REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET
DE LA RECHERCHE SCIENTIFIQUE
UNIVERSTE DE BATNA 2
FACULTE DE TECHNOLOGIE
DEPARTEMENT DE GENIE INDUSTRIEL

THESE

PRÉSENTEE AU LABORATOIRE D'AUTOMATIQUE ET PRODUCTIQUE

POUR L'OBTENTION DU DIPLÔME DE DOCTORAT

Spécialité : Génie industriel

Option : informatique industrielle et productique

Par

BERGHOUT Tarek

Master en Génie Industriel, option informatique industrielle et productique

THEME

Proposition d'un système distribué de diagnostic et pronostic basé sur les services Web et Extreme Learning Machine.

Soutenue le : 29/04/2021.
Devant le Jury composé de :

BENOUDJIT	Nabil	Professeur	Président	Université de	Batna-2
MOUSS	Leila Hayet	Professeur	Rapporteur	Université de	Batna-2
KADRI	Ouahab	MC-A	Co-Rapporteur	Université de	Batna-2
TARI	Abdelkamel	Professeur	Examinateur	Université de	Bejaia
BOUZGOU	Hassen	Professeur	Examinateur	Université de	Batna-2
BENBOUZID	Mohamed El Hachemi	Professeur	Invité	Université de	Bretagne Occidentale

Année universitaire : 2021

Remerciements

L'ensemble des travaux présentés dans cette thèse a été effectué au Laboratoire d'Automatique et Productique (LAP.) de l'université de Batna 2. Je tiens à exprimer ma profonde gratitude à ma directrice de thèse, **Leïla Hayet MOUSS** Professeur à l'Université de de Batna-2 et Directrice du LAP, qui m'a encadré tout au long de cette thèse et qui m'a fait partager ses brillantes intuitions. Qu'elle soit aussi remerciée pour sa gentillesse, sa disponibilité permanente et pour les nombreux encouragements qu'elle m'a prodigués. C'est aussi l'occasion de la remercier pour les connaissances qu'elle nous a donné durant notre formation: Graduation & Post graduation au sein du département de Génie Industriel de l'université de Batna-2.

Je remercie également **Ouahab KADRI**, Maître de Conférences à l'Université de Batna-2, pour avoir accepté de co-encadrer ce travail de recherche, dirigé et m'avoir fait bénéficier de son expérience et de ses précieux conseils.

J'adresse mes sincères remerciements à Monsieur **Mohamed EL Hachemi BENBOUZID**, Professeur à l'Université Occidentale de Brest, pour ses conseils, remarques et encouragements durant la réalisation de ce projet de recherche.

Je remercie vivement **Toufik BENTRCIA**, Docteur à l'Université de Batna-2 pour son aide précieuse lors de la préparation et de la présentation de mes contributions dans mon projet de recherche.

Je remercie Monsieur **Nabil BENOUDJIT**, Professeur à l'université de Batna-2 d'avoir accepté le double rôle d'examinateur et de président du jury de cette thèse.

J'adresse mes sincères remerciements à Messieurs **Abdelkamel TARI** Professeur à l'Université Bejaia, **Hassen BOUZGOU**, Professeur à l'Université de Batna-2 **Mohamed EL Hachemi BENBOUZID** Professeur à l'Université Occidentale de Brest qui ont accepté d'évaluer ce travail et de participer au jury de soutenance

Mes remerciements s'adressent aussi à Monsieur **Lotfi SAÏDI,** Docteur à l'Université de Sousse, en Tunisie, Monsieur **Elhoussin ELBOUCHIKHI,** Docteur à l'ISEN, Brest, pour leur aide inestimable dans mes contributions à ce projet de recherche.

Mes sincères remerciements à toutes les personnes qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé mes réflexions et ont accepté de me rencontrer et de répondre à mes questions durant la préparation de ce travail de recherche.

Résumé **Cette thèse traite de l'utilisation d'un outil de Machine Learning appelé "Extreme Learning Machine" dans le domaine du pronostic et de l'évaluation de la santé (*e.g. estimation de la durée de vie utile restante (RUL)*). Cet algorithme implique des méthodes de programmation linéaires qui reposent principalement sur les moindres carrés dans les paradigmes non linéaires des réseaux de neurones artificiels pour produire des estimateurs de santé très rapides et précis. D'autres parts, cette étude vise à distribuer le système de prédiction sur le web à l'aide de services Web pour résoudre les problèmes de répartition géographique de surveillance décentralisée. Sur la base de ce nouvel outil, de nombreux algorithmes d'apprentissage ont été développés dans le cadre de ce travail et comparés à d'autres algorithmes présents dans la littérature en termes de temps et de précision. La plupart des algorithmes développés sont inspirés des théories récentes de Deep Learning et ce en raison de leur bonne réputation. Les données étudiées dans ce travail de recherche sont tirées du logiciel C-MAPSS simulateur de système de propulsion aérodynamique développé par la NASA. Les résultats obtenus ont prouvé l'efficacité des nouveaux algorithmes et les recommandent pour une utilisation future dans le domaine de l'évaluation de la santé.**

Mots-clés **Pronostic ; diagnostic ; ELM ; C-MAPSS ; RUL ; Services Web ; deep learning**

Abstract **This thesis deals with the use of a machine learning tool called "Extreme Learning Machine" in the field of prognostic and health assessment (*e.g. Remaining useful life (RUL) estimation*). This algorithm involves linear programming methods that rely primarily on least squares within the nonlinear paradigms of artificial neural networks to produce very fast and accurate health estimators. On the other hand, this study aims to distribute the prediction system on the web using Web services to solve the geographic distribution and decentralized monitoring problems. On the basis of this new tool, many learning algorithms have been developed within the framework of this work and compared to others in the literature in terms of time and precision. Most of our contributions in these developed algorithms were inspired from the basis of recent Deep Learning theories due to their successful reputation. The data which were studied in this work are taken from the C-MAPSS software which is a commercial modular aero-propulsion system simulator developed by NASA. Obtained results proved the effectiveness of the new algorithms and recommend them for future use in health assessment field.**

Keywords **Prognosis ; diagnosis ; ELM ; C-MAPSS ; RUL ; Web Services ; deep learning.**

مُلَخَّصٌ

تُعَالِجُ هَذِهِ الأُطْرُوحَةُ تَقَصِّي إِمْكَانِيَةَ اسْتِخْدَامِ خَوَارِزْمِيَةٍ إِي إِل إِم فِي مَجَالِ التَّعْلِيمِ الأُوتُومَاتِيكِيِّ لِلْمَاكِينَاتِ الحَاسِبَةِ وَ ذَلِكَ لِغَرَضِ تَطْبِيقِهَا أَثْنَاء التَّقْيِيمِ الذَاتِيِّ لِلْأَنْظِمَةِ الشغَّالَةِ قَبْلَ وبَعْدَ ظُهُورِ أَعْطَابٍ مُخْتَلِفَةٍ عَلَى مُسْتَوَاهَا. تَتَضَمَّنُ هَذِهِ الخَوَارِزْمِيَةُ طُرُقَ البَرْمَجَةِ الخَطِّية التِي تَعْتَمِدُ بِشَكْلٍ أَسَاسِيٍّ عَلَى المُرَبَّعَاتِ الدُنْيَا دَاخِلَ النَّمَاذِجِ غَيْرِ الخَطِّيَةِ لِلشَّبَكَاتِ العَصَبِيَّةِ الاصْطِنَاعِيَةِ لِإِنْتَاجِ دَوَالٍ مُقَارِبَةٍ سَرِيعَةٍ وَدَقِيقَةٍ لِلْغَايَة. مِنْ نَاحِيَةٍ أُخْرَى ، تَهْدِفُ هَذِهِ الدِّرَاسَةُ إِلَى نَشْرِ نِظَامِ التَّنَبُؤِ عَلَى الوِيبْ بِاسْتِخْدَامِ خَدَمَاتِ الوِيبْ فِي حَلِّ مَشَاكِلِ التَّوزِيعِ الجُغْرَافِيِّ وَ كَذَا اللاَّمَرْكَزِيَّةِ فِي مُرَاقَبَةِ سَيْرُورَةِ هَذِهِ الأَنْظِمَةِ. بِنَاءً عَلَى أَسَاسْ هَذِهِ الأَدَاةِ الجَدِيدَةِ، تَمَّ تَطْوِيرُ العَدِيدِ مِنْ خَوَارِزْمِيَاتِ التَّعْلِيمِ وَتَمَّتْ أَيْضًا مُقَارَنَتُهَا مَعَ خَوَارِزْمِيَّاتٍ أُخْرَى مِنْ حَيْثُ الوَقْتُ والدِّقَّةُ. تَمَّ اسْتِلْهَامُ مُعْظَمُ مُسَاهَمَاتِ الأُطْرَوحَةِ فِي هَذِهِ الخَوَارِزْمِيَاتِ المُطَوَّرَةِ مِنْ أُسُسٍ وَ نَظَرِيَاتِ التَعْلِيمِ المُعَمَّقِ الحَدِيثَةِ نَظَرًا لِسُمْعَتِهَا النّاجِحَةِ. أُخِذَتِ البَيَانَاتُ التِّي تَمَّتْ دِرَاسَتُهَا فِي هَذَا البَحْثِ مِنْ بَرْنَامَج "سِي مَابْسْ" وَهُوَ عِبَارَةٌ عَنْ بَرْنَامَج كُمْبْيُوتَرْ مُحَاكٍ لأَنْظِمَةِ الدَّفْعِ الجَوَّيةِ ذَوَاتِ التُرْبِينَاتِ النَّفَاثَةِ الذِّي تَمَّ تَطْوِيرُهُ مِنْ طَرَفِ وَكَالَةِ نَاسَا الأَمْرِيكِيَّةِ لِأَبْحَاثِ الفَضَاءِ. أَثْبَتَتِ النَّتَائِجُ المُتَحَصَّلُ عَلَيْهَا فَعَالِيَةَ الخَوَارِزْمِيَاتِ الجَدِيدَةِ كَمَا تُوصِي هته النَتَائِجُ بِاسْتِخْدَامِهَا المُسْتَقْبَلِيِّ فِي مَجَالِ التَّقْيِيمِ الذَاتِيِّ.

كُلِمَاتٌ مِفْتَاحِيَةٌ تَشْخِيصُ الأَعْطَالِ، خَوَارِزْمِيَاتِ الشَّبَكَاتِ العَصَبِيَّةِ الاصْطِنَاعِيَةِ، خَدَمَاتِ الوِيبْ، أَنْظِمَةِ الدَّفْعِ الجَوَّيةِ.

Proposition d'un système distribué de diagnostic et pronostic basé sur les services Web et Extreme Learning Machine.

Sommaire

Liste des figures

Chapitre I

Chapitre II

Chapitre III

Chapitre IV

Chapitre VI

Liste des tableaux

Chapitre II

Chapitre IV

Chapitre V

Chapitre VI

Liste des Algorithmes

Chapitre IV

Chapitre V

Chapitre VI

Liste des abréviations

ACP = Analyse en Composantes Principales (Principal Component Analysis).
C-MAPSS = Commercial Modular Air Propulsion System Simulation.
CNN = Convolutional Neural Networks.
CPU = Central Processing Unit.
DBN = Deep Belief Neural network.
DFF = Dynamic Forgetting Factor.
ELM = Extreme Learning Machine.
ELM-LRF = ELM with Local Receptive Fields.
K-means = k-moyennes.
K-ppv = k Plus Proche Voisin (k Nearest Neighbors).
MAE = Mean Absolute Error.
OS-ELM = Online Sequential Extreme Learning Machine.
PHM = Prognostic and Health Management.
PSNR = Peack Signal to Noise Ratio (Rapport Signal Sur Bruit).
RBM = Restricted Boltzmann Machine (Machine de Boltzmann Restreinte).
ReLU = Rectifier Linear Unit.
R-ELM = Regularized ELM.
RMSE = Root Means Squared Error (Racine Carrée de l'Erreur Quadratique Moyenne).
RNA = Réseau de Neurones Artificiels (Artificial Neural Networks).
RNN = Recurrent Neural Network (Réseau de Neurone Récurrent).
R-OSELM = Regularized OS-ELM.
RUL = Remaining Useful Life.
SMW = Sherman Morison and Woodbury.
SRC = Sparse Representation Classification.
SVD = Singular Value Decomposition
SVM = Support Vector Machine
USS = Updated Selection Strategy.

Introduction générale

1. Positionnement de la recherche

Suite au développement massif des technologies industrielles, la complexité des procédures de manipulation des systèmes industriels en termes de surveillance, de maintenance et de répartition géographique devient plus accrue [1,2].

L'évaluation de la santé de ces systèmes sous la répartition géographique est devenue le sujet de nombreuses études récentes qui visent à simplifier et à centraliser les opérations de gestion de la santé des systèmes sans les moindres coûts.

L'une des principales solutions à ces problèmes consiste à impliquer ou à intégrer des plateformes de prédiction et d'évaluation de la santé (systèmes de diagnostic ou de pronostic) des systèmes d'exploitation des unités entières dans le Cloud en fonction des services offerts par le Web. En projetant les systèmes de surveillance sur le Cloud, le processus de surveillance sera alors disponible à tout moment, quelque soit la répartition géographique [3,4].

Cependant, la non-similitude entre les équipements et leur flexibilité augmente la complexité du développement d'algorithmes de surveillance adaptatifs en temps réel d'une part et le coût d'utilisation des services web en raison de la multi-variété des algorithmes conçus. La complexité et la multitude des algorithmes (si nous supposons que ce sont des estimateurs précis) utilisés peuvent retarder l'arrivée des informations nécessaires à la situation actuelle des équipements d'exploitation ainsi que la réactivité du processus d'intervention des maintenances.

Dans ce contexte, nous pouvons conclure que l'un des principaux problèmes de la surveillance des processus industriels réside fondamentalement dans la précision, la fiabilité et du degré de complexité des systèmes de diagnostic et de prédiction des causes de défaillances [5].

Par conséquent, la recherche actuelle repose sur trois fondements principaux:

- Réduction de la complexité de l'algorithme de prédiction.
- Suivi de l'état de santé des équipements en fonction des variables physiques et environnementales.
- Centralisation du système de prédiction en le projetant sur le Cloud.

Le principal objectif de cette thèse est de simplifier à la fois le processus de modélisation de tout système en utilisant des paradigmes non complexes et précis pour pouvoir aborder des estimateurs plus fiables et contribuer à la réduction des coûts de calcul et aux coûts des services de Cloud lors de la projection du système de surveillance sur le web.

2. Formalisation et outils

De nombreux modèles de prédiction tels que les modèles physiques, mathématiques, statistiques et numériques, ont été développés pour tenter de résoudre le problème de l'estimation précise en considérant les coûts de calcul et l'intervention humaine pendant leur conception ou l'exploitation (prédiction des nouvelles données inconnues).

Les modèles physiques sont les modèles les plus authentiques dans lesquels il est permis de simuler le comportement d'un tel système avec un niveau de précision plus élevé [6]. Ils modélisent mathématiquement l'interaction intérieure du composant du système en s'appuyant sur une connaissance approfondie de sa conception et de son attitude. En utilisant plusieurs techniques tels que (réseaux de pétri, espaces d'états, chaines de Markov, etc.), la modélisation est simplifiée à un certain niveau. Cependant si la complexité du système (en

termes de conception) augmente, le processus de modélisation en incluant chaque élément sera quasiment impossible. Le processus de conception sera alors limité à certaines conditions définies par l'expert en fonction de leur importance, ce qui est le résultat d'une mauvaise généralisation [7].

