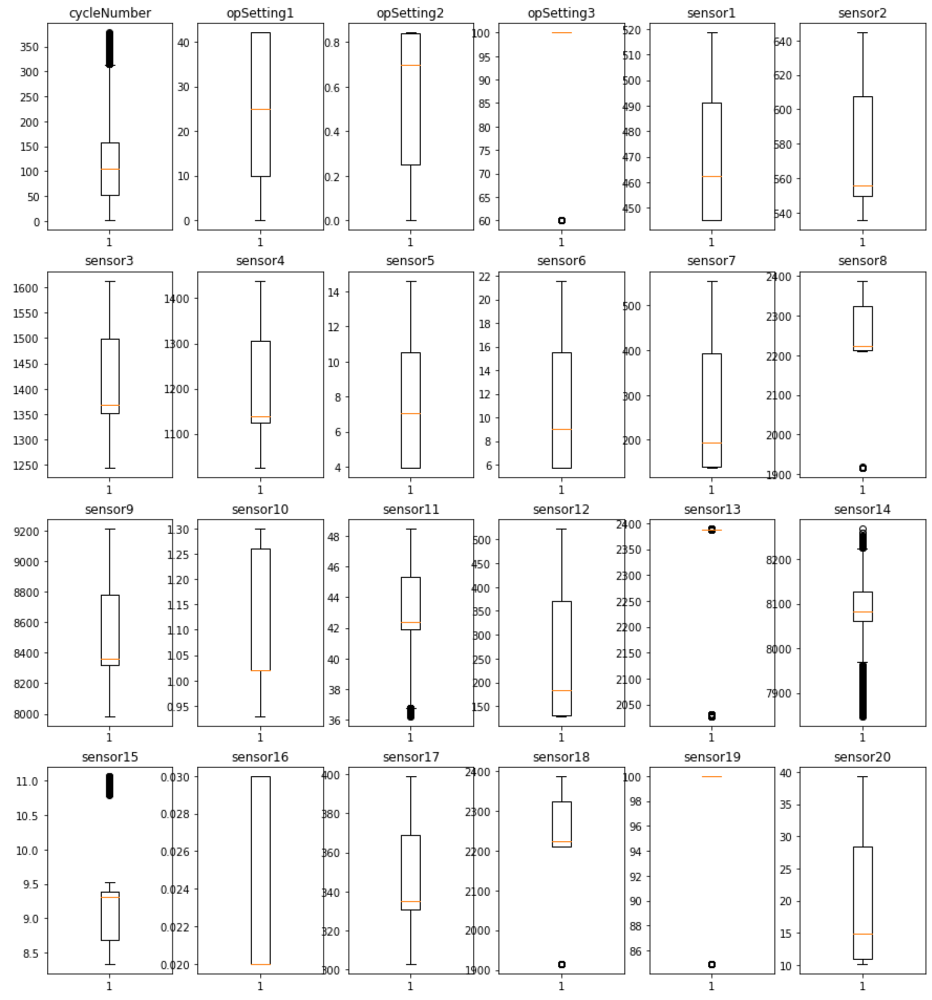


Figure 1: box plot of FD002 dataset

From the results obtained any one of them can be implemented to test the relation of the dependencies with the target variables. The box plot of the FD002 dataset is mentioned for further understanding. from the box plot, we can observe that the plots with constant represents the variables that have a poor dependency on the respective variables in the dataset. The variables are 'Opsetting3','sensor8', 'sensor13','sensor15','sensor19','sensor16' & 'sensor18'.

5.2 Data pre-processing:

Data pre-processing is one of the important pre-model deployment steps. The concept behind the data-preprocessing is to prepare the dataset for model training and testing. In data pre-processing, we first drop all the data variables that are least correlated with the target variables. In our proposed system one of the highlighted portions of innovation is the implementation of the "Kalman filter". The Kalman Filter is an efficient optimal estimator (a set of mathematical equations) that provides a recursive computational methodology for estimating the state of a discrete data-controlled process from measurements that are typically noisy while providing an estimate of the uncertainty of the estimates. Kalman filters are used to optimally estimate the variables of interest when they can't be measured directly, but an indirect measurement is available. They are also used to find the best estimate of states by combining measurements from various sensors in the presence of noise.

