Amélioration du pronostic par apprentissage profond pour des applications de maintenance prédictive

Guillaume Chambaret [*,**], Laure Berti-Equille [*,***], Frédéric Bouchara[*] Emmanuel Bruno[*]
Vincent Martin[**] Fabien Chaillan[**]

[*] DIAMS/SIIM, Laboratoire d'Informatique et Systèmes, Université de Toulon
[**] Naval Group Research, Ollioules [***] ESPACE-DEV, IRD, Montpellier

Résumé. Dans cet article, nous nous intéressons à l'amélioration de la prédiction de la durée restante de fonctionnement utile d'un système complexe dont l'état est représenté par des séries temporelles de données multivariées. Notre contexte d'application est le domaine de la maintenance prédictive pour l'industrie navale. Nous présentons et évaluons deux approches différentes en mesurant l'amélioration de la prédiction de la durée de vie restante utile (*Remaining Useful Life* ou RUL) au moyen de quatre approches d'apprentissage automatique utilisant des réseaux de neurones profonds. La première méthode que nous proposons s'appuie sur un ré-échantillonnage de la base d'apprentissage afin de réduire localement les erreurs. La deuxième méthode proposée s'intéresse à la détection automatique et l'utilisation d'un point de rupture dans le signal multivarié pour améliorer la phase d'entraînement. Nous montrons que les techniques de détection de points de rupture permettent une amélioration significative de la performance de prédiction des durées de vie restantes avec des gains allant jusqu'à 27 % sur l'erreur moyenne absolue (MAE) quel que soit le réseau utilisé, ce qui démontre la généricité et l'intérêt de notre approche.

1 Introduction

Pour l'industrie navale de Défense et au-delà, le principe de la maintenance prédictive est de développer des outils permettant d'automatiser efficacement la détection, la classification, et l'explication des avaries des bâtiments et de leurs équipements (qui sont eux mêmes des systèmes complexes) par un processus d'accompagnement continu dans leur cycle de vie. Ainsi, l'objectif final est de prédire le temps pendant lequel un système restera utilisable en fonction de son historique et de son état courant. On parle généralement de *i-maintenance*, terme largement popularisé au sein des industries tout secteur confondu. En effet, à la différence de la maintenance préventive qui s'appuie sur des inspections régulières et sur le remplacement systématique des composants, il est possible, par une surveillance continue et par des techniques de prédiction, d'anticiper une usure anormale ou au contraire de repousser un remplacement inutile (augmentant ainsi la disponibilité des équipements).

Dans ce cadre, nous nous intéressons en particulier à l'approche prédictive qui se focalise sur le pronostic, c'est-à-dire l'estimation de la durée de vie restante d'un équipement pour lequel on est certain de l'apparition d'une défaillance. Des défis se posent dès qu'il est question du

pronostic selon une approche basée sur les données. Citons, en particulier, le fait de ne travailler qu'à partir des dates de défaillances, et donc d'ignorer le mode de dégradation amenant à la défaillance. Cet article introduit deux contributions aux approches prédictives basées sur la préparation des données précédent un apprentissage automatique. Il s'agit d'améliorer les performances de l'apprentissage en utilisant deux méthodes distinctes :

(1) Le rééchantillonage des signaux disponibles pour l'entraînement afin de mieux pondérer les dégradations pour lesquelles la durée de vie est mal anticipée ;

(2) La transformation de la durée de vie restante utilisée pour l'apprentissage au moyen de techniques de segmentation : l'idée étant d'ajouter une information sur un potentiel début de dégradation (ou point de rupture).

