Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Variational encoding approach for interpretable assessment of remaining useful life estimation

Nahuel Costa [*,1], Luciano Sánchez [1]

Computer Science Department, University of Oviedo, Gijón, 33202, Asturias, Spain

A R T I C L E   I N F O

MSC:
0000
1111

Keywords:
Remaining useful life
Prognostics and health management
Interpretability
Variational inference
Recurrent neural networks

A B S T R A C T

A new method for evaluating aircraft engine monitoring data is proposed. Commonly, prognostics and health management systems use knowledge of the degradation processes of certain engine components together with professional expert opinion to predict the Remaining Useful Life (RUL). New data-driven approaches have emerged to provide accurate diagnostics without relying on such costly processes. However, most of them lack an explanatory component to understand model learning and/or the nature of the data. To overcome this gap we propose a novel approach based on variational encoding. The model consists of a recurrent encoder and a regression model: the encoder learns to compress the input data to a latent space that serves as a basis to build a self-explanatory map that can visually evaluate the rate of deterioration of aircraft engines. Obtaining such a latent space is regularized by a new cost function guided by variational inference and a term that penalizes prediction errors. Consequently, not only an interpretable assessment is achieved but also a remarkable prognostic accuracy, outperforming most of the state-of-the-art approaches on the popular simulation dataset C-MAPSS from NASA. In addition, we demonstrate the application of our method in a real-world scenario with data from actual Turbofan engines.

1. Introduction

Prognostic technologies are crucial in any physical system. In aircraft engines this is a must since throughout their life cycle they are subjected to different conditions that cause degradation and ultimately lead to failure. For this reason, data is routinely collected from various built-in sensors to monitor performance and avoid operating in undesirable conditions. Over the years, the amount of information collected has increased and this has paved the way for making more complex analyses in favor of maintenance that extends the useful life of these systems. However, traditional strategies such as scheduled preventive maintenance or corrective maintenance of failures [1] are increasingly unable to meet growing industrial demand in terms of efficiency and reliability. In this regard, Prognostics and Health Management (PHM) technologies are proving to have promising capabilities for application in industries [2]. As a result, metrics like remaining useful life (RUL) of systems have been established as key elements to improve maintenance schedules and avoid engineering, safety and reliability failures. Consequently, this would make it possible to determine engine deterioration, increase engine flight time and reduce maintenance costs.

1.1. Literature review

In the last decade, several techniques have been proposed to model the degradation of these complex systems, from which two currents arise: model-based approaches and data-driven approaches. Among the former, works such as [3,4] stand out, although these techniques require extensive prior knowledge about the physical systems, information that is often not available in practice. Precisely for this reason, data-driven approaches have become so popular in recent years, as they are able to model degradation features based purely on historical records from which the underlying causalities and correlations can be modeled. That is, knowledge can be inferred from sensor data to predict valuable system information such as RUL [5].

Especially, the use of Machine Learning models has had a great impact given that they are able to model highly nonlinear, complex and multi-dimensional systems with little prior prognostic experience. If we focus on RUL estimation, initial work was oriented towards the application of multi-layer perceptrons (MLP) as in [6], where the authors reported higher prediction results than model-based approaches. In [7,8] both diagnostics and prognostics were approached with PCA and hidden Markov models. Over the years, other techniques have

* Corresponding author.
E-mail address: costanahuel@uniovi.es (N. Costa).
[1] All authors have participated equally in all tasks.

https://doi.org/10.1016/j.ress.2022.108353
Received 4 November 2021; Received in revised form 10 January 2022; Accepted 21 January 2022
Available online 18 February 2022

been also explored: some researchers have integrated fuzzy logic to capture more information for Engine Health Monitoring (EHM) [9,10], others applied Support Vectors [11] or Gradient Boosting trees [12]. Nevertheless, despite being all of them considered relevant work for the sake of RUL estimation, the greater impact has undoubtedly been produced by the use of Deep Learning models [13]. This is due to the fact that the raw data obtained from machine health monitoring share a high dimensionality, similar to that of other problems in which these models have had a significant impact and are known to perform remarkably well, especially in Computer Vision and Natural Language Processing (NLP).

Certainly, RUL estimation is a hot-topic, partly thanks to the application of these new deep models where two trends clearly stand out: the use of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). High-impact work can be easily found in both directions, calling for the use of CNNs for feature extraction [14–16] and recurrent networks for modeling the temporal nature of the data [17–19]. From there, promising modifications have been proposed [20–22], where both architectures are combined for better prediction capabilities. In addition, approaches that go beyond RUL estimation are beginning to emerge such as [23] where a semi-supervised method is developed to avoid relying on data labeling or [24] where the authors present a model to mine different levels of degradation trends.

However, there is a clear gap between all Deep Learning oriented approaches: although they do achieve remarkable results, models are treated as black boxes where inputs are used to obtain some output, in this case, the RUL. It is challenging to find algorithms that go beyond providing good numerical performance and this is vitally important. Despite the fact that the current is to dispense with prior knowledge about the system to be monitored, in the end, these models are designed to be used by people outside academia. Therefore, it is of great interest to be able to provide a tool that gives certain interpretability of the models' decisions as well as some insights about the nature of the data. In fact, these are attributes of particular interest, if not demanded, for decision making in safety-critical applications [25].

1.2. Suitable approaches and limitations

To meet the goals stated one can think of unsupervised learning techniques as a possible way to approach this. Especially, when it comes to reveal insights about the nature of the data, Representation Learning approaches such as autoencoders come in handy. Autoencoders are models designed to reconstruct the input data while learning a compressed representation of it, the so-called latent space. Their applications are quite widespread in anomaly diagnosis, being the most common case that in which the probability distribution of non-anomalous data is learned in order to detect, through reconstruction errors, patterns that do not correspond to that distribution [26–28].

The key element of these models is that their performance is based directly on internal representations, which in turn can be used to better understand the problem itself. Accordingly, they have been used to identify anomalous elements within a set of systems with similar characteristics, such as fleets of vehicles or aircraft engines [29,30]. In these cases, the compression capacity of the autoencoder is exploited, thus enhancing the interpretability of the latent space: assuming that anomalies are infrequent, those points in the latent space furthest away from the most populated clusters can be identified as such. However, the main limitation is that, unlike other dimensionality reduction methods like PCA [31,32], the relative distances between the input patterns are not necessarily preserved in the projection of the encoder, therefore this cluster analysis is not always possible. This problem was solved with Variational Autoencoders (VAEs). In VAEs, variational inference is added to the error function through the Kullback–Leibler term, which guarantees that data with similar patterns will also be encoded nearby in the latent space. VAEs are a recent but well-known alternative with numerous applications in anomaly analysis [33–35].

The problem of determining the RUL of a system, on the other hand, has been studied in less depth. This problem has many points in common with the diagnosis of anomalies and has also been solved by using autoencoders [36,37]. Despite these similarities, both problems have a fundamental difference: in anomaly diagnosis, the aim is to look for individuals in unlikely areas of the latent space. In RUL prediction, on the contrary, the objective for a complete and interpretable diagnosis should be to project the evolution of the system in the latent space over time in order to know how fast it is moving towards anomalous zones. The presence of anomalies is indeed correlated with the RUL since anomalous latent states usually correspond to low RUL values. However, two systems can be in the same initial condition but have a different temporal evolution, so that the successive states of the system cannot be studied independently as is done in anomaly detection. Instead, RUL estimation must be linked to the temporal analysis of complete state trajectories in the latent space.

In this line of research, we have recently proposed a new VAE architecture where the input and output layers are recurrent [38], as VAE applications are mainly oriented to the image domain and not so much to time series data, which is the case of RUL estimation. This architecture allows obtaining projections of state sequences and solves to some extent the problem of applying VAEs to RUL estimation since variational inference guarantees that systems with similar degradation patterns are going to be projected in close areas of the latent space. The recurrent VAE thus allows differentiation of systems with anomalous trajectories, however, this method is not a complete solution to the RUL estimation problem, mainly for two reasons:

1. It does not produce a numerical estimation of the system lifetime. It only separates low RUL systems from high RUL systems, but does not quantify what the RUL value is at each time step.
2. There is no guarantee that the time evolution of the trajectory projections are correctly separated (see Fig. 1), so it does not provide a solution to the problems of fleet health prognosis.

