Unsupervised Prognostics based on Deep Virtual Health Index Prediction

Martin Hervé de Beaulieu[1], Mayank Shekhar Jha[2], Hugues Garnier[3] and Farid Cerbah[4]

[1,2,3] *Université de Lorraine, CRAN, CNRS UMR 7039, 54506 Vandoeuvre-les-Nancy, France*
martin.herve-de-beaulieu@univ-lorraine.fr
mayank-shekhar.jha@univ-lorraine.fr
hugues.garnier@univ-lorraine.fr

[4] *Dassault Aviation 92552 Saint-Cloud, France*
Farid.Cerbah@dassault-aviation.com

ABSTRACT

Prediction of the Remaining Useful Life (RUL) for industrial systems has been facilitated by the acquisition of large amounts of real-time data and the use of deep learning methods. However, the vast majority of these methods rely on the availability of extensive RUL-labeled data, which is not the case for most of real industrial applications. The goal of this paper is to show how unsupervised learning can provide alternative ways to address this issue. The proposed method is essentially made of two steps. First, a Virtual Health Index (VHI) is extracted in an unsupervised manner from the raw sensor data using a Deep Convolutional Neural Network (CNN) autoencoder. Secondly, an Long-Short Term Memory (LSTM) Encoder-Decoder predicts the future values of the VHI, until an End-of-Life (EOL) pattern is recognized (using a sliding window DTW algorithm). The suggested method is tested on the C-MAPSS dataset and offers promising results with a great potential to be applicable on real-life use cases.

1. INTRODUCTION

The increase in data collection and storage capabilities has led to the use of deep learning methods in predictive maintenance strategies. Two main indicators are used in the Prognostic and Health Management (PHM) community (Lee et al., 2014). The first is the Health Index (HI). Health Index is an indicator which represents the State of Health (SOH). It is built from the measured data collected by the sensors placed on the system (Lei et al., 2018). It is a critical indicator, as it should reveal the degradation process hidden within the different signals. A common approach for constructing such an HI from multiple signals is to fuse them into a single indicator, using for example a deep neural network based compression technique (C. Hu, Youn, Wang, & Yoon, 2012). Such an HI is called "Virtual Health Index" (VHI). It worth noting that a VHI is an implicit representation of the degradation but it cannot be interpreted from a physical point of view.

Autoencoder models are very efficient in extracting VHI from raw sensors data, being well adapted for the treatment of multivariate non-stationary data. It has been shown that auto-extracted features are preferable over the handcrafted features in the case of bearing vibrations (Y. Hu, Palmé, & Fink, 2016), exhibiting monotonicity and clear trendability. Such an AI-based extraction of VHI can then be employed to perform RUL prediction (Gensler, Henze, Sick, & Raabe, 2016), in a two-stage framework similar to the method we present in this paper. Moreover, CNN is a particular deep learning model which has shown great ability on feature extraction, especially in the domain of image classification, speech recognition and time series prediction (LeCun, Bengio, et al., 1995). CNN has also been successfully applied to health monitoring and prognostics (Babu, Zhao, & Li, 2016) (Li, Ding, & Sun, 2018), in the context of direct mapping of raw sensor data to health state indicators. Therefore, there is good reason to think that the combination of CNN and autoencoders would offer very good VHI extraction capabilities from raw sensor data.