2 État de l'art

Depuis quelques années le développement du *Prognostics and Health Management* (PHM) (Gouriveau et al., 2016; Hashemian, 2010) a mis en évidence une multitude d'approches pour optimiser la maintenance à partir des données sans apport de modélisation du fait de diverses contraintes (temps, expertise, complexité du système). Depuis une décennie plusieurs article d'état de l'art ont mis l'accent sur les techniques d'apprentissage automatique pour prédire la durée de vie restante Schwabacher et Goebel (2007). Plus récemment, d'autres reviews introduisant l'usage de l'apprentissage profond, pour modéliser les phénomènes non-linéaires, mettent en évidence de meilleures performances sur les mêmes jeux de données que ceux utilisés dans les travaux plus anciens. (Khan et Yairi, 2018; Ellefsen et al., 2019a). Citons en particulier les architectures de type auto-encodeurs (Tao et al., 2015) qui permettent de reproduire un signal et donc corréler l'erreur de reconstruction à un phénomène de dégradation. Bien que facile à implémenter, ces derniers ont néanmoins tendance à capturer trop d'informations.

Les réseaux récurrents (Gugulothu et al., 2017) sont largement popularisés pour les phénomènes de dégradation car il permettent de prendre efficacement en compte la dimension séquentielle et détectent bien les changements. Ils ont néanmoins tendance à sur-apprendre des phénomènes ponctuels.

Les réseaux convolutifs (Janssens et al., 2016), basés sur la fusion de produits de convolutions, essentiellement popularisés dans le domaine du traitement d'image permettent de bien extraire les variables pertinentes d'un signal multidimensionnel, mais ils sont parfois longs à entraîner.

Enfin, il convient de préciser que des modèles génératifs tel que les RBM (Restricted Boltzmann Machine) (Ellefsen et al., 2019b) ou encore les DBN (Deep Belief Network) (Mao et al., 2017) peuvent également convenir au pronostic car ils permettent de créer des modèles de représentation même en présence de données manquantes.

Notre travail vise à améliorer la qualité de l'estimation de la durée de vie restante par apprentissage automatique quelle que soit l'architecture d'apprentissage profond utilisée. Il s'agit d'un avantage déterminant d'un point de vue opérationnel, car certaines architectures sont plus performantes que d'autres selon les jeux de données et notre approche est "agnostique" et complémentaire à un modèle qui aurait été défini et optimisé au préalable.

3 Mise en oeuvre du pronostic par apprentissage profond

Jeu de données. Par souci de reproductibilité et du fait de nos contraintes de confidentialité, le jeu de données utilisé est le jeu de données public " Turbofan Engine Degradation Simulation Data Set " [1] fourni par la NASA. Il est constitué de plusieurs séries temporelles associées à la supervision d'un moteur TurboFan de son démarrage à sa défaillance. Au total, on comptabilise 100 séries temporelles d'entraînement de tailles variables et autant de séries pour le test. Ces dernières sont tronquées aléatoirement. Les durées de vie restante de test (temps réel) sont données dans un fichier annexe. Il convient de les déterminer le plus précisément possible pour la politique de maintenance. Les variables utilisées (température, enthalpie, etc.) pour le pronostic sont décrites dans (Saxena et al., 2008).

Pré-traitement. Initialement, nous pré-traitons les données de manière classique afin de se ramener à un problème d'apprentissage automatique. Ce pré-traitement consiste, dans un premier temps, à recentrer les valeurs observées par une z-normalisation sur l'ensemble des variables (de la totalité des séries temporelles) telle que, pour une variable v non constante, sa normalisation, notée z, est définie par : $\forall t, \quad z(t) = \frac{v(t)-\mu(v)}{\sigma(v)}$. Les paramètres de la normalisation (écart-type et moyenne) sont conservés pour la normalisation des variables du jeu de test. Par la suite, on utilise une fenêtre glissante sur les séries temporelles avec une taille $L = 32$ (taille de la plus courte série de test). Dans un premier temps, les fenêtres sont étiquetées au moyen d'un RUL linéaire. En notant T les différentes séries temporelles d'entraînement, et Z, le vecteur multivarié des variables normalisées, le jeu d'entraînement est finalement composé des paires fenêtres-labels définies de la façon suivante : $\bigcup_T \left\{ \left([Z_{t-L} \dots Z_t], RUL(t) := \text{len}(T) - t \right), \quad t = L \dots \text{len}(T) \right\}$.

