A Time Window Neural Network Based Framework for Remaining Useful Life Estimation

Pin Lim, Chi Keong Goh
Rolls Royce Singapore Pte Ltd
Advance Technology Center
6 Seletar Aerospace Rise, Singapore 797575
Email: Pin.Lim@Rolls-Royce.com,
ChiKeong.Goh@Rolls-Royce.com

Kay Chen Tan
Department of Electrical and Computer Engineering
National University of Singapore, Singapore 119077
Email: eletankc@nus.edu.sg

***Abstract*—This paper develops a framework for determining the Remaining Useful Life (RUL) of aero-engines. The framework includes the following modular components: creating a moving time window, a suitable feature extraction method and a multi-layer neural network as the main machine learning algorithm. The proposed framework is evaluated on the publicly available C-MAPSS dataset. The prognostic accuracy of the proposed algorithm is also compared against other state-of-the-art methods available in the literature and it has been shown that the proposed framework has the best overall performance.**

***Index Terms*—Neural Networks (NN), Moving Time Window, Feature Extraction, RUL Estimation, C-MAPSS, Prognostics.**

I. Introduction

Traditionally, maintenance of mechanical systems have been carried out based on regular schedules which is costly and introduces the possibility of human error [1], [2]. This has led to the increasing popularity of Condition Based Maintenance (CBM), which allows for maintenance based on the current health of the system, thus cutting costs and overall downtime of the system. Resulting in the increased focus on prognostics as it is one of the key enablers of CBM. Prognostics is the study of how systems degrade and the estimation of Remaining Useful Life (RUL) of the system [3].

Despite the increased research focus on prognostic in both industry and academia, prognostics still remains as a relatively challenging problem. One such problem is that historically, prognostic algorithms tend to be tailored exclusively for a specific problem. This leads to difficulty in comparing different algorithms and replicating the results for different applications. Recent efforts to provide a common ground for researchers to collaborate include the release of public domain datasets as well as special programs such as the Joint Strike Fighter (JSF) program. These platforms allow researchers to develop and compare prognostic algorithms for a common problem. A detailed summary of other commonly faced problems in the field of prognostics can be found in [4]

This paper aims to investigate the problem of estimating RUL for aero-engines. A suitable dataset for use is the publicly available NASA C-MAPSS Dataset [5]. Certain learning methods such as Neural Netowrks (NN), Decision Trees, Support Vector Machines, etc. take instantaneous datapoints as inputs. Such implementation therefore assumes each datapoint to be independent of each other. This paper proposes the investigation on the effectiveness of a moving time window and Feature Extraction (FE) methods to account for time dependency between datapoints. The entire framework includes the following modular components: creating a moving time window, a suitable FE method and a multi-layer NN as the main machine learning algorithm.

TABLE I: Dataset details (Simulated from C-MAPSS)

	C-MAPSS			
Dataset	**FD001**	**FD002**	**FD003**	**FD004**
Train Trajectories	100	260	100	248
Test Trajectories	100	259	100	248
Operating Conditions	1	6	1	6
Fault Modes	1	1	2	2

The rest of the paper is organized as follows. Section II introduces the dataset used in this paper as well as the data preparation carried out prior to any experiments. Section III briefly describes the basic learning network which will be used in this paper. Section IV then elaborates on the various FE techniques considered for the proposed framework. Section V investigates the moving time window method used to create high-dimensional data containing discriminating information. Section VI finally combines all the modular components proposed in the earlier sections and illustrate the performance of the proposed framework.

II. Dataset

In this paper the NASA C-MAPSS Dataset [5] was used to validate the effectiveness of the proposed method. This dataset contains simulated data produced using a model based simulation program (named Commercial Modular Aero-Propulsion System Simulation, C-MAPSS) developed by NASA. The C-MAPSS dataset is further divided into 4 subsets as shown in Table I, details on the conditions and model used for simulating these dataset are described in [6].

The data is arranged in an n-by-26 matrix where n corresponds to the number of data points in each dataset. Each

978-1-5090-0620-5/16/$31.00

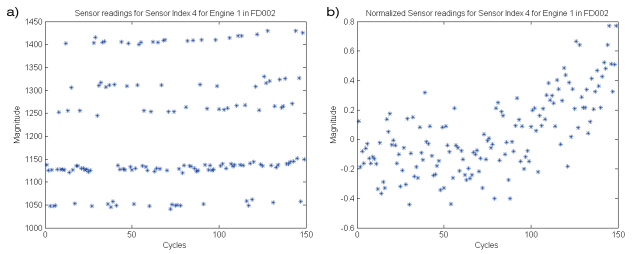

Fig. 1: Sensor values before (a) and after (b) normalization

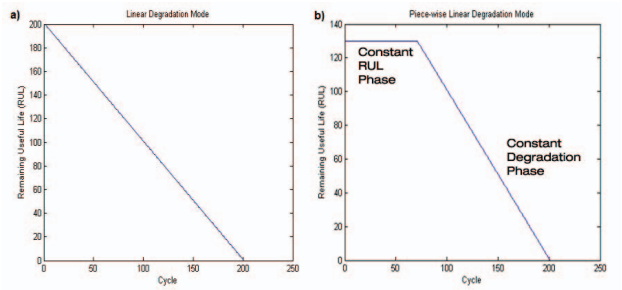

Fig. 2: (a)Linear degradation model and (b) Piece-wise linear model

row is a snapshot of data taken during a single operational cycle and each column represents a different variable. The first two variables represent the engine number and cycle number respectively. The following three variables are operational settings which corresponds to the operating conditions in Table I and have a substantial effect on engine performance. The remaining variables represent the 21 sensor readings that model the engine degradation through time.

