Temporal Classification of Turbofan Engine Health using Elman Recurrent Network

Navar M. M. Nascimento*†, Guilherme A. Barreto†, Cairo L. Nascimento Jr.*, Pedro P. Rebouças Filho†

*Instituto Tecnológico de Aeronáutica, São José dos Campos - SP, Brazil
Emails: navar@lapisco.ifce.edu.br, cairo@ita.br

†Graduate Program on Teleinformatics Engineering (PPGETI),
Federal University of Ceará, Fortaleza - CE, Brazil
Emails: gbarreto@ufc.br, pedrosarf@ifce.edu.br

***Abstract*—Prognosis and health management (PHM) plays an essential role in condition-based maintenance routines. For such purposes, academy and industry have devoted considerable efforts into providing efficient, safe and reliable solutions. In this regard, we aim at contributing to this field by proposing a temporal classifier for engine's health state identification based on the Elman recurrent neural network. The evaluation of the proposed approach involves a benchmarking data set originated from the C-MAPSS, a flexible turbofan engine simulation by NASA. A comprehensive performance comparison with state of the art approaches is then carried out. The proposed system is able to identify engine's total degradation 125 steps in advance, with 86.21% of confidence and low false negative rate, i.e. less than 2% of engines faulty conditions are identified as normal. With a temporal-based classification, the proposed approach reaches over 95% of accuracy on turbofan diagnosis.**

***Index Terms*—Prognosis, Turbofan engine, C-MAPSS, fault detection, machine learning, Elman network**

I. Introduction

Health prognosis of an industrial equipment is a key process in condition-based maintenance strategies, since it contributes to reduce related risks and maintenance costs of the equipment under supervision, improving its availability, reliability and security issues. In recent years, data-driven approaches have gained momentum for prognosis purposes in order to improve system's health management [1]. In this regard, it is common to evaluate data-driven approaches for health prognosis, especially those based on machine learning methods, using benchmarking data sets [2], with special attention devoted to the C-MAPSS[1] database provided by the National Aeronautics and Space Administration (NASA) [3].

A common goal of data-driven approach to equipment health prognosis is the estimation of the remaining useful life (RUL) of its components [4]–[6]. Bearing this in mind, in this paper we aim at contributing to the portfolio of machine learning tools for equipment prognosis by introducing a neural-based temporal classifier for engine's health state identification based on the Elman recurrent network. The evaluation of the proposed approach involves the aforementioned C-MAPSS data sets and a comprehensive performance comparison with state of the art approaches. We report results showing that the proposed system is able to identify engine's total degradation 125 steps in advance, with 86.21% of confidence and low false negative rate, i.e. less than 2% of engines faulty conditions are identified as normal. With a temporal-based classification, the proposed approach is able to reach over 95% of accuracy on turbofan diagnosis.

The remainder of the paper is organized as follows. In Section II details on the C-MAPSS database are presented. We also discuss state of art approaches for the problem of interest. In Section III an exploratory analysis is carried out in the PHM'08 database. In Section IV the experimental settings and simulations are described. The obtained results are reported in Section V. The paper is concluded in Section VI.

II. Details on the C-MAPSS Data Set

NASA plays an important role on the problem of interest to the current paper, since it provides the most important database for prognosis and health management on turbofan engines [1]. The NASA's concern on this field is such that Frederick *et al.* [3] provide a user's guide for the C-MAPSS simulation tool[2] with which advanced control and diagnostic algorithms can be developed and quickly tested on a generic turbofan engine simulation. NASA's Prognostics Center of Excellence (PCoE) maintains a repository[3] full of different data sets for such purposes.

Among them one have attracted special interest from system modeling community: the PHM'08 data set for health state segmentation/classification. It has been developed for the Prognosis and Health Management competition occurred in 2008 and since then has been used for performance benchmarking of machine learning algorithms due to its complexities and high fidelity to real world scenarios [1].

Ramasso and Gouriveau [7], working on this data set and using a neurofuzzy classifier, reported 66.25% of accuracy. This work provides an important baseline of comparison for research on this data set for machine learning methods.

[1]Acronym for *Commercial Modular AeroPropulsion System Simulation*.