Métriques. Notons, pour une fenêtre i, d_i la différence entre le pronostic prédit RUL_p et le pronostic réel RUL_r soit $d_i = RUL_p(i) - RUL_r(i)$. Les métriques classiquement utilisées pour évaluer la qualité du pronostic sur le jeu de test de taille N sont : MAE $= \frac{1}{N}\sum_i |d_i|$; MSE $= \frac{1}{N}\sum_i d_i^2$; et nous utilisons également le Score introduit par Saxena et al. (2008) et défini tel que $S = \sum_{d_i<0}\left(e^{-\frac{d_i}{13}} - 1\right) + \sum_{d_i\geq 0}\left(e^{\frac{d_i}{10}} - 1\right)$, car il permet de pénaliser les prédictions tardives, synonyme de remplacements non anticipés d'un équipement dégradé.

Comparaison d'architectures. Afin d'évaluer la meilleure approche pour le pronostic, nous avons sélectionné plusieurs modèles de prédiction de RUL par apprentissage profond issus de l'état de l'art. Plus spécifiquement, il s'agit de : (1) un modèle de réseau convolutif monodimensionnel (CNN1D) , (2) un réseau récurrent avec unités LSTM, (3) un réseau récurrent avec unités GRU , et (4) un perceptron multicouche (MLP) entièrement connecté. Afin de définir notre référence (*baseline*), ces modèles sont prédimensionnés sur un RUL linéaire introduit en section 3. Pour l'ensemble des architectures, la fonction de coût sélectionnée est la MSE. Les modèles sont entraînés sur 80 epochs, un arrêt prématuré (*early stopping*) est appliqué sous contrôle du jeu de validation. Les résultats des trois métriques obtenus sur le jeu de test sont donnés pour chaque modèle dans le tableau 1.

1. FD001-CMPASS, voir `https://ti.arc.nasa.gov`

Architecture	Détails de la configuration	MAE	MSE	Score
CNN1D	32/16 filtres de taille kernel : 8/4, dropout : 0.1, Batch-normalisation momentum : 0.8, Flatten, Dense (64), activation : SELU, linéaire dernière couche, optimisation : Adam (lr = 0.0001)	16.3	514.7	4137.1
GRU	128 unités GRU, dropout : 0.1, Dense (64/1), activation : tanh, linéaire dernière couche, optimisation : Adam (lr = 0.00005)	**15.8**	**492.3**	**2271.3**
MLP	Dense (256/128/64/1), dropout : 0.1, Flatten, activation : tanh, linéaire dernière couche, optimisation : Adam (lr = 0.0001),	19.0	581.6	4225.6
LSTM	128+64 unités LSTM, Dense (64/1), dropout : 0.1, Dense (64), activation : tanh, linéaire dernière couche, optimisation : Adam (lr = 0.00005)	15.9	533.5	17485.0

TAB. 1 – *Résultats de la comparaison entre les 4 architectures de départ*

4 Améliorations proposées

4.1 Ré-échantillonnage des paires fenêtres-labels

On se propose, tout d'abord, d'améliorer la qualité des labels par une modification de la distribution des données. La durée de vie restante utilisée décroît linéairement. Toutefois, on souhaite donner davantage de poids à certaines prédictions. Notons que pour les modèles proposés, les contributions à l'erreur de prédiction sont extrêmement variables selon les modèles comme l'indique la figure 1 pour la MAE.