Concerning the second reason, it should be noted that RUL estimation, in real-world cases, is an online process: the useful life of each system is continuously updated as new data is received. For this reason, it is not enough that the new points are located in the vicinity of the previous ones: the successive projections of each system in the latent space, as time progresses, must form a continuous trajectory, which can be extrapolated into the future. In this way, it will be possible to diagnose continuous degradations over time (such as wear, efficiency losses, etc.) that affect the RUL, but which do not correspond to occasional events and therefore cannot be identified by anomaly detection analysis.

In this study, we solve the two open problems mentioned above by the combined use of a new neural architecture based on a recurrent variational encoder and a fresh way of regularizing training. To this end, we propose a new cost function related to the association of the Kullback–Leibler term with a second term that favors that the projections of successive states of the engines in the latent space constitute a continuous trajectory. This second term, as will be further explained, penalizes the successive RUL prediction errors over time, having a positive influence both on the ability of the new network to predict the lifetime of the engines and on the quality of the latent space. Thus, we take full advantage of the use of novel recurrent network architectures without giving up Representation Learning properties due to the construction of a latent space with suitable properties to provide a visual, hence explainable and interpretable diagnosis. The method is first validated with the popular C-MAPSS dataset from NASA and subsequently tested on a real environment.

The structure of this paper is organized as follows: Section 2 introduces the settings carried out to approach this problem. A detailed description of the proposed method for achieving an explainable diagnosis of aircraft engines is described in Section 3. The experimental set-up is explained in Section 4. Experimental results concerning both a benchmark problem and a real-world problem are presented in Section 5 while conclusions are drawn in Section 6.

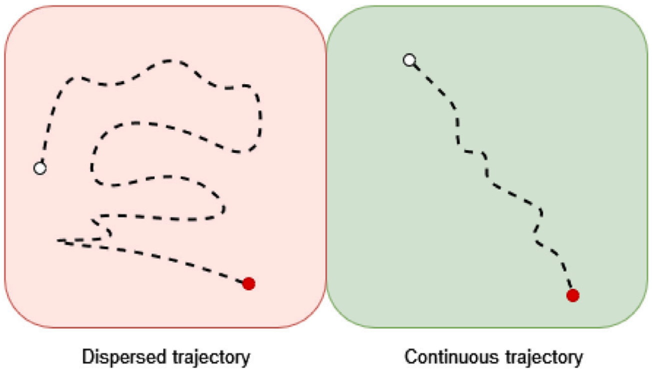

Fig. 1. In a vanilla VAE, training is regularized to prioritize generative purposes. This results in a dispersed latent projection of the system trajectory as in the figure on the left, in which there is no clear evolution between the state of the equipment at the beginning (white dot) and at the end of evaluation (red dot). On the contrary, a projection like the one in the figure on the right is what we are aiming for.

2. Problem settings

Before delving into the details of the model and results, it is of particular interest to highlight some important issues which are of great impact in achieving optimal performance and will help to better understand the problem itself.

2.1. RUL estimation

RUL stands for Remaining Useful Life and is a popular metric in prognostics, especially in aircraft monitoring [39]. Normally, sensors such as turbine pressure or compressor temperature are used to collect flight information about the engine. This data form a multi-valued series. The jth element of the series is a vector of h elements, each of which is the reading of one of the available sensors taken at the jth time instant. Having this information for several engines, a dataset could be formed from which to train a model to estimate the number of remaining time cycles in which a new unseen aircraft works well before failure, i.e its RUL.

In this paper, the proposed method is evaluated on the popular NASA's engine degradation dataset [40], known as C-MAPPS. Although it was published several years ago, it is still relevant today, (perhaps motivated by the fact that there are hardly any other similar datasets in the field), being the standard problem on which to test new RUL estimation models. This dataset contains simulated data of Turbofan engines produced by Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), a model-based simulation program. It is composed of multivariate temporal data obtained from twenty-one sensors and is further divided into 4 sub-datasets. As can be seen in Table 1, in each sub-dataset a training set and a test set are provided, from which there is a slight difference. The training set comprises run-to-failure data. That is, although each engine unit starts with different degradation states that are unknown, these are considered healthy and as time progresses, the engine units degrade to failure, therefore the last data sample corresponds to the time cycle in which the engine unit is declared unhealthy (RUL = 0). On the contrary, sensor records in the test sets terminate at some point before system failure and the actual RUL value for these engines is provided. The aim of this problem is to estimate the RUL of each engine in the test sets. It should be noted that training on a particular sub-dataset might be not applicable on another sub-dataset because the operating and failure conditions are different. There are promising approaches such as [41] in which adaptive methods are adopted to avoid these differences between training and test sets and thus avoid offline training. However, this is out of the scope of this work. Since there are four different sub-datasets, we train our model on each training set and evaluate on the test sets as they have exactly the same conditions.

Table 1
Data sets details.

	FD001	FD002	FD003	FD004
Train trajectories	100	260	100	249
Test trajectories	100	259	100	248
Operating conditions	1	6	1	6
Fault conditions	1	1	2	2

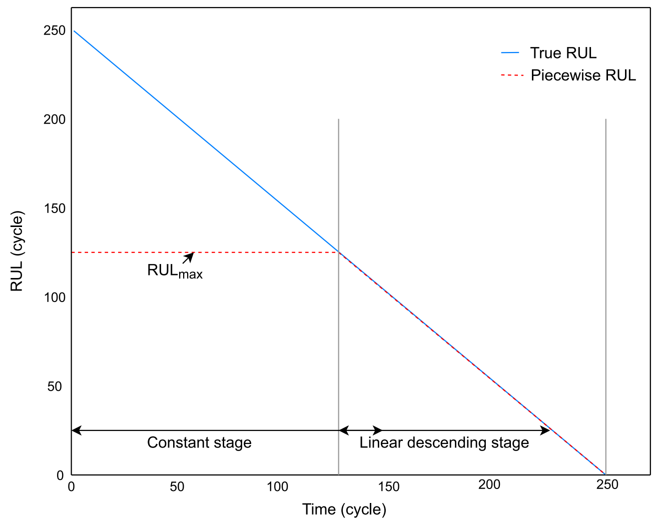

Fig. 2. RUL target function.

2.1.1. RUL target function

In prognostic problems, as the system always tends to deteriorate, it is quite usual to assume degradation behavior. Thus, a target RUL can be constructed based on these assumptions to guide the model training and enhance its predictions in a supervised manner. The most naive approach would be to assume that RUL decreases linearly over time, however, when analyzing the sensor signals, there is a common pattern: many sensors seem rather constant at the beginning until a breakpoint occurs that makes the engine degrade linearly with usage. The piecewise linear degradation model proposed in [42] follows this idea and is the most extended target function used in the literature. It simply limits the maximum value of the RUL function as illustrated in 2. We use this degradation model to obtain the RUL label with respect to each training sample at each time-step. The maximum RUL is set at 125 cycles. This is used to make fair comparisons with respect to other models that used the same methodology, but it should be noted that this is just a guideline value. Different equipment in the system has different lifetimes and different degradation trajectories, therefore this value may be too high or too low for different individuals. In [43] the authors propose a new methodology to construct the target RUL for each individual in order not to rely on a single value. However, there is still no consensus on the best way to teach the algorithm the behavior of the system. Precisely, this can be considered a bottleneck and that is why it is desirable to provide learning that does not depend exclusively on this function. In this work, we learn the nature of the data in an unsupervised manner with variational inference and fine tune it with the labels to improve predictions. Thereby, what the model learns is guided more by the nature of the data than by the labels themselves.

2.2. Metrics

In order to establish a fair comparison with the rest of the approaches the same metrics used in most similar works are chosen. On the one hand, there is the original metric proposed by NASA in PHM 2008 Data Challenge, which is described in Eq. (1), where N is the number of engines in the test set, S is the computed score, and d = (Estimated RUL - True RUL).

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

The main objective of this function is to differentiate late predictions from early predictions. The former are more penalized because it is understood that it is too late to perform maintenance while early predictions are not a major problem. Although maintenance resources could be wasted, priority is given to penalizing false negatives. This has some drawbacks since, if there is an outlier leading to a late prediction, this would dominate the overall performance score, thus masking the true overall accuracy of the algorithm. In addition, the level of confidence with which the algorithm is able to estimate the RUL value before the failure point is also not taken into account.

Due to these shortcomings, the use of RMSE is also proposed as it gives equal weight to early and late predictions:

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

The use of RMSE together with the scoring function (Eq. (1)) would avoid favoring an algorithm that artificially lowers the score by underestimating, which is quite likely due to the reasons exposed, by resulting in a higher RMSE. In summary, both metrics complement each other by providing more information about the accuracy of the model.