Each trajectory within the train and test trajectories is assumed to represent the life-cycle of an engine. Each engine is simulated with different initial health conditions which are considered to be healthy (no faults). For each trajectory within the training sets, the last data entry corresponds to the moment the engine is declared unhealthy (functional failure). On the other hand the trajectories within the test sets terminate at some time prior to failure and the aim is to predict the Remaining Useful Life (RUL) of each engine in the test set. The actual RUL value of each test trajectories were also included in the dataset for verification purposes.

A. Data Preparation

1) Data Normalization: Based on the operational setting values, the data points can be separated into 6 distinct operating conditions. Each of these operating conditions results in disparate sensor values (Fig. 1). Therefore prior to any testing and training, the data points are normalized to be within the range of [-1,1] using Eq. (1). As normalization was carried out within each sensor and each operating condition, this will ensure equal contribution from all features across all operating conditions.

$$Norm(x^{(c,f)}) = 2\frac{(x - x_{min}^{(c,f)})}{x_{max}^{(c,f)} - x_{min}^{(c,f)}} - 1, \forall c, f \qquad (1)$$

where c represents the operating conditions and f represents each of the original 21 sensors.

2) Feature Selection: By plotting the sensor values against time for all sensors, it was observed that several of the sensor measurements showed no significant variations as the engine degrades. In addition, some sensor readings also seems to be heavily corrupted by noise and did not show any consistent variation with time. Feature selection was therefore carried out to filter away any redundant sensor readings for each data-set.

The feature selection method implemented in this paper is identical to that described in [7]. The method makes use of a NN, in this case a Multi-Layer Perceptron (MLP), for group feature selection. The details of this method is clearly illustrated in [7] and will not be repeated here. The feature selection method was implemented across all data-sets with normalized data values and the chosen sensor indexes were consistent across all data-sets. The final chosen set of 14 sensor indexes (out of a total of 21) are {2,3,4,7,8,9,11,12,13,14,15,17,20,21}. Comparing with other literature, the set of sensor indexes chosen by [8] is a subset of those chosen by this method. However [8] chose these sensors simply based on visual inspection. This method thus provides a rigorous method in obtaining discriminating features within a dataset.

B. Degradation model

Unlike usual regression problems, an inherent challenge for data driven prognostic problems is determining the desired output values for each input data point. This is because in real world applications, it is impossible to accurately determine the health of the system at each time step without an accurate physics based model. A sensible solution would be to simply assign the desired output as the actual time left to functional failure [9], [10] (Fig. 2a).

However, recent publications proposed an alternative approach of deriving the desired output values based on a suitable degradation model [11]–[13]. For this data-set, a piece-wise linear degradation model which limits the maximum value of the RUL function (Fig. 2b) has shown to be effective. The maximum value was chosen based on the observations of the data and its numerical value is different for each data-set. In the following sections, the desired RUL values is derived based on the piece-wise linear degradation model. However, it should be noted that in cases where knowledge of a suitable degradation models is unavailable, the linear model is the most natural choice to use.

C. Performance Evaluation

1) Scoring Function: The original authors of the dataset [6] proposed the use of a scoring function to rank the accuracy of the algorithms. This scoring function is illustrated in Eq. (2) below.

$$s = \sum_{i=1}^{N} s_i, s_i = \begin{cases} e^{-\frac{d_i}{13}} - 1 & \text{for } d_i < 0 \\ e^{\frac{d_i}{10}} - 1 & \text{for } d_i \geq 0 \end{cases} \qquad (2)$$

Where s is the computed score, N is the number of engines in each data-set, and $d_i = \widehat{RUL}_i - RUL_i$ (Estimated RUL-True RUL). The characteristic of this scoring function is that it favors early predictions more than late predictions; this is in line with the risk adverse attitude in aerospace industries.

2) Root Mean Square Error: In addition to the scoring function, the Root Mean Square Error (RMSE) of the estimated RULs is also used as a performance measure to establish a common comparison with current literature.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2} \tag{3}$$

III. Multi-layer Neural Net as Base Regressor

A. Method Description

In this paper a basic MLP is used as a regressor to estimate the RUL of each individual engines. The MLPs were implemented in MATLAB environment using the NN Toolbox. The network structure used was kept consistent for all data-sets with a hidden layer of 20 neurons. The hidden layer size was chosen based on the structure which yielded the lowest validation error. The inputs to the neural network are the selected sensor values at each time instant and the output of the neural network would be the desired RUL value or the predicted RUL value during the training and testing phase respectively.

B. Results

The results shown in this section are obtained by training the network on each training set (FD001 to FD004) and evaluating on the respective testing set. The performance of the basic neural network is shown in Fig. 7. The RMSE of data-sets FD002 and FD004 are relatively higher due to the increased difficulty of these data-sets. This could be due to two reasons. First, the increased number of operating conditions and fault modes might have complicated the problem and secondly the larger number of engines would lead to a greater cumulative error.