The second important indicator in PHM is the Remaining Useful Life (RUL) which is defined as the remaining operating time before the End Of Life (EOL) of the system is reached (Si, Wang, Hu, & Zhou, 2011). A large number of contributions have been published in the recent years in order to propose methods for RUL prediction, based on deep learning approaches, using various types of neural networks. The vast majority of these papers relies on the direct mapping between the sensor values and the RUL prediction. These approaches are supervised and thus require large RUL-labeled dataset for training. However, in practice, these labels are, most of the time, not available. In fact, to obtain the genuine value of the RUL, it is necessary to take measurements of the degradation level until the EOL of the machinery. This is time consuming and extremely costly, as it requires numerous (possibly complete) failure tests. Consequently, such methods, although theoretically sound, are hardly applicable in an industrial environment. A few projects have already been carried out with a perspective of avoiding of labeled data. A RUL estimation method based on a Health Index obtained by using the reconstruction error has been proposed (Malhotra et al., 2016). In this paper, an LSTM-based encoder-decoder is trained to reconstruct the time series corresponding to the healthy state of the system. With increasing degradation, the reconstruction error also increases, resulting in a Health Index. Other suggestions also involve Unsupervised Kernel Regression (UKR) (Khelif, Malinowski, Chebel-Morello, & Zerhouni, 2014) or Adversarial Regressive Domain Adaptation (ARDA) (Jiang, Xia, Wang, Fang, & Xi, 2022) for RUL prediction.

The current paper addresses the problem of RUL prediction in the absence of RUL-labeled data. To this end, an unsupervised method of RUL prediction in two steps is proposed. First, a VHI is extracted using a pruned CNN autoencoder in a non-supervised manner. Secondly, long-range prediction of this VHI is performed by an LSTM encoder-decoder. Such a structure leverages the recurrent nature of the data, and has already proven its efficiency on similar use cases (Yu, Kim, & Mechefske, 2019). Prediction continues until an end-of-life pattern is recognized. For this recognition task, Dynamic Time Warping (DTW) similarity measure is employed. As a result, the RUL can be estimated without using any labeled data during the training phase. This paper extends and improves upon our previous work (Herve de Beaulieu, Jha, Garnier, & Cerbah, 2022), incorporating two major new contributions, namely: the VHI extraction using a CNN autoencoder and the EOL pattern recognition with DTW.

The rest of the paper is organized as follows. The problem statement and background are presented in Section 2. The architecture of the proposed method is introduced in Section 3. Application results to the C-MAPSS dataset, including technical choices and data processing are presented in Section 4. Conclusion and future work are discussed in Section 5.

2. Problem statement and background

In this section, the fundamental background necessary for the understanding of the proposed approach will be briefly introduced.

2.1. Problem statement

Let us suppose that we have N identical category equipment, e.g. N engines with index $1 \leq i \leq N$ for which we collect data from K sensors with index $1 \leq k \leq K$. Each data record is made until the End of Life (EOL), denoted as T_i, of the equipment is reached, with index $1 \leq t \leq T_i$. The set of data is thus a collection of objects $\{X^{(i)} | i \in [1, N]\}$ with each data sample $X^{(i)} \in \mathbb{R}^{T_i \times K}$. Therefore, the k-th column ${X_k}^{(i)}$ corresponds to the vector of values of sensor k for all time steps $1 \leq t \leq T_i$ and the t-th row $X^{t^{(i)}}$ corresponds to the vector of values of all sensor $1 \leq k \leq K$ for the given time step t. Finally, the scalar $X_k^{t^{(i)}}$ is the single value recorded by sensor k at time step t on equipment i. The problem can then be formulated as follows: from an initial subset (or window) $X^{t_1{}^{(i)}}$ to $X^{t_2{}^{(i)}}$ with $1 \leq t_1 \leq t_2 \leq T_i$, the corresponding set of VHI values $VHI^{t_1{}^{(i)}}$ to $VHI^{t_2{}^{(i)}}$ must be extracted. Based on this VHI set, the RUL value for the last time step of the window t_2, denoted as $RUL^{t_2{}^{(i)}}$ must be predicted.

2.2. CNN AutoEncoder

An encoder-decoder is a neural network structure which is composed of two elements. First, an encoding function f_{θ_e} compresses the input data to a subspace called latent space. Second, the compressed representation is expanded through a decoding function g_{θ_d}. The learning process can then be formalized as follows:

$$y = g_{\theta_d}(f_{\theta_e}(x)). \tag{1}$$

with y the output of the encoder-decoder and x the input data. A special case of encoder-decoder is the autoencoder, where the output y is learned to be the reconstruction of the input x, which we note $y = x'$. The loss (also called "reconstruction error") is therefore obtained via a function of x and x':

$$J_{AE}(\theta_e, \theta_d) = \sum L(x, x') = \sum L(x, g_{\theta_d}(f_{\theta_e}(x))) \tag{2}$$

where L is a loss function such as the mean squared error (Bengio, Courville, & Vincent, 2013). Note that an autoencoder is trained in an unsupervised manner since the cost calculation does not require any labeled data.