La modélisation mathématique et l'analyse statistique du comportement des systèmes dépendent généralement d'une analyse d'échantillons qui peut ou non permettre une généralisation, surtout lorsqu'elle se situe dans une programmation dynamique où les données évoluent très rapidement.

Par conséquent, nous avons constaté que la plupart des recherches actuelles tendent à étudier le comportement des données basé sur les outils de «Machine Learning», en particulier lorsque les données sont disponibles en raison de l'évolution de la technologie des capteurs [8].

Théoriquement, les outils de «Machine Learning» permettent de modéliser mathématiquement le comportement de tout système sans même le voir ou avoir une connaissance préalable de ses sous-systèmes. En étudiant uniquement la distribution des données et en améliorant leur représentation. En se basant uniquement sur de celles-ci (sans expertise approfondie sur le système étudié), le processus de prédiction ne sera pas plus complexe que les autres modèles.

Les outils «Machine Learning» avec différents paradigmes tels que les algorithmes d'ensemble, hybride et Deep Learning ont montré un grand succès dans le processus de modélisation basé sur les données. Néanmoins, ils ne peuvent pas être parfaits et présentent quelques faiblesses[9]. Nous pouvons citer:

- Ils sont gourmant en temps en raison de l'utilisation des algorithmes itératifs de l'apprentissage tels que la rétropropagation pour les réseaux de neurones et les réseaux bayésiens, la divergence contrastive, le Intelligence en essaim et les algorithmes génétiques.

- Le problème des critères d'arrêt des locaux minima qui reste un problème ouvert pour les algorithmes de l'apprentissage itératifs.
- Le taux d'apprentissage et beaucoup des hyper-paramètres similaires qui tentent de trouver le meilleur réglage en fonction d'une certaine quantité de données.

Dans ce contexte, où les principaux problèmes traités sont les coûts de calcul, la complexité algorithmique et la projection sur le Web, nous nous dirigeons vers le développement d'un outil d'apprentissage le plus adapté capable de traiter ces problèmes avec moins d'interventions humaine et de coûts de calcul et permettant simplement une projection sur le web à moindre coût. Les résultats de la recherche nous amènent à utiliser les outils de l'apprentissage les plus récents appelés Extreme Learning Machine (ELM) connus pour leur vitesse et leur précision.

3. Contributions de la thèse

L'ELM apparaît comme un outil très sophistiqué basé sur l'imitation du cerveau humain en abordant l'apprentissage biologique de manière plus cohérente que les anciens algorithmes itératifs tels que la rétropropagation ou la divergence contrastive. L'ELM peut faire apprendre de manière flexible avec n'importe quel type de réseau neuronal artificiel (Multicouches, monocouche, feedforward, etc) et avec n'importe quelle architecture (ordinaire, deep Learning). En effet, il peut se diriger vers des paradigmes très complexes tels que le Deep Learning et les rend très simples à manipuler.

Par conséquent, en impliquant l'ELM dans l'évaluation de la santé d'un système, ce serait un saut géant qui pourrait améliorer le processus de modélisation et de faire la surveillance de nombreuses manières comme:

- L'ELM est en mesure de traiter à la fois l'apprentissage hors ligne et en ligne sans réglage itératif, les problèmes de minimum local ou le réglage du taux d'apprentissage, ce qui lui permet de minimiser le temps de modélisation et les autres coûts de calcul (les coûts matériels tels que le CPU et le GPU).
- L'ELM est capable de traiter très facilement l'apprentissage adaptatif basé sur les variations de données et les variables environnementales comme la température ambiante, la pression atmosphérique, etc..
- La simplicité des règles d'apprentissage de l'ELM basée sur la méthode des moindres carrés permet son utilisation sur des ordinateurs plutôt que dans des stations très coûteuses contrairement à d'autres algorithmes.
- ELM est capable de gérer les «Big-Data», même pour ceux dont leurs échantillons sont contaminés par le bruit.
- En introduisant un outil de régularisation simple, l'ELM peut éviter le «overfitting» facilement sans affecter le temps d'apprentissage.
- La chose magique à propos des théories d'apprentissage de l'ELM est que pour les problèmes généraux non complexes, un réseau neuronal à une seule couche cachée peut être suffisant pour satisfaire la généralisation et l'approximation universelle sans avoir besoin d'une structure multicouches.
- En couplant l'ELM avec des services web, il n'est plus nécessaire d'avoir de grands espaces sur les serveurs ou même des fonctionnalités de sécurité plus élevées en raison de la taille plus petite de l'algorithme et de la grande capacité d'apprentissage qui contribue directement à la réduction des coûts financiers.

D'une part, et en fonction du cycle de fonctionnement d'un tel système, les caractéristiques physiques peuvent être exposées à certaines variations provoquées par une modification soit

des variables environnementales, soit de l'état physique de santé. Ces changements conduiront le modèle d'apprentissage conçu à une erreur de prédiction. D'autres parts, les mesures enregistrées des capteurs à plusieurs niveaux de santé (sain, critique et malsain) peuvent être dynamiquement perturbées par les mêmes contraintes précédentes [10].

Ainsi, en essayant d'adapter le modèle d'apprentissage aux contraintes mentionnées précédemment, plusieurs algorithmes ont été développés pour extraire les meilleurs indicateurs même en cas d'évolution de santé en proposant les approches développées suivantes:

- L'ELM pour la compression des mesures des capteurs afin de réduire la complexité du Big-Data avec prise en compte de l'apprentissage adaptatif.
- Les réseaux de neurones adaptatifs «Deep Belief Neural Network» (DBN) basés sur l'ELM pour l'estimation de la durée de vie restante pour résoudre les problèmes liés à l'estimation de l'état de santé d'un système.
- Les réseaux de neurones adaptatifs de débruitage de type DBN basés sur l'ELM pour l'estimation de la durée de vie restante pour résoudre les problèmes liés à la l'estimation de l'état de la santé selon les contraintes de mesures bruitées.

4. Organisation du manuscrit

Pour des raisons illustratives et pour simplifier la compréhension de la manière d'utilisation de l'ELM dans le domaine du diagnostic et de l'évaluation de la santé d'un système, cette thèse est organisée en deux parties.

La première partie, « **Extreme Learning Machine : vers un nouvel outil pour l'évaluation de la santé** », traite de l'évolution de «Machine Learning» dans les modèles d'évaluation de la santé et de la raison du passage à l'outil ELM. Elle est composée de trois chapitres distincts. Le

chapitre I est consacré à la présentation de l'état de l'art de l'évolution de la philosophie diagnostic vers le pronostic et l'évaluation de la santé en passant par des concepts fondamentaux. **Le chapitre II** explique comment exploiter les modèles de «Machine Learning» Dans ce dernier est présenté également les différents types de classifications, les défis et les tendances possibles de modèles de prédiction. **Le chapitre III** est consacré aux règles d'apprentissage et les variantes principales de l'ELM sur lesquels nous nous sommes concentrés dans notre travail.

Comme le processus de prédiction ne se concentre pas uniquement sur l'apprentissage supervisé et que le prétraitement des données est la caractéristique clé pour une estimation précise, la deuxième partie « **Applications de l'Extreme Learning Machine dans le domaine du pronostic et de l'évaluation de la santé** » introduit respectivement plusieurs nouveaux outils d'apprentissage automatique développés sur la base de l'ELM pour satisfaire plusieurs conditions d'apprentissage supervisé et non supervisé. Cette partie est composée de trois chapitres également.

Le chapitre IV présente un nouvel autoencodeur d'apprentissage en ligne adaptatif développé pour améliorer la compression des données dans des caractéristiques variant dans le temps. **Le chapitre V** quant à lui illustre l'utilisation de l'apprentissage adaptatif basé sur l'estimation du RUL à condition que les données ne soient pas contaminées par le bruit. Enfin **le chapitre VI** présente un nouvel outil d'apprentissage adaptatif pour l'estimation du RUL dans le cadre de la réduction du bruit dans les données. Enfin, nous terminons par une conclusion et quelques perspectives de travail qui nous semblent intéressante d'explorer à la suite de cette thèse.

Partie 01:

Extreme Learning Machine : vers un nouvel outil pour l'évaluation de la santé

Du diagnostic vers le pronostic pour l'évaluation de la santé d'un système industriel

Résumé:

Ce chapitre présente les concepts de base du diagnostic et pronostic du point de vue des industriels.

Il présente également un bref aperçu sur les méthodes utilisées aussi bien pour le diagnostic que le pronostic et propose une méthodologie d'une analyse structurée jusqu'à l'obtention d'une conclusion fiable sur l'état de santé ou la cause de défaillance d'un système industriel.

1. Introduction

Les fonctions de diagnostic et pronostic jouent un rôle très important dans les différentes opérations de maintenance telles que la réparation, la prévention ou l'amélioration. L'étude du principe et l'historique de fonctionnement des systèmes ou des équipements peut contribuer à l'optimisation des programmes d'intervention et de minimisation des coûts et des ressources de la maintenance.

Pour optimiser la politique de maintenance et augmenter la fiabilité d'un système quelconque, il est nécessaire de suivre une démarche appropriée et bien définie pour une prédiction immédiate ou préalable, des causes des défaillances probables. Dans la littérature cette démarche est connue sous le nom de diagnostic ou pronostic, selon les types de prédiction. Ce chapitre est réservé à la présentation des concepts de base du diagnostic et du pronostic.

2. Concepts de base

Avant d'aller plus loin, il est nécessaire d'identifier les concepts de base du diagnostic et du pronostic liés au contexte de notre travail. Par conséquent, cette section est dédiée spécialement aux concepts les plus pertinents utilisés dans notre travail.

- Système

> *D'un point de vu général, un système peut être défini comme «un ensemble d'éléments reliés entre eux d'une manière prédéfinie pour assurer une fonction requise».*

D'après la figure I.1, dans n'importe quel système, la sortie est le résultat de transformation de l'entrée dont des conditions ou des commandes prédéfinies. Selon [11], un système industriel est composé de deux processus élémentaires interconnectés, associés à des actionneurs et des chaines de mesures fournissent à tout instant des informations complètes ou partielles sur l'état du système.

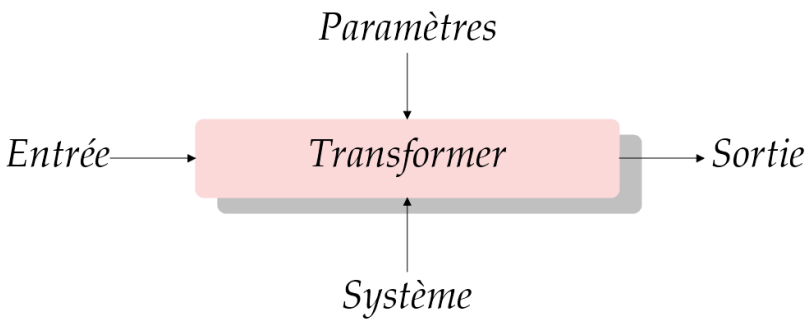

Figure: I.1. Structure générale d'un système.

- **Fonctionnement normal et anormal:**

> *Un système est dit dans un état de fonctionnement normal lorsque les variables le caractérisant (variable d'état, variable de sortie, variable d'entrée, paramètres des systèmes) demeurent au voisinage de leur valeurs nominales. Il est dit dans un état de fonctionnement anormal dans le cas contraire* [11].

- **Surveillance:**

> *La surveillance est une technique informationnelle qui repose sur le diagnostic des variations de paramètres de fonctionnement du système pour l'évaluation de son état. La surveillance consiste notamment à décider de la nécessité d'une reconfiguration ou d'une opération de maintenance afin d'éviter/corriger la défaillance* [12,13].

- **Symptôme:**

> *Un symptôme est un signe qui correspond à une ou plusieurs observations qui révèlent un dysfonctionnement Il s'agit d'un effet qui est la conséquence d'un comportement anormal* [14].

- **Défaillance:**

> *Une défaillance est l'altération ou la cessation de l'aptitude d'un ensemble à accomplir une ou des fonctions requises avec les performances définies dans les spécifications techniques.*

Une défaillance est un dysfonctionnement du système, le processus présente alors un fonctionnement indésirable du point de vue performances [11].

- **Dégradation:**

> *Une dégradation représente une perte de performance d'une des fonctions assurées par un système. Si les performances sont au-dessous du seuil d'arrêt défini dans les spécifications fonctionnelles de cet équipement, il n'y a plus dégradation mais défaillance* [15].
>
> *La dégradation peut être liée à des facteurs directs, tels que l'usage, la durée d' l'usage, …, ou à des facteurs indirects, tels que l'humidité, la température...etc.* [14].

3. Diagnostic

3.1. Définition

Dans la littérature, plusieurs interprétations sont adoptées pour la définition du mot diagnostic. La première apparition du mot diagnostic a été dans le domaine médical, où il est défini comme:

> *Action de déterminer une maladie d'après ses symptômes.*

Après cela, l'utilisation de la philosophie du diagnostic s'étend pour s'adapter à plusieurs domaines (tels que l'économie, le sport et l'industrie) avec différentes démarches de l'application.

Dans le domaine industriel, Gilles Zwingelstein l'un des pères de diagnostic industriel utilise la définition de la norme AFNOR-CEI pour présenter la signification de mot diagnostic:

> *Le diagnostic est l'identification de la cause probable de la (ou des) défaillances(s) à l'aide d un raisonnement logique fondé sur un ensemble d'informations provenant d'une inspection, d'un contrôle ou d'un test.* [13].

Depuis, dans le même domaine, cette définition a connu de nombreuses évolutions, selon les besoins des chercheurs et les types de méthodes utilisées[1]. On peut trouver une autre définition du mot diagnostic:

> *Détermination des causes à partir des effets c'est un problème de classification, puisque, il s'agit de comparer un vecteur de données qui représente l'état actuel du système avec des vecteurs références qui représentent l'historique du fonctionnement du système* [16].

Dans le contexte de notre travail, nous prenons en considération cette définition:

> *Le diagnostic de tout système peut être considéré comme une opération consistant à classer des observations liées à des causes spécifiques à une classe spécifique de symptômes, qui peuvent indiquer le mode actuel de défaillance après un dysfonctionnement.*

3.2. La démarche de diagnostic

Le processus de diagnostic nécessite une méthodologie prédéfinie à suivre pour la détection et localisation des défaillances. La **figure I.2** illustre les principales étapes de la démarche de diagnostic.

3.2.1. Validation des mesures

La procédure de diagnostic nécessite de disposer d'informations sur l'état de fonctionnement du système à surveiller [11]. L'opération de la validation des mesures consiste à la mise en forme des caractéristiques associées aux fonctionnements normaux et anormaux, à partir de moyens des mesures apportées ou observations réalisées lors des rondes par le personnel de surveillance [13].

[1] Le lecteur peut utiliser par exemple une méthode de diagnostic basée sur la reconnaissance de formes, où la définition peut avoir une autre interprétation orientée vers le domaine du traitement de données (ex: classification d'une observation, approximation d'une fonction, etc).