3. Model

The proposed model consists of three components: an encoder network, a regression model and a latent space. The encoder learns to compress the data into the latent space so that it is described by the parameters that initialize the probability distribution to which the data belongs. Variational inference is added to the loss function through the Kullback–Leibler divergence, which measures how much one probability distribution diverges from another, to learn the above-mentioned parameters. A second term is also added to penalize wrong estimations of the regression model. In the end, all this allows a latent space to be learned in which similar data is located in nearby areas from which to efficiently perform other tasks.

The workflow followed for this problem is depicted in Fig. 3. The model is trained with data from aircraft engines to learn a simplified representation of their trajectories. Thus, the resulting encoder acts as a feature extractor compressing into the latent space the data according to their properties, which are different stages of degradation in the engines. The latent space contains the compressed representation of the aircraft, particularly in 2 dimensions. This representation will be used after training is completed to visually evaluate the degradation patterns of the engines. Finally, a numerical prediction of the RUL that best represents each engine that is fed to the model is provided by training a regression model directly with the features learned in the latent space. This section explains the main differences between our model and a VAE and how the encoder output can be leveraged to create the visual diagnosis we propose. Emphasis is placed on the implementation of the recurrent architecture for dealing with time series, as well as how latent features lead to perform RUL estimation.

3.1. Variational encoding

Variational encoding refers to the process of compressing input data based on variational inference, a key element in our research, as stated in the introduction. This process is the basis for the operation of VAEs, therefore it is important to know how they work in order to clarify the differences with respect to our model. In a VAE the training process is regularized to avoid overfitting and to ensure that the latent space has the necessary properties that enable the generative process. To obtain them, the encoder must map the data in the latent space in such a way that similar data is close to each other. This allows the decoder not only to reconstruct the data efficiently but also to generate new instances from points in the latent space that do not correspond to the encoding of any training sample.

VAEs compress the input data into a latent vector, which is a simplified representation, described as $p(x) = p(x|z)p(z)dz$, where the domain of z is continuous and therefore intractable. For this reason variational inference is used since this intractability can be solved via the lower bound of the log-likelihood [44], $\mathcal{L}_{vae}$,

$$\mathcal{L}_{vae} = E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) \tag{3}$$

The first term is the reconstruction of x that tends to make the coding–decoding scheme as efficient as possible by maximizing the log-likelihood $\log p_\theta(\mathbf{x}|\mathbf{z})$ with sampling from $q_\phi(\mathbf{z}|\mathbf{x})$, modeled by the encoder, whose output is the parameters of a multivariate Gaussian: a mean and a diagonal covariance matrix. In other words, the main goal of the encoder is to map the input data into a lower-dimensional space, acting as a feature extractor. The second term tends to regularize the organization of the latent space by causing the distributions returned by the encoder to approach a standard normal. It regularizes the latent variables (represented by z) by minimizing the KL divergence between the variational approximation and the prior distribution of z.

The data is reconstructed from the conditional probability distribution p(x–z), learned by the encoder. For generative purposes, the regularization produced in the latent space facilitates random sampling and interpolation for the creation of new data. This is why VAEs are understood as generative models and their use is widespread as such.

Nevertheless, we do not strive to generate new aircraft data, but to diagnose it by making use of latent representations. VAEs latent space, in contrast, is not usually used for clustering or visualization despite it has promising properties for this. In fact, there are works in which this has been taken advantage of, as in [45] in what they refer to as the latent-feature discriminative model. The authors trained a VAE and then fed a classifier with the outputs obtained from the resulting encoder. Still, this is not further explored in the literature since VAEs are mainly oriented to generative tasks and this causes the regularization of the latent space to lead the encoder to project the data as compressed as possible, resulting in obvious overlaps.

This is a barrier to our objectives because these overlaps make it difficult to estimate the RUL. First visually: although aircraft with similar RUL values will be close on the map, they will not be clearly differentiated from those that are far away. Then, because any model built on top of this will be guided by this representation and will most likely result in prediction failures. Therefore, a vanilla VAE does not meet our needs and we must adapt the use of variational inference for our problem: what we want is to enhance the latent organizational properties of variational encoding by using a regressor, not a classifier, and not once the model is trained, but while it is being trained.

The image on the left in Fig. 4 represents why the approach mentioned in [45] is not suitable for the problem at hand. It corresponds to the latent space the encoder learns after training the model without any restrictions thence the regularization of the latent space for the generation of new data is prioritized. This causes the input data to be placed in areas where instances whose features are not similar (different RUL values) are not clearly differentiated or even overlap. As this approach suggests, a simple regressor was trained on the frozen encoder weights, however, this overlap severely penalizes the performance in RUL estimation, making it unable to compete with other state-of-the-art methods. There are promising works that propose to solve this by including regression errors in the training process as in [46], although

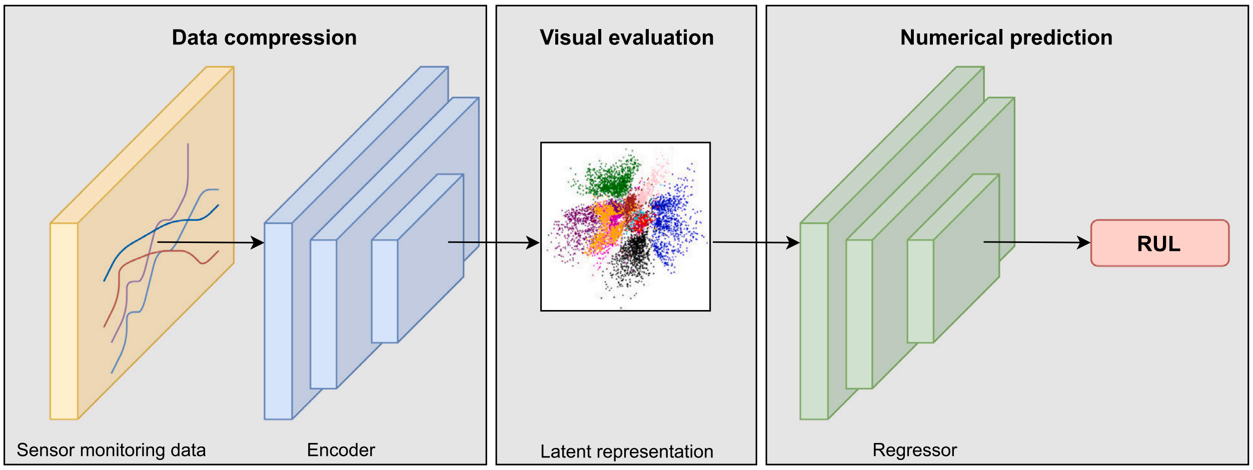

Fig. 3. Workflow followed for the proposed approach: aircraft data is fed into the encoder, which learns a latent representation based on deterioration patterns in order to build a graphical map reflecting the evolution of their trajectories. The regressor directly influences obtaining such a latent space and allows to report numerically the RUL of each engine.

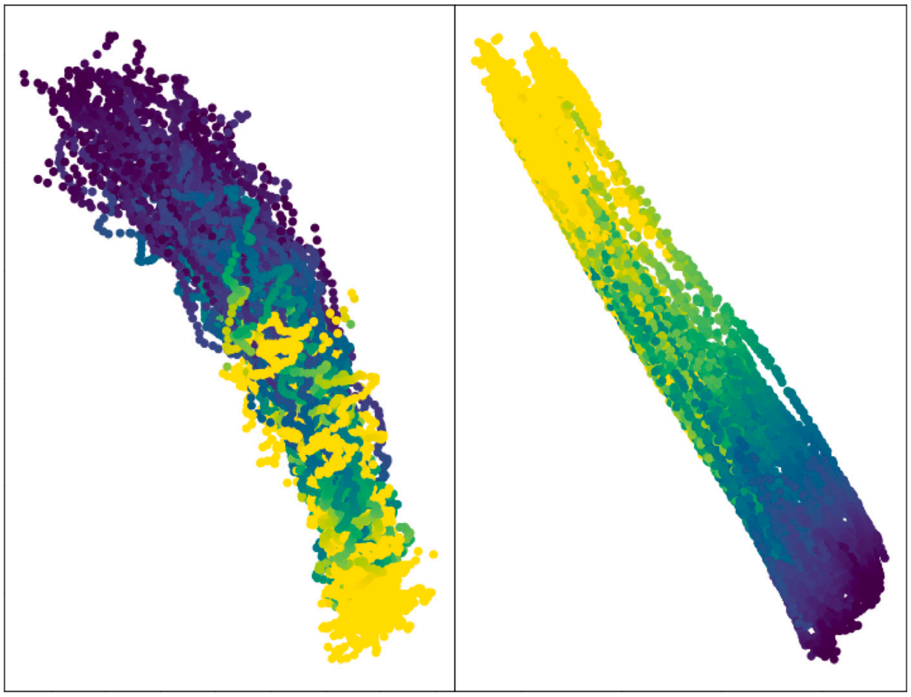

Fig. 4. Latent representations learned by the encoder for FD001. The figure on the left shows the regular training of a VAE, while the figure on the right shows the result with our model, which does not include the decoder but a regression model that adds a penalty for wrong predictions.

the decoder is still used, which may wrongly interfere with our ultimate goal: the diagnosis of the aviation history of the engines.