The performance values of the basic NN will serve as a benchmark to evaluate the effectiveness of the proposed method.

IV. Feature Extraction

FE methods are known to improve the accuracy of the method by extracting discriminating information from the data and reducing the input dimension of the data at the same time. In this paper, a total of 3 FE methods were experimented to determine the effectiveness of each FE method on this dataset.

A. Principal Component Analysis (PCA)

PCA is a commonly used method of FE in numerous machine learning and pattern recognition problems [14]–[16]. The principle of PCA is to project the original data onto m number of basis vectors. These vectors are chosen based on orthogonal directions which give the highest variance to result in uncorrelated features. A detailed explanation of PCA and MATLAB codes for implementation are published in [16].

B. Kernel PCA

A caveat of traditional PCA is that it assumes linear separability of the data. In order to tackle data which are non-linearly separable, an alternative FE method known as Kernal PCA (KPCA) is proposed. KPCA works by mapping the data into a higher dimension using a non-linear function (Φ), PCA is then done to determine the eigenvectors and eigenvalues for the covariance matrix C Eq. (4, 5)

$$C = \frac{1}{n}\sum_{j=1}^{n} \Phi(x_i)\Phi(x_j)^\top \tag{4}$$

$$\lambda \mathbf{V} = C\mathbf{V} \tag{5}$$

However in actual applications, Φ does not need to be explicitly calculated. Instead a kernel matrix K of dimension $n \times n$ is determined based on Eq. (6) and the eigenvalue decomposition of this matrix is then computed using Eq. (7) and used for FE. Full details of the derivation of KPCA can be found in [17].

$$K_{ij} = \Phi(x_i) \cdot \Phi(x_j) \tag{6}$$

$$n\lambda \boldsymbol{\alpha} = K\boldsymbol{\alpha} \tag{7}$$

Due its ability to handle non-linearly separable data, KPCA has found numerous applications such as facial recognition [18], robust speech FE [19] and novelty detection [20].

1) Nyström Approximation: A significant drawback of KPCA is the computational load required to store the kernel matrix K, the problem is increasingly prevalent as n increases for large datasets. Numerous literature [21]–[25] have proposed methods to circumvent this problem by approximating the kernel matrix K.

The original Nyström approximation for integral equations was extended to approximate the kernel matrix by [22]. It is shown in both [22], [25] that the original kernel matrix can be approximated by a subset of landmark points. Following the notation in [25], for a sample set $X = \{x_i\}_{i=1}^{n}$ and corresponding $n \times n$ kernel matrix K. The Nyström approximation states that a subset of landmark points $Z = \{z_i\}_{i=1}^{m}$ will approximate the eigensystem of the full kernel matrix $K\Phi_k = \Phi_k\Lambda_k$. Where the eigenvectors and eigenvalues can be obtained using

$$\Phi_k \simeq \sqrt{\frac{m}{n}} E\Phi_z\Lambda_z^{-1}, \Lambda_K \simeq \frac{n}{m}\Lambda_z \tag{8}$$

where $E_{ij} = k(x_i, z_j), E \in \mathbb{R}^{n\times m}$ and Φ_z and Λ_z are respectively the eigenvectors and eigenvalues of W, where $W_{ij} = k(z_i, z_j), W \in \mathbb{R}^{m\times m}$. The original kernel matrix K can then be reconstructed using Eq. (8)

$$\begin{aligned} K &\simeq \left(\sqrt{\frac{m}{n}} E\Phi_z\Lambda_z^{-1}\right)\left(\frac{n}{m}\Lambda_z\right)\left(\sqrt{\frac{m}{n}} E\Phi_z\Lambda_z^{-1}\right)^\top \\ &= EW^{-1}E^\top \end{aligned} \tag{9}$$

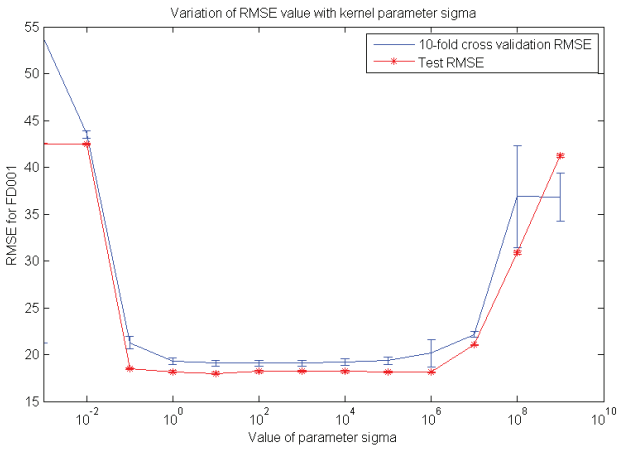

Fig. 3: Variation effects of parameter σ for RBF kernel

The FE can then be carried out without evaluating the kernel matrix K. Using Eq. (8) and (9)

$$\begin{aligned} X_{new} &= K \cdot \Phi_k \\ &= \sqrt{\frac{n}{m}} E \Phi_z \end{aligned} \tag{10}$$

The implementation used in this paper is identical to that described in [25] which proposes to make use of the centers from K-Means clustering as the subset of landmark points instead of the original method of random sampling from the dataset.