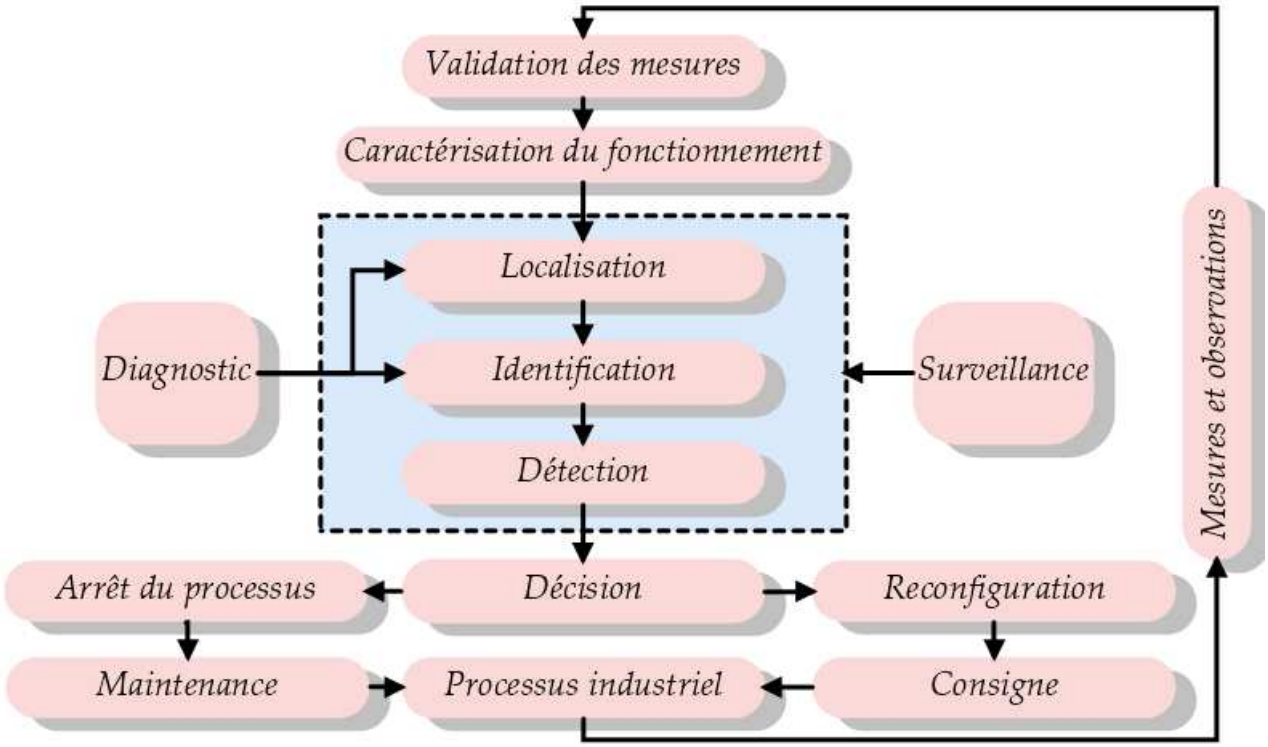

Figure: I.2. Différentes étapes de diagnostic industriel [11,13,14]**.**

Les méthodes de validation des mesures peuvent être classifiées en deux grandes catégories : les méthodes classiques et les méthodes avancées.

a. Les méthodes classiques

Les méthodes classiques sont des démarches de validation qui dépendent des caractéristiques des données mesurées sans modification ou test de crédibilité.

b. Les méthodes avancées

Le but de méthodes avancées est de fournir des informations très fiable et crédible à partir de mesures originales après la vérification de ces cohérences et des vraisemblances.

Un ensemble d'outils de validation utilisés dans les deux types de méthodes est illustré dans la **figure I.3**. Un aperçu de ces méthodes et de leur utilisation se trouve dans [13] (voir Ch III, § 3.3).

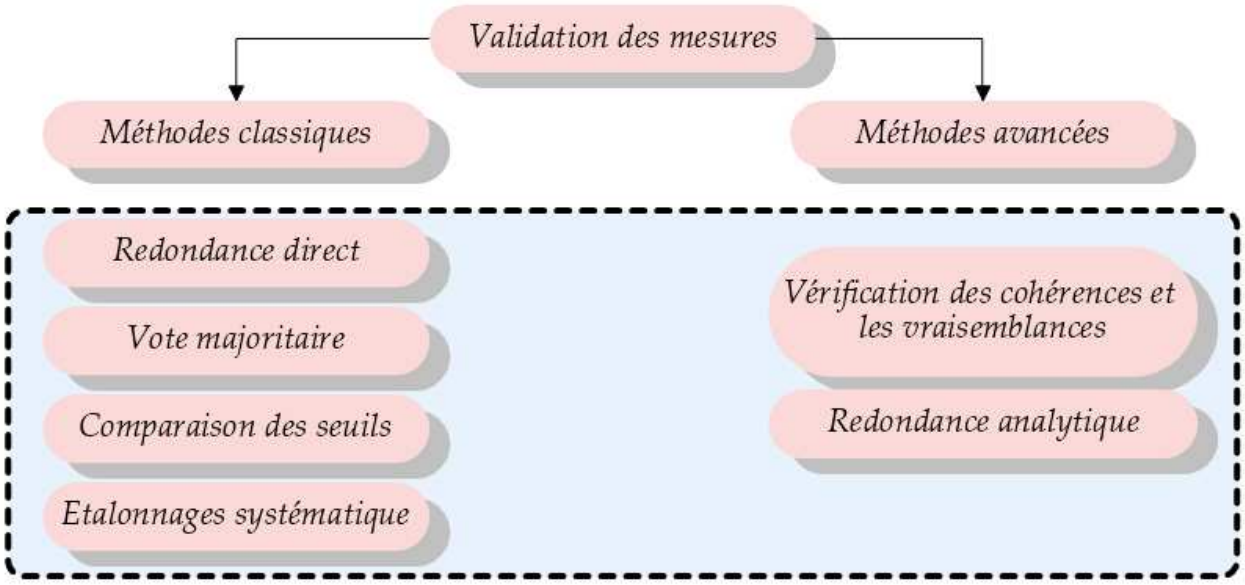

Figure: I.3. Méthodes communes de validation des mesures.

3.2.2. Caractérisation du Fonctionnement

Cette étape a été dédiée au traitement des données brutes mesurées après leur validation. Plusieurs techniques dans le domaine temporel ou fréquentiel peuvent être utilisées selon les types de signaux accumulés (signaux vibratoires, signatures acoustiques, signatures ultrasonores), pour l'extraction des paramètres qui doivent caractériser le fonctionnement (normal ou anormal).

- ✓ **Localisation:** La localisation ou l'isolation d'une défaillance consiste à remonter les symptômes pour retrouver l'ensemble des éléments défaillants. Ce problème est difficile à résoudre. En effet, il est possible de déterminer une défaillance, ou une panne résultant d'un défaut. Par contre, le problème inverse est plus difficile à résoudre, puisque une panne peut résulter d'un ou plusieurs défauts [14].
- ✓ Identification: Elle consiste à déterminer les causes qui sont à l'origine d'une situation anormale. Ces causes peuvent être internes ou bien externes au système.

- ✓ Détection: Elle consiste à classifier les défaillances d'un système selon des situations observées, à l'une des catégorie de fonctionnement ; normale ou anormale [15].
- ✓ Décision: La décision finale du système de diagnostic est faite selon le résultat de système de surveillance. Si le système de surveillance signale qu'il y a un fonctionnement anormal, la décision est de faire l'opération de maintenance appropriée. Le test d'hypothèse est l'une des approches les plus utilisées pour confirmer avec un degré de confiance donné la fiabilité de la décision. Une description générale de ces tests se trouve dans [13] (voire Ch4**).**

3.3. Classification des méthodes de diagnostic

L'évolution de la technologie dans le domaine industriel produit une diversité des approches et méthodes de diagnostic selon le type de système abordé (thermique, mécanique, électronique,…) [17]. Plusieurs classifications sont proposées dans la littérature selon les types de traitement ou d'analyse de données. Dans cette partie nous allons présenter les classifications les plus récentes. Cette classification est inspirée de [11],[13] et [17].

3.3.1. Classification selon la connaissance reliée au système

a. Les méthodes internes

Les méthodes internes de diagnostic sont des méthodes basées sur la connaissance profonde de système étudié. A partir de cette connaissance, nous pouvons modéliser et simuler son fonctionnement. Cette approche est considérée parmi les meilleures approches dédiées à l'estimation de l'état réel du système durant le fonctionnement pour les systèmes non complexes. Lorsque le système étudié devient plus complexe (en terme de nombre de sous-composants, nature de l'interaction (non-linéarité)...etc.), le processus de modélisation devient très difficile voire impossible.

b. Les méthodes externes

Les méthodes externes sont utilisées lorsque l'historique de fonctionnement (cause-effet) est valable. Pour ce type de méthodes, le système est considéré comme une boite noire, et la relation entre l'entrée et la sortie est inconnue. La relation (cause → effet) va modéliser avec une formule mathématique qui permet de relier l'entrée et la sortie.

3.3.2. Classification selon le mode de raisonnement

a. Les méthodes inductives

Ces méthodes correspondent à une approche ascendante où l'on identifie toutes les combinaisons d'événements élémentaires possibles qui entrainent la réalisation d'un événement unique indispensable [18]. La méthode de l'**A**rbre **d**e **D**éfaillance (AdD) est l'une des méthodes qui représente par excellence une démarche inductive.

b. Les méthodes déductives

Pour les méthodes déductives, la démarche est inversée puisqu'on part d'un événement indésirable et on recherche ensuite par une approche descendante, pour identifier toutes les causes possible [18]. L'analyse AMDE (**A**nalyse de **M**odes de **D**éfaillance et leur **E**ffets) est une méthode déductive où les conséquences des causes des défaillances des composants élémentaires sont systématiquement identifiées.

Une explication bien détaillé de la démarche de construction de la AdD et AMDE se trouve dans [11,13], [19] et [20].

3.3.3. Classification selon le type de modélisation

a. Les méthodes qualitatives :

Ces méthodes sont utilisées pour modéliser la relation cause→effet, si la modélisation physique de comportement de ***fonctionnement normal*** du système/sous système étudié est compliqué voire impossible. Généralement les outils de «Machine Learning » tels que les

Réseaux de Neurones Artificiels (RNA), les techniques de « Swarm Intelligence (SI) », Arbre de Décision (Itérative Dichotomiser 3 : ID3 [21]), etc., sont utilisés dans ce type de diagnostic. L'effet de défaillance peut être une donnée nominale (classe de fonctionnement, ex : 'bon fonctionnement', 'défaillance de type quelconque', 'fonctionnement anormal'), ou numérique qui décrit cet effet lui-même (ex: température, pression, bruit acoustique…etc.)

b. Les méthodes quantitatives :

Ce type de méthodes est réservé à la modélisation de système/sous système étudié. Alors, une analyse mathématique doit être engagée pour le prétraitement, traitement, évaluation et la prise de discision.

La **figure I.4** illustre la démarche de diagnostic qualitative et quantitative dans un processus industriel.

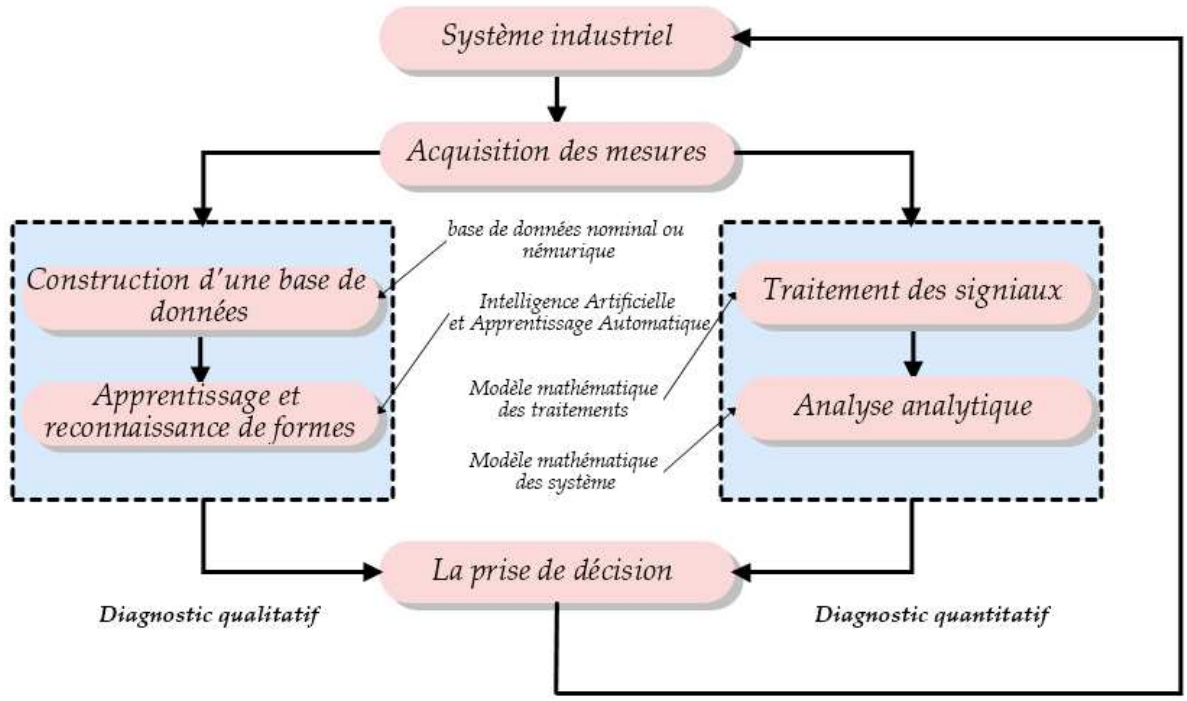

Figure: I.4. Méthodes de diagnostic qualitatives et quantitatives.

3.4. Choix de la méthode de diagnostic

Le choix de la méthode de diagnostic adéquate pour un système/sous système dépend de deux critères principaux à savoir la disponibilité des données et la complexité. Si le système n'est pas complexe et on peut le modéliser, l'approche «à base de modèles» peut être utilisée, et le diagnostic devient quantitatif interne. Sinon, si le système devient très difficile à modéliser et seul les données de l'historique de fonctionnement sont valables, une modélisation mathématique de la relation (cause → effet) est la seule solution valable, le type de diagnostic devient qualitatif externe. Aussi le système d'évaluation de la santé se concentrera sur l'étude des formes (une étude qualitative (patterns)) et des signes importants entraînés par des mesures de capteurs (mesures effectuées à l'extérieur du système (externe)) pour évaluer l'état de santé du système.

4. Pronostic

Beaucoup de définitions du terme pronostic ont été proposées et il n'en existe pas de totalement consensuelle. Un trait marquant peut cependant être identifié :

4.1. Définition

> *Le pronostic est souvent assimilé à un processus de prédiction (une situation future doit être appréhendée). Deux grandes catégories du pronostic peuvent être considérées comme fédératrices. Il désigne selon les cas un processus visant à déterminer la durée de vie restante d'un système, c'est-à-dire sa RUL (durée de fonctionnement avant défaillance), ou la probabilité pour que le système fonctionne durant un certain temps.*

> *D'un point de vue général, le pronostic est un diagnostic prédictif basé sur la même démarche que le diagnostic. La seule différence réside dans la philosophie du pronostic utilisée pour prédire les causes et localiser les organes ayant entraînés une dégradation particulière pour éviter définitivement la défaillance. Si les performances sont au-dessous du seuil d'arrêt défini dans les spécifications fonctionnelles de l'équipement, la procédure de diagnostic doit être mise en place pour la détection après la défaillance.*

Comme le diagnostic contribue à minimiser les coûts de maintenance corrective, l'opération de pronostic permet d'améliorer et minimiser les coûts totaux de maintenance à partir de l'amélioration des programmes de maintenance préventive conditionnelle.