Instead, the path we decide to take includes the omission of the decoder to focus learning on obtaining an interpretable latent space. Thereby, the main difference with respect to a VAE is that we replace the decoder with a regression model, as shown in Fig. 3, and the training is done differently. Our proposed model is trained to minimize a loss function composed of two objectives:

$$\mathcal{L}_x = -D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + RMSE \tag{4}$$

The first objective corresponds to the regularization of the latent space through variational inference, as explained before in Eq. (3) and the second one is the Root Mean Square Error (RMSE) between the known RULs and the RULs predicted by the model. Including the regressor optimization in the loss function adds a constraint to the model, as it will strive not only for a continuous latent space, but also for a space in which different types of trajectories, and so with different associated RUL values, are sufficiently separated to be clearly differentiated so that the evolution of degradation in an aircraft can be observed. The right part of Fig. 4 demonstrates the effectiveness of training the model in this way. The architecture of the regressor is simple: on top of the encoder base, a fully connected layer with a tanh activation function and another layer with a single neuron containing the RUL prediction are added.

As for the encoder, we decide to implement it with recurrent networks given that most recent studies make use of them to model the time complexity of historical aircraft data [47,48]. Among the different types of RNNs that can be found, LSTM networks are the most popular. These networks process data from backward to forward conserving information from the past through hidden states. However, Bidirectional LSTM networks are in high demand because they provide not only information about the past but also about the future: data is first processed from past to future and then from future to past, thus preserving the information from both periods. This is quite helpful because the network is aware of what the data may look like in its future stages, which helps it to understand what kind of information to predict (different stages of engine deterioration). All in all, we decide to implement the model with this type of network.

In summary, the encoder, built with bidirectional LSTMs, approximates the Gaussian distribution $p_\theta(\mathbf{z})$ by feeding the output into two linear modules to estimate its mean and covariance. This means that the compression of the engine data by the encoder results in a two-dimensional latent space whose axes would be the mean and covariance of the approximate distribution. Consequently, the learned latent space is expected to group engine trajectories into different clusters according to their underlying nature, illustrating a simplified representation. Furthermore, the regressor influences directly over the organization of the latent space and can report explicitly which RUL value is the one that best represents the cycles belonging to each engine unit that is fed to the model.

3.2. Interpretable diagnosis

The diagnostic tool introduced in this work is a map that shows the actual state of the engine and also the rate of change from healthy to deteriorated. What we pursue is a map in which each point represents the status of an engine associated with a window of events during its flight history so that points of degraded aircraft are grouped in nearby areas and, on the contrary, points belonging to healthy aircraft are located in more distant areas. As the actual health status of the training aircraft will be known, since we used the RUL target function, a color scale can be established to clearly differentiate healthy aircraft from deteriorated or totally deteriorated aircraft, coloring each point according to its corresponding RUL. Thinking about how variational inference works, this can be easily put into practice: once the model has been trained with the engine data, each input can be encoded into the latent space, being represented in terms of the mean and variance of the approximate distribution learned. This means that the data can be projected into the latent space and each point will be clustered near those with a similar degradation pattern. An example of one of the maps produced by this algorithm is shown in Fig. 4, right side. Aircraft with high RUL values are painted in yellow while aircraft with low RUL values are painted in dark purple. It can be observed that there is a clear progression in the colors along the map since events with no or low deterioration are located in the upper part of the map (high RUL values) while the most deteriorated ones are located in the lower part (low RUL values). This representation can be used later: when new unseen engine events are used as inputs, the encoder will place them according to their characteristics, giving information about their RUL depending on the proximity to other nearby points whose diagnosis is known. This is why it is considered explainable, since the method's decisions are based entirely on the learned representation and can therefore be justified; and interpretable, because a simple glance at the map gives insight into the status of each engine unit. Other Deep Learning methods can also reveal interpretable information in intermediate layers, however, extra processing is needed in order to find the most suitable layers or to transform the content of these layers into human readable information. An example of this is the embedding projector of tensorflow [49], which applies different dimensionality reduction methods such as UMAP, T-SNE or PCA to provide a visualization of the embedding layer. In contrast to this, our method provides a direct 2-D compression, which does not need any further processing. More details on the interpretation of this map are given in the following section.

4. Experimental design

Table 1 shows the different levels of difficulty of the datasets according to the last two rows. Each dataset can operate under different operating conditions and the system failure can be caused by two components: the turbine and the compressor. Thus, FD001 and FD003 operate under the same conditions although FD003 includes engines whose failure could be caused by either of the two mentioned components. Then, FD002 operates under 6 operating conditions as does FD004, while in FD004, as in FD003, the failure conditions cover both turbine and compressor failure. In this sense, it is believed that according to their characteristics, the level of difficulty of the datasets, in increasing order, is: FD001, FD003, FD002, FD004. Some studies focus on a particular dataset [50,51] and others explore in detail the impact of different hyperparameters such as the number of sensors to use or the upper limit for the target function for each dataset [43]. Still, this is a benchmark problem and the interest lies in finding a pre-processing procedure that can be applied to similar problems, rather than finding the ideal series of steps for a particular dataset.

For this reason, the decision we make, since the different failure conditions do not have a major impact on pre-processing, is to focus on those samples where the operating conditions are different. In those cases, even a simple exploratory analysis would yield little or no information concerning the signals because apparently operating conditions change between cycles, which makes analyzing and predicting RUL much more complex. It is important to take this into account when normalizing data, although it is something that seems to be overlooked in other papers since min–max normalization is usually used [15]. Instead, we take another path by using a condition-based standardization. With this approach, all records of the same operating condition are grouped together and scaled using a standard scaler. The application of this type of scaling will bring the average of the grouped operating conditions to zero. As this technique is applied for each operating condition separately, all signals will receive an average of zero, making them comparable [52].

On the other hand, although sensor data have a general trend, it is known that they are subject to local oscillations, mainly caused by high-frequency sensors, which lead to noise [23,43]. To ease the processing of the series, an exponential weighted moving average is carried out. It takes the current value and the previous filtered value into account when calculating the filtered value:

$$X'_j = \alpha * X_j + (1 - \alpha) * X'_{j-1}$$

where X'_j is the filtered value of X_j and α the strength of the filter. Lower values for α will have a stronger smoothing effect and consequently, stationarity is lost. Nevertheless, stronger smoothings lead to better model performance. It is important to note that what we intend to model is not the detection of failure points, but the changes in the degradation rate, i.e. those breakpoints where after some time, the engine parts deteriorate at a different rate than they did before. For this reason, the smoothing we apply does not adversely affect the data. Furthermore, the sole purpose of the filter is to reduce oscillations in the sensor measurements, therefore in no case is smoothing applied that would compromise the trend of the data.

In time series problems it is quite recurrent to split the data into sequences for better prediction performance. That is, multivariate series are not processed for each engine but are sliced into fixed-size windows as shown in Fig. 5. At each time step, data is picked from sensors within the time window to form a high-dimensional feature vector used as inputs to the network to predict the RUL. Thus, each input sample in our network contains thirty single-cycle data which is extracted from the following six sensors: T30, T50, P30, EPRA, PS30, phi and the aim is to find patterns in those time-windows that can lead to an adequate RUL estimation. There may be cases in which the partitioning of the sequences for a particular engine in the last few cycles may not have enough data to complete the length of the window. In those cases a

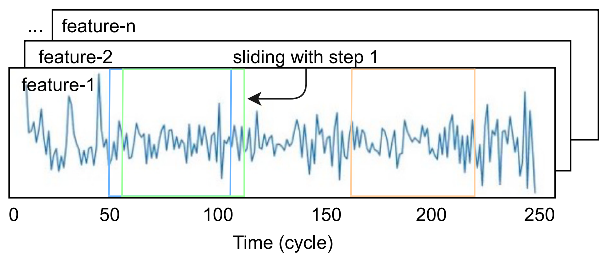

Fig. 5. Time window framing.

masked value is used and will be treated in the first layer of the model by simply ignoring those values. In this way, as much information as possible is used.