CNN autoencoder is a particular case of autoencoder where the encoding and decoding functions are achieved by CNN structures. It consists of performing a convolution product between the input I and a kernel F (also called filter). For time series, a one-dimensional convolution is applied along the time. The expression of the process is as follows:

$$S(t) = (I * F)(t) = \sum_{t=1}^{T} I(t)F(t - T) \tag{3}$$

with S the output (also called feature map), I the input, F the kernel, T the overall length of the input and $*$ the convolution product (Goodfellow, Bengio, & Courville, 2016).

Since the time series are K-dimensional, the K different variables (e.g. K different sensors) are treated as multiple channels. An independent kernel is thus assigned to each channel, resulting in K feature maps. By varying the number of kernels, the dimensionality of the data can then be extended or reduced. In particular, in the case of CNN autoencoders, care should be taken to reduce the number of dimensions to obtain a latent space of reduced size.

2.3. LSTM Encoder-Decoder

Recurrent neural network (RNN) is a neural network structure adapted to sequential learning. It uses prior knowledge along with current input to make the prediction of the desired output. Therefore, RNN models are widely used for sequential data learning such as time series. A recurrent model is made of multiple standard cells whose states are affected by both past states and current input. The most used version of this kind of cell is called "Long-Short Term Memory" (LSTM) (Hochreiter & Schmidhuber, 1997). LSTM have been widely used for RUL prediction as well (Wu, Yuan, Dong, Lin, & Liu, 2018), (Zhang, Zhang, Shao, Niu, & Yang, 2020).

Similarly to CNN autoencoder, LSTM encoder decoder is nothing else than a special case of encoder-decoder where the encoding and decoding functions are performed by RNN models (Cho et al., 2014). Therefore, the RNN-encoder transforms the input time series into a fixed-dimensional vector representation (usually referred to as "context vector") while the RNN-decoder maps this context vector to the target time series.

For a univariate source time series $X = \{x_1, x_2, ..., x_T\}$, h_t is the hidden state of the RNN-encoder at time t. The RNN-encoder captures relevant information as the source time series is scanned and when its last time step T is reached, the hidden state h_T is the vector representation of the entire source time series X. The RNN-decoder, in a mirroring operation, takes the final encoding hidden state h_T as initial decoding hidden state. It then constructs the desired output time series. The desired output can be a reconstruction of the input ($X' = \{x'_1, x'_2, ..., x'_T\}$), in this case it is called RNN-autoencoder and it is an unsupervised process. The output can also be a prediction of the future values of the source time series (Du, Li, Yang, & Horng, 2020). Then, it is a supervised learning task requiring a training set containing the labels of future time series values.

Regardless of the type of output chosen, the model is trained to minimize an error (which is called "reconstruction error" in the case of RNN-autoencoder) between the target and the result of the model. It can be written as:

$$E = \sum_{i=1}^{N} \sum_{t=1}^{T} (y_t - y'_t) \tag{4}$$

for N time series of length T, with y_t the target value and y'_t the value obtained from the model.

Different recurrent structures and options can be used for the encoding and decoding process, such as stacked-RNN and/or bidirectional RNN (Yu et al., 2019).

2.4. Sliding DTW pattern recognition

Dynamic Time Warping (DTW) is a similarity measuring technique for time series traditionally used in speech recognition (Sakoe & Chiba, 1978). Let us consider two time series $X = (x_1, x_2, ...x_N)$, and $Y = (y_1, y_2, ...y_M)$ of lengths N and M. A local cost matrix C representing all pairwise distances between X and Y is built. To do so, a local cost measure c is computed between each pair of elements of the sequences X and Y. Thus, the local cost matrix can be denoted as $C \in \mathbb{R}^{N \times M}$ and is defined by $C(n, m) = dist(x_n, y_m)$ with $n \in [1 : N], m \in [1 : M]$ and $dist$ being a local distance measure (e.g. $dist(x_n, y_m) = \|x_n - y_m\|$).