Dans le cadre de notre travail on définit le pronostic comme:

> *Le pronostic est une formule mathématique relative au résultat de la meilleure approximation de la fonction (observation → indicateur de santé).*

4.2. Indicateurs de la santé d'un système

Plusieurs indicateurs peuvent être utilisés selon le type de méthodes utilisées dans le système de diagnostic ou pronostic, nous présentons les plus utilisés.

a. Le résidu

Dans les méthodes internes, le calcul de l'erreur entre la valeur désirée (obtenue à partir du modèle mathématique physique) et la valeur réelle (actuelle) peut être utilisé comme un indicateur de la santé de système. Plusieurs formules mathématiques de l'erreur peuvent être utilisées comme le RMSE (Root Mean Squered Error), le MAE (Mean Absolute Error), PSNR (Peak Signal to Noise Ratio) …etc.

b. Le temps de vie restante

Le temps de vie restante (RUL : Remaining Useful Life) est l'indicateur principal de la santé des équipements. En d'autres termes, l'erreur elle même peut être utilisée pour la détermination du RUL. Il peut être déterminé en utilisant les méthodes externes ou internes. Dans le cadre de notre travail, nous avons préconisé les méthodes externes car elles sont les plus étudiées et les plus pertinentes.

4.3. Types de prédiction du RUL

Dans le cas de la prédiction du RUL, plusieurs types de résultats peuvent apparaître. Ils sont classifiés selon l'erreur de prédiction $(\tilde{y} - y)$ en deux catégories: prédiction ultérieure et prédiction antérieure.

a. Prédiction ultérieure

Si la prédiction du RUL a une erreur positive (le RUL estimé est supérieur au RUL désiré). La prédiction est considérée comme une prédiction ultérieure. Cette prédiction a un inconvenant majeur de mettre le système étudié au risque d'endommagement avec une probabilité très élevée. C'est pour cette raison que plusieurs chercheurs dans la littérature se sont focalisé fondamentalement sur la minimisation de l'erreur de ce type de prédiction [22].

b. Prédiction antérieure

La prédiction dans le cas inverse où le RUL estimé est inférieur au RUL désiré, permet d'éviter l'endommagement des systèmes opératoires. Néanmoins, elle reste parfaitement inefficace parce qu'elle consomme plusieurs ressources de maintenance [7].

La **figure I.5** est un exemple d'une prédiction de RUL selon une dégradation linéaire ou les types d'estimation sont adressés.

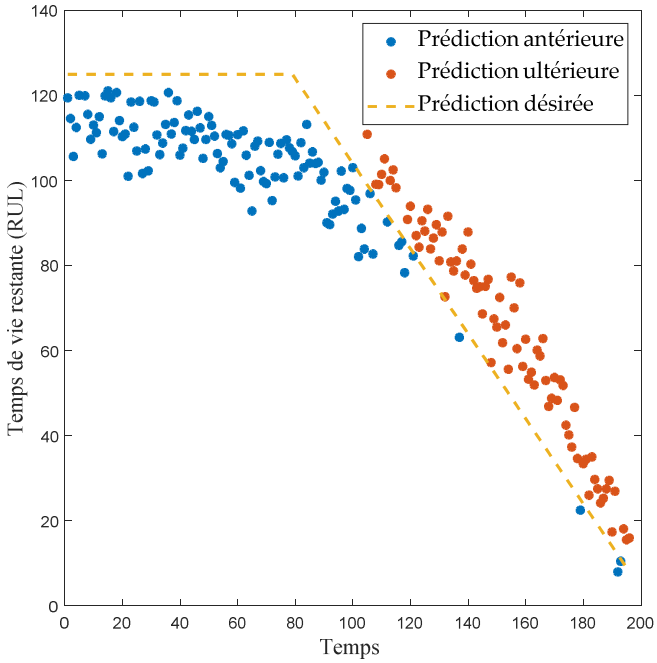

Figure: I.5. Types de prédiction du RUL

Selon la détérioration linéaire de la fonction RUL souhaitée, l'erreur des prédictions antérieure et ultérieure est la quantité de prédiction manquée de l'estimateur du RUL souhaité.

4.4. L'estimation du RUL

Compte tenu du fait que le pronostic est, par essence, un processus incertain, il est intéressant de pouvoir juger de sa qualité dans le but d'imaginer des actions plus adéquates ; dans ce sens, plusieurs indicateurs peuvent être construits. Ce sont les **mesures de performance du système de pronostic** ; les principales mesures mises en avant dans la littérature sont l'*exactitude*, la *précision* et *l'opportunité* des estimations [23].

a. L'exactitude :

C'est un indicateur de performance du modèle de prédiction qui représente une mesure de la proximité de la date de défaillance prévue avec la date de défaillance réelle.

Le calcul de cette métrique représente un point critique dans le processus de pronostic. La question qui se pose tout d'abord est si la prédiction est "assez bonne". La réponse dépend fortement de la rigidité des critères d'évaluation imposés [15].

Pour cela, le calcul de cette grandeur s'appuie sur l'existence de données historiques qui décrit des profils de dégradations des systèmes similaires au système étudié.

Si un ensemble de N mesures sont identifiées par leurs valeurs réelles du RUL, on peut définir l'exactitude du système de prédiction en utilisant la formule mathématique **[I.1]**.

$$S = \sum_{i=1}^{N} e^{(\tilde{y}_i - y_i)/\mu} \qquad \textbf{[I.1]}$$

où:

$\tilde{y}$ et y : sont respectivement le RUL prédit et désiré.

μ : est un facteur de normalisation. Une constante dont les valeurs dont la valeur est basée sur l'importance de la valeur réelle dans l'application[2].

La fonction exponentielle est utilisée ici pour donner une fonction asymétrique par rapport à l'origine comme illustré dans la **figure I.6**.

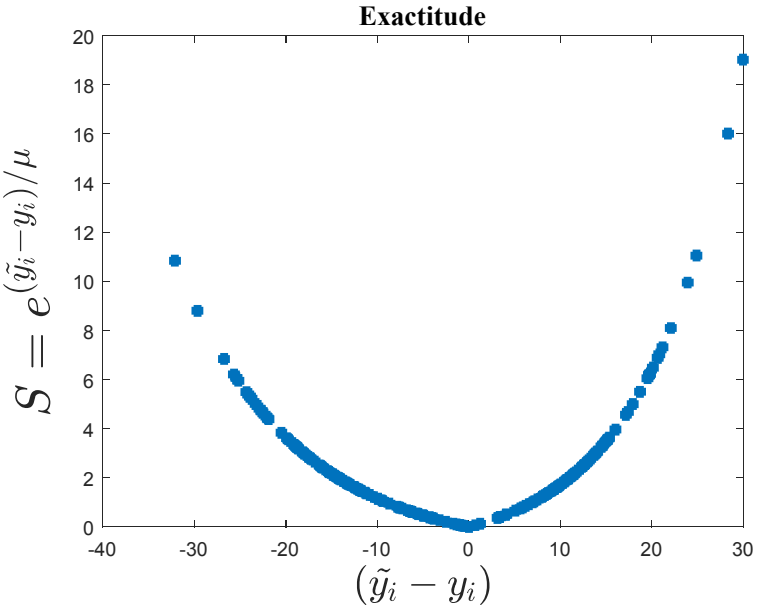

Figure: I.6. Exemple du comportement de la fonction de précision

Si nous exposons les valeurs d'erreur des types de prédiction du RUL (antérieure et ultérieure **figure I.6**) à la fonction d'exactitude, elle se comportera de la même manière illustré dans la **figure I.7**.

[2] Ce paramètre peut être attribué différemment pour chaque type de prédiction (antérieure et ultérieure) pour des raisons de maintenance. De manière générale, les prédictions ultérieures sont plus nuisibles et exposent le système à plus de danger. Les prévisions antérieures ne sont pas nuisibles, mais plus de consommateurs pour les ressources de maintenance. Par conséquent, la prédiction latente devrait être plus pénalisée.

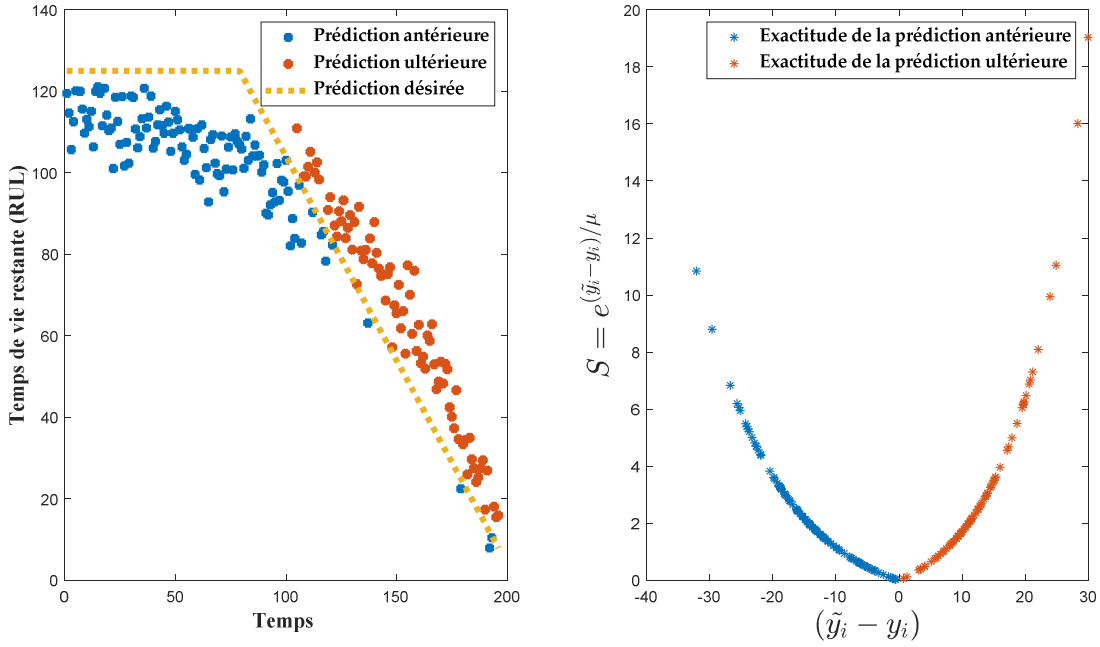

Figure: I.7. Comportement des types de prédiction dans une fonction d'exactitude

b. Précision

La précision est une mesure de dispersion des prédictions du RUL. Elle permet d'évaluer comment les valeurs prédites sont groupées autour de l'intervalle dans lequel survient la défaillance. La précision dépend fortement du niveau de confiance et de la distribution des prédictions ([I.2], [I.3]et [I.4]) [23]:

$$S = \left(\frac{1}{N} \sum_{i=1}^{N} e^{(r_i / r)} \right) e^{\frac{\frac{1}{N}\sum_{i=1}^{N}(E_i - \overline{E})^2}{\sigma}} \quad \textbf{[I.2]}$$

$$\overline{E} = \frac{1}{N} \sum_{i=1}^{N} E_i \quad \textbf{[I.3]}$$

$$E_i = 100(\tilde{y}_i - y_i) / y_i \quad \textbf{[I.4]}$$

Les grandeurs r et σ sont des facteurs de normalisation, et r_i est l'intervalle de confiance de la prédiction pour l'expérimentation. De même, une fonction exponentielle est employée ici pour définir les relations entre l'écart-type de la prédiction, l'intervalle de confiance et la précision. La précision a une valeur entre 1 et 0 (1 indiquant la précision la plus élevée et 0 la plus basse).

c. Opportunité (Timeliness)

Il existe d'autres métriques comme **l'horizon de pronostic (horizon of prédiction)** qui mesure la différence entre l'instant où la première prédiction entre dans un intervalle de confiance prédéfini et la date réelle de la défaillance. Plus la valeur de l'horizon est grande, plus le pronostic est performant. Egalement la **Convergence, une autre métrique** qui quantifie l'évolution dans le temps des mesures traditionnelles, comme l'exactitude et la précision.

4.5. Classification générale des modèles de prédiction du RUL

Selon [24], les modèles de prédiction du RUL peuvent être classés en trois catégories: à savoir les modèles basés sur la survie, ceux basés sur la dégradation et les modèles basés sur la similarité.

4.5.1. Les modèles à base de la survie

L'analyse de survie est une méthode statistique utilisée pour modéliser les données (temps → événement). Elle est utile lorsque nous n'avons pas d'historique complet de fonctionnement normal jusqu'à la défaillance, mais que nous avons à la place:

- Les données sur la durée de vie des systèmes similaires. Par exemple, nous pourrions avoir le nombre de tours qu'un moteur a tourné avant d'avoir besoin de maintenance, ou le nombre d'heures de fonctionnement d'une machine avant de tomber en panne. Dans ce cas, nous utilisons «les modèles de survie à base de fiabilité ». Compte tenu des informations historiques sur les temps de défaillance d'un ensemble de système similaire au système étudié. Ce modèle estime la distribution de probabilité des temps de défaillance.

- La durée de vie et les variables environnementales (des informations tels que les régimes dans lesquels les systèmes doivent être utilisés, exemples: température ambiante, pression à l'extérieur, etc.). Dans ce cas, nous utilisons «les modèles de survie à base de variables environnementales. ». C'est un modèle de survie à risque proportionnel qui utilise les durées de vie et les variables environnementales pour calculer la probabilité de survie d'un système.

4.5.2. Les modèles à base de dégradation

Ces modèles extrapolent le comportement passé pour prédire la condition future. Ce type de calcul du RUL s'adapte à un modèle linéaire ou exponentiel au profil de dégradation d'un indicateur de condition, compte tenu des lois de dégradation d'un système. Ils utilisent ensuite cette loi pour calculer statistiquement le temps restant jusqu'à ce que l'indicateur atteigne un certain seuil prescrit. Ces modèles sont particulièrement utiles lorsqu'une valeur connue de l'indicateur d'état indique une défaillance. Les deux types de modèles de dégradation disponibles sont:

- Modèle de dégradation linéaire: Il décrit le comportement de dégradation comme un processus stochastique linéaire. Les modèles de dégradation linéaire sont utiles lorsque le système ne subit pas de dégradation cumulative (grandit progressivement).

- Modèle de dégradation exponentiel: Il décrit le comportement de dégradation comme un processus stochastique exponentiel. Les modèles de dégradation exponentielle sont utiles lorsque le composant de test subit une dégradation cumulative (croît de façon exponentielle).

Après avoir créé un modèle de dégradation, l'initialisation du modèle doit être faite à l'aide de données historiques concernant l'intégrité d'un ensemble de systèmes similaires, telle que plusieurs machines fabriquées selon les mêmes spécifications.

Les deux types des modèles de dégradation sont illustrés dans la **figure I.8**.