2) Parameter Variation: In this paper, the gaussian function is used as the kernel function in KPCA. Therefore, the elements of the kernel matrix K can be evaluated using

$$k(x_i, x_j) = e^{\frac{-\|x_i - x_j\|^2}{2\sigma^2}} \tag{11}$$

It can be seen that the parameter σ affects the distance metric calculation and is therefore an important parameter affecting the eventual accuracy of the algorithm. The selection of parameter σ value was carried out based on a 10-fold cross validation process where the value of sigma ranged from 10^{-9} to 10^{9}. The cross validation process was carried out on the training set of FD001 and the optimal value of σ was assumed to be applicable for the remaining datasets. Throughout the experiment, the number of features extracted is fixed at 10.

In order to ensure direct correlation between validation error and true testing error, testing errors were also computed for the same range of σ values. Fig. 3 shows the variation of RMSE validation and testing error for the range of σ values (RMSE scores for $\sigma < 10^{-2}$ are omitted due to relatively poorer performance)

It is observed that the performance of the MLP remained relatively stable across a wide range of σ values ($10^{0} - 10^{6}$). Therefore, in the remainder of the paper the value of σ is fixed at 10^{1}. Furthermore it was also observed that there is a strong direct correlation between the validation RMSE and testing RMSE, this useful correlation allows for tuning of parameters and network structures using a validation set.

C. K-Means Based Feature Extraction

In this section, K-Means clustering is used as an unsupervised learning module to select suitable vectors as FE of the whole dataset. This method has recently been proposed as a fast and easily implemented alternative to sparse FE which is suitable for large scale data. It has also been shown to provide excellent results in image processing [26], [27].

The main idea of this method is that K-Means can be used to create a "dictionary" of m centers, such that $D \in \mathbb{R}^{d \times m}$. The original data vectors $x_i \in \mathbb{R}^d$ can then be expressed as a code vector of these centers such that the error of reconstruction is minimized [26].

In the original article [26], special changes have been proposed to address some of the shortfalls of K-Means feature representations. However in this paper, a simpler version will be used as it is deemed to be sufficient for this purpose. In general, The K-Means FE methodology can roughly be separated into three main components

1) **Normalize datapoints:** In this paper, the datapoints have already been normalized in section II-A1
2) **Carry out K-Means clustering method:** While the original authors proposed the use of a spherical K-Means with special initialization sequence, based on the results shown in this paper indicate that the classical K-Means with random initialization is sufficient to show improvement over traditional FE methods for this dataset.
3) **Maps original data to a feature vector:** There are a few functions proposed by the original authors which could be used to map the original data into a feature vector. In this case, a soft threshold non-linear mapping is used due to its ease of implementation. mathematically the function can be expressed as $F(x;\ D, \alpha) = max\{0, D^\top x - \alpha\}$

Similar to the KPCA FE method (section IV-B), the number of centers extracted in the K-Means clustering process determines the eventual dimension of the mapped feature vectors.

1) Parameter Variation: As seen in section IV-C, the choice of using soft threshold non linear mapping generates a parameter α which is capable of tuning the performance of the overall algorithm. In order to select an optimal value for this parameter, a 10-fold cross validation was carried out on the training set of FD001. The experiment was repeated for the range of values of α from 10^{-10} to 10^{10}. However, due to the relatively poorer performance of the algorithm for $\alpha > 1$, these results have been omitted from Fig. 4. Based on the results, the validation RMSE of the method remains relatively stable and low for $\alpha \leq 0.1$. Therefore, in the remainder of this paper, the value of α is fixed at 10^{-5}.

D. Results of FE methods on single instant data

In this section, the relative performance of the three FE methods (Section IV-A, IV-B, IV-C) and the base regressor are presented together for comparison. In order to determine the optimal dimension of the reduced feature set, a 10-fold cross validation was carried out for a range of features from 1 to 14