The goal is then to find the alignment path which runs through the low-cost areas of the local cost matrix. A warping path is formally defined as a sequence of points $p = (p_1, p_2, ..., p_L)$ with $p_l = (p_n, p_m) \in [1 : N] \times [1 : M]$ for $l \in [1 : L]$. Any warping path must respect three conditions:

1. *Boundary condition*: $p_1 = (1, 1)$ and $p_L = (N, M)$. This basically means that the starting and ending points of the warping path are effectively the first and the last points of the two sequences X and Y.
2. *Monotonicity condition*: $n_1 \leq n_2 \leq ... \leq n_L$ and $m_1 \leq m_2 \leq ... \leq m_L$. This condition ensures that the points are correctly time-ordered.
3. *Step size condition*: $p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}$ for $l \in [1 : L - 1]$. This criterion prevents the warping from doing big shifts in time while aligning the two sequences.

The total cost of a warping path p is expressed as:

$$C_p(X, Y) = \sum_{l=1}^{L} dist(x_{n_l}, y_{m_l}) \tag{5}$$

The DTW distance between two time series X and Y is then the total cost of the optimal warping path denoted as p^*, which is the warping path having the minimal total cost among all the possibilities. The major benefit of DTW is that it minimizes the effects of shifting and distortion in time by allowing a flexible transformation of time series with the intention of detecting similar shapes. A simplified comparison between euclidean and DTW distances is illustrated in Figure 1. The Python library used in this work is from (Giorgino, 2009).

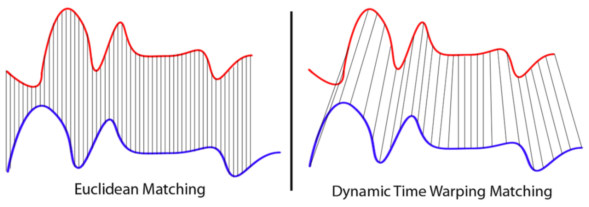

Figure 1. Simplified comparison between the Euclidean distance and the DTW distance.

The DTW is used in the proposed approach in order to recognise a pattern Y in a time series X. Therefore, DTW is applied by using a sliding time window mechanism. A window denoted $W(X)$ (i.e. a sub-sequence of X) is sliding over the full sequence X, with a stride s. At each step, the DTW distance is computed between the sliding window $W(X)$ and the pattern (i.e. template time series) Y.

3. Proposed method

The main advantage of the method proposed in this paper is to provide a prediction of RUL in an unsupervised way, i.e. without needing RUL-labeled data. To do this, we proceed in two steps.

Step 1: a CNN autoencoder model is used to extract from the raw sensor data a univariate Virtual Health Index (VHI) which is a compression of the multi-variate input data. The excellent feature extraction abilities of such a structure enable to highlight a so called "End-of-Life-pattern" which characterizes the moment when the EOL of the studied equipment occurs.

More specifically, for the suggested method, the CNN encoder model is made of 9 convolution layers. In order to extract deep features, the first layers increase the depth of the data by expanding the number of channels from $K = 7$ to $K = 56$ and then compress it to obtain a univariate VHI in the latent space ($K = 1$). The CNN decoder is built as a mirror of the encoder. The hyper parameters of the convolutions layers are chosen so that the initial length of the time series remains unchanged over the convolution operations. Specifically : stride $s = 1$; kernel length $k_l = 23$; symmetrical padding $p = 11$. Between each convolution, a ReLU activation function is apply to ensure non-linearity. Figure 2 shows the overall CNN autoencoder structure.