[2]www.grc.nasa.gov/www/cdtb/software/mapss.html

[3]https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

Ramasso [8] evaluated Hidden Markov models (HMM) and reached 69% of accuracy, but using a slightly different version of the PHM'08 data set (one with only a single operational setting and two types of faults, having labeled data as well). Tamilselvan and Wang [9] achieved over 90% of accuracy, being a state of art approach on this problem. Zhao *et al.* [10] also reports 90% of accuracy on their paper. However, there are quite a few missing details about the used methodology for a fair comparison. For example, the authors fail to explain how the operational setting are taken into account, if this was the case; and, if the health state definitions used were the same as proposed by Ramasso and Gourivea [7].

III. Exploratory analysis

There a five different data sets from turbofan engine simulation model [1] available from NASA PCoE. Frederick *et al.* [3] and Saxena *et al.* [2] explain in details how the data can be generated. These databases were first introduced during the PHM'08 challenge and since then are public available from NASA's data repository.

Among the five NASA's turbofan degradation databases, we have chosen the one named "*PHM08 Challenge Data Set*", still publicly available for download by the time the current research was developed. There is also a further one named "*5T*" in the review of Ramasso and Saxena [2]. However, this data set poses an additional difficulty with respect to other turbofan data sets, because there is no ground truth to measure performance.

A. Description of the PHM'08 data set

The data set is already split into train an test sets. The guidelines of the competition are strict into using only the training set for model selection and evaluation. We recommend to addresses NASA's guideline for further details [3].

The database contains records of 21 sensors in 218 units. A sample from these records are exhibited in Fig. 2b, wherein the time series of the engine decay is represented in sensors 3, 4, 11 and 12. These records are from unit number 1. There are different approaches in literature to address this data set, some of them intend to predict when the failure will occur, while others aim at detecting the engine's health state [1]. Our approach aims at creating a reliable and expert system to detect turbofan's health state.

B. Preprocessing steps

Most of the approaches reported on literature are based on supervised learning, a scenario where labels are known. This is not the case for the PHM'08 data set. However, researchers have addressed this issue already. Ramasso and Gouriveau [7] proposed an algorithm for signal segmentation into four health state (HS), which has received a ground truth years ahead [11]. Tamilselvan and Wang [9] used a slightly different ranges for labeling HS at the signals. We follow the same procedures, which are described in the following.

- The last 50 time cycles in each signal are labeled as HS-4. It corresponds to a on-failure state on the engine.;
- The next 51 to 125 time cycles are HS-3. It is to a caution state because there is an imminent failure to occur;
- The followed 126 to 200 are a transition, HS-2. And, there is not much different from a normal health state.
- Time cycles longer than 200 from the signal ending are considered normal steady state, HS-1.

TABLE I: Summary of the PHM'08 database with the number of samples, sensors and units and, the subsets created by clustering the operational settings.

	Full database	Subset of operational settings					
	Train database	OS-1	OS-2	OS-3	OS-4	OS-5	OS-6
N. Samples	45918	6882	6771	6954	11680	6750	6881
N. Units	218						
N. Sensors	21						

The result of this process is shown in Fig. 1b. However, engines have different degrees of initial manufacturing, which means that the number of time cycles until total degradation varies from unit to unit within the range $[128, 357]$. Among all engines, 25% have been degraded before 180 time cycles, 50% before 210 time cycles and 75% before 240 time cycles.

Wang *et al.* [12] identified that the operational settings (OS) can be grouped into six clusters, as in Fig. 2a. The point is to identify the OS of readings each time cycle. This procedure results in dispersion such as the one in Fig. 2b. The evolution of readings on sensor 1, for operational setting 1, indicates the decay of engine's health.

Furthermore, Wang *et al.* [12] claims that some of the 21 sensor contain redundancy and useless information, so they propose to work with sensors 2, 3, 4, 7, 11, 12 and 15 only. Indeed, after grouping the operational settings, some of theses sensors did not show a trend as the one exhibited in Fig. 2b. The readings of these sensors will be used as the feature set for all classifiers.