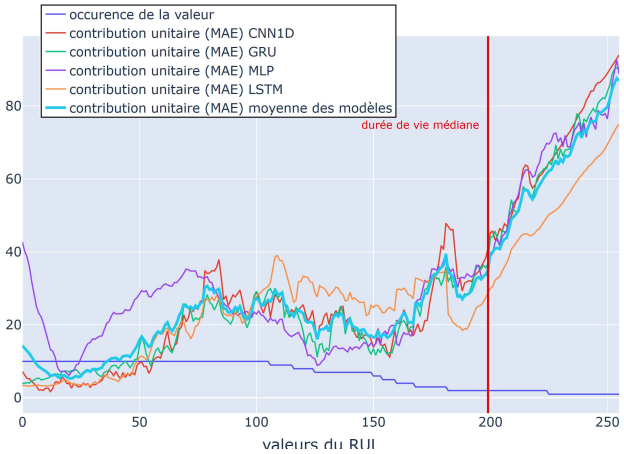


FIG. 1 – *Contribution à l'erreur en valeur absolue (MAE) selon les valeurs de RUL*

Lorsque que l'on observe les signaux, loin en amont de la défaillance, la durée de vie restante peut être considérée comme constante et égale à la durée de vie médiane égale à 199 (espérance d'un tirage aléatoire sur les durées de vie). Par la suite, on intervient uniquement sur les paires fenêtres-labels pour lesquelles le label est inférieur à 199. On augmente alors le jeu de données en tirant aléatoirement des paires de façon à réduire localement les erreurs. Pour ce faire, on sélectionne deux lois dont les pics coïncident avec la courbe lissée des erreurs quadratiques : l'une étant (1) une distribution gaussienne centrée en 95 avec un écart-type de 17 et l'autre étant (2) un mélange composé de la première loi pondérée deux fois et d'une autre distribution gaussienne centrée en 180 avec un écart-type de 10,6. Les écarts-types sont

obtenus par interpolation sur les erreurs quadratiques avec la largeur à mi-hauteur (LMH) tel que : $\sigma = \frac{LMH}{2\sqrt{2\log(2)}}$. Par la suite, on augmente le jeu de données d'entraînement selon un ratio de 5 à 15% de fenêtres-labels dupliqués avec la loi considérée. En pratique, un premier tirage est effectué à 15% puis partitionné aléatoirement afin d'obtenir les ratios inférieurs. Le tableau 2 ci-après donne les résultats obtenus :

Ajout gaussien	RUL linéaire			5 % d'augmentation			10% d'augmentation			15% d'augmentation		
	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score
CNN1D	16.3	515	4137	**14.6**	**329**	**654**	**15.1**	**358**	**728**	**15.9**	**403**	**943**
GRU	15.8	492	2271	16.3	432	1063	18.6	552	2600	19.0	603	3423
LSTM	15.8	536	17488	15.7	364	740	17.2	442	1014	19.0	561	1588
MLP	19.0	582	4226	19.6	551	1569	21.9	704	2676	24.3	858	4405
Ajout par mélange	RUL linéaire			5 % d'augmentation			10% d'augmentation			15% d'augmentation		
	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score
CNN1D	16.3	515	4137	**14.4**	**328**	**700**	**15.1**	**367**	**869**	16.4	466	4428
GRU	15.8	492	2271	17.9	494	1892	18.5	541	1897	19.2	590	2181
LSTM	15.8	536	17488	16.8	408	1077	17.5	459	1057	19.9	586	1577
MLP	19.0	582	4226	19.4	541	1528	21.7	691	2597	24.8	885	4647

TAB. 2 – *Résultats avec ré-échantillonnage*

Le ré-échantillonnage a globalement dégradé la qualité des prédictions sur le modèle de RUL linéaire à l'exception du CNN1D (voir les valeurs de MAE, MSE et score des 3 premières colonnes du tableau comparées aux valeurs obtenues par nos méthodes). Mais les performances obtenues par ré-échantillonage avec CNN1D sont néanmoins meilleures que celles de la meilleure architecture initiale. Le réseau convolutif, particulièrement utile pour la reconnaissance de forme a bénéficié des nouvelles occurences qui lui ont permis d'affiner le calcul de RUL. Il serait donc intéressant, pour des travaux ultérieurs sur le CNN1D, d'étendre le nombre de pics pour un ré-échantillonnage plus complet du jeu d'entraînement.