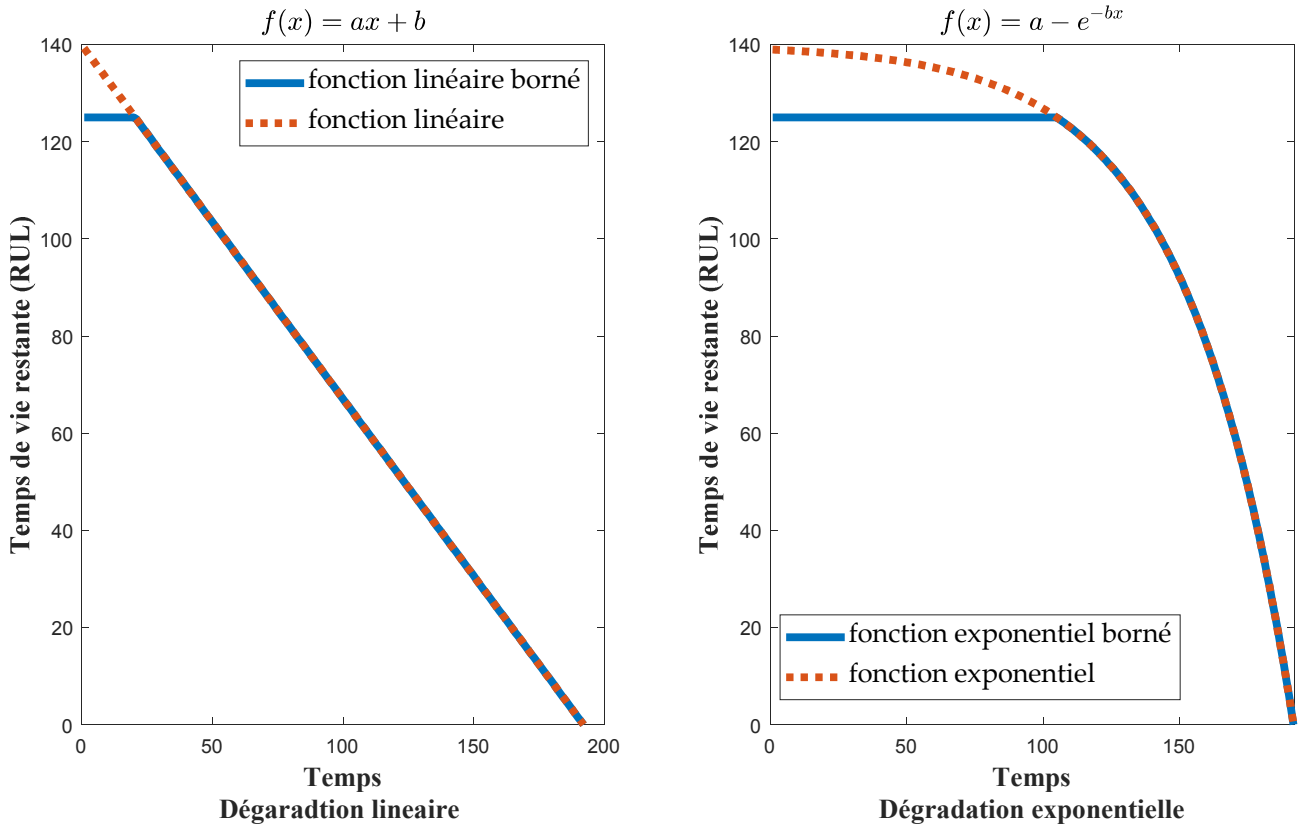

Figure: I.8. Types des modèles de dégradation.

Sur la **figure I.8**, la dégradation linéaire et exponentielle peuvent avoir deux interprétations différentes (fonction de dégradation bornée et ordinaire) selon le type de système étudié. A titre d'exemple, les batteries perdent progressivement leur énergie de charge dès qu'elles sortent du chargeur, donc la meilleure fonction de dégradation peut être la fonction linéaire ordinaire. Autre exemple concernant les machines tournantes, elles fonctionnent normalement à un certain moment, et elles commencent à perdre des performances, donc la meilleure description de ce type de détérioration serait une fonction linéaire bornée.

4.5.3. Les modèles à base de similarité

Ils se basent sur la prédiction du RUL d'un système à partir d'un comportement connu des machines similaires à partir d'une base de données historique. Ces modèles comparent une

tendance des valeurs des indicateurs de performances aux mêmes informations extraites d'autres systèmes similaires.

Les modèles de similarité sont utiles lorsque:

- Les données de fonctionnement normal jusqu' à la défaillance des systèmes similaires, sont valable. Ces données commencent pendant un fonctionnement sain et se terminent lorsque la machine est dans un état proche d'une défaillance.

- Les données montrant des comportements de dégradation similaires sont valable, c'est à dire que les données changent d'une manière caractéristique à mesure que le système se dégrade. La **figure I.9** présente la classification la plus récente du modèle de prédiction des RUL.

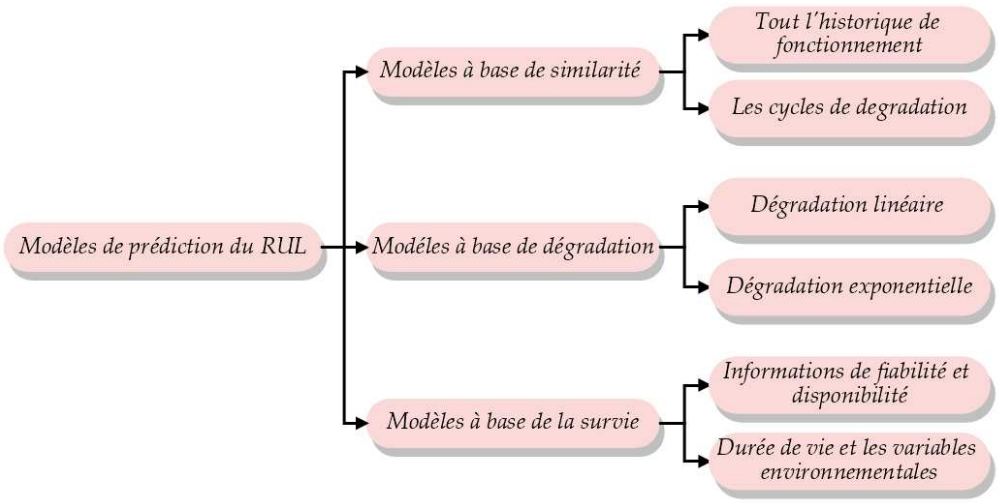

Figure: I.9. Classification des modéles de préduction du RUL.

5. Positionnement des fonctions de diagnostic et pronostic

La **figure I.10** indique la position occupée par les opérations de diagnostic et de pronostic dans différents scénarios de fonctionnement d'un système industriel.

Le système de pronostic a été intégré pour la surveillance en temps réel durant les conditions de fonctionnement du système industriel. Le temps consommé par la surveillance dans ce cas

ne cesse pas l'activité du système. Contrairement à l'algorithme de diagnostic qui prend un moment pour localiser et identifier la cause de défaillance. En effet, le processus de diagnostic a démarré après l'apparition de défaillance. Le temps de réparation dépendra de la précision de l'algorithme de prédiction.

Dans ce schéma, différentes évolutions de l'état de la santé durant le fonctionnement d'un système sont présentées avec un indicateur de performance. Dans le cas de fonctionnement normal, le système de pronostic essaie de détecter en temps réel s'il existe des symptômes de dégradation pouvant conduire à une défaillance. Si un symptôme est détecté, une politique de maintenance doit s'appliquer pour résoudre le problème. Sinon, si cette dégradation évolue vers un mode de défaillance, alors le système de diagnostic sera responsable de la détection de la cause. Contrairement à l'algorithme de diagnostic qui est un détecteur juste-à-temps (instantanément), la politique de pronostic fonctionne en temps réel (permanent). Le schéma actuel explique que l'intégration de ces deux fonctions produira des systèmes d'évaluation et de surveillance de la santé plus précis.

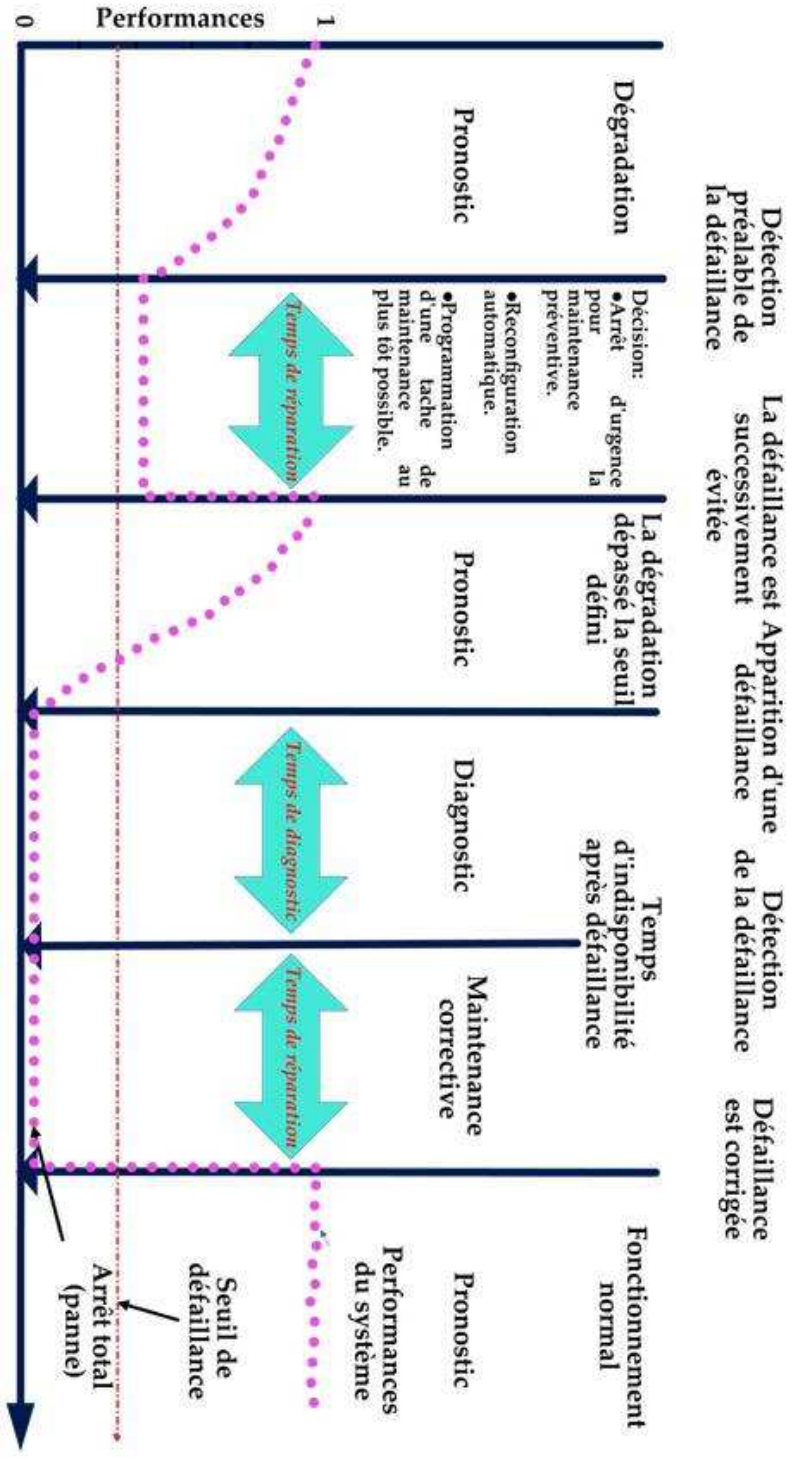

Figure: I.10. Exemple de positionnement des tâches de diagnostic et de pronostic .

6. Conclusion

Dans ce chapitre, les notions de diagnostic et pronostic ont été expliquée dans le contexte de notre travail. Le travail actuel consiste à utiliser les techniques récentes de « Machine Learning » pour l'estimation de l'état des systèmes dont le but de détecter et isoler les causes de défaillances. Ainsi, la prédiction dans ce cas, devient une prédiction par reconnaissance des formes. Pour le système de diagnostic, un algorithme de classification doit être utilisé pour modéliser la relation (effet → cause) ou bien (symptôme → cause). Pour le système de pronostic, l'algorithme de prédiction doit être utilisé pour définir la fonction mathématique (symptôme → indicateur de santé). Dans ce contexte, le chapitre suivant est dédié à une étude plus détaillée des modèles de prédiction et leur évolution en fonction des besoins des systèmes de diagnostic et pronostic.

Machine Learning pour le diagnostic et pronostic: état de l'art

Résumé:

Ce chapitre présente une solution générale de toute prévision basée sur les outils de Machine Learning pour l'évaluation de la santé d'un système, en abordant les étapes les plus importantes de la construction d'un estimateur précis.

Ce chapitre présente également les différents types de classification des outils de l'apprentissage les plus utilisés dans la littérature en fonction des besoins des opérations de pronostic et de gestion de la santé.

Enfin, ce chapitre identifie les problèmes des outils de Machine Learning et expliquant les raisons de l'utilisation de nouveaux outils tels que Extreme Learning Machine pour les modélisations futures du comportement des données et de leur utilisation.

1. Introduction

La prédiction de l'état de santé des équipements ou des sous-systèmes complexes basés sur des méthodes conventionnelles telles que les approches de modélisation physiques est devenue une tâche très difficile. En raison de la nécessité d'une connaissance approfondie des composants du système et de leurs interactions internes, la complexité du processus de modélisation a augmenté et est devenue presque impossible. Même si le modèle final peut être préparé dans des conditions limitées, les résultats peuvent induire en erreur la prédiction ou ne pas être suffisamment précis en raison d'une mauvaise généralisation.

De nos jours, dans les applications de Machine Learning pour l'estimation de l'état de santé des systèmes, un saut important a été fait en raison de la disponibilité de données hétérogènes, ce qui a motivé les chercheurs à renforcer le paradigme de prédiction conventionnelle avec une variété d'approches de prédiction.

Les améliorations continues des modèles de Machine Learning rendent leur utilisation dans le pronostic et le management de la santé (**P**rognostic and **H**ealth **M**anagement: PHM) plus pertinente. Ils permettent de modéliser le comportement des systèmes en extrayant uniquement les informations importantes de leurs données récupérées même sans connaissance préalable de leurs caractéristiques internes. Contrairement aux paradigmes conventionnels, les techniques de Machine Learning visent à réduire la complexité de la modélisation avec moins d'intervention humaine et moins de coûts de calcul.

Ce chapitre présente l'évolution de l'utilisation des outils de Machine Learning dans le domaine des PHM selon une variété d'axes.

2. Concepts de Machine Learning

Cette section est dédiée à lever les ambiguïtés et barrières entre nous et les concepts de machine learning les plus importants que nous utiliserons dans notre étude.

- ✓ **Machine Learning:**

> *Technologie de l'intelligence artificielle qui permet aux ordinateurs d'apprendre, de comprendre et de prendre des décisions sur la base d'algorithmes construits pour analyser et bénéficier des données historiques disponibles* [25].

- ✓ **Données étiquetées et non étiquetées:**

> *Les données étiquetées constituent un groupe d'échantillons étiquetés avec une ou plusieurs informations. Par exemple, dans notre cas, les étiquettes peuvent indiquer si une observation porte une information sur un fonctionnement normal ou anormal, ou bien une information sur la santé du système étudié. Les étiquettes peuvent être obtenues d'après un historique prédéfini du système étudié ou bien à partir d'une connaissance approfondie du fonctionnement de se système. Les données non étiquetées constituent un groupe d'échantillons qui n'ont aucun étiquetage* [26].

- ✓ **Apprentissage supervisé:**

> *Dans le cas d'un apprentissage supervisé, le modèle d'apprentissage doit trouver ces paramètres pour satisfaire la fonction d'approximation qui permet de lier une observation à son étiquette. Par exemple, si nous supposons qu'il existe une fonction linéaire qui prend un ensemble d'observations (mesures) à un ensemble de modes de fonctionnement (normal, anormal ... etc.), l'apprentissage dans ce cas consiste à régler les paramètres de la fonction linéaire pour satisfaire également l'approximation et généraliser cette fonction sur cet ensemble* [27].