Step 2: an LSTM encoder-decoder is used in order to predict the future values of the VHI. This structure is made of deep-stacked LSTM (3 layers of 120 hidden units) whose role is to extract the deep temporal features of the VHI time series and to output a prediction window of length P. This has already been presented in (Herve de Beaulieu et al., 2022). The final objective is to estimate the RUL. To do so, for each prediction window, the presence of the EOL pattern is checked. As long as the pattern announcing the end of life is not detected, the prediction continues, using previously predicted values as input for subsequent predictions. The detection of the EOL-pattern is achieved by measuring the DTW distance between the prediction windows and the EOL-pattern that have been extracted by the CNN autoencoder. The prediction continues step by step until the EOL-pattern is recognized with a sufficiently high similarity (meant as a threshold). When that time is reached, this means that the equipment has reached its EOL. The RUL can thus be inferred recursively by counting the number of prediction cycles that were necessary. The proposed RUL inference process is summarized by Figure 3. Min-similarity threshold, stride and window lengths are set initially in an empirical manner and the optimized by using a validation set.

Of course, the closer the input data is to the end of life, the lower the length to be predicted and therefore the more accurate the deduced RUL. This results in a higher variability in the early predictions (temporally distant from the EOL) than in the latter.

```
Input: VHI window of length $T$ denoted as $X$, EOL
       pattern denoted as $Z$.
Output: RUL value (scalar).
$i = 0$;
$Y \leftarrow$ VHI predicted window of length $P$ from $X$;
while $DTW(Y, Z) > threshold$ do
    $i \leftarrow i + 1$;
    $X \leftarrow X[S:] + Y[0:S]$ ;        /* stride S */
    $Y \leftarrow$ new VHI predicted window from new input $X$;
end
$RUL \leftarrow S \times i + P$;
```

Figure 3: RUL prediction process for one input VHI window

4. Experimental results

4.1. C-MAPSS dataset

The C-MAPSS dataset is used as experimental data to test the proposed method. This dataset is composed of turbofan engines degradation trajectories which are obtained from a simulator developed by NASA (Saxena, Goebel, Simon, & Eklund, 2008). C-MAPSS dataset is divided into four different subsets (named FD001 to FD004), each made of a training set containing several complete degradation data (i.e. multivariate data collected from sensors) and a test set containing truncated degradation data (from the same distribution as training set), the objective being to predict the RUL based on this incomplete test data. Depending on the subset of C-MAPSS, fault modes can vary from one in FD001 and FD002 to two in FD003 and FD004. Similarly, the data is obtained from simulations carried out under a variable number of operating conditions. These conditions are based on different combinations of altitude (0 to 42000 feet), throttle resolver angle (20 to 100) and Mach (0 to 0.84). More details about

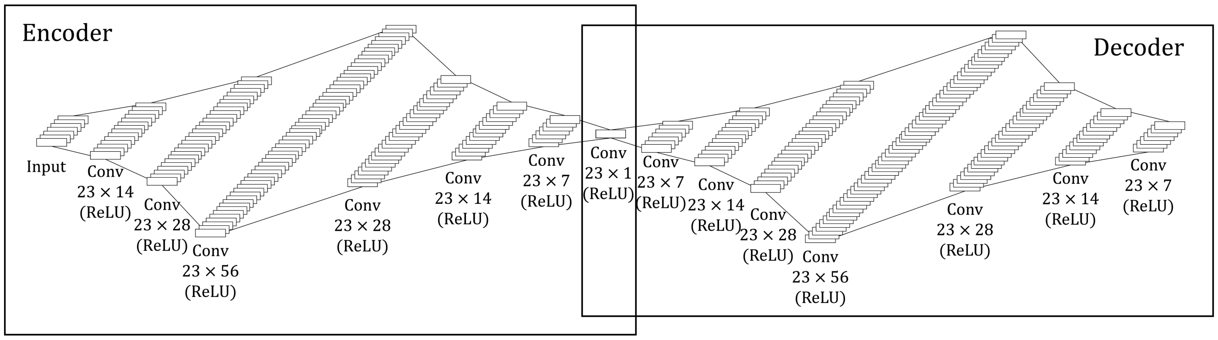

Figure 2. Proposed deep CNN autoencoder structure for VHI extraction.