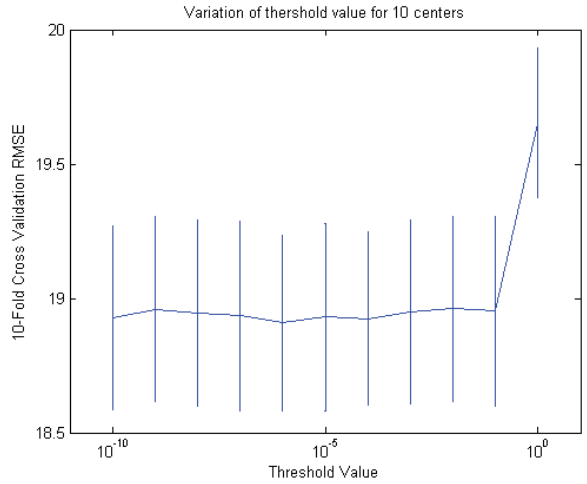

Fig. 4: Variation effects of threshold value

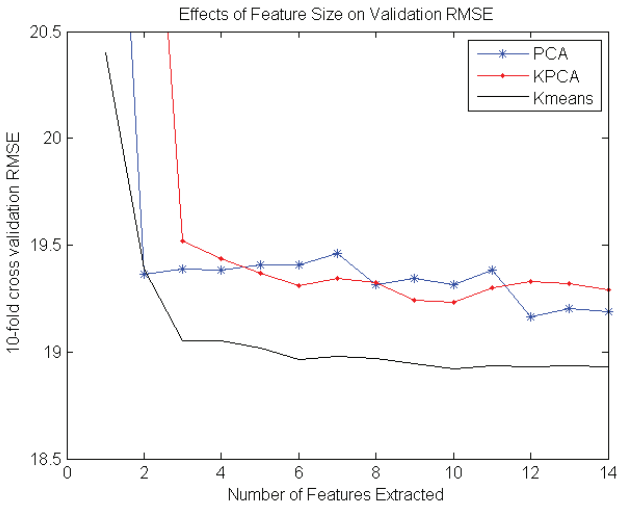

Fig. 5: 10-fold cross validation RMSE for different number of extracted features on FD001 training set

on FD001 training set. The validation RMSE values plotted in Fig. 5 are the mean RMSE values obtained from the 10-fold cross validation. The aim of this section is to determine the optimal FE method and its corresponding parameter values for application in the final framework.

It can be observed that the validation RMSE of each FE method remains rather stable for 3 or more extracted features. In addition, the mean validation RMSE value obtained using the K-Means method is slightly better than the other two FE methods.

The same experiment was repeated using test set of FD001 to verify the phenomenon observed in Fig. 5. The results of this experiment is shown in Fig. 6 where each point on the graph represents the mean RMSE value of 10 independent trials. In contrast, the mean RMSE of all 3 methods on the test sets shows much more haphazard fluctuations against the

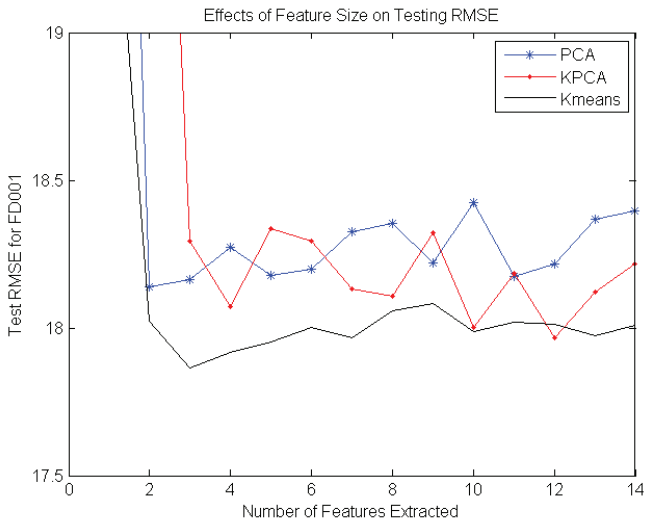

Fig. 6: Test RMSE for FD001 test set for different number of extracted features

number of feature extracted. However, similar to the results in Fig. 5, K-Means FE method outperforms the other two methods in most cases.

Based on the validation results in Fig. 5, a feature size of 10 was determined as the default feature size for the remainder of the paper. This is chosen as all three FE methods have approximately stabilized. Upon fixing the number of extracted features, the relative performance of each FE method was compared across all datasets. The results of 5 independent trials for each method and each dataset was presented in a form of a boxplot in Fig. 7. For each trial, the training set was dimensionally reduced using the respective FE method and used to train the base regressor described in Section III. The accuracy of the overall method was then evaluated on the respective test set. In addition, the performance of the base regressor without any FE is also presented in Fig. 7 under the column "None".

By comparing the results across all datasets, it can be observed that the performance of the MLP with K-Means FE significantly outperforms the rest in all but one dataset (FD004). In addition, the performance of the FE methods is peculiar for FD004 as the base regressor outperforms the remaining three FE methods. A possible reason for this could be due to the higher number of operating modes and fault conditions in FD004, causing some critical information to be lost during FE. However, in general, it can be concluded that the K-Means FE method performed best for the three remaining datasets.

V. Time Window Processing

In numerous previous studies [9], [28], [29] as well as in Section III, the inputs to the learning algorithm are selected sensor values at each specific time step. This usually implies that the outputs from the learning algorithm is dependent