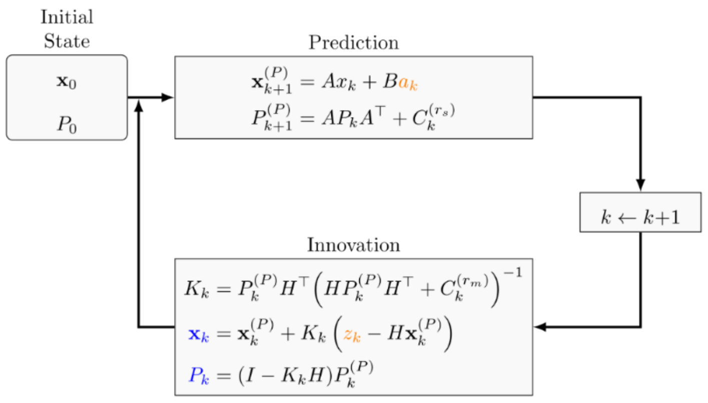

Initial State
$\mathbf{x}_0$
P_0
Prediction
$\mathbf{x}_{k+1}^{(P)} = Ax_k + Ba_k$
$P_{k+1}^{(P)} = AP_kA^\top + C_k^{(r_s)}$
$k \leftarrow k+1$
Innovation
$K_k = P_k^{(P)}H^\top\left(HP_k^{(P)}H^\top + C_k^{(r_m)}\right)^{-1}$
$\mathbf{x}_k = \mathbf{x}_k^{(P)} + K_k\left(z_k - H\mathbf{x}_k^{(P)}\right)$
$P_k = (I - K_kH)P_k^{(P)}$

Figure 2: working of Kalman filter

The Kalman filter reduces the noise in the data as the presence of noise is highly prone in datasets with a large number of dependent variables. Next, we move towards the normalization process of the data pre-processing.
Normalization is the process that normalizes various unscaled data into single scaled data. Unscaled data contribute to low accurate predictions. So, therefore providing a scaling to one's dataset gives a high probability of getting accurate and precise results. Here, we have used the MinMax scaler to scale the dataset. We can understand from the name that the scaling is carried out with the manipulation of the minimum and maximum values. The working expression of the MinMax is provided below to understand the further provisions of the data pre-processing.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Figure 3: Expression of MinMax scaler

Another unique data-preprocessing step for this unique dataset is finding the RUL- Remaining Usable Life for the training dataset. The importance of this step is highly essential because the target variable in our case is the RUL so, to get the predicted RUL- Remaining Usable Life values. The expression to find the RUL is as follows:

RUL = EOL – Current Cycle

RUL = Remaining Usable Life

EOL = End of Life in cycles

From the above expression, the RUL can be found to help train the model. Therefore, with this our data pre-processing terminates and we can proceed onto model training, testing, and later deployment.

5.3 Model selection:

The dataset provided to us is a multivariate time series model. The best model to fit in this pretense is the deployment of the LSTM model. LSTM stands for Long Short-Term Memory. Long Short-Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It can handle the vanishing gradient problem faced by RNN. A recurrent neural network also known as RNN is used for persistent memory. An LSTM module has a cell state and three gates which provides them with the power to selectively learn, unlearn or retain information from each of the units. The cell state in LSTM helps the information to flow through the units without being altered by allowing only a few linear interactions. Each unit has an input, output and a forget gate which can add or remove the information to the cell state. The forget gate decides which information from the previous cell state should be forgotten for which it uses a sigmoid function. The input gate controls the information flow to the current cell state using a point-wise multiplication operation of 'sigmoid' and 'tanh' respectively.