In order to compare with the state of the art approaches, as Tamilselvan and Wang [9], we have followed the same methodology, creating subsets of data by clustering operational settings. This leads to six different data sets to work with. In Table I we provide a summary of all generated data sets. The next steps concerns the description of the experimental procedures used in the simulations.

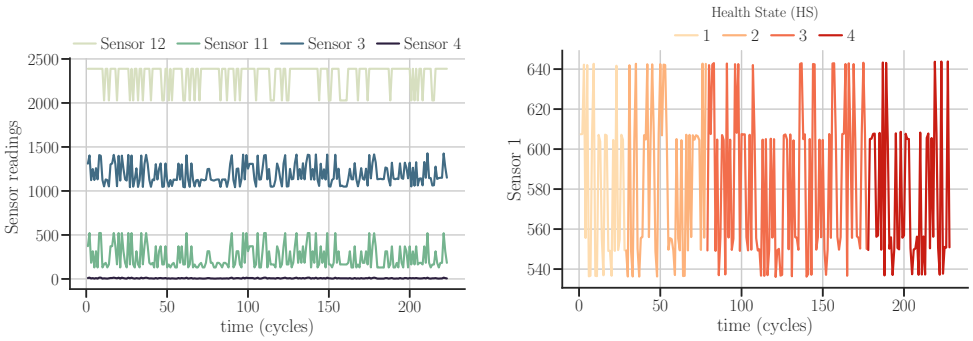
IV. Experimental settings and simulations

In this section we describe the methodology that has been followed to compare the performances of various machine learning techniques for turbofan engine diagnosis.

A. evaluated classifiers and their configurations

There are several paradigms that can be used for pattern classification purposes, each one with its own motivation, assumptions and discriminative power. In our work, we aim at evaluating different paradigms, including non-parametric and parametric, linear and nonlinear classifiers.

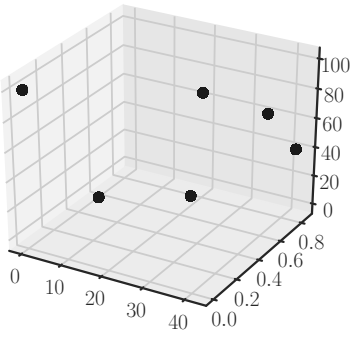
The k-nearest neighbors (KNN) is a non-parametric method that can be used for classification purposes, predicting class labels by calculating distances (e.g. euclidean distances) from



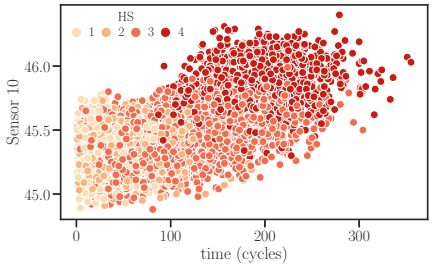
(a) Readings of sensors in unit number 1.

(b) Degradation evolution of sensor 1 in unit number 1. The Health states proposed by Tamilselvan and Wang [9].

Fig. 1: Example of sensor readings during the engine degradation.



(a) Clusters of operational settings (OS).



(b) Readings of sensor 1, grouped by OS 1. The shades of red indicate the engine's health decay.

Fig. 2: Diagrams from NASA's C-MAPSS technical report.

the current input pattern to all examples in the training data set. Then, the input pattern is assigned to the most frequent class label among those associated with the k closest examples. Another widely used non-parametric method is the *Random Forest*, which is in essence an ensemble of decision trees.

The simplest type of neural network based method is the Perceptron. It was proposed by Rosenblatt [13] and is based on the neuron model of McCulloch and Pitts [14]. It is basically a linear classifier with an adaptive learning rule for weight adjustments. Among the various flavors of artificial neural network architectures, the multilayer perceptron (MLP) network is a powerful tool for nonlinearly separable pattern classification problems. The MLP is comprised of a system of interconnected perceptrons, capable of representing a nonlinear mapping between the input and output variables. Connection weights link the inputs to output signals, through a layer of hidden nonlinearly neurons [15].

Of particular interest to this work is the Elman recurrent network [16], a dynamic MLP-based network in which the outputs of the hidden neurons are fed back to the augmented input layer. This feedback path gives rise to a short-term memory mechanism capable of handling temporal dependencies present in the input signal.