4.2 Modifications de la durée de vie restante par segmentation

Dans cette section, on se propose d'amener davantage d'information via un diagnostic pour décider de l'instant où la durée de vie restante décroît. Prenons l'exemple de la série normalisée d'entraînement pour le moteur #9 présentée en figure 2. La présence d'une rupture semble

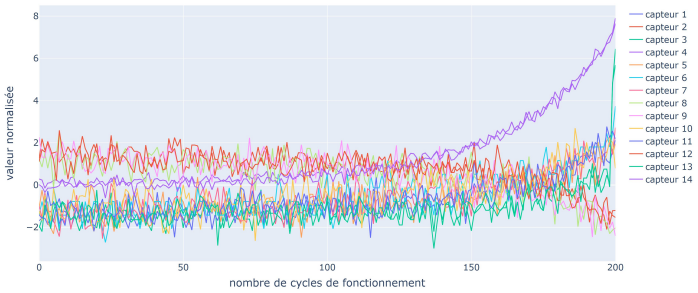


FIG. 2 – *Valeurs normalisées en fonction du nombre de cycles pour le moteur #9*

évidente vers 130 cycles, néanmoins, il convient de formaliser l'approche. Nous supposerons

pour la suite l'existence d'un unique point de rupture. Notre problématique consiste alors à le déterminer via une méthode de recherche et une fonction de coût. En pratique, la fonction de coût c est calculée sur deux parties de la série temporelle (à gauche et à droite du point observé), puis comparée à la fonction de coût de l'ensemble du signal. On définit alors la fonction de divergence au point t sur un segment $[a, b]$ par : $\text{div}(t, [a, b]) = c([a, b]) - c([a, t]) - c([t, b])$. Le point de rupture contenu dans $[a, b]$ correspond alors au maximum de la divergence.

Méthodes de recherche du point de rupture. Dans le cadre de nos travaux, on propose d'utiliser deux méthodes de recherche : (1) une méthode qui consiste à fixer pour une série T, $a = 0$ et $b = len(T)$. En d'autres termes, on calcule la divergence en chaque point, quitte à comparer des segments de taille variable, il s'agit d'une **segmentation exacte** ; et (2) une méthode de **segmentation locale** (Truong et al., 2020) qui consiste à calculer la divergence sur des segments consécutifs de même longueur tels que $[a, a + l/2]$ et $[a + l/2 + l]$ pour $a = 0..len(T) - l$ afin de trouver le point de rupture en dehors des tendances. Par la suite, on fixe arbitrairement $l = 80$.

Fonction de coût. La fonction de coût sélectionnée détecte les changements dans la moyenne du signal intégré sur un espace de Hilbert $\mathcal{H}$ ayant pour noyau $k(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ et son morphisme associé : $\Phi : \mathbb{R}^d \mapsto \mathcal{H}$. Pour un signal z multivarié sur un segment $[a, b]$ de moyenne μ sur ce dernier. La fonction de coût est définie telle que : $c([a, b]) = \sum_{t \in K} \|\Phi(Z_t) - \bar{\mu}\|^2_{\mathcal{H}}$ avec pour noyau, une fonction de base radiale (*Radial Basis Function*), $k(x, y) = \exp\left(-\gamma \|x - y\|^2\right)$ et γ égal à l'inverse de la médiane des distances.

Segmentation sur série monovariée. Une dernière approche consiste à essayer de reconstruire le signal multivarié jusqu'à défaillance et à observer le comportement de ce dernier au moyen d'une série monovariée. L'intérêt de cette approche réside essentiellement dans l'élimination du bruit pour obtenir, *a priori*, une meilleure segmentation. Pour procéder, nous définissons un encodeur/décodeur-récurent (LSTM(20) x2) entraîné sur des fenêtres de taille 20, prises sur les 10 premiers % de longueur de chaque série T d'entraînement. On étudie ensuite les résidus sur les prédictions de l'encodeur pour les 90 % restants. On obtient alors une courbe d'erreur croissante en accord avec la dégradation. On applique les méthodes précédentes sur cette dernière (même paramétrage).