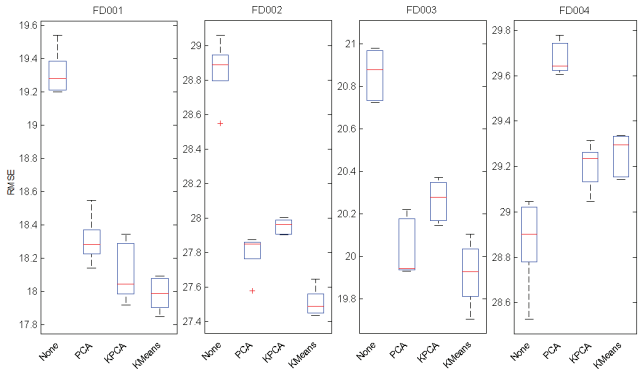

Fig. 7: RMSE of various FE methods and basic MLP across all test datasets

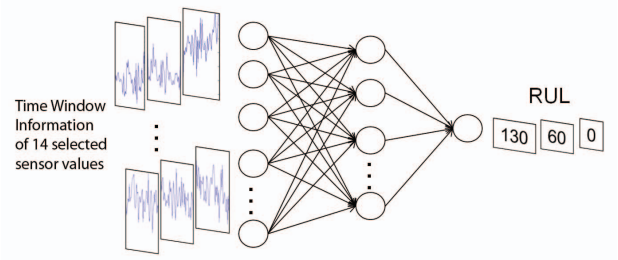

Fig. 8: Illustration of proposed time window method

only on the current input values. Similarly, methods such as neural networks assumes the data are independent and identically distributed (i.i.d.). Therefore, in order for the learning algorithm to take into consideration prior input values, this paper proposes using a time window based method. Another alternative to achieving the same effect is the use of algorithms which exhibit dynamic temporal behavior such as recurrent neural networks (RNNs) [11].

For a specific time window size, the method collects all the past desired sensor values within the time window and presents it as inputs to the learning algorithm. Fig. 8 illustrates the methodology of the proposed method. An important note is that the time window used in this methodology is a moving time window where each subsequent window differs by only one datapoint. As the input to the network at each instance is a time window, the input dimension of the network naturally increases as a multiple of the time window size. This would inevitably lead to higher computational load and a more complex problem. Therefore, a solution to this problem is to use FE methods to reduce the input dimensions which will be covered in Chapter VI.

A. *Effects of Time Window Size*

In this section, the effect of the time window size was considered. The range of time window sizes increased in units of 10 cycles until the maximum allowable value of each respective testset. The maximum allowable value is determined by the smallest operating history present in the test set. For

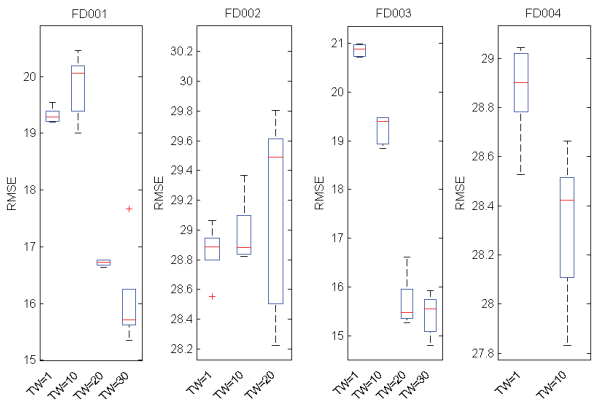

Fig. 9: Comparison of effects of time window size considered for different datasets. ('$TW = x$', the value of x refers to the length time window considered)

example, for an engine with an operating history of 20 cycles, it is impossible to create a time window of 30 cycles thus limiting the maximum size of the time window to 20 cycles. The results are shown in Fig. 9, and similar to previous experiments, the box plots are obtained from 5 independent trails for each experimental condition on their respective test set.

It can be observed that the improvement due to the time window input is the most significant in FD001 and FD003. On the other hand, FD004 shows slight improvement while in the case of FD002 the performance has deteriorated. It should however be noted that due to the maximum window size of 20 and 18 cycles respectively for FD002 and FD004, the improvement might not have been evident as compared to FD001 and FD003 which allows for a maximum time window of 30 cycles. This could also be a reason for the relatively poorer performance of the algorithms on FD002 and FD004, as having a shorter operating history imply that the system is still in its early stage thus making it more difficult to predict the RUL accurately.

In order to establish an accurate comparison, engines with operating history less than 30 cycles were deleted. This process resulted in the elimination of 6 engines and 11 engines from the testsets of FD002 and FD004 respectively. Each dataset is therefore left with 253 and 237 engines remaining out of the original 259 and 248 engines respectively. Fig. 10 illustrates the effects of the time window size on engines with operating history more than 30 cycles. The general trend observed is that a larger time window considered would result in a more accurate estimation of RUL. However, the point in which significant improvement was observed is different for each dataset. In FD001, FD003 and FD004 significant improvement was observed from a time window size of 20 onward. FD002 on the other hand only showed significant improvement from a time window size of 30. Time windows larger than 30 were not tested as it would result in elimination of more engines from all datasets. In addition, as the time window size increases,

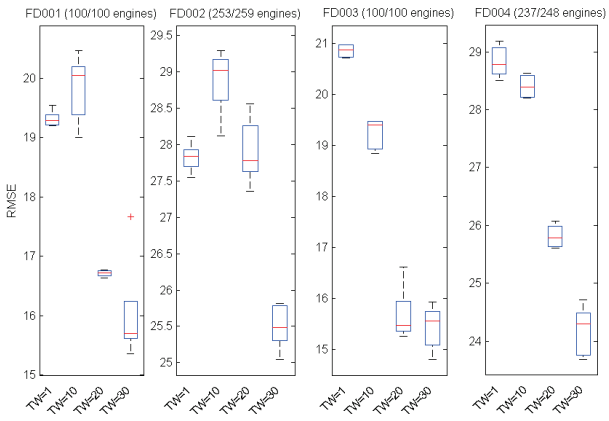

Fig. 10: Comparison of effects of time window size considered for different datasets on engines with operating history of more than 30 cycles. ('$TW = x$', the value of x refers to the length time window considered)

Fig. 11: Overall flow process of proposed framework

the practical applicability of the method decreases. Based on these considerations, the default time window size considered for the remainder of the paper is set at 30 cycles.