Bayesian Gaussian classifiers use Bayes probability rule to classify new samples as belonging to the most likely class given the current data [17]. An unknown example is assigned to the class with the highest score given by the maximum value of the *a posteriori* probability density function. For this purpose, one must assume that the data in separate classes follows a Gaussian probability density model, with their own mean vector and covariance matrix.

Support vector machines (SVM) are kernel-based classifiers introduced by Cortes and Vapnik [18]. Originally, SVM was designed to solve binary problems, but it can be extended to multiclass problems in several ways using pairwise approaches, such as one-versus-one and one-versus-all, to mention a few possibilities. These kernel machines provide a powerful way of designing nonlinear classifiers by projecting data samples into a new high-dimensional feature space in which they can be separated by a hyperplane.

The minimal learning machine (MLM), recently proposed by Souza Junior *et al.* [19] for supervised learning problems, is a distance-based regression and classification method. In this regard, it builds a map not between the input and output vectors, but rather between the distance matrices of the input and output data vectors. Data representation via distance matrices may reveal hidden structure and relationships among original data samples that cannot be easily seen in the original data representation. The found hidden structures can then be

used for classification purposes.

The configurations of all the the classifiers evaluated in this paper are listed below.

- KNN - Distance metric: euclidean; number of neighbors: 5 to 10.
- Random Forest - number of estimators: 10; maximum depth: 10; maximum number of ramifications: 10; Meta-algorithm: *bagging*.
- Perceptron LMS - $\eta = 0.01$; maximum number of epochs: 1000; error tolerance: 10^{-4}.
- Perceptron SGD - $\eta_0 = 0.005$; maximum number of epochs: 1000; error tolerance: 10^{-4}; regularization: L_2; $\alpha : 0.0001$.
- MLP - Cost function: cross-entropy; activation function: hyperbolic tangent; $\eta_0 = 0.001$; error tolerance: 10^{-5}; training algorithm: ADAM [20] and L-BFGS. [21]; number of neurons ranging from 2 to 1000; number of hidden layers: 1 to 2;
- MLM - Number of reference points ranges from 10 to 20.
- SVM - $\gamma \in [2^{-15}, 2^{33}]$; $C \in [2^{-5}, 2^{15}]$; kernels: linear, polynomial and RBF.

B. Experimental settings

The training runs for tuning of hyperparameter and model selection are carried out using the *train* file in the given data, while the test of the best model is made upon the *test* file. All classifiers are trained on the six data sets, representing the operational conditions of the engine. The goal is to end up with a well-tuned model for each one of these subsets. A simulation using 10-fold cross validation is performed to emulate a Monte Carlo simulation on all classifiers. We used a random search [22] with 50 candidates for hyperparameter choosing.

The metrics used for evaluation is both accuracy and confusion matrix. The configurations of the classifiers are chosen based on these metrics and only the best results are reported in the next section. The experiments were conducted in a PC Intel i7 with 3.1 GHz and 8Gb of RAM running under a Linux Ubuntu operating system installed in a solid-state drive. We coded the experiments using the Python language and the simulations and results are available online[4].

V. Results and discussions

In this section the results are presented following the chronological order in which the experiments were conducted. Each stage of the research has risen hypothesis and important questions that motivated the subsequent actions regarding the design of the experiments, until we eventually reached our ultimate goal.

The results achieved by all classifiers on the OS-1 and OS-2 subsets of data are exhibited in Tables II and Tab. III, respectively. Results from subsets 0S-3 to OS-6 will not be presented due to limited space, but they lead to similar conclusions as the ones discussed presented here. Anyway, these undiscussed results can be reached at the aforementioned web repository. Note that we provided the results on the training set, as a means to evaluate overfitting.

A closer look at the results on the OS-1 subset, the KNN classifier performed 14% better on the training stage than on the testing set. The Random Forest classifier achieved 98.51% on the training, but achieved only 58% on the testing set, indicating that it has been overfitted despite any precautions. Among the Gaussian classifiers, the linear variant with same covariance matrix computed over the whole set of training data, performed slightly better than the Naive Bayes classifier[5] and 13% better than the quadratic Gaussian classifier. Among the linear classifiers, the perceptron trained with the Hinge loss function performed 11% better than its similar with the Widrow-Hoff learning rule. The MLP network achieved 62.06% of accuracy, performing similarly to the Gaussian classifiers; however, the MLP still has room left to improvements, e.g. by increasing the number of hidden layers.