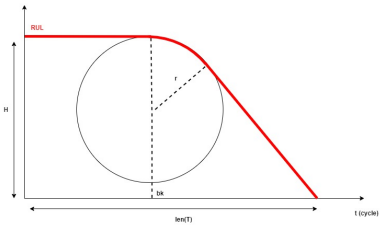


FIG. 3 – *RUL rectifié pour la série T*

Correction de la durée de vie restante. Une fois le point de rupture déterminé par segmentation, il convient de modifier notre définition de durée de vie restante pour la série T. Pour ce faire, on se propose de la définir avec une variation linéaire précédent la défaillance, raccordée par un arc de cercle tangent (voir figure 3). La dégradation n'est pas abrupte, on obtient une phase stationnaire suivie de la dégradation avec la zone linéaire avant défaillance. Pour la zone stationnaire, on choisit après fine-tuning la valeur $H = 125$ à $t = 0$. Ainsi, pour une série T, en notant b_k le point de rupture (*breakpoint*), la durée de vie

restante rectifiée est définie de la façon suivante :

$$RUL(t) = \begin{cases} H & \text{si } t \in [0, t[\qquad r = (1+\sqrt{2})(\text{len}(T) - b_k - H) \\ H - r + \sqrt{r^2 - (t - b_k)^2} & t \in [\text{b}_k, d[\qquad d = \text{b}_k + \frac{(1+\sqrt{2})(\text{len}(T) - bk - H)}{\sqrt{2}} \\ \text{len}(T) - t & \text{si } t \in [d, len(T)[\end{cases}$$

On donne ci-dessous un comparatif des résultats obtenus pour les différentes segmentations :

série multivariée	RUL linéaire			segmentation exacte			segmentation locale		
	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score
CNN1D	16.3	515	4137	**10.8**	**219**	**285**	13.3	344	435
GRU	15.8	492	2271	**11.3**	**264**	**329**	13.0	344	445
LSTM	15.9	536	17488	**10.9**	**240**	**292**	13.3	329	393
MLP	19.0	582	4226	**15.8**	**365**	**584**	16.7	423	568
série monovariée	RUL linéaire			segmentation exacte			segmentation locale		
	MAE	MSE	Score	MAE	MSE	Score	MAE	MSE	Score
CNN1D	16.3	515	4137	12.5	296	384	**11.7**	**248**	**296**
GRU	15.8	492	2271	**11.5**	**264**	**315**	12.6	322	411
LSTM	15.9	536	17488	**11.6**	**269**	**332**	14.5	413	727
MLP	19.0	582	4226	15.9	371	493	**15.1**	**346**	**428**

TAB. 3 – *Résultats avec RUL corrigé par segmentation*

Pour la segmentation multivariée, on remarque une amélioration notable de la qualité des prédictions par ajout d'information via la segmentation. Pour la segmentation exacte, cela représente des améliorations de 27% (MAE), 49% (MSE) et 94 (Score) % par rapport aux performances données sur le RUL linéaire. Néanmoins, pour l'ensemble des modèles, la segmentation locale est moins pertinente que la segmentation exacte avec 16% (MAE), 38% (MSE) et 92% (Score) d'amélioration par rapport aux architectures initiales. Ceci s'explique par le fait que la perte d'information liée à la sélection des points de rupture sur des portions de la série ne permet pas de capturer efficacement le démarrage de la dégradation.

Dans le cas monovarié, la segmentation contribue également à l'amélioration des prédictions pour toute arhitecture. Notons que la compression de l'information par l'auto-encodeur pour les réseaux récurrents (GRU et LSTM) a dégradé la qualité des prédictions par rapport au cas multivarié. Néanmoins, dans le cas d'une segmentation locale, le recours à cette compression a permis d'améliorer les performances du CNN1D et du MLP pour l'ensemble des métriques considérées, tant par rapport à la durée de vie restante linéaire que par rapport au cas multivarié.