VI. Final Proposed Framework

In this section, the final complete algorithm is presented which involves the combination of forming a time window and then reducing the input dimension using an appropriate FE method. Fig. 11 illustrates the entire structure of the proposed framework

A. Results

1) Comparison of feature extraction methods: Fig. 12 plots the relative performance of different FE methods with a time window size of 30 cycles. The number features extracted per time window is calculated based on $m = 10 * window\ size$. The value of 10 was selected based on the observations in Section IV-D where the default number of features extracted per cycle was set at 10.

The results show that across all datasets K-means based FE has the best overall performance. It is also the only method which consistently outperforms the original time window method without FE, especially for FD004 where the opposite phenomenon was observed in Fig. 7.

2) Comparison of feature size: In order to verify that the optimal number of features extracted per cycle deduced in section IV-D can be generalized for the case of a time window, the relative performance of K-Means FE with different number of centers ($m = 90$, $m = 300$) for a time window size of 30 is shown in Fig. 13. Based on the results it can be concluded that Applying K-Means FE with more centers ($m = 300$) results in

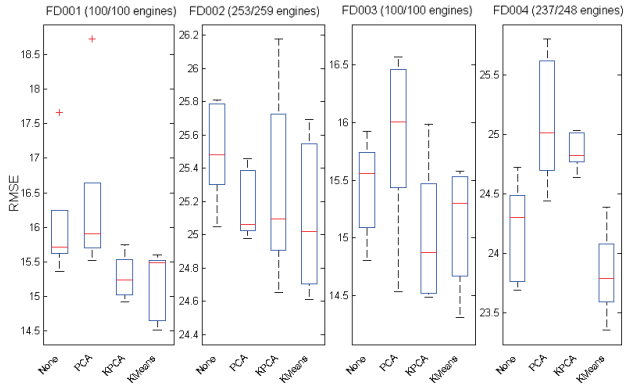

Fig. 12: Comparison of performance of respective FE methods on a time window size of 30. (Number of features extracted, $m = 10 * window\ size = 300$)

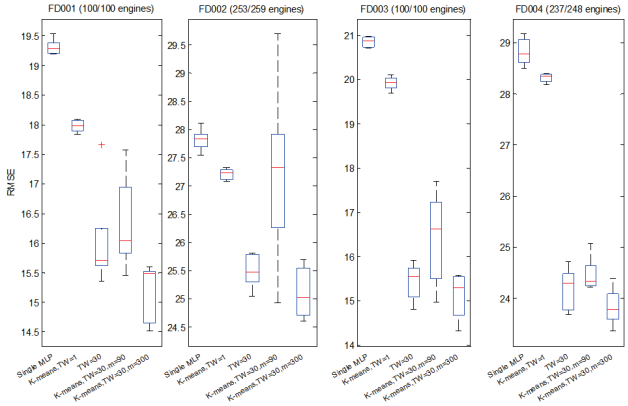

Fig. 13: Illustration of the change in RMSE values as each modular components are added to the proposed framework

lower RMSE values with smaller variance. The optimal value of m for a time window size of 30 is therefore set at 300.

Fig. 13 also illustrates the evolution of RMSE as different modular components are introduced to the basic MLP. It can be seen that the introduction of each modular component at each stage of the paper has resulted in improvements in the RMSE for all datasets. Overall, the final proposed framework has significantly lower RMSE value compared to the basic MLP.

3) Comparison with other papers: In this section, the performance of the proposed method is compared to other state-of-the-art methods reported in literature. Since most literatures have only reported results on the testset of FD001 in terms of RMSE, the results can be easily tabulated in Table II. The RMSE value for the proposed method in Table II is the mean value of 5 independent runs. The remainder of the values are identical to those reported in their respective original papers.

Based on the results, the proposed method is the best overall performer when taking into consideration the whole FD001 dataset. The RMSE value of the proposed method is

TABLE II: RMSE values of various algorithms on FD001 test dataset

Methods	No. of Engines	RMSE
ESN trained by Kalman Filter [28]	100 of 100	63.4565
Modified approach with classified sub-models of the ESN [28]	80 of 100	7.0221
Support Vector Machine Classifier [29]	100 of 100	29.823
Proposed method (Time Window = 30, K-Means m = 300)	100 of 100	15.1593

significantly lower than other method However, the modified ESN method proposed by [28] showed a lot of promise on 80 out of 100 engines. The main drawback is that the algorithm was unable to determine the RUL of the remaining 20 engines. Taking this into consideration, the proposed method is currently the best overall performer to the best of the authors' knowledge.

VII. Conclusion

This paper presents a new framework for predicting the RUL of aero-engines. While in this paper the algorithm has been tested on a simulated dataset simulating the degrading health of a commercial aero engine, the method can theoretically be applied to any other forms of similar mechanical systems. The framework makes use of a moving time window and K-Means based FE to improve the performance of a basic MLP. Furthermore, it was also shown that the proposed framework is the best overall performer when compared to other methods in the literature. The paper also clearly details the steps taken to arrive at the final choice of components and parameters used in the framework.