Table 1. Features of the C-MAPSS dataset.

	C-MAPSS subsets			
	FD001	FD002	FD003	FD004
Engine for training	100	260	100	249
Engine for testing	100	259	100	248
Operating conditions	1	6	1	6
Fault modes	1	1	2	2

the different subsets are given in Table 1.

4.2. Data preparation

21 different sensor variables are available in the C-MAPSS dataset. To reduce the computational burden, only the seven most indicative sensors are kept, based on the observation of the input time series (Wang, Yu, Siegel, & Lee, 2008). Therefore, the input set is composed of sensors number 2, 3, 4, 7, 11, 12, 15. The detailed list of sensors, along with the units and description, is available in Appendix A. All sensor time series are normalized following a standard scaling defined as:

$$Norm(x^s) = \frac{x^s - \mu^s}{\sigma^s} \tag{6}$$

where s is the selected sensor, μ^s and σ^s are the mean and the standard deviation.

Before providing this data to the CNN autoencoder, a zero pre-padding operation is applied, in order to force all time series to have the same length (Dwarampudi & Reddy, 2019). Once the VHI has been obtained, the univariate VHI is unpadded to go back to its original length. Figure 4 gives an example of padded and normalized multi-variate sensor time series given as input to the CNN autoencoder.

4.3. Performance indicator of the RUL prediction

The evaluation of the performance of the RUL prediction made by the proposed method is handled using a Root Mean Square Error (RMSE). Indeed, it is the most used performance indicator in RUL prediction literature, especially on C-MAPSS. In the future, it may be considered to complete this metric with other indicators such as the score function (Saxena et al., 2008) or the Mean Absolute Percentage Error (Malhotra et al., 2016). RMSE is computed as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2}, \tag{7}$$

for n predictions, where d_i is the difference between the predicted and the actual RUL:

$$d_i = R\hat{U}L_i - RUL_i. \tag{8}$$

4.4. Results

4.4.1. Step 1 - VHI extraction using CNN autoencoder

At the end of the training phase, an EOL pattern is recognised by the deep CNN autoencoder. It is a local minimum whose shape and position are always identical for all the turbines of the training set. Let us keep in mind that, as mentioned in Section 1, VHI does not have any physical interpretation. It should be considered as an indication of the EOL (based on the location of the pattern), not as a physical measure of the equipment SOH. Therefore, the fact that the VHI increases at the end of life does not indicate an improvement in SOH. Such an EOL pattern can be seen in Figure 4 and in Figure 5. On the latter, the true RUL has also been plotted in order to show that the pattern is effectively corresponding to the EOL. Therefore, this pattern will serve as template to be detected on the test data. Note that after having obtained the VHI, the zero-pre-padding is removed to keep the original length. As explained in section 4.1, the test set is composed of time series which are not reaching the end of life. Thus, this test data does not yet reveal the pattern.

4.4.2. Step 2 - VHI long-range prediction and RUL inference

Therefore, from the unfinished test VHI, future predictions are obtained by the LSTM encoder-decoder. As described in Section 3, future VHI values are predicted in a rolling window process, re-using the previous predictions as new inputs for the next ones. Figure 6 shows the result of such a predic-

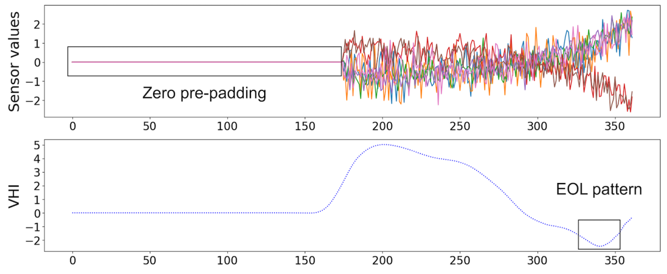

Figure 4. Top: an example multi-variate input data for one turbine from the training set. Bottom: the corresponding VHI extracted by the CNN autoencoder.