In the experiments performed, although there are not so many hyperparameters to adjust, the impact of those that are present is very noticeable in the final performance of the model. Thus, the time window length, the intensity of smoothing or the internal number of neurons of the recurrent layers are key elements. Hyperparameter tuning is an arduous task that in this case was driven by our own experience along with the use of the Hyperopt Bayesian optimization library [53] to find the final configuration. On the other hand, an accurate choice of the LR is particularly essential to improve the optimization process, therefore for this parameter an adaptive LR optimizer, Adam, and the Cyclic Learning Rate technique proposed in [54] were used to help to select the optimal LR with which to start the training. In addition, to achieve the best possible performance of our model, we used callbacks to customize our experiments in terms of relegating the training stop condition to the validation error instead of the number of epochs. These tweaks led us to find the best results for the functions to be optimized.

The choice of the sensors is not arbitrary, we only use the following six: T30, T50, P30, EPRA, PS30 and phi, which are precisely the ones available in the real problem we introduce later on. Note that in both datasets the engines are Turbofan aircraft engines. Surprisingly, we found that out of the twenty-one provided sensors, of which most of them are used in similar works, using only these not only reduces computational costs but also gives sufficient information to predict the RUL efficiently. Moreover, for datasets FD001 and FD003 the EPRA sensor is not necessary since it measures the engine thrust under different operating conditions while FD001 and FD003 operate under the same condition and so this sensor does not provide relevant information, the captured values simply remain constant.

Finally, 20% of the training data was used for validation, resulting in 17692 training samples and 4128 validation samples for FD001 and FD003, 37432 training samples and 8787 validation samples for FD002 and finally 43523 training samples and 10505 validation samples for FD004. All models and experiments were implemented in TensorFlow. Further details regarding the experimental setup and the source code to reproduce the experimental results are available in the following public git repository: https://github.com/NahuelCostaCortez/Remaining-Useful-Life-Estimation-Variational

5. Experimental results

The experimental validation of the proposed framework has two parts. First, C-MAPSS datasets are used to compare our framework with other state-of-the-art approaches for RUL estimation. Secondly, C-MAPSS engines as well as actual engines from the real problem we present are diagnosed based on their projections in the latent space. We begin by introducing the numerical results, the diagnostic map achieved is then presented and finally, the experimental validation is discussed.

5.1. Results on C-MAPSS

In this section we demonstrate that our framework can compete with state-of-the-art methods for RUL estimation. Both the training and test sets used are the same for all methods, since both sets are provided in the original dataset, as stated in Table 1. It is worth mentioning that the baseline methods we present, which collect the most impactful approaches to date, do not provide any representation of the data, but merely predict the RUL corresponding to the next time step. This makes us appreciate the importance of Representation Learning as it provides a piece of more illustrative information than a simple numerical or categorical result.

The comparison results of the proposed framework with other popular approaches on the test sets are listed in Table 2 where the selected metrics of all methods, included ours, labeled as RVE (Recurrent Variational Encoder), are listed for every dataset. Results in which our method outperforms the others are highlighted in bold. It can be quickly noted that with datasets FD001 and FD003, although the metrics are considered good, they are not the best. However, the interest lies mostly in FD002 and FD004 as the increasing number of operating conditions and failure modes make these two datasets contain more complicated multiscale degradation features. RVE significantly improves prediction accuracy in these two for both Score and RMSE, due to its good feature extraction capability in the face of these complex fault prediction problems. The comparison also includes a row labeled "VAE+RNN", which corresponds to the adaptation of a recurrent VAE to this problem. The superiority of our model can be clearly seen. Although both use variational inference, the numerical differences are explained by the different latent spaces obtained: one dispersed and the other one continuous (recall Figs. 1 and 4), allowing the latter to improve the predictive capabilities of the model.

RUL estimations for the life-time of some testing engine units corresponding to the different datasets are shown in Fig. 6. It is very common to see figures like these in papers working with C-MAPSS, exhibited to obtain an understanding of the model's performance. The RUL constructed from the piece-wise function is represented in orange, of which C-MAPSS provides the RUL corresponding to the last cycle. RUL values predicted at each time instant by our method are presented in blue. It is clearly seen that the network is able to model this degradation to, finally, accurately predict the real RUL of the engine. However, this is not enough to explain the performance of the model and this is where we differ from other methods.

These kind of figures seem very clear and promising but despite being good predictions, there is a gap when it comes to explainability of the model´s decisions and internal representations. A gap that can be filled with techniques such as the one we propose. As explained in Section 3, the latent space build by the encoder serves as a basis for creating a map that allows us to understand the evolution of the data over time and Fig. 7 is a sample of this. Each map represents the latent space obtained for the set of cycles traveled by each aircraft shown in Fig. 6. That is, for example, for plane #7 the compressed representation of the first thirty cycles corresponds to the first upper left red dot, while the compressed representation of the last thirty would be the last lower right red dot. The remaining points correspond to the representations of each data sample seen during training. The encoder learns to locate in the latent space each data window passed to it according to its characteristics. Thus, in all the exposed maps, in which the RUL is labeled in the color bar, it can be seen that when the airplane is operating in favorable conditions (high RUL values), its latent representation is located in the upper left zone and, as it begins to degrade, this location moves to the right until the data indicating that the airplane is degraded (low RUL) are located in the lower rightmost area.

In this way, a model that can be fed with data from the trajectories of an aircraft that has flown at least thirty cycles is achieved. From there, the model can be fed each time a new data sample is available