- ✓ **Apprentissage non-supervisé:**

> *En général, l'apprentissage non-supervisé est un apprentissage qui ne dépend que de l'étude de données non-étiquetées. Il peut être utilisé pour une multitude d'applications telle que la classification ou le clustering à base d'outils tels que le SVM, k-means et le k-ppv, ou l'extraction et la représentation des paramètres des observations utilisant des méthodes telles que les autoencodeurs* [27].

✓ **Observation:**

> *Est une entité unique qui contient un ensemble de caractéristiques combinées (paramètres qui ont une sorte de relation) pour décrire certaines informations. Les caractéristiques peuvent être numériques, nominales ou ordinales. Pour qu'une information doit être suffisante pour un apprentissage supervisé elle doit contenir un couple de données (paramètres, étiquètes).*

✓ **Classification:**

> *La classification vise à trouver un modèle de l'affectation d'un objet à une classe. Ce modèle est basé sur la reconnaissance du représentant de cet objet. Les applications typiques de classification comprennent des tâches de reconnaissance de formes comme l'image, caractère ou parole* [12]. *Dans le domaine de l'automatique, la classification est également utilisée pour des applications de diagnostic.*

✓ **Régression:**

> *La régression est une approche d'estimation permettant d'interpoler, voire d'extrapoler ou de prédire une sortie à variables réelles, à partir des entrées de l'espace d'apprentissage. Cette approche est basée sur la construction d'un modèle capable de calculer la fonction d'estimation, qui passe parfaitement par tous les points de l'espace* [12].

✓ **Hyper-paramètre:**

> *Les hyper-paramètres sont plutôt des propriétés qui régissent l'ensemble du processus de l'apprentissage. Ils comprennent des variables qui déterminent la structure du modèle de prédiction (ex: nombre de neurones cachés, nombre de couches cachées) et les variables qui déterminent la façon dont le modèle est entraîné (par exemple, le taux d'apprentissage, le paramètre de régularisation, etc.). Les hyper-paramètres du modèle sont définis avant le processus d'apprentissage, puis ajustés après le processus de l'apprentissage selon la meilleure approximation*[28]. *Les hyper-paramètres sont importants car ils contrôlent directement le comportement de l'algorithme d'apprentissage, ayant un impact important sur les performances du modèle en cours de l'apprentissage.*

✓ **Overfitting:**

> *C'est un problème de modélisation qui se produit lorsqu'une fonction d'approximation est trop proche d'un ensemble limité de points de données. Le overfitting du modèle prend généralement la forme d'un modèle trop complexe pour expliquer les particularités des données à l'étude* [29].

> *Ainsi, tenter de rendre le modèle trop conforme aux données légèrement inexactes peut infecter le modèle avec des erreurs substantielles et réduire son pouvoir prédictif* [30].

- ✓ **Le risque empirique:**

> *La minimisation des risques empiriques est un principe de la théorie de l'apprentissage statistique qui définit une famille d'algorithmes d'apprentissage et est utilisé pour donner des limites théoriques à leurs performances. L'idée centrale est que nous ne pouvons pas savoir exactement dans quelle mesure un algorithme fonctionnera dans la pratique (le véritable «risque») parce que nous ne connaissons pas la véritable distribution des données sur lesquelles l'algorithme fonctionnera, mais nous pouvons plutôt mesurer ses performances sur un ensemble connu de données de l'apprentissage (le risque «empirique»).*

- ✓ **Le risque structurel:**

> *La minimisation des risques structurels est un principe inductif utilisé dans Machine Learning. le but de l'apprentissage est généralement de trouver un modèle qui offre de bonnes performances de généralisation sur une distribution sous-jacente des données* [31]. *Le principe de la minimisation du risque structurel résout ce problème en équilibrant la complexité du modèle et son succès à ajuster les données d'apprentissage* [32].

3. Solution générale de la prédiction à base de Machine Learning

Dans le cas général, la démarche suivie pour résoudre n'importe quel problème de prédiction avec les outils de Machine Learning est représenté dans la **figure II.1**.

Comme nous pouvons le voir, le modèle de prédiction peut être construit en passant par quatre étapes fondamentales à savoir la préparation des données, l'apprentissage, le prétraitement des résultats et l'évaluation du modèle.

3.1. La préparation des données

Le processus de préparation en lui-même peut passer par plusieurs étapes. Dans cette section, seuls les plus importants sont présentés.

3.1.1. Sélection des caractéristiques:

La première étape importante dans la préparation des données est la sélection ou la validation des paramètres, où seuls les éléments les plus significatifs d'une seule observation doivent être élus pour le processus de l'apprentissage.

Le processus de sélection doit suivre une certaine étude de corrélation pour confirmer que la variation des cibles reflète la variation de l'entrée d'origine. Des outils tels que les coefficients de corrélation [33] et la matrice Jacobéenne des dérivatives [34] peuvent être utilisés pour cette tâche.

3.1.2. Nettoyage de données:

Le nettoyage de données est l'opération de détection et de correction (ou suppression) d'erreurs présentes sur des données stockées dans des bases de données ou dans des fichiers.

Les données présentes dans les bases de données peuvent avoir plusieurs types d'erreurs comme des erreurs de mesures, des informations manquantes, des imprécisions,… etc.

La partie impropre de la donnée traitée peut être remplacée, modifiée ou supprimée. Le processus de nettoyage identifie les données erronées et les corrigés automatiquement avec un programme informatique ou les proposer à un humain pour qu'il effectue les modifications. Des outils d'exploration de données tels que l'ACP et des algorithmes de reconstruction tels

que les autoencodeurs et le « Compressed Senssing[3] » [35] peuvent être utilisés pendant le remplissage des données manquantes.

[3] « Compressed Senssing » est une technique de Machine Learning qui tente de résoudre des problèmes d'apprentissage basée sur la minimisation de la norme L_1 [35].

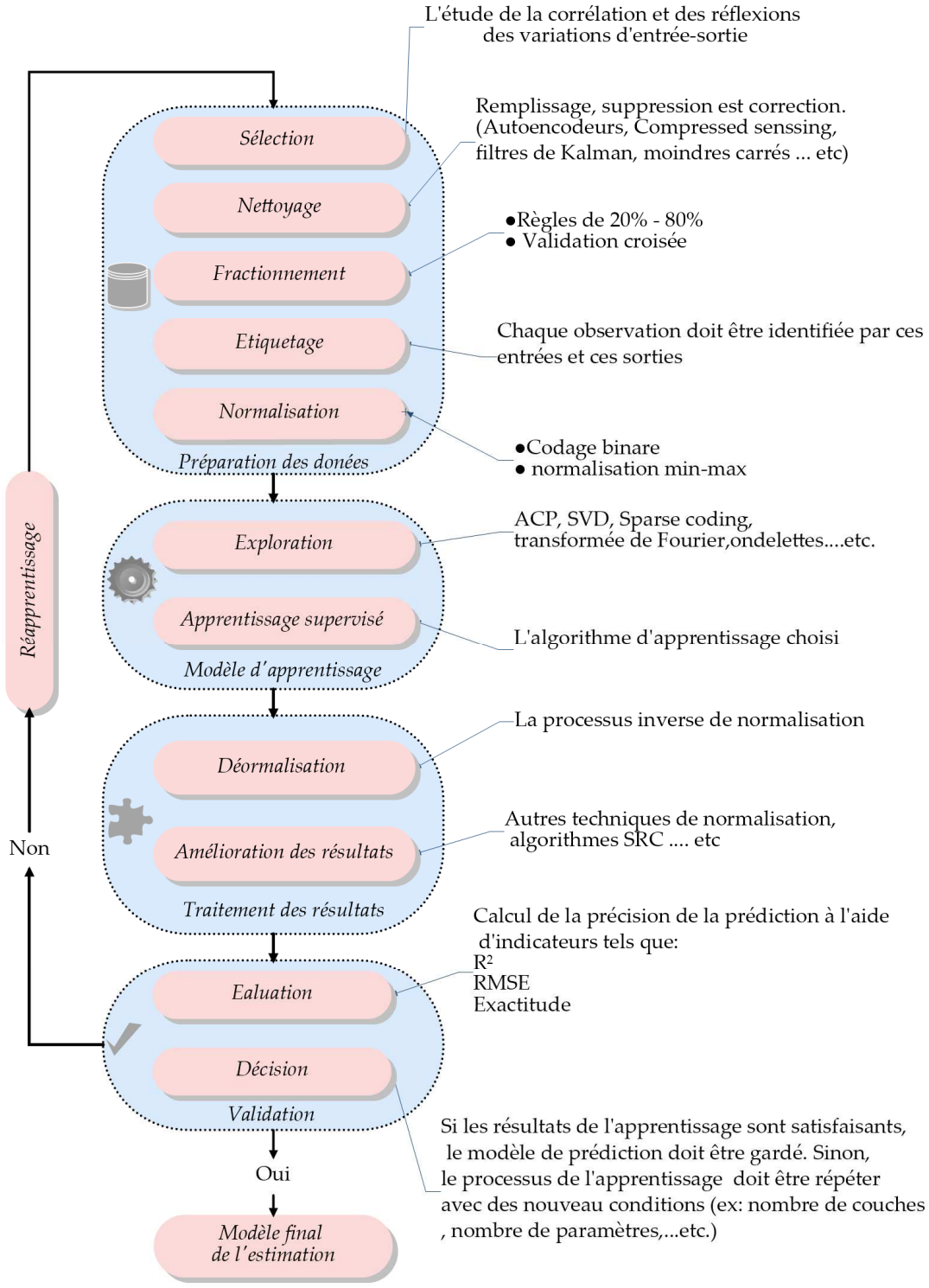

Figure: II.1. Démarche générale de la prédiction par les outils de Machine Learning.

3.1.3. Fractionnement des données

Avant que le processus de l'apprentissage ne se déclenche, les données doivent être divisées en sous-ensembles d'apprentissage et de test pour garantir la généralisation de la fonction d'approximation après validation sur les deux ensembles. Dans les cas généraux où la distribution des données est moins compliquée, des règles bien connues de (20%, 80%) sont utilisées pour l'identification des ensembles de test et d'apprentissage respectivement. Cependant, lorsque la complexité du modèle augmente, des méthodes de devisions telles que la validation croisée seront utilisées pour identifier le meilleur ensemble d'apprentissage.

3.1.4. Définition d'étiquettes

Les étiquettes de l'apprentissage seront identifiées par des experts. Certaines des étiquettes de données peuvent être trouvées dans l'ensemble de données lui-même et d'autres seront définies par l'utilisateur avant le processus de l'apprentissage en fonction du comportement de la fonction cible qui ne peut être proposé que par les experts eux-mêmes. Par exemple, selon [24], la fonction de dégradation d'un système est en fait inconnue, mais, son comportement peut être supposée par l'utilisateur avec deux cas; comportement exponentiel ou linéaire (voir chapitre I, § 4.4).

3.1.5. Normalisation

Cette étape utilise deux chemins différents pour les modèles de classification et de régression. Dans certains modèles de classification tels que les réseaux de neurones, les étiquettes des classes doivent être transformées en forme binaire pour satisfaire les limites des fonctions d'activation et les entrées doivent également être ajustées pour créer un équilibrage des données pendant le processus de l'apprentissage à l'aide d'outils tels que (normalisation min-max).

En ce qui concerne les modèles de régression, les entrées et les cibles peuvent utiliser les mêmes outils de normalisation pour maintenir les modèles d'apprentissage sous leurs limites.

3.2. Apprentissage du modèle de prédiction

Le processus de l'apprentissage peut se faire en deux étapes essentielles: l'exploration des données et le réglage des hyper-paramètres.

3.3. Exploration des données

Dans ces étapes, les relations entre les entrées elles-mêmes peuvent être déterminées et utilisées pour améliorer sa représentation avant le début du processus de formation. Des outils tels que (ACP, SVD et le sparse coding, ...etc.) peuvent être intégrés avec les techniques de représentation des caractéristiques pour garantir une distribution plus significative.

3.4. Réglage des hyper-paramètres (apprentissage)

Selon les outils de Machine Learning utilisés, les hyper-paramètres de la fonction d'approximation sont ajustés de manière généralisée pour identifier la relation entre les entrées et les cibles.

3.5. Traitement des résultats

Après la reconstruction des résultats finaux de l'algorithme d'apprentissage en inversant le processus de normalisation. Les résultats peuvent être améliorés à l'aide de plusieurs algorithme de corrections tels que les algorithmes de normalisation [36] et les algorithmes SRC (Sparse Representation Classification) [37].

3.6. Validation

En utilisant des fonctions métriques d'évaluation telles que RMSE (Root Mean Squered Error) et l'exactitude, les experts peuvent apprécier si le modèle est prêt pour l'application ou si le processus de l'apprentissage doit être répété jusqu'à ce que les résultats optimaux soient atteints.

4. Classification des méthodes de diagnostic et de pronostic à base de «Machine Learning»

De nombreuses classifications des modèles d'apprentissage de la détection des défaillances et du diagnostic prédictif peuvent être trouvées dans la littérature. Cette section présente les types les plus utilisés.

4.1. Classification selon les types de données

4.1.1. Modèles d'apprentissage statique

Les modèles d'apprentissage statiques sont des modèles qui dépendent de données statiques qui ne sont plus exposés à des changements liés à l'état physique du système. Généralement, un modèle statique est entraîné hors-ligne. Autrement dit, nous entraînons le modèle exactement une fois, puis nous utilisons ce modèle entraîné pendant un certain temps [38].

Dans le cadre de notre étude, ces modèles sont les plus utilisés pour les opérations de diagnostic (déterminer la cause après l'apparition de la défaillance utilisant la relation : cause → effet.

La **figure II.2** présente un aperçu de la forme de variation des données (mesures des capteurs) dans un cycle de fonctionnement d'un moteur où dans le même temps, il aborde les types de données et leur distribution selon la loi normal. Cette figure représente un ensemble de mesures de capteurs obtenues pendant la période de temps spécifique d'un moteur d'avion. Dans la première partie (**Figure II.2-a**), nous pouvons observer que la structure des données est totalement changée au cours du temps, contrairement à la seconde (**Figure II.2-b**) qui ne représente pas trop de grands changements.