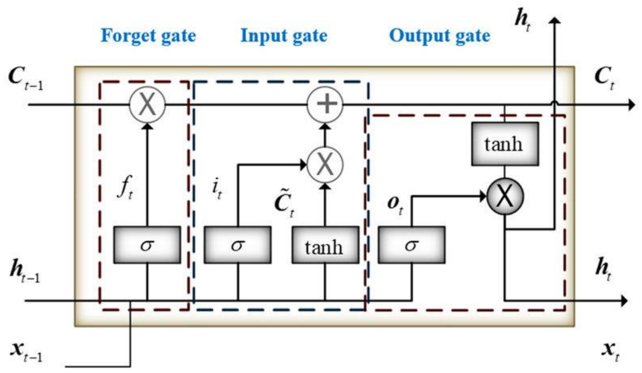


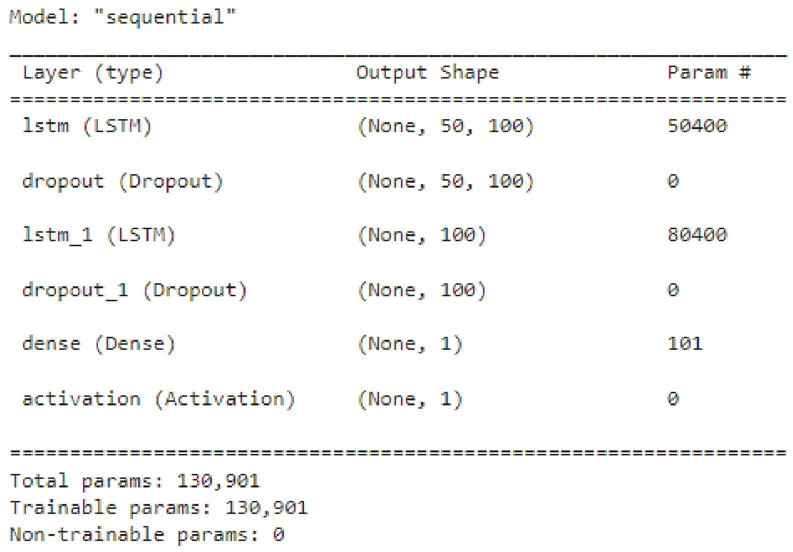
Figure 4: Structure Of LSTM

Finally, the output gate decides which information should be passed on to the next hidden state.

5.3.1 Model Training:

The model was trained with the help of the dependent variables and the targeted variables – Remaining Usable Life. So, initially, the main step to be executed for any model training is the setting up of the model network. The 'input shape' and 'return sequences' of the model are

first inputted into the network. Various model additional features of the model are to be added to the model network using “model.add ()”. Next, we set up the compile instructions of the model using ‘model.compile ()’.
The compilation of the model is instructed to deliver the losses in the “MSE- Mean Squared Error”. The optimizer of the algorithm is also given as “rmsprop”. The RMSprop optimizer is like the gradient descent algorithm with momentum. The RMSprop optimizer restricts the