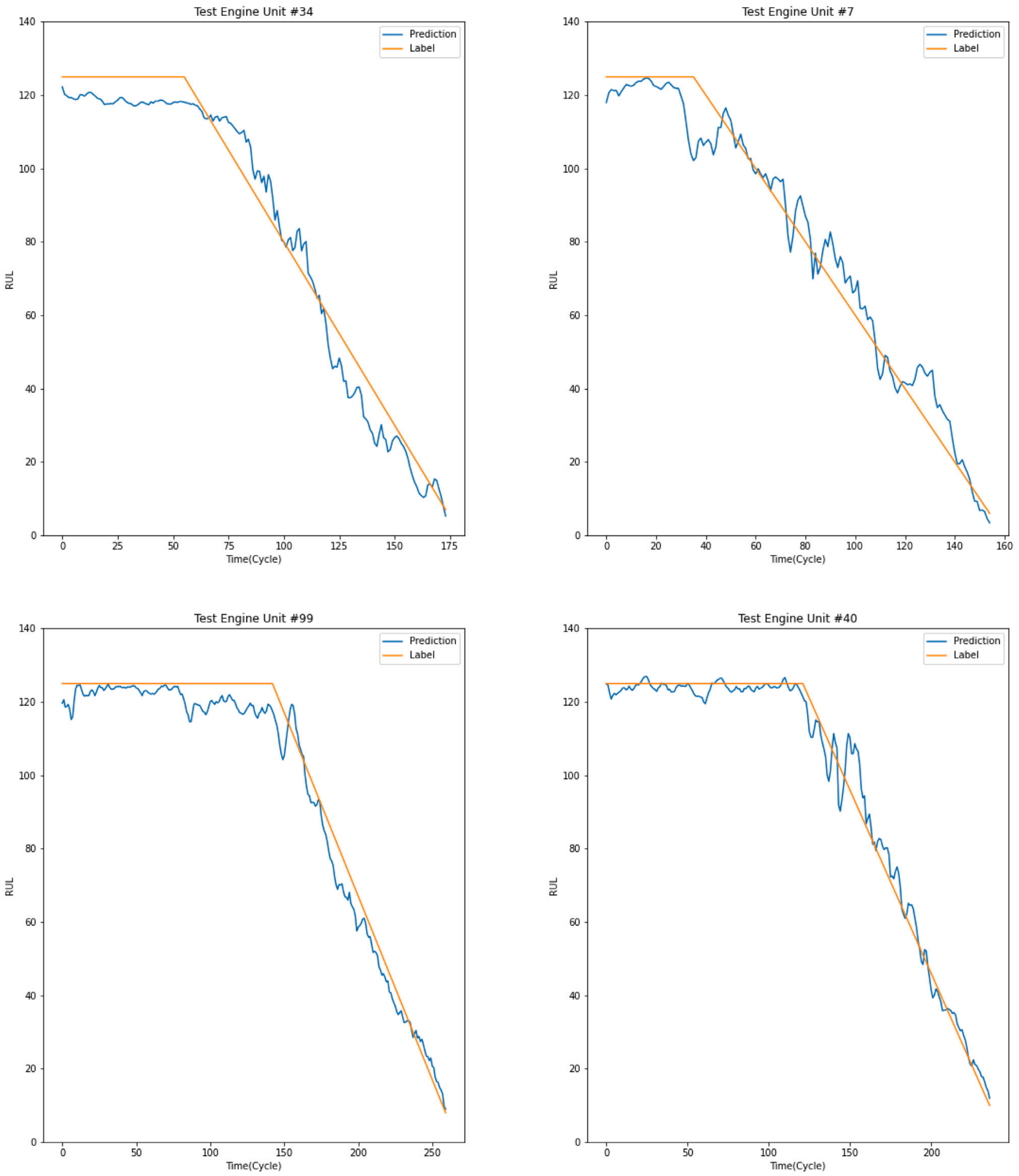

Fig. 6. Four examples of life-time RUL predictions for testing units: #34 corresponding to FD001, #7 corresponding to FD002, #99 corresponding to FD003 and #40 corresponding to FD004.

in order to obtain the two diagnostics we are looking for. First, a visual diagnosis is presented in which, based on the proximity to samples whose condition is known, the state of the aircraft at that particular moment is perceived. In https://github.com/NahuelCostaCortez/Remaining-Useful-Life-Estimation-Variational/tree/main/images/gifs there are some gifs available corresponding to the engines from Figs. 6 and 7 in which the speed of deterioration suffered by these airplanes along the cycles can be appreciated. Second, a quantitative diagnosis is obtained that explicitly reports the RUL value that determines the remaining life time of the aircraft.

5.2. *Results on a real-world problem*

To illustrate how the proposed model may work in a more realistic context, an example for actual engines is presented below. The data is sampled on Turbofan engines under actual conditions of use. Unfortunately, for confidentiality reasons, we are unable to disclose the

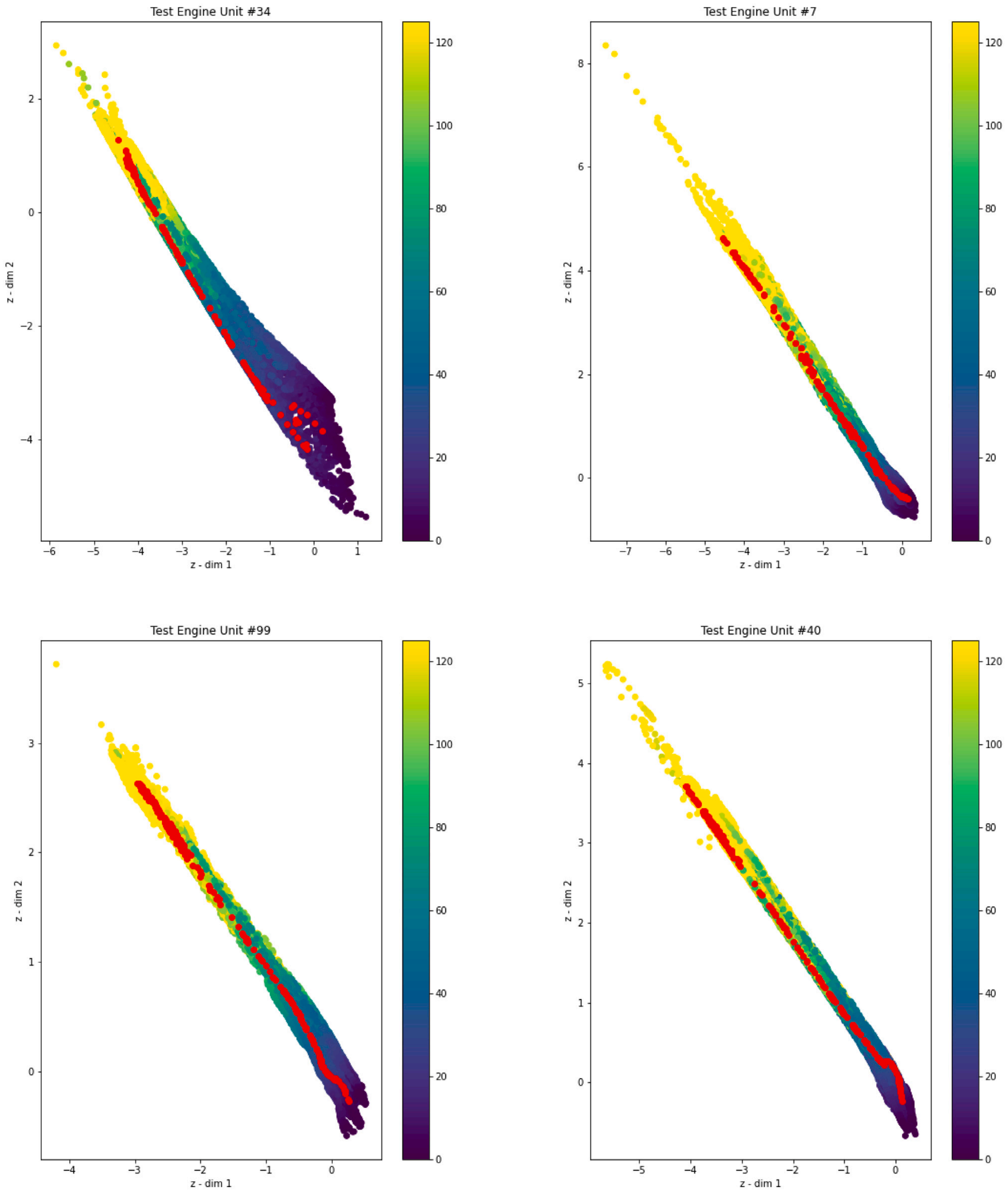

Fig. 7. Latent predictions for every time-step of the samples presented in 6.

name of the company or make the data set public. Nevertheless, a brief description of the engines is provided in Table 3.

The pre-processing applied is the same as the one explained in Section 4. The objective is to convert the data from each engine into inputs that can be processed by the network. During training, as in the NASA´s dataset, the model learns different deterioration patterns which leads the encoder to project the engine units into the latent space according to their degradation, maintaining coherence in the distances between healthy and compromised engines. This projection is again used as a basis to find out, given undiagnosed units, how their degradation evolves as the number of flights increases. Fig. 8, as the above-mentioned gifs, pictures this idea: six airplanes have been chosen to project their state into the latent space in two different time steps: t = 0 would correspond to feeding the network with the data corresponding to the cycles from 0 to windows length and so it is the same with t = 1000, starting from data corresponding to the cycle 1000.

Table 2
Evaluation metrics of different approaches for RUL estimation on C-MAPSS datasets.

	FD001		FD002		FD003		FD004	
	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
MLP [17]	37.56	18 000	80.03	7 800 000	37.39	17 400	77.37	5 620 000
SVR [17]	20.96	1380	42.00	590 000	21.05	1600	45.35	371 000
RVR [17]	23.80	1500	31.30	17 400	22.37	1430	34.34	26 500
CNN [17]	18.45	1299	30.29	13 600	19.82	1600	29.16	7890
Deep LSTM [17]	16.14	338	24.49	4450	16.18	852	28.17	5550
Semi-supervised [23]	12.56	231	22.73	3366	12.10	251	22.66	2840
DCNN [55]	12.61	273.7	22.36	10 412	12.64	284.1	23.31	12466
MS-DCNN [55]	**11.44**	**196.22**	19.35	3747	**11.67**	**241.89**	22.22	4844
VAE+RNN	15.81	326	24.12	4183	14.88	722	26.54	5634
RVE	13.42	323.82	**14.92**	**1379.17**	12.51	256.36	**16.37**	**1845.99**

Table 3
Summary of the main properties of the engines provided by the manufacturer.