The results from the OS-2 data set, shown in Table III, follow the same performance patterns as those achieved for the OS-1 data set, with the Random Forest overfitting the training data and performing poorer than other classifiers in the testing set. The Elman has achieved the same performance, along with MLP in its both configurations. The SVM classifier using the RBF Kernel has achieved over 61% of accuracy.

Comparing to related results reported on literature, Ramasso [8] reached 69.25%, using a simplified version of these data sets, while Ramasso and Gouriveau [7] uses the same one. In both reports, the authors based their approaches on the HMM model, which is a probabilistic approach. Tamilselvan and Wang [9] reached 90.72% and 95.80% on the OS-1 and OS-2 data sets, respectively. The author used DBN which is also a probabilistic model. As already mentioned w could not made a fair comparison to Zhao *et al.* [10] since the lack of methodological details. From now on, the results to be reported originated from experiments designed to closely follow the approach by Tamilselvan and Wang [9].

By analyzing the confusion matrix of the MLP classifier in Table IV one can easily observe that most of the misclassification involves the health state 1 (HS-1) and the health state 2 (HS-2).

The model's misclassification rates are the following ones: HS-1 $\Rightarrow$ HS-2 (24.06%), HS-1 $\Rightarrow$ HS-3 (74.06%), and HS-1 $\Rightarrow$ HS-4 (0.27)%. It should be noted, however, that to misclassify HS-1 as either HS-2 or HS-3 does not incur in a serious problems because HS-1, HS-2 and HS-3 are not faulty states. The false positive rate for the MLP classifier is very small (0.27%)

The false positive rate is 0.92%, 0.53% and 0% for HS-2, HS-3 and HS-4 respectively. So, very few of faulty samples are misclassified as in a Normal health state. It is much lower than the false negative error, showing a robustness in false

[4]https://github.com/navarmn/Turbofan-Airplane-fault-prognosis

[5]Also a Gaussian classifier with the assumption of independence between input attributes. This is equivalent to a Mahalanobis distance-based classifier using a diagonal covariance matrix.

TABLE II: Results from all classifiers on the OS-1 data set. Features are the sensors reading just as reported by Wang [12]

Classifier	Acc - Train (%)	Acc (%) - Test
KNN	69.71±0.34	55.18±1.87
Random Forest	98.51±1.06	58.01±1.72
Gaussian Naive-Bayes	61.87±0.23	61.55±2.13
Gaussian Linear discriminant	62.42±1.06	**62.35±1.25**
Gaussian Quadratic discriminant	38.95±9.71	39.39±10.07
Perceptron - LMS	45.61±7.07	45.07±6.98
Perceptron - HL	56.83±3.44	56.30±3.73
MLP	63.18±0.29	**62.70±1.45**
MLP-2 hidden	64.27±1.13	**63.15±0.17**
Elman	62.27±2.15	**63.06±3.78**
MLM	61.15±1.04	60.97±0.23
SVM-Linear	60.13±1.15	59.15±0.54
SVM-Polynomial	62.87±1.63	61.97±1.15
SVM-RBF	61.57±2.17	**62.54±2.07**
Reports on literature		
Ramasso [8] - HMM	-	69.25
Ramasso and Gouriveau [7] - HMM + Fuzzy	-	66.25
Zhao *et al.* [10] - SVM	-	90
Tamilselvan and Wang [9] - DBN	-	90.72

TABLE III: Results from all classifiers on the OS-2 data set. Features are the sensors reading just as reported by [12]