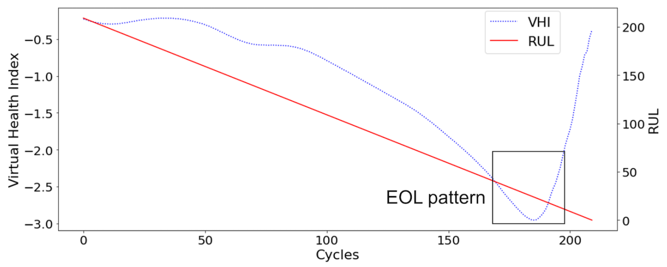

Figure 5. An example of extracted VHI (dotted line) along with RUL labels of one turbine from the training set.

tion process for one turbine, from time step $t = 45$. Here, the whole process of prediction only relies on the 50 values preceding the time step $t = 45$. The rest of the prediction process is self-feeding, by reusing past predictions. On the same figure is displayed the reference pattern, at the instant where it has been recognized with a satisfying DTW distance score value, using the sliding DTW algorithm introduced in Section 2.4. This value is set empirically, in a hyperparameter optimization loop conducted on a validation set. The VHI prediction process is applied for each turbine available, at each time step, thus resulting for each turbine in a sequence of RUL values. Such a trajectory is shown in Figure 7, for the same turbine as in Figure 6. The RMSE of the RUL trajectory is calculated for each turbine of the test set, leading to an average RMSE of 40.1 and a standard deviation of 21.7.

5. Conclusion

An unsupervised RUL prediction method has been proposed in this paper that avoids the dependence on RUL-labeled data, therefore offering great interest for real-world applications. A VHI has been extracted from sensor data in an unsupervised manner, using a deep CNN autoencoder, highlighting a clear end-of-life pattern. Then, using an LSTM encoder decoder, future values of unseen VHI have been predicted, until the EOL pattern is recognized using a DTW distance measure. This pattern detection algorithm allows to deduce precisely the RUL value from the VHI without needing the RUL labels. The proposed method has been tested with success on the C-MAPSS dataset.

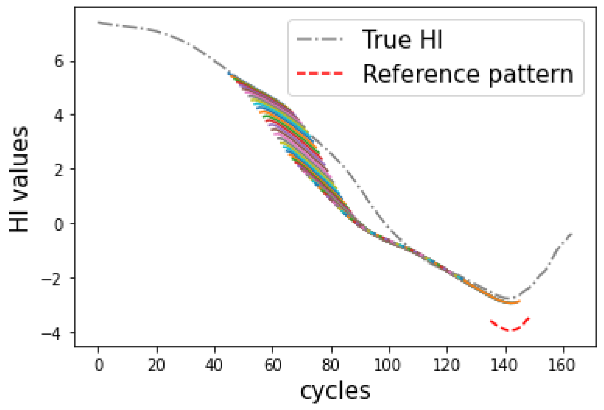

Figure 6. Overview of the set of predicted sliding windows from time step $t = 45$, until the reference pattern (in red) is recognized with a high enough DTW measure of similarity.

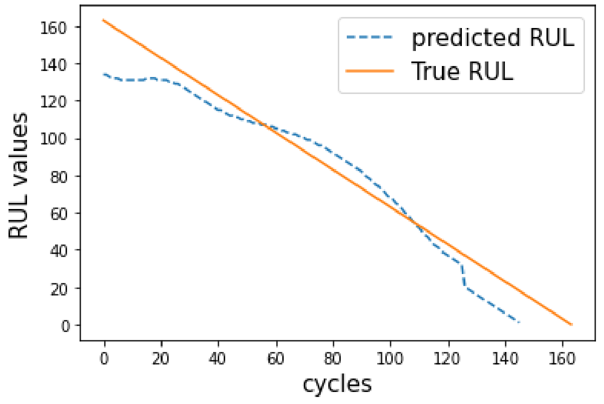

Figure 7. Complete RUL trajectory for all the time steps for one turbine.