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 50, 100)           50400

 dropout (Dropout)           (None, 50, 100)           0

 lstm_1 (LSTM)               (None, 100)               80400

 dropout_1 (Dropout)         (None, 100)               0

 dense (Dense)               (None, 1)                 101

 activation (Activation)     (None, 1)                 0

=================================================================
Total params: 130,901
Trainable params: 130,901
Non-trainable params: 0
```

Figure 5: LSTM model summary of FD002 dataset

oscillations in the vertical direction. Therefore, we can increase our learning rate and our algorithm could take larger steps in the horizontal direction converging faster. To give a clear understanding of the model network, the model summary is being attached below
So, now that the model is all set to train, we can execute the training of the model. Once, the model has been trained, we can now calculate the training metrics of the model training process. By the usage of ‘the model. evaluate ()’ we can now calculate the training metric score of the model. The model score for the various dataset was fair, which led to finalizing this as the fair fit for the model.

5.3.2 Model Testing:

Now, we proceed with the model testing of the deep learning deployment. This is the most essential stage of any model deployment. This stage lets the programmer understand whether the model one has fitted will work properly or not for the given instances. Using the function model. predict ()' which provides an array of values for our test data. We can then plot the ‘True RUL’ and ‘Predicted RUL’. The plot will help the user to analyze the fitting of the model. From the plots shown below, we can understand the fitting of the model. By using the library ‘matplotlib’ we can plot the said required graphs. We can infer from the below-attached graphs that the deep learning model of LSTM fits perfectly well for our required problem statement. The model accuracy kept increasing after further execution. so, we can understand that this model is perfectly fitted. After each execution, the model predictions of the remaining usable life of the NAS turbofan dataset kept getting more accurate and precise.

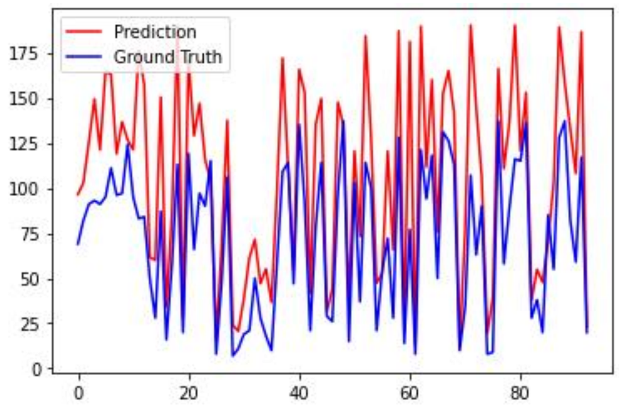


Figure 9: FD001 True RUL vs Predicted value

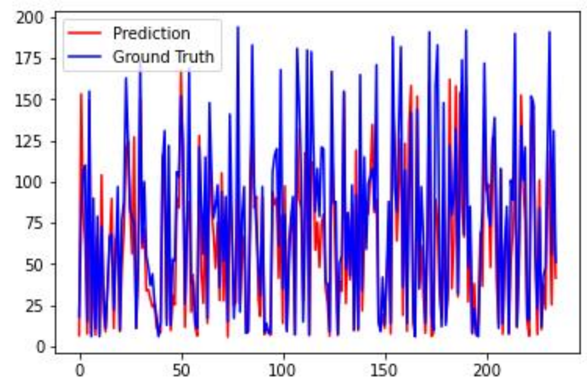


Figure 8: FD002 True RUL vs Predicted value

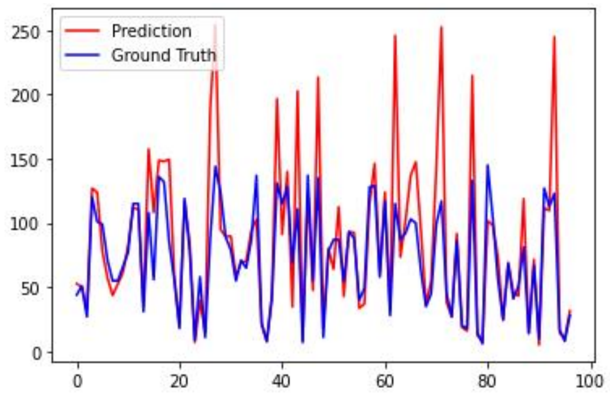


Figure 7: FD003 True RUL vs Predicted value

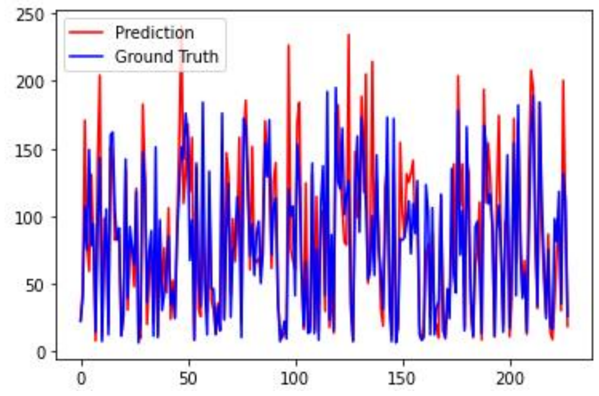


Figure 6: FD004 True RUL vs Predicted value

5.3.3 Model Deployment:

Model deployment is the most important process of any kind of machine learning or deep learning model for a particular problem or application. Deploying the model unto a production site either through local hosting or cloud-based applicational setups. This deployment process explores the scalability of any innovative machine learning or deep learning solution. For this explained said proposed system, we have deployed our model in local hosting applicational deployment. To deploy our model locally we have opted to choose flask as the framework for the website deployment. Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is a Python web framework. Flask's large ecosystem of extensions makes it easier for developers to build common web app features such as authentication, relational database access, and APIs even though support is not built into the core Flask library.