Classifier	Acc - Train (%)	Acc (%) - Test
KNN	69.10±0.54	55.10±0.99
Random Forest	98.42±2.03	56.34±1.36
Gaussian Naive-Bayes	60.15±0.23	56.74±2.12
Gaussian Linear discriminant	63.15±1.06	**62.93±1.36**
Gaussian Quadratic discriminant	35.54±7.71	29.46±10.59
Perceptron - LMS	54.74±3.04	54.95±6.79
Perceptron - SGD	53.83±3.12	54.76±6.30
MLP	62.90±0.27	**62.06±0.12**
MLP-2 hidden	63.17±1.15	**61.15±0.58**
Elman	61.27±1.15	**62.19±1.95**
MLM	60.27±1.14	59.57±0.38
SVM-Linear	61.23±1.38	61.15±1.44
SVM-Polynomial	61.58±0.83	60.75±0.85
SVM-RBF	60.37±0.98	**61.23±1.57**
Reports on literature		
Ramasso [8] - HMM	-	69.25
Ramasso and Gouriveau [7] - HMM + Fuzzy	-	66.25
Zhao *et al.* [10] - SVM	-	90
Tamilselvan and Wang [9] - DBN	-	90.72

positive errors. This is very important, because in a real-world scenario scenario this type of error means the expert system is indicating to the maintenance team that the aircraft is ready to go, but it is not, especially if it is on HS-3 and HS-4. It may eventually lead to several damage in both airplane and personnel. This type of error is considered very serious and can not be tolerated while designing a classifier for such purposes.

It should be noted that, due to the nature of the problem, the degradation grows in magnitude from HS-1 to HS-4, passing through all health states. There is no scenario where the degradation happens in HS-1, for instance. What might happen is the degradation grows faster in some engine than in others, but always going through all this predefined states. It means the following errors are reinterpreted as: (i) 23.62% of HS-4 classified as HS-3 is not very critical because those are faulty stage regions, as shown in Figure 2b; so as the 79.95% of errors from misclassification of HS-2 into HS-3, since those are transition stages. This analysis indicates the MLP is being conservative in its classifications and taking care for faulty

TABLE IV: The confusion matrix of the MLP network on the OS-1 data set. The features are the sensor readings suggested by Wang *et al.* [12].

Labels	Predictions			
	HS-1	HS-2	HS-3	HS-4
HS-1	**1.6%**	24.06%	74.06%	0.27%
HS-2	0.92%	**18.55%**	79.95%	0.58%
HS-3	0.53%	5.71%	**86.21%**	7.55%
HS-4	0.0%	0.0%	23.62%	**76.38%**

classes more than the normal one.

Furthermore, another interpretation from the confusion matrix in Table IV is that the classifier predicts with 76.38% of confidence that the failure will happen in no more than 50 engine cycles. And with 86.21% of confidence the total degradation of the engine will occur up to 125 time cycles. 18.55% of confidence in 200 engine's time cycle and 1.6% in more than 200 engine's time cycle. The furthest the engine is from its degradation, the harder is to predict it with confidence. The classifiers also performs like an expert engineer, making no strong assumptions without prior knowledge. Thus, this can be a powerful indicator for the maintenance team, because even if they do not have a precise knowledge of the time at which the failure will occur, they have a good indicator based on the engine's operational cycle.

In short, by the proposed approach we were able to detect with 86.21% of confidence the engine's total degradation with more than 100 operational cycles in advance, with less than 2% of false negatives. But even so, we have questioned ourselves if there was room for improvements in the classifiers' performances.

A detailed analysis of all the features was carried out and different sensors were chosen, removing the ones with lower variance. Thus, we came up with the following set of sensors' readings as the new feature set: 1, 2, 3, 6, 8, 10, 11, 12, 13, 14, 19, 20. Then, a new round of performance comparison involving all the classifiers was executed. The obtained results are reported in Table V. The SVM and MLM classifiers were not taken into consideration in this step, since they did not show any improvements over the MLP classifier. By the same token, the one-hidden-layered MLP classifier was chosen instead of the two-hidden-layered architecture.

The results are slightly different than that using the sensors' reading proposed by Wang *et al.* [12] and from now on our proposed set of sensors' readings will be the one used in this work. In summary, the MLP classifier kept basically the same performance, while the Elman network had a 2% improvement in the accuracy rate. This apparently minor result then led us to elaborate further on the design of the Elman-based classifier, aiming to improve the performance of this classifier even further.