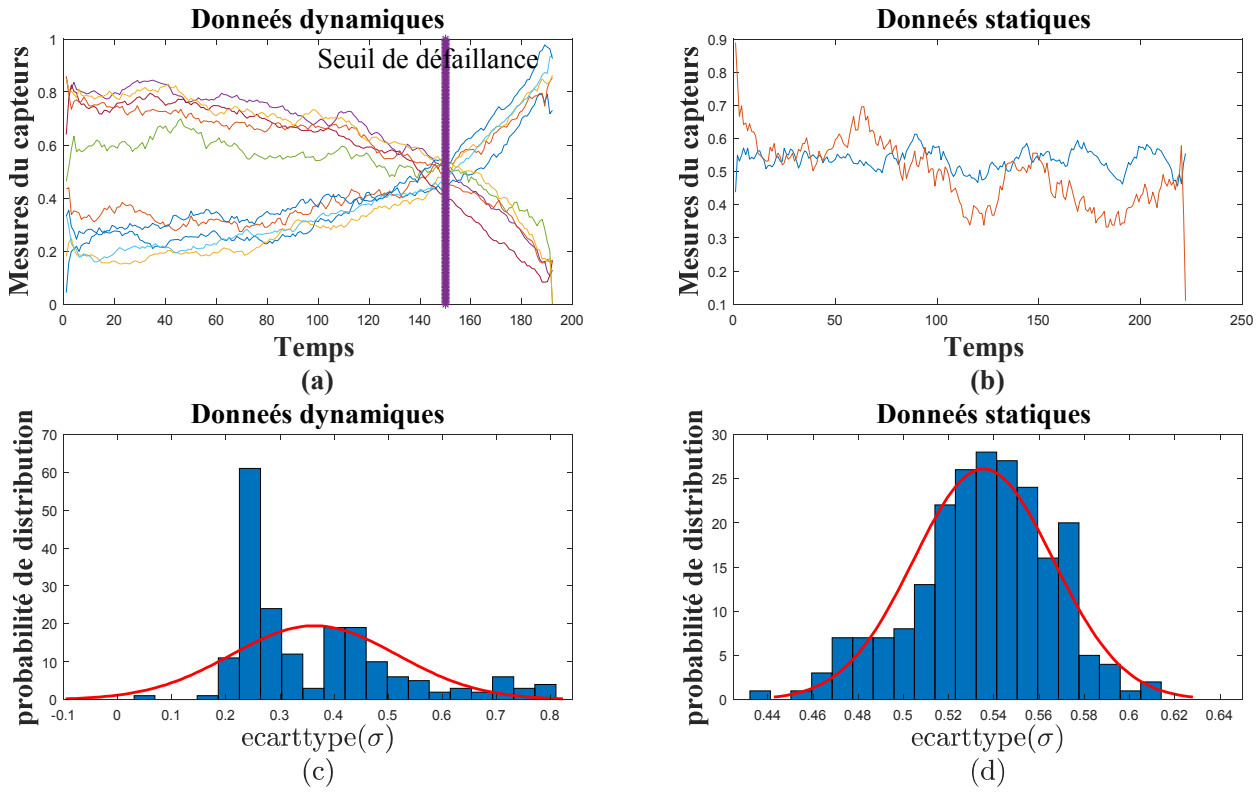

Figure: II.2. Exemples de données statiques et dynamiques

En plus, la distribution normale montre que les inférences statistiques telles que l'écart type sont très large dans les données dynamiques par rapport aux données statiques (voir **figure II.**2-c et **figure II.2-d**).

Parmi les caractéristiques d'une dégradation d'un équipement ou d'un système industriel, on trouve le seuil de défaillance (**Figure: II.2-a**). Si la dégradation des performances des systèmes dépasse ce seuil, les échantillons récupérés dans ce cas ne changeront pas trop comme avant ce seuil. Le problème de la détection de la cause de défaillance devient un problème de diagnostic. Dans la littérature, l'utilisation de la méthode des moindres carrés combinée à un réseau de neurones artificiels pour traiter le diagnostic des défaillances basé sur des données de classification statique dans [39], est un exemple utile qui traite de la programmation statique.

4.1.2. Les modèles d'apprentissage dynamique

Les modèles d'apprentissage dynamiques sont les modèles les plus utilisés pour la régression et généralement pour les systèmes industriels dont l'objectif est l'adaptation en temps réel. Notamment, ils sont appelés dynamiques car qu'ils changent leur caractéristiques et leurs paramètres pour s'adapter à de nouveaux états du système. Généralement les données traitées dans ce cas sont des séries temporelles (**Figure II.2-b**).

La **figure II.3** représente les étapes du cycle de vie d'un système industriel, utilisant la courbe en baignoire en fonction du temps t et de taux de défaillance $\lambda(t)$ (la probabilité que le système se trouve dans un cas de défaillance). Nous pouvons distinguer trois périodes de vie de l'équipement: période de jeuneuse, maturité et vieillesse. Les modèles d'apprentissage dynamique doivent s'adapter avec les périodes.

Si nous supposons que le modèle d'apprentissage est correctement formé et évalué à l'aide de données similaires du même système au cours de la première période de jeunesse, l'estimation du RUL pour les nouveaux échantillons est précise dans cette période. Cependant, si nous voulons utiliser le même modèle d'apprentissage pour estimer l'état de santé du système étudié au cours de la deuxième période de maturité ou de la troisième période de vieillissement, le modèle ne se comportera jamais de manière positive en raison des changements physiques des caractéristiques du système. La solution appropriée dans ce cas est *d'adapter* le modèle d'apprentissage aux nouvelles caractéristiques. Ce type d'apprentissage et appelé «apprentissage adaptatif». D'une autre part, les variables environnementales (tels que la température ambiante, pression, bruit acoustique, etc.) peuvent affecter l'algorithme de prédiction, alors que l'adaptation de l'algorithme d'apprentissage doit être étendu aussi pour prendre en considération ces paramètres [40].

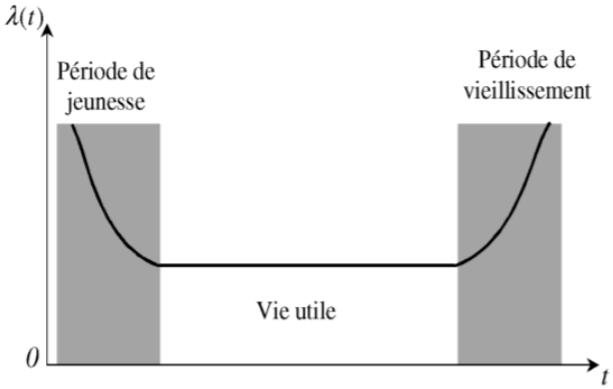

Figure: II.3. Courbe en baignoire [41]

La méthode des moindres carrés récursifs avec un mécanisme *d'oubli dynamique* et une différence temporelle est le meilleur exemple qui concerne l'apprentissage adaptatif [42]. Cette méthode a la possibilité d'oublier progressivement les anciennes données (qui concernent certains états physiques du système) pour passer aux nouvelles.

4.2. Classification selon le type d'accumulation de données

4.2.1. Les algorithmes hors ligne

Dans «Machine Learning», les systèmes qui utilisent l'apprentissage hors ligne ne modifient pas l'approximation de la fonction cible (target function) lorsque la phase d'apprentissage initiale est terminée [43].

Les algorithmes d'apprentissage hors ligne permettent un processus d'apprentissage avec un seul bloc des données à la fois. Après validation, le modèle de prédiction est généralisé à la fois pour l'ensemble des données d'apprentissage lui même et pour l'ensemble des données de test. Il est supposé qu'il soit également généralisé sur d'autres échantillons similaires inconnus à un certain degré de confiance.

Ces algorithmes sont fortement recomposés pour l'analyse de données statiques telles que la classification binaire et l'ajustement d'une courbe simple. Des algorithmes tels que SVM, ELM de base et moindres carrés sont des algorithmes courants qui traitent des applications

d'apprentissage hors ligne. Cependant, ces algorithmes ne sont pas en mesure de traiter des données variant dans le temps dont leurs caractéristiques varient fortement au cours du temps.

Parmi les algorithmes hors ligne les plus performants dans le domaine du diagnostic ou le pronostic des défauts dans les recherches récentes, nous pouvons citer: SVM [44], Moindres carrés [39,45] et ELM de base. Les contributions de ces algorithmes à la détection des défauts et à l'estimation de l'état de santé d'un système sont résumées dans le **tableau II.1**.

Les modèles hors ligne actuels (**Tableau II.1**) montrent un grand succès dans la littérature en termes de précision de prédiction. Cependant, leur implémentation dans une analyse en temps réel où les échantillons sont modifiés dynamiquement au fil du temps n'est pas possible.

Cela ne signifie pas que l'analyse hors ligne n'est pas importante, en tant que mesure de fait, c'est l'étape essentielle de l'évaluation de la santé. En estimant le niveau de la santé hors ligne, ce qui aiderait à estimer la précision maximale que le modèle devrait atteindre dans sa version en ligne.

En effet, la mise à jour itérative peut être plus exposée au risque structurel que l'apprentissage hors ligne.

Tableau: II.1. Modèles hors linge récents pour le diagnostic et le pronostic.

Références	Méthodes	Contrebutions	Domaine d'application
(Matias,T. *et al*. 2015) [25]	• Moindres carrés. • Ondelette.	• La détection de défaut combine deux classificateurs indépendants formés à la fois dans les domaines de fréquentiel et temporel. • L'ondelette est utilisée pour l'écaillage et la transformation. • La méthode des moindres carrés est l'outil essentiel utilisé pour cette mission.	• Classification (Détection de la défaillance).
(Ordóñez. *et al*. 2019) [44]	• SVM • ARIMA	• Extractions des coefficients de modèle d'apprentissage SVM selon l'utilisation des modèles d'auto-régression.	• Régression (estimation du RUL).
(Li, Y. *et al.* 2017) [34]	• LSSVM • IMPE	• Amélioration des paramètres de LSSVM utilisant IMPE.	• Classification (Détection de la défaillance).
(Yang, Z. *et al* .2016) [46]	• ELM	• Intégration d'un mécanisme de seuil auto-adaptatif pour un apprentissage robuste.	• Régression (estimation du RUL).
(Zheng, C . *et al*. 2018) [41]	• ELM • Fenêtre temporelle	• Application de la méthode ELM après l'extraction du paramètres avec la fenêtre temporelle (time window).	• Régression (estimation du RUL).

4.2.2. Les algorithmes en ligne

En algorithmique, l'apprentissage en ligne est une méthode d'apprentissage dans laquelle les données deviennent disponibles dans un ordre séquentiel et est utilisée pour mettre à jour le modèle d'apprentissage pour les données futures à chaque étape.

Par opposition aux techniques d'apprentissage hors ligne qui génèrent les meilleurs prédicateurs par apprentissage sur l'ensemble des données d'apprentissage à la fois. L'apprentissage en ligne est une technique courante utilisée dans les domaines de «Machine Learning» où il est impossible de faire l'apprentissage sur l'ensemble des données, nécessitant des algorithmes hors ligne.

Il est également utilisé dans des situations où il est nécessaire que l'algorithme s'adapte dynamiquement à nouvelles variations dans les données, ou lorsque les données elles-mêmes sont générées en fonction du temps, par exemple, la prédiction du prix des stocks.

Les algorithmes d'apprentissage en ligne tels que les moindres carrés récursifs, les filtres de Kalman, la rétropropagation pour l'apprentissage des réseaux de neurones artificiels ou bien les réseaux de Bayes et la divergence contrastive sont les plus utilisés dans ce type d'analyse où les signaux d'entrées sont séquentiellement entraînés dans le temps.

Un ensemble d'algorithmes en ligne récents qui abordent l'apprentissage séquentiel via un apprentissage adaptatif pour la détection des défauts et l'évaluation de la santé sont résumés dans le **tableau II.2**.

Ces algorithmes permettent l'apprentissage en ligne ainsi que la possibilité d'adaptation dynamique des modèles de l'apprentissage qui conduit à l'apprentissage adaptatif. L'un des avantages les plus importants de ces algorithmes réside dans leur capacité de traiter à la fois l'apprentissage en ligne et adaptatif.

Tableau: II.2. Modèles en linge récents pour le diagnostic et le pronostic

Références	Méthodes	Contrebutions	Domaine d'application
(Zhang, C. *et al.* 2019) [47]	• LSTM • le filtrage des particules	• Modélisation des réseaux de neurones récurrents et correction des erreurs basées sur le filtrage des particules.	• Régression (estimation du RUL).
(Djeziri, M.A. *et al*. 2019) [48]	• Wiener process.	• Utilisation des données augmentées en simulation pour prendre en compte toutes les tendances possibles du processus de dégradation d'un système, • Estimation du RUL	• Régression (estimation du RUL).
(Lu, F. *et al.* 2019) [49]	• filtre de Kalman • OS-ELM • Fonction sigmoïde.	• Intégration de filtre de Kalman dans les règles d'apprentissage de OS-ELM ou lieu de la méthode du moindre carrées récursif. • Fonction d'activation sigmoïde.	• Régression (estimation du RUL).
(Alam, M.M. *et al*. 2014) [50]	• des fenêtres de mise à l'échelle • RNA • modèles autorégressifs	• Amélioration des paramètres de l'RNA utilisant des coefficients des modèles autorégressifs après l'extraction des échenillent des données avec des fenêtres de mise à l'échelle	• Régression (estimation du RUL).
(Berghout, T. *et al*. 2020) [5]	• OS-ELM • Autoencoders • Différence temporelle	• Estimation en temps réelle a base de l'intégration des technique de le «Deep Learning» dans la méthode séquentielles OS-ELM en considèrent les différences temporelles entres les séquences de données et la programmation dynamique.	• Régression (estimation du RUL).

Contrairement aux algorithmes hors ligne, le mécanisme d'apprentissage actuel permet une interaction avec les séquences de données inondées en temps réel telles qu'elles soient.

4.3. Classification selon l'architecture des modèles

4.3.1. Les méthodes hybrides

Les méthodes hybrides combinent deux ou plusieurs méthodes de Machine Learning pour obtenir des performances et résultats optimaux. En fait, les méthodes hybrides bénéficient de l'avantage que deux méthodes ou plus pour atteindre de meilleures performances.

Parfois, les méthodes hybrides contiennent une unité pour la prédiction et une pour l'optimisation de l'unité de prédiction pour atteindre une sortie précise. Par conséquent, nous pouvons affirmer que les méthodes hybrides contiennent différentes méthodes avec une flexibilité et capacité élevée par rapport aux méthodes simples. Les méthodes hybrides sont devenues plus populaires en raison de leur potentiel et de leur capacité élevés [51].

Les modèles de prédiction hybrides sont des modèles d'apprentissage qui ont une fonction objective globale obtenue à partir de combinaisons d'un ensemble de modèles de prédiction différent [52]. Chaque algorithme de prédiction particulier est strictement différent des autres algorithmes en terme d'architecture et de paramètres initiaux d'apprentissage. Le résultat de chaque système de prédiction peut être considéré comme l'entrée d'un autre algorithme, et le nombre de sous systèmes est défini par l'utilisateur.

La **figure II.4** présente une idée générale sur l'architecture de l'algorithme de prédiction hybride où les exemples des interactions entre les différents algorithmes sont bien illustrés.

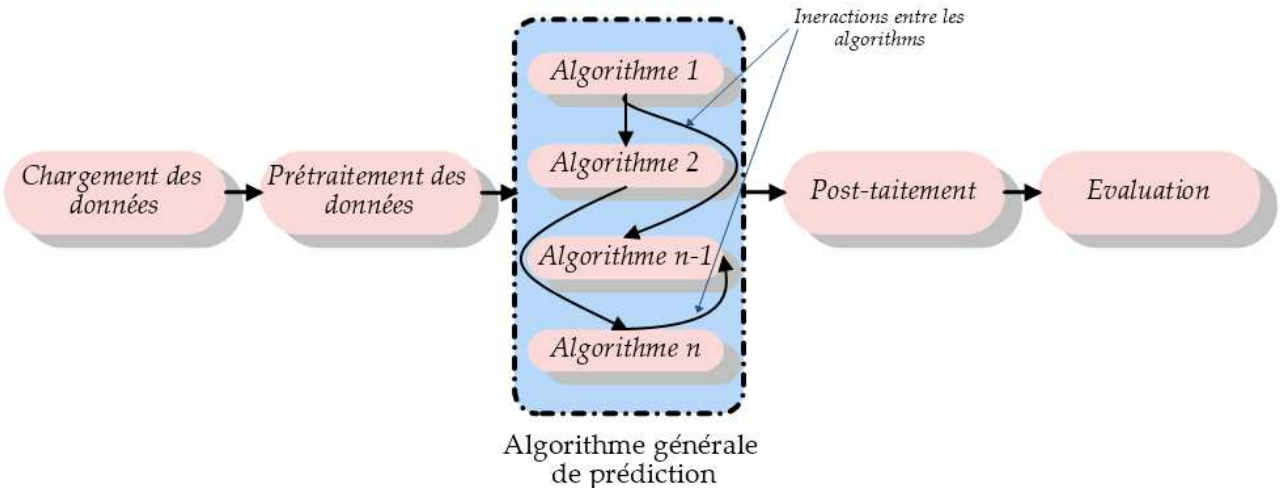

Figure: II.4. Architecture générale d'un modèle de prédiction de type hybride

Dans le contexte de la gestion de la santé et lorsque les chercheurs ont pu rassembler différents approximateurs universels et mécanismes d'optimisation pour généraliser une fonction objective unique à la fois sur les échantillons d'apprentissage et de test, de nombreux paradigmes de prédiction ont été développés pour tenter de répondre à une estimation précise.