As a perspective, this unsupervised RUL prediction problem will be comprehensively extended to handle various conditions and/or various fault modes. In particular, the other datasets included in C-MAPSS will be used, as they offer up to 6 different operating conditions based on 3 variable settings mixed randomly during each flight and up to 2 fault modes.

References

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications* (pp. 214–228). Dallas, Texas, USA.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Du, S., Li, T., Yang, Y., & Horng, S.-J. (2020). Multivariate time series forecasting via attention-based encoder–decoder framework. *Neurocomputing*, *388*, 269–279.

Dwarampudi, M., & Reddy, N. (2019). Effects of padding on LSTMs and CNNs. *arXiv preprint arXiv:1903.07288*.

Gensler, A., Henze, J., Sick, B., & Raabe, N. (2016). Deep learning for solar power forecasting—an approach using autoencoder and LSTM neural networks. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 002858–002865).

Giorgino, T. (2009). Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, *31*, 1–24.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (`http://www.deeplearningbook.org`)

Herve de Beaulieu, M., Jha, M., Garnier, H., & Cerbah, F. (2022). Long range health index estimation based unsupervised RUL prediction using encoder-decoders. In *11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes.* Pafos, Cyprus.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hu, C., Youn, B. D., Wang, P., & Yoon, J. T. (2012). Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life. *Reliability Engineering & System Safety*, *103*, 120–135.

Hu, Y., Palmé, T., & Fink, O. (2016). Deep health indicator extraction: A method based on auto-encoders and extreme learning machines. In *PHM 2016, Denver, USA* (pp. 446–452).

Jiang, Y., Xia, T., Wang, D., Fang, X., & Xi, L. (2022). Adversarial regressive domain adaptation framework for infrared thermography-based unsupervised remaining useful life prediction. *IEEE Transactions on Industrial Informatics*.

Khelif, R., Malinowski, S., Chebel-Morello, B., & Zerhouni, N. (2014). Unsupervised kernel regression modeling approach for RUL prediction. In *PHM Society European Conference* (Vol. 2).

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The Handbook of Brain Theory and Neural Networks*, *3361*(10), 1995.

Lee, J., Wu, F., Zhao, W., Ghaffari, M., Liao, L., & Siegel, D. (2014). Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical systems and signal processing*, *42*(1-2), 314–334.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical systems and signal processing*, *104*, 799–834.

Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, *172*, 1–11.

Malhotra, P., TV, V., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., & Shroff, G. (2016). Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder. *arXiv preprint arXiv:1608.06154*.

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *26*(1), 43–49.

Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *International Conference on Prognostics and Health Management* (pp. 1–9). Denver, Colorado, USA.

Si, X.-S., Wang, W., Hu, C.-H., & Zhou, D.-H. (2011). Remaining useful life estimation–a review on the statistical data driven approaches. *European Journal of Operational Research*, *213*(1), 1–14.

Wang, T., Yu, J., Siegel, D., & Lee, J. (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *International Conference on Prognostics and Health Management* (pp. 1–6). Denver, Colorado, USA.

Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, *275*, 167–179.

Yu, W., Kim, I. Y., & Mechefske, C. (2019). Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. *Mechanical Systems and Signal Processing*, *129*, 764–780.

Zhang, H., Zhang, Q., Shao, S., Niu, T., & Yang, X. (2020). Attention-based LSTM network for rotatory machine remaining useful life prediction. *IEEE Access*, *8*, 132188–132199.

APPENDIX

A. SUMMARY OF THE SELECTED SENSORS OF FD001 DATASET

Sensor ID	Description	Unit
s2	Total temperature at LPC outlet	°R
s3	Total temperature at HPC outlet	°R
s4	Total temperature at LPT outlet	°R
s7	Total pressure at HPC outlet	psia
s11	Static pressure at HPC outlet	psia
s12	Ratio of fuel flow to Ps30	-
s15	Bypass Ratio	-