Model/properties	Type I engine	Type II engine
Thrust ratings	between 14,750 and 15,000 lb	between 14,000 and 21,500 lb
Flying hours	11m+	11m+
N° fans	1	2
Fan diameter	48 in.	58 in.
Two-shaft, high-bypass-ratio engine	Yes	Yes

The RUL provided by the model is shown in the colorbar. Fixing the latent projection obtained after training gives us some insight into the progression of the health status of these units: The latent projection of engine e1, e2, e3 and e4 during the time steps shown remain over the upper left quadrant, next to other aircraft with similar characteristics: RUL around two hundred cycles, with no signs of near degradation. On the contrary, there is a clear progression in samples e5 and e6, which move clearly downward, being placed together with engine units close to their end of life (low values of RUL), thus obtaining an accurate and explainable diagnosis beyond a possible label indicating the predicted health.

In the figure presented only two time steps have been selected to show the update of the health status of the engines according to the data from their sensors. However, it is noteworthy that once enough data is available to be fed into the network in each subsequent trip this update can be performed because we are using recurrent networks. This is where the interest really lies because this update allow us coping with the non-stationarity of the data distribution and in the end this can be used as a diagnostic tool. As mentioned in the introduction, this is an online process, being the useful life of each system continuously monitored. Particularly in the company, these motors have periodic maintenance cycles and also have parallel systems that warn in case of detecting any anomalous operation. The fact of having a diagnostic system of these characteristics, however, represents an invaluable economic saving for the company. This is because the aim of the method is to prevent such anomalies from occurring. To this end, the degradation speed of the engines is modeled, so that the acceleration in the normal degradation speed of an engine can be easily detected. In this way, the probability of the aircraft having an unexpected event is highly reduced.

An example of interpretation of the method is as follows: Arrows were used in Fig. 8 to depict the evolution of each sample, but as demonstrated in the previous sub-section, a footprint of every step is recorded (that is, a latent projection is available for t = 1, t = 2, t = 3 and so on) so there is a clear evolution over time and the rate of these updates may trigger alerts in a real-world scenario: as long as the engine projection remains in the healthy range, its evolution will be considered positive; on the contrary, if the projection moves towards the red zone rapidly, it may be a clear sign of deterioration, information that will be used by the mechanics to make a decision regarding its follow-up, either to make it more exhaustive or to take the aircraft to the workshop for a more complete overhaul, to name some alternatives. This translates into a prolongation of the useful life of these engines by being able to anticipate the breaking point at which severe deterioration may occur.

6. Concluding remarks and future work

We have proposed a novel architecture based on variational encoding with a new way of regularizing latent representations to address aircraft engine diagnostics. These are obtained through variational inference and are shaped by a term in the cost function that penalizes erroneous RUL estimates. The result is a latent space capable of projecting the history of engines trajectories continuously and without abrupt jumps, like other models such as VAEs. As a consequence, the latent space learned by the encoder is used as a diagnostic tool. It learns a two-dimensional representation of engine data with different deterioration stages to, given an unseen engine, project its encoding near engines with similar degradation patterns. Thus, prevailing an explainable diagnosis.

We have demonstrated that, besides providing a visual assessment of the rate of degradation in aircraft engines, our method can also accurately estimate the RUL. To this end, we used the popular C-MAPSS simulation dataset on which we outperformed most of the current state-of-the-art methods. We have shown that the learned latent space can comprehensively model aircraft degradation history and consequently improve prediction capabilities. Furthermore, we include a report of its application to data belonging to actual engines to illustrate its performance in a real-world scenario.

Lastly, in future works we aim to explore the suitability of the model in other areas related to condition monitoring and predictive maintenance. Additionally, it would be of interest to motivate the model to learn latent features that, beyond differentiating the stages of degradation, can also explain the different causes of failure.

CRediT authorship contribution statement

Nahuel Costa: Writing – original draft, Validation, Supervision, Software, Methodology, Investigation. **Luciano Sánchez:** Writing – review & editing, Supervision, Funding acquisition, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been partially supported by the Ministry of Economy, Industry and Competitiveness (“Ministerio de Economía, Industria y Competitividad”) from Spain/FEDER under grant PID2020-112726-RB-100 and by Principado de Asturias, grant SV-PA-21-AYUD/2021/50994.

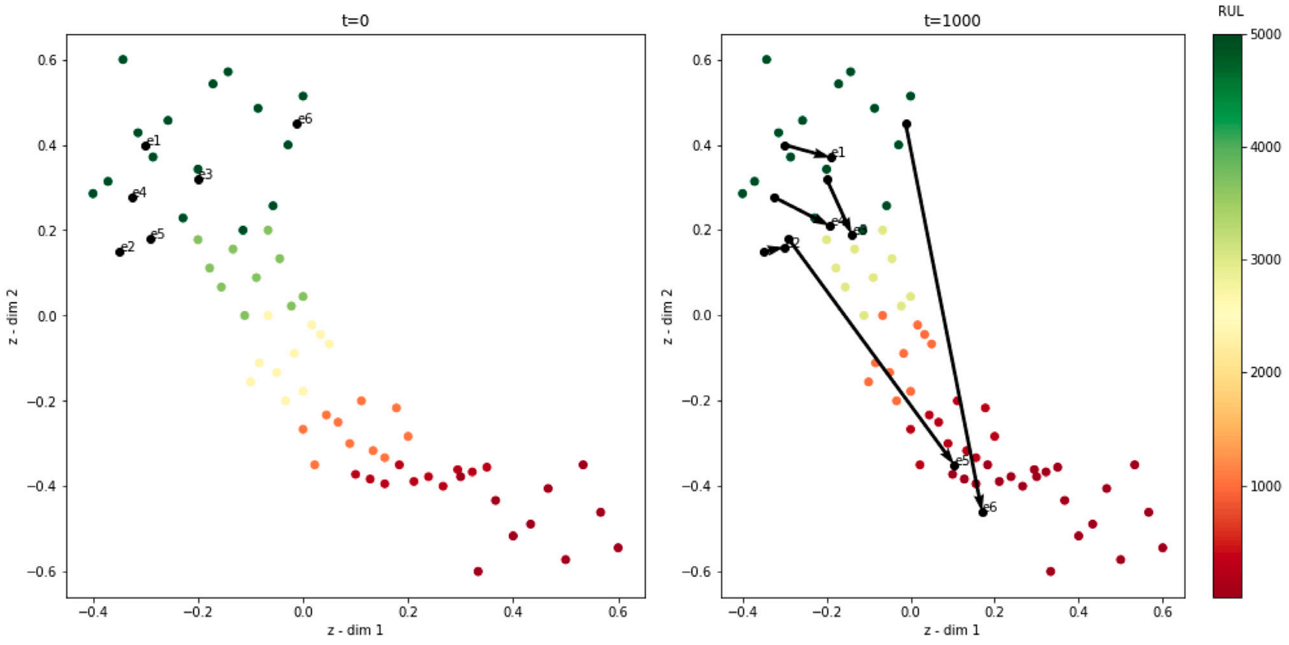

Fig. 8. RUL evolution of six selected engines. As the cycles progress, the aircraft are placed in areas close to other aircraft with similar degradation patterns whose diagnosis is known, thus making it easy to identify those that degrade more rapidly, as in the case of e5 and e6.

References

[1] Azadeh A, Asadzadeh S, Salehi N, Firoozi M. Condition-based maintenance effectiveness for series–parallel power generation system—A combined Markovian simulation model. Reliab Eng Syst Saf 2015;142:357–68.

[2] Zhao Z, Liang B, Wang X, Lu W. Remaining useful life prediction of aircraft engine based on degradation pattern learning. Reliab Eng Syst Saf 2017;164:74–83.

[3] Ali JB, Chebel-Morello B, Saidi L, Malinowski S, Fnaiech F. Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network. Mech Syst Signal Process 2015;56:150–72.

[4] Jouin M, Gouriveau R, Hissel D, Péra M-C, Zerhouni N. Degradations analysis and aging modeling for health assessment and prognostics of PEMFC. Reliab Eng Syst Saf 2016;148:78–95.

[5] Si X-S, Wang W, Hu C-H, Zhou D-H. Remaining useful life estimation–a review on the statistical data driven approaches. European J Oper Res 2011;213(1):1–14.

[6] Tian Z. An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring. J Intell Manuf 2012;23(2):227–37.

[7] Kwan C, Zhang X, Xu R, Haynes L. A novel approach to fault diagnostics and prognostics. In: 2003 IEEE international conference on robotics and automation (Cat. No. 03CH37422), Vol. 1. IEEE; 2003, p. 604–9.