The confusion matrices for the MLP and Elman networks are reported in Figures VI and VII, respectively. As expected, the performance pattern follows basically the same as before. Misclassifications tend to occur between states HS-2 and HS-3. The Elman network misclassified HS-1 and HS-2 as HS-3

TABLE V: Results from all classifiers on the OS-1 data set (testing data). Features are the readings from sensors [1,2,3,6,8,10,11,12,13,14,19,20]

Classifier	Acc after (%)	Acc before (%)
KNN	57.90±0.27	55.10±0.99
Random Forest	60.51±2.03	56.34±1.36
Gaussian Naive-Bayes	58.28±3.23	56.74±2.12
Gaussian Linear discriminant	63.61±1.33	62.93±1.36
Gaussian Quadratic discriminant	61.39±2.23	29.46±10.59
Perceptron - LMS	56.40±6.25	54.95±6.79
Perceptron - SGD	56.48±2.81	54.76±6.30
MLP	63.70±0.20	**62.06±0.12**
Elman	65.58±0.38	**64.07±1.15**
Reports on literature		
Ramasso [8] - HMM	-	69.25
Ramasso and Gouriveau [7] - HMM + Fuzzy	-	66.25
Zhao *et al.* [10] - SVM	-	90
Tamilselvan and Wang [9] - DBN	-	90.72

TABLE VI: The confusion matrix of the MLP classifier on the OS-1 data set. The selected features are the sensors' readings [1,2,3,6,8,10,11,12,13,14,19,20]

Labels	Predictions			
	HS-1	HS-2	HS-3	HS-4
HS-1	**3.16%**	38.05%	58.79%	0%
HS-2	3.84%	**26.37%**	69.79%	0%
HS-3	0.74%	9.88%	**83.55%**	5.82%
HS-4	0.0%	0.0%	18.07%	**81.93%**

with higher rates. However, it should be noted that the power of this network comes from its short-term memory ability. Thus, using standard (i.e. static) classification methodology for designing a recurrent network is not adequate to handle time series data.

Thus, we decided to follow a temporal classification approach with the hope of improving the performance of the Elman-based classifier. This works in two simple steps. Firstly, we collect the classification results of the Elman network over a predefined time window, e.g. 20 time steps. Then, the final classifier's decision is taken as the most frequent class label occurring within the time window. This simple change has improved considerably the overall accuracy rate of the Elman network since it was able to capture the temporal dependencies leading to the increase of the degradation state of the engine.

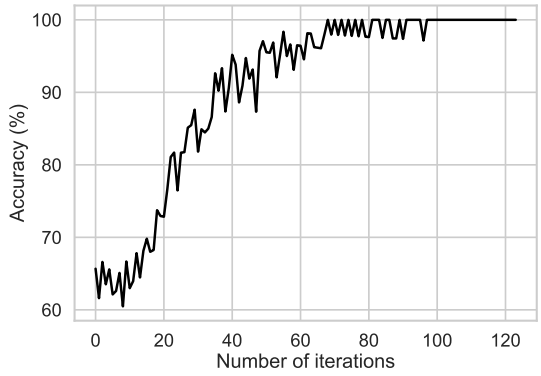
In Figure 3 the accuracy of the Elman network as a function of the time window is exhibited. With a time window of size 40, this temporal classifier achieved 95% of accuracy, a number equivalent to that reported by Tamilselvan and Wang [9], but using a much simpler approach. The temporal Elman-based classifier was then selected as the best performing one among all the classifiers evaluated in this research.

As a final remark, we arrived at the following conclusion. We tried to simplify too much a complex dynamic problem by adopting standard static classification methodology to handle it. Even well-known powerful nonlinear classifiers, such as the SVM and the MLP, performed poorly in identifying the health states of a turbofan engine. At the end, a simple but efficient approach using a simple recurrent network consistently outperformed all static classifiers.

TABLE VII: The confusion matrix of Elman on OS-1 subset of data. The features are the sensors [1,2,3,6,8,10,11,12,13,14,19,20]

Labels	Predictions			
	HS-1	HS-2	HS-3	HS-4
HS-1	**1.68%**	17.43%	77.05%	3.81%
HS-2	1.34%	**15.52%**	78.98%	4.15%
HS-3	0.66%	10.38%	**77.84%**	11.10%
HS-4	0.04%	3.26%	41.10%	**55.59%**

Fig. 3: Accuracy of Elman classifier, on subset of OC-1, ranging the number of iterations from 1 to 125.