So, first and foremost when it comes to model deployment using flask, the said decided model requirement i.e., the libraries required for executing the model and also the libraries that are required for getting the data for the model execution and returning the predicted RUL-Remaining usable life of the NASA turbofan engine. The local hosting production site gets the values for the dependent variables in the form of 'forms' and get is back routes these data to the back-end kept deep learning algorithm which is kept in '.h5' format. The back end is built in a certain structural way that the data given as input is got in the same format as how the model was trained and tested in the actual coding of the LSTM model. The model receives the data and predicts the final targeted value. The accuracy of the targeted value keeps increasing as the model is executed multiple times so that it learns from the past inputs. The value is then promptly delivered to the front end of the local production site. The local production site is held by the structural coding of "HTML" and "CSS". The HTML repository is responsible for holding up the local production site. The styling of the site is all implemented with the help of CSS coding. The CSS coding is purely responsible for the suitable fonts and designs on the site. The value predicted is then available on the local site. This helps the production site workers to prevent the turbofan engine from going to failure.

6. Final deliverable

This problem of finding the Remaining Usable Life of the NASA turbofan dataset to prevent the engine from going to failure is a major poignant issue when it comes safe operation of the aircraft. The proposed solution is highly effective in dealing with this issue. From the data analysis to the model deployment every part of the deep learning implementation is carried out with such a provision to avoid any kind of prediction inaccuracy. The process of using the "Kalman filter" from the 'pykalman library' seems to be a better implementation to avoid any kind of noise in the data. The provided dataset is vast. The dependencies of the dataset are huge. Thus, the dataset is easily prone to huge noise in the data. The normalization of the data using the MinMax scaler is better suited in this instance to scale the data onto a single scaling unit. Thus, increasing the accuracy of the prediction of the deep learning LSTM model. The LSTM is better suited for this dataset. The model fitted perfectly after the training. Even, after the testing, the accuracy of the model was analyzed using test metrics and the visualization of the predicted values against the original value. Hence, contributing to a better fit of a model to prevent engine failure. The model accuracy kept increasing after multiple executions. The model is a better learning model. And finally deploying the model onto flask a python framework specially designed for model deployment. The flask front end and back end is developed with simple programming concept contributing to the easy understanding of any novice user. To summarize the deep learning method proposed system is highly efficient for this cause of preventing turbojet engine failure.

7. Innovation

Innovation is the key ingredient in the proposed system for an existing problem. This proposed system was formulated keenly by learning all the existing solutions. From the existing models, the following things are to be formulated as the innovative keystone of the proposed system.

- The data filtering process to avoid noisy data by using the Kalman filter is a key idea that contributes to the high accuracy model. The data set is huge, so therefore the presence of noise in the data is evident. Using the filter imported from the "pykalman" library is the better fit for removing the noise.

- The fitting of the LSTM model to this pre-processed noiseless data is an even more bonus in boosting the accuracy of the model. The LSTM model is a precise model that learns and unlearns certain things in the information based on the provided instance to increase efficiency. Though, the ARIMA- Autoregressive integrated moving average model is a better fit when it comes to time predicting series, LSTM plays well with huge testing and training data than the latter.

- The deployment of the model onto flask contributing to a local hosting service is one of the key features of the system. The local hosting system is set up in the form of the forms contributing to the easy implementation of any novice production site user.