5 Conclusion

Dans cet article, nous nous sommes intéressés à l'amélioration de la prédiction de la durée du fonctionnement utile restante d'un système complexe décrit par des séries temporelles multivariées. Nous avons présenté et évalué deux approches permettant l'amélioration de la prédiction de la durée de vie restante pour quatre architectures d'apprentissage profond. La première méthode proposée est basée sur un ré-échantillonnage de la base d'apprentissage afin de réduire localement les erreurs. Après définition formelle de notre proposition, nous avons mesuré que cette approche dégrade les performances sauf dans le cas du réseau convolutif. La deuxième méthode proposée s'appuie sur la détection automatique d'un point de rupture dans le signal pour améliorer la phase d'entraînement. Le recours à ce dernier nous permet d'adapter le modèle de prédiction de la durée de vie restante après ré-entraînement. Cette seconde approche améliore significativement les performances et démontre la généricité de notre approche.

Références

Ellefsen, A. L., V. Æsøy, S. Ushakov, et H. Zhang (2019a). A comprehensive survey of prognostics and health management based on deep learning for autonomous ships. *IEEE Transactions on Reliability 68*(2), 720–740.

Ellefsen, A. L., E. Bjørlykhaug, V. Æsøy, S. Ushakov, et H. Zhang (2019b). Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. *Reliability Engineering & System Safety 183*, 240–251.

Gouriveau, R., K. Medjaher, et N. Zerhouni (2016). *From Prognostics and Health Systems Management to Predictive maintenance 1 : Monitoring and Prognostics*. John Wiley & Sons.

Gugulothu, N., V. Tv, P. Malhotra, L. Vig, P. Agarwal, et G. Shroff (2017). Predicting remaining useful life using time series embeddings based on recurrent neural networks. *2nd ACM SIGKDD Workshop on Machine Learning for Prognostics and Health Management, Halifax, Canada*.

Hashemian, H. M. (2010). State-of-the-art predictive maintenance techniques. *IEEE Trans. on Instrumentation and Measurement 60*(1), 226–236.

Janssens, O., V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, et S. Van Hoecke (2016). Convolutional neural network based fault detection for rotating machinery. *J. of Sound and Vibration 377*, 331–345.

Khan, S. et T. Yairi (2018). A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing 107*, 241–265.

Mao, W., J. He, Y. Li, et Y. Yan (2017). Bearing fault diagnosis with auto-encoder extreme learning machine : A comparative study. *Proceedings of the Institution of Mechanical Engineers, Part C : J. of Mechanical Engineering Science 231*(8), 1560–1578.

Saxena, A., K. Goebel, D. Simon, et N. Eklund (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *IEEE Intl. Conf. on Prognostics and Health Management*, pp. 1–9.

Schwabacher, M. et K. Goebel (2007). A survey of artificial intelligence for prognostics. In *AAAI Fall Symposium : Artificial Intelligence for Prognostics*, pp. 108–115.

Tao, S., T. Zhang, J. Yang, X. Wang, et W. Lu (2015). Bearing fault diagnosis method based on stacked autoencoder and softmax regression. In *IEEE 34th Chinese Control Conf. (CCC)*, pp. 6331–6335.

Truong, C., L. Oudre, et N. Vayatis (2020). Selective review of offline change point detection methods. *Signal Processing 167*, 107299.

Summary

In this article, we are interested in improving the prediction of the remaining useful operating time of a complex system whose state is represented by multivariate time series. We present and evaluate two approaches for measuring the improvement of the *Remaining Useful Life* (RUL) prediction using four different state-of-the-art machine learning approaches based on deep learning. The first method that we propose is based on re-sampling the training data set in order to reduce the errors locally. The second proposed method relies on automatically detecting and using break-points in the signals to improve the training step. We show that break-point detection techniques allow a significant improvement of the RUL prediction performance with gains of more than 27% on the mean absolute error (MAE) regardless of the neural architecture used, which demonstrates the genericity of our approach.