References

[1] N. Z. Gebraeel, M. A. Lawley, R. Li, and J. K. Ryan, "Residual-life distributions from component degradation signals: a bayesian approach," *IIE Transactions*, vol. 37, no. 6, pp. 543–557, 2005.

[2] M. A. Zaidan, A. R. Mills, and R. F. Harrison, "Bayesian framework for aerospace gas turbine engine prognostics," in *Aerospace Conference, 2013 IEEE*. IEEE, 2013, pp. 1–8.

[3] S. Uckun, K. Goebel, and P. J. Lucas, "Standardizing research methods for prognostics," in *Prognostics and Health Management, 2008. PHM 2008. International Conference on*. IEEE, 2008, pp. 1–10.

[4] T. Brotherton, G. Jahns, J. Jacobs, and D. Wroblewski, "Prognosis of faults in gas turbine engines," in *Proceedings of the 2000 IEEE Aerospace Conference*, Big Sky, MT, Mar. 2000, pp. 163–171.

[5] A. Saxena and K. Goebel. (2008) Phm08 challenge data set. NASA Ames Research Center. Moffett Field, CA. [Online]. Available: http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/

[6] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proceedings of the 2008 IEEE International Conference on Prognostics and Health Management*, Denver, CO, Oct. 2008, pp. 1–9.

[7] D. Chakraborty and N. R. Pal, "Selecting useful groups of features in a connectionist framework," *IEEE Trans. Neural Netw.*, vol. 19, pp. 381–396, Mar. 2008.

[8] T. Wang, J. Yu, D. Siegel, and J. Lee, "A similarity-based prognostics approach for remaining useful life estimation of engineered systems," in *Proceedings of the 2008 IEEE International Conference on Prognostics and Health Management*, Denver, CO, Oct. 2008, pp. 1–6.

[9] L. Peel, "Data driven prognostics using a kalman filter ensemble of neural network models," in *Proceedings of the 2008 IEEE International Conference on Prognostics and Health Management*, Denver, CO, Oct. 2008, pp. 1–6.

[10] P. Baraldi, F. Mangili, and E. Zio, "A kalman filter-based ensemble approach with application to turbine creep prognostics," *IEEE Trans. Rel.*, vol. 61, pp. 966 – 977, Dec. 2012.

[11] F. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Proceedings of the 2008 IEEE International Conference of Prognostics and Health Management*, Denver, CO, Oct. 2008, pp. 1–6.

[12] P. Lim, C. K. Goh, K. C. Tan, and P. Dutta, "Estimation of remaining useful life based on switching kalman filter neural network ensemble," in *Proceedings of the 2014 Annual Conference of the Prognostics and Health Management Society*, Fort Worth, TX, 2014, pp. 2–9.

[13] ——, "Multimodal degradation prognostics based on switching kalman filter ensemble."

[14] L. Zhang, R. Lukac, X. Wu, and D. Zhang, "Pca-based spatially adaptive denoising of cfa images for single-sensor digital cameras," *Image Processing, IEEE Transactions on*, vol. 18, no. 4, pp. 797–812, 2009.

[15] S. Borguet and O. Léonard, "Coupling principal component analysis and kalman filtering algorithms for on-line aircraft engine diagnostics," *Control Engineering Practice*, vol. 17, no. 4, pp. 494–502, 2009.

[16] J. Shlens, "A tutorial on principal component analysis," *Systems Neurobiology Laboratory, University of California at San Diego*, 2005.

[17] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Artificial Neural NetworksICANN'97*. Springer, 1997, pp. 583–588.

[18] Q. Wang, "Kernel principal component analysis and its applications in face recognition and active shape models," *arXiv preprint arXiv:1207.3538*, 2012.

[19] T. Takiguchi and Y. Ariki, "Robust feature extraction using kernel pca," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1. IEEE, 2006, pp. I–I.

[20] H. Hoffmann, "Kernel pca for novelty detection," *Pattern Recognition*, vol. 40, no. 3, pp. 863–874, 2007.

[21] A. J. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," 2000.

[22] C. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems 13*. Citeseer, 2001.

[23] D. Achlioptas, F. McSherry, and B. Scholkopf, "Sampling techniques for kernel methods," in *Advances in Neural Information Processing Systems 14: Proceedings of the 2002 [sic] Conference*, vol. 1. MIT Press, 2002, p. 335.

[24] T.-J. Chin and D. Suter, "Improving the speed of kernel pca on large scale datasets," in *Video and Signal Based Surveillance, 2006. AVSS'06. IEEE International Conference on*. IEEE, 2006, pp. 41–41.

[25] K. Zhang, I. W. Tsang, and J. T. Kwok, "Improved nyström low-rank approximation and error analysis," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1232–1239.

[26] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 561–580.

[27] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223.

[28] Y. Peng, H. Wang, J. Wang, D. Liu, and X. Peng, "A modified echo state network based remaining useful life estimation approach," in *Prognostics and Health Management (PHM), 2012 IEEE Conference on*. IEEE, 2012, pp. 1–7.

[29] C. Louen, S. X. Ding, and C. Kandler, "A new framework for remaining useful life estimation using support vector machine classifier," in *Control and Fault-Tolerant Systems (SysTol), 2013 Conference on*. Nice: IEEE, Oct. 2013, pp. 228 – 233.