[8] Zhang X, Xu R, Kwan C, Liang SY, Xie Q, Haynes L. An integrated approach to bearing fault diagnostics and prognostics. In: Proceedings of the 2005, American control conference, 2005.. IEEE; 2005, p. 2750–5.

[9] Khawaja T, Vachtsevanos G, Wu B. Reasoning about uncertainty in prognosis: a confidence prediction neural network approach. In: NAFIPS 2005-2005 annual meeting of the North American fuzzy information processing society. IEEE; 2005, p. 7–12.

[10] Zio E, Di Maio F. A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system. Reliab Eng Syst Saf 2010;95(1):49–57.

[11] Khelif R, Chebel-Morello B, Malinowski S, Laajili E, Fnaiech F, Zerhouni N. Direct remaining useful life estimation based on support vector regression. IEEE Trans Ind Electron 2016;64(3):2276–85.

[12] Singh SK, Kumar S, Dwivedi J. A novel soft computing method for engine RUL prediction. Multimedia Tools Appl 2019;78(4):4065–87.

[13] Xu Z, Saleh JH. Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. Reliab Eng Syst Saf 2021;107530.

[14] Babu GS, Zhao P, Li X-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. In: International conference on database systems for advanced applications. Springer; 2016, p. 214–28.

[15] Li X, Ding Q, Sun J-Q. Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab Eng Syst Saf 2018;172:1–11.

[16] Li X, Zhang W, Ma H, Luo Z, Li X. Data alignments in machinery remaining useful life prediction using deep adversarial neural networks. Knowl-Based Syst 2020;197:105843.

[17] Zheng S, Ristovski K, Farahat A, Gupta C. Long short-term memory network for remaining useful life estimation. In: 2017 IEEE international conference on prognostics and health management (ICPHM). IEEE; 2017, p. 88–95.

[18] Yu W, Kim IY, Mechefske C. An improved similarity-based prognostic algorithm for RUL estimation using an RNN autoencoder scheme. Reliab Eng Syst Saf 2020;199:106926.

[19] Shi Z, Chehade A. A dual-LSTM framework combining change point detection and remaining useful life prediction. Reliab Eng Syst Saf 2021;205:107257.

[20] Li J, Li X, He D. A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction. IEEE Access 2019;7:75464–75.

[21] Xia T, Song Y, Zheng Y, Pan E, Xi L. An ensemble framework based on convolutional bi-directional LSTM with multiple time windows for remaining useful life estimation. Comput Ind 2020;115:103182.

[22] Canizo M, Triguero I, Conde A, Onieva E. Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study. Neurocomputing 2019;363:246–60.

[23] Ellefsen AL, Bjørlykhaug E, Æsøy V, Ushakov S, Zhang H. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliab Eng Syst Saf 2019;183:240–51.

[24] Xiang S, Qin Y, Luo J, Pu H, Tang B. Multicellular LSTM-based deep learning model for aero-engine remaining useful life prediction. Reliab Eng Syst Saf 2021;216:107927.

[25] Zio E. Prognostics and health management (PHM): Where are we and where do we (need to) go in theory and practice. Reliab Eng Syst Saf 2022;218:108119.

[26] Sakurada M, Yairi T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis. 2014, p. 4–11.

[27] Zhou C, Paffenroth RC. Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. 2017, p. 665–74.

[28] Zhao Y, Deng B, Shen C, Liu Y, Lu H, Hua X-S. Spatio-temporal autoencoder for video anomaly detection. In: Proceedings of the 25th ACM international conference on multimedia. 2017, p. 1933–41.

[29] Guo X, Liu X, Zhu E, Yin J. Deep clustering with convolutional autoencoders. In: International conference on neural information processing. Springer; 2017, p. 373–82.

[30] Martinez-Garcia M, Zhang Y, Wan J, Mcginty J. Visually interpretable profile extraction with an autoencoder for health monitoring of industrial systems. In: 2019 IEEE 4th international conference on advanced robotics and mechatronics (ICARM). IEEE; 2019, p. 649–54.

[31] Camacho J, Pérez-Villegas A, García-Teodoro P, Maciá-Fernández G. PCA-based multivariate statistical network monitoring for anomaly detection. Comput Secur 2016;59:118–37.

[32] Sanchez-Fernández A, Fuente MJ, Sainz-Palmero G. Fault detection in wastewater treatment plants using distributed PCA methods. In: 2015 IEEE 20th conference on emerging technologies & factory automation (ETFA). IEEE; 2015, p. 1–7.

[33] An J, Cho S. Variational autoencoder based anomaly detection using reconstruction probability. Spec Lect IE 2015;2(1):1–18.

[34] Sun J, Wang X, Xiong N, Shao J. Learning sparse representation with variational auto-encoder for anomaly detection. IEEE Access 2018;6:33353–61.

[35] Xu H, Chen W, Zhao N, Li Z, Bu J, Li Z, Liu Y, Zhao Y, Pei D, Feng Y, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: Proceedings of the 2018 world wide web conference. 2018, p. 187–96.

[36] Ren L, Sun Y, Cui J, Zhang L. Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. J Manuf Syst 2018;48:71–7.

[37] Yu W, Kim IY, Mechefske C. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. Mech Syst Signal Process 2019;129:764–80.

[38] Costa N, Sánchez L. Remaining useful life estimation using a recurrent variational autoencoder. In: International conference on hybrid artificial intelligence systems. Springer; 2021, p. 53–64.

[39] Xiongzi C, Jinsong Y, Diyin T, Yingxun W. Remaining useful life prognostic estimation for aircraft subsystems or components: A review. In: IEEE 2011 10th international conference on electronic measurement & instruments, Vol. 2. IEEE; 2011, p. 94–8.

[40] Saxena A, Goebel K. Phm08 challenge data set. In: NASA Ames prognostics data repository. NASA Ames Research Center; 2008.

[41] Ayhan B, Kwan C, Liang SY. Adaptive remaining useful life prediction algorithm for bearings. In: 2018 IEEE international conference on prognostics and health management (ICPHM). IEEE; 2018, p. 1–8.

[42] Heimes FO. Recurrent neural networks for remaining useful life estimation. In: 2008 international conference on prognostics and health management. IEEE; 2008, p. 1–6.

[43] Miao H, Li B, Sun C, Liu J. Joint learning of degradation assessment and RUL prediction for aeroengines via dual-task deep LSTM networks. IEEE Trans Ind Inf 2019;15(9):5023–32.

[44] Kingma DP, Welling M. An introduction to variational autoencoders. 2019, arXiv preprint arXiv:1906.02691.

[45] Kingma DP, Mohamed S, Rezende DJ, Welling M. Semi-supervised learning with deep generative models. In: Advances in neural information processing systems. 2014, p. 3581–9.

[46] Zhao Q, Adeli E, Honnorat N, Leng T, Pohl KM. Variational autoencoder for regression: Application to brain aging analysis. In: International conference on medical image computing and computer-assisted intervention. Springer; 2019, p. 823–31.

[47] Wu Y, Yuan M, Dong S, Lin L, Liu Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. Neurocomputing 2018;275:167–79.

[48] Elsheikh A, Yacout S, Ouali M-S. Bidirectional handshaking LSTM for remaining useful life prediction. Neurocomputing 2019;323:148–56.

[49] Tensorflow, Embedding projector. URL https://projector.tensorflow.org/.

[50] Nguyen H, Tran KP, Thomassey S, Hamad M. Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management. Int J Inf Manage 2021;57:102282.

[51] Aydemir G, Acar B. Anomaly monitoring improves remaining useful life estimation of industrial machinery. J Manuf Syst 2020;56:463–9.

[52] Pasa GD, de Medeiros IP, Yoneyama T. Operating condition-invariant neural network-based prognostics methods applied on turbofan aircraft engines. In: Proceedings of the annual conference of the PHM society, Vol. 11. 2019.

[53] Bergstra J, Yamins D, Cox DD. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In: Proceedings of the 12th Python in science conference, Vol. 13. Citeseer; 2013, p. 20.

[54] Smith LN. Cyclical learning rates for training neural networks. In: 2017 IEEE winter conference on applications of computer vision (WACV). IEEE; 2017, p. 464–72.

[55] Li H, Zhao W, Zhang Y, Zio E. Remaining useful life prediction using multi-scale deep convolutional neural network. Appl Soft Comput 2020;89:106113.