VI. Conclusions

In this paper, we reported a comprehensive performance comparison involving multiple classifiers for detection of the health states of turbofan engines, on a benchmarking data set. By following the methodology of other reports on literature we were able to design experiments and evaluate a considerable number of classification paradigms, including neural networks and kernel methods.

The best performing classifier was based on the Elman recurrent network. This system was able to detect engine's total degradation with 125 step in advance, with 86.21% of confidence, with less than 2% of false negative rates, *i.e.*, engine's faulty conditions identified as normal. By using slightly modification in the Elman neural network prediction we could reach over 95% of accuracy for engine's health state identification, similar to the state of art approach of Tamilselvan and Wang [9], using a much simpler approach.

Currently, we are evaluating deep learning based architectures, e.g. LSTM-based deep networks, to design efficient temporal classification strategies on the same task.

Acknowledgments

This study was financed by the following Brazilian research funding agencies: CAPES (Finance Code 001) and CNPq (grant 309451/2015-9).

References

[1] E. Ramasso and A. Saxena, "Performance benchmarking and analysis of prognostic methods for cmapss datasets." *International Journal of Prognostics and Health Management*, vol. 5, no. 2, pp. 1–15, 2014.

[2] 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 (PHM'2008)*, 2008, pp. 1–9.

[3] D. K. Frederick, J. A. DeCastro, and J. S. Litt, "User's guide for the commercial modular aero-propulsion system simulation (C-MAPSS)," NASA Glenn Research Center, Tech. Rep. NASA/TM–2007-215026, 2007.

[4] A. K. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[5] A. Heng, S. Zhang, A. C. Tan, and J. Mathew, "Rotating machinery prognostics: State of the art, challenges and opportunities," *Mechanical Systems and Signal Processing*, vol. 23, no. 3, pp. 724–739, 2009.

[6] M. Pecht, "A prognostics and health management for information and electronics-rich systems," in *Engineering Asset Management and Infrastructure Sustainability*. Springer, 2012, pp. 19–30.

[7] E. Ramasso and R. Gouriveau, "Prognostics in switching systems: Evidential markovian classification of real-time neuro-fuzzy predictions," in *Proceedings of the 2010 IEEE International Conference on Prognostics and Health Management (PHM'2010)*, 2010, pp. 1–10.

[8] E. Ramasso, "Contribution of belief functions to hidden markov models with an application to fault diagnosis," in *Proceedings of the 2009 IEEE International Workshop on Machine Learning for Signal Processing (MLSP'2009*, 2009, pp. 1–6.

[9] P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification," *Reliability Engineering & System Safety*, vol. 115, pp. 124–135, 2013.

[10] D. Zhao, R. Georgescu, and P. Willett, "Comparison of data reduction techniques based on SVM classifier and SVR performance," in *Signal and Data Processing of Small Targets 2011*, vol. 8137. International Society for Optics and Photonics, 2011, p. 81370X.

[11] E. Ramasso and R. Gouriveau, "Remaining useful life estimation by classification of predictions based on a neuro-fuzzy system and theory of belief functions," *IEEE Transactions on Reliability*, vol. 63, no. 2, pp. 555–566, 2014.

[12] 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 (PHM'2008)*, 2008, pp. 1–6.

[13] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[14] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[15] S. Haykin, "Neural networks: principles and practice," *Bookman*, 2001.

[16] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[17] N. M. Nasrabadi, "Pattern recognition and machine learning," *Journal of electronic imaging*, vol. 16, no. 4, p. 049901, 2007.

[18] C. Cortes and V. Vapnik, "Support Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[19] A. H. de Souza Júnior, F. Corona, G. A. Barreto, Y. Miche, and A. Lendasse, "Minimal learning machine: a novel supervised distance-based approach for regression and classification," *Neurocomputing*, vol. 164, pp. 34–44, 2015.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, no. 4, pp. 525–533, 1993.

[22] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.