8. Scalability to Solve Industrial Problems

The scalability of this proposed system in solving the problems in the aircraft sector is high. As the model works well with the training and testing dataset of the NASA turbofan jet engine dataset. The well-executed LSTM model is deployed onto a local hosting production site and the results of the predictions are fair suited. The model can be deployed onto a cloud-based computing service which can elevate the stance of the deployment in huge. The cloud-based deployment will help us to understand and learn which parameters majorly contribute to the failure after a period of execution. The above facts of the proposed system will help us confirm the better scalability of the deep learning model in solving industrial problems. Not only the cloud but also the training of the model with the refined noiseless dataset is highly helpful. The model itself has a great scope in predicting more accurate results by nature.

9. Conclusion

The turbofan jet engine is an inevitable part of the safe execution of any kind of aircraft. The safe working of the aircraft lies on the shoulder of this jet engine. Failure to this turbofan leads to the destruction of the machinery. Artificial intelligence is a highly needed sector of technology in the future era. The scalability of Artificial intelligence in solving this aircraft engine to not fail will be an easy task. This method largely contributes to the growing field of preventive maintenance. To avoid this without any manual labor the concept of deep learning is a better fit for solving the issue. The deep learning algorithm LSTM – long short term memory is a highly effective deep learning model for this problem. This Keras-supported model is highly efficient in the large dataset and this seems to be an exact fit for the model in the prevention of failure of the jet engine. The model gives a better result in its predictions and the deployment is also executed in such a way that any kind of production user can see through it and easily analyze the model deployment. The scalability of the model is high as this model can be deployed into a cloud-based system increasing the possibility of this issue of preventive maintenance even simpler. To conclude this problem is a highly essential one for this growing age of development. This concept could not only be implemented in turbofan jet engines but also in any kind of similar problems and applications.

10. references:

1. NASA, "Prognostics Center of Excellence Data Repository", http://ti.arc.nasa.gov/projects/data_prognostics, last accessed on January, 2007.

2. H. Madsen, G. Kariniotakis, H. A. Nielsen, T. S. Nielsen, and P. Pinson, "A Protocol for Standardizing the Performance Evaluation of Short-Term Wind Power Prediction Models,"
3. Technical University of Denmark, IMM, Lyngby, Denmark, Deliverable ENK5-CT-2002-00665, 2004.

4. "NN5: Forecasting Competition for Artificial Neural Networks & Computational Intelligence", http://www.neural-forecasting- competition.com/index.htm, last accessed on July 2008, 2008.

5. "Time-Series Prediction Competition at 2nd European Symposium on Time Series Prediction (ESTSP'08)", http://www.estsp.org/index.php?pg=welcome, last accessed on July 2008, 2008.

6. Wei, J.; Bai, P.; Qin, D.T.; Lim, T.C.; Yang, P.W.; Zhang, H. Study on vibration characteristics of fan shaft of geared turbofan engine with sudden imbalance caused by blade off. J. Vib. Acoust. 2018, 140, 1–14. [CrossRef]

7. Gers FA, Schmidhuber JA, Cummins FA. Learning to forget: continual prediction with lstm. Neural Comput 2000;12(10):2451–71. https://doi.org/10.1162/089976600300015015

8. Peng C, Chen Y, Chen Q, Tang Z, Li L, Gui W.' A Remaining Useful Life Prognosis of Turbofan Engine Using Temporal and Spatial Feature Fusion. Sensors (Basel)'. 2021 Jan 8;21(2):418. doi: 10.3390/s21020418. PMID: 33435633; PMCID: PMC7827555.