A titre d'exemple, Xiong Li *et al*. [53] ont utilisé un modèle du type semi-Markov caché pour faire l'apprentissage d'un algorithme SVM pour la prédiction du RUL. García *et al*. [54] ont conçu une technique hybride intelligente basée sur SVM pour la recherche des meilleurs coefficients d'apprentissage lors de la prédiction du RUL des moteurs d'avion. Saidi *et al.* [55] ont combiné Spectral Kurtosis (SK) à SVM pour pouvoir construire un estimateur du RUL avec une représentation de données plus significatives pour le pronostic des roulements d'arbre à grande vitesse des éoliennes. Zheng *et al.* [56] se sont basé sur un modèle hybride pour construire un modèle d'apprentissage de l'algorithme ELM. Ils ont adopté une fenêtre temporelle pour la mise à l'échelle des paramètres d'entrée comme un prétraitement, et une étape de sélection appropriée pour garantir une prédiction précise du RUL. Ordóñez *et al.* [44] ont proposé un modèle autorégressif hybride combiné à un SVM amélioré utilisant un

algorithme génétique pour construire plusieurs algorithmes d'estimation pour la prédiction précoce du RUL. Chen *et al.* [57] ont utilisé une approche de similarité basée sur SVM pour la prédiction du RUL avec le même ensemble de données C-MAPPS.

Il ressort clairement de la brève littérature mentionnée ci-dessus, que les méthodes hybrides sont de plus en plus populaires en raison de leur potentiel élevé et de leur capacité à augmenter l'estimation et à optimiser les performances.

Cependant, la conception des différents algorithmes de l'apprentissage pour résoudre les problèmes d'optimisation peut augmenter considérablement le temps de calcul pendant le réglage des paramètres à base de la recherche aléatoire, car il faut aussi beaucoup d'intervention humaine. En effet, cette complexité peut devenir plus difficile compte tenu des données dynamiques de grande dimension telles que l'analyse de séries temporelles.

Aussi, si nous avons toujours besoin de reconstruire un modèle hybride, nous devons toujours examiner en profondeur l'optimisation de la fonction objective en considérant un schéma d'apprentissage plus rapide.

4.3.2. Les méthodes d'ensemble

Les méthodes d'ensemble peuvent utiliser un ensemble d'algorithmes similaires au lieu d'un seul. À travers cette technique, la précision du modèle est sensiblement améliorée. Les méthodes d'ensemble sont considérées comme des algorithmes d'apprentissage supervisé.

Les méthodes d'ensemble bénéficient de différents algorithmes d'apprentissage pour augmenter la précision et pour atteindre une précision de test plus élevée. La méthode ensemble permet d'utiliser différents algorithmes d'apprentissage pour rendre l'apprentissage flexible [51].

La **figure II.5** est une illustration de l'architecture générale des algorithmes de type ensemble.

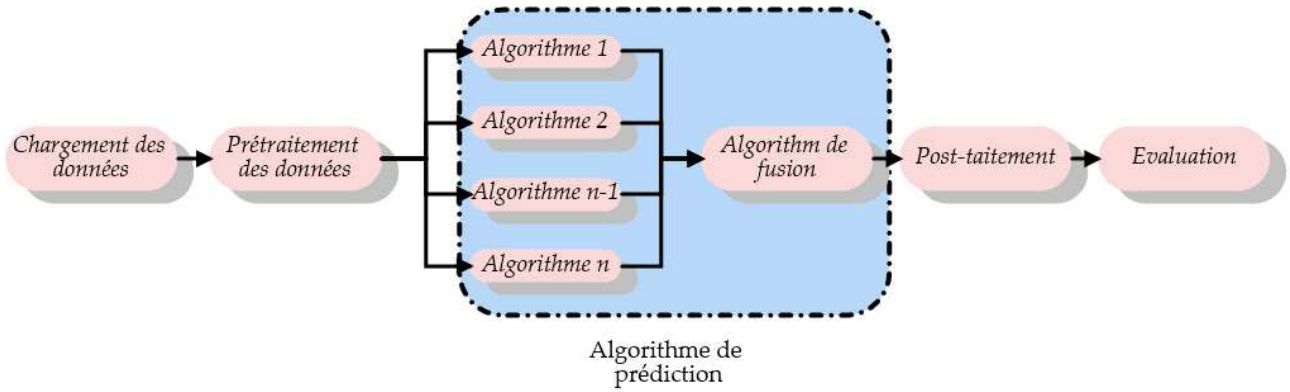

Figure: II.5. Architecture approximative d'un modèle d'ensemble

Dans le cas de l'apprentissage, les algorithmes sont initialisés avec des paramètres différents dont le but de scanner le maximum possible de combinaisons nécessaires à la meilleure approximation. Après, le résultat du réglage des paramètres fait partie de la décision de l'algorithme de fusion qui est responsable du processus de sélection.

Dans les domaines qui traitent de la gestion des opérations de maintenance, les réseaux de neurones avec des algorithmes qui traitent des lots de données séquentielles (mini-batches) sont capables d'interagir avec des séquences de données accumulées et de mettre à jour le modèle de prédiction pour satisfaire les changements dans les nouvelles données arrivées.

Dans [58], une nouvelle approche basée sur les données (data-driven) est introduite par Ben Ali *et al*. L'apprentissage d'un réseau neuronal SFAM (Simplified Fuzzy Adaptive Resonance Theory Map) est fait avec la distribution probabiliste de Weibull (WD) pour éviter les fluctuations du domaine temporel lors de la prédiction du RUL. Al-Dulaimi *et al.* [59] ont développé un réseau de neurones artificiel hybride (RNA) profond (Deep hybrid neural network) qui prend deux voies pour l'estimation du RUL. L'extraction de caractéristiques multidimensionnelles basée sur LSTM (Long Short Term Memory), le RNA convolutionnel et le chemin de prédiction via un algorithme de fusion. Wen *et al.* [60] ont construit de nouvelles représentations des entrées basées sur l'ensemble CNN résiduel et ont validé leur modèle de

formation avec la méthode de validation croisée après avoir construit plusieurs modèles pour prédire le RUL avec précision. Xiang *et al*. [61] ont simplifié le processus de prédiction du RUL des moteurs d'avion par l'utilisation directe de mesures de capteurs bruts sans expertise préalable dans le traitement du signal utilisant un nouveau RNA convolutionnel profond (Deep CNN) combiné avec un extracteur de caractéristiques de fenêtre temporelle.

Les approches de prédiction utilisées dans ces travaux sont basées sur un apprentissage interagissant avec des séquences de données. De nombreux modèles d'apprentissage ont été développés avec différents paradigmes tentant d'accéder à des représentations de données plus significatives grâce à une prédiction précise.

l'un des inconvénients des algorithmes d'ensemble est qu'ils ont beaucoup plus d'hyper paramètres, ce qui nécessite plus de temps, en particulier pour les algorithmes basés sur le gradient. De même pour l'algorithme de fusion, où il ne doit pas s'agir d'un mécanisme de recherche de grille qui peut rendre le processus de prise de décision difficile.

Dans le cadre de notre travail (simplicité, vitesse et précision de l'apprentissage), s'orienter vers moins de réglage d'hyper-paramètres et d'algorithmes non itératifs simples et précis est le seul moyen de réduire la complexité de ce type d'architecture.

4.3.3. Les modèles de Deep Learning

Les algorithmes du Deep Learning ont récemment émergé dans le domaine de Machine Learning. Depuis lors, plusieurs algorithmes ont été introduits et sont utilisés dans divers domaines d'application.

Aujourd'hui, l'utilisation du Deep Learning est devenu essentielle en raison de leur intelligence, leur apprentissage efficace, leur précision et leur robustesse dans la construction de modèles de prédiction [62]. Cette section fournit une liste complète des algorithmes de Deep Learning les plus populaires, ainsi que leurs applications dans le domaine du diagnostic et du pronostic.

La **figure II.6** présente les étapes essentielles de l'apprentissage de tous les modèles de Deep Learning. Il s'agit d'étapes d'apprentissage non supervisé et supervisé.

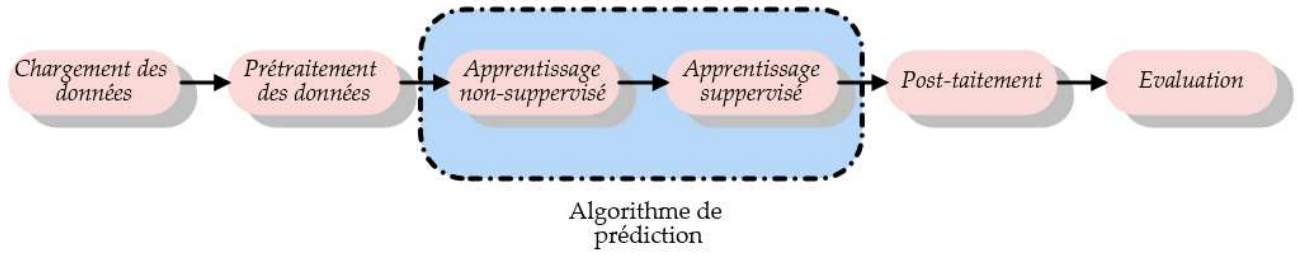

Figure: II.6. Architecture des modèles à base du Deep Learning

Le Deep Learning permet d'améliorer l'approximation supervisée lors de la recherche des formes (patterns) les plus importants dans des données impliquées par un apprentissage non supervisé.

Dans la littérature, les algorithmes du Deep Learning les plus récemment utilisés peuvent être classés en quatre catégories essentielles: Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Deep Belief Neural Networks (DBN) et les autoencodeurs.

4.3.3. 1. Réseau de Neurones Convolutionel (CNN)

Le CNN est l'une des architectures les plus connues des techniques de Deep Learning. Cette technologie peut être utilisée dans tous les domaines, mais est généralement utilisé pour les applications de traitement d'image. CNN contient trois types de couches avec différentes couches convolutionnelles, regroupées et entièrement connectées. Dans chaque CNN, il y a deux étapes pour le processus d'apprentissage, l'étape de passe avant (feedforward) et l'étape passe à l'arrière (backward).

Dans la passe avant, le CNN prend un ensemble de nœuds et le transforme en couche convolutionnelle avec des filtres ou des champs récepteurs locaux (Local Receptive Fields) considèrent la fenêtre de chevauchement (the overlap). Après l'activation de ces

transformations à l'aide de la fonction d'activation prédéterminée, le nouvel espace de données sera à nouveau sous-échantillonné (pooling) en utilisant un type d'échantillonnage bien défini comme (la moyenne, la min ou le max). Les couches convolutionnelles et les couches de sous-échantillonnage sont localement connectées. Les résultats finaux du sous-échantillonnage seront entièrement connectés à la couche de sortie.

Dans la passe arrière, les algorithmes d'apprentissage essaieront de régler ces connexions (poids) en fonction de l'erreur d'approximation pour satisfaire la fonction cible. Généralement les CNN sont formés en utilisant l'algorithme de rétro-propagation.

La **figure II.7** illustre l'architecture fondamentale d'un réseau de neurones CNN.

Les outils d'apprentissage en profondeur et en particulier les CNN sans aucun doute, ont contribués avec succès à la création de programmes de l'estimation précise pour les modèles de l'évaluation de la santé d'un système tel que les travaux qui ont été réalisés en [60,63–65].

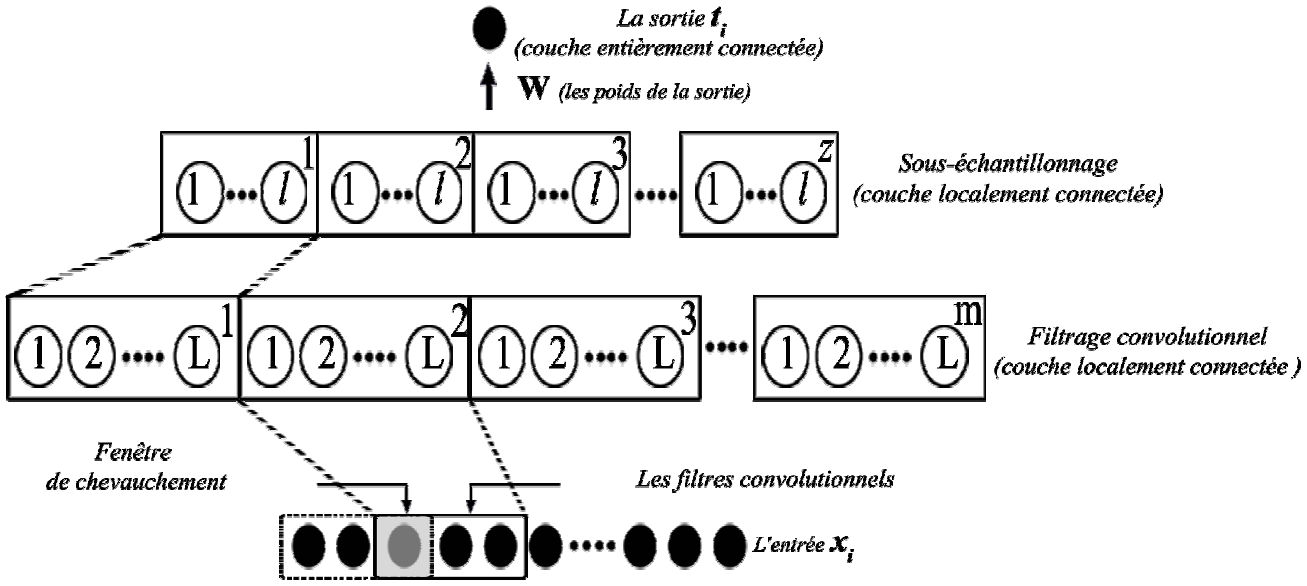

Figure: II.7.Architecture simple d'un réseau de neurones CNN

4.3.3. 2. Réseau de Neurones Récurent (RNN)

Les réseaux neuronaux récurrents sont une sorte de réseau neuronal spécialisé dans le traitement de données séquentielles. C'est-à-dire un ensemble de points de données disposés dans un ordre particulier de telle sorte que les points de données connexes se suivent, les données audio et vidéo, les séquences d'ADN, les données des capteurs, le texte en langage naturel,…. etc. [66], en raison de leur efficacité.

Contrairement aux CNN qui prennent des entrées de taille fixe et produisent des sorties de taille fixe, les RNN sont utiles car ils nous permettent d'avoir des séquences de longueur variable comme entrées et sorties [67].

Les RNN sont capables de faire des prédictions en répétant le même processus sur une séquence entière [66]. La **figure II.8** montre la structure de base du réseau récurrent. Il prend à la fois le vecteur d'entrée de données de séquence *(x)* et les informations d'état caché *(a)* et les utilise pour prédire la sortie de données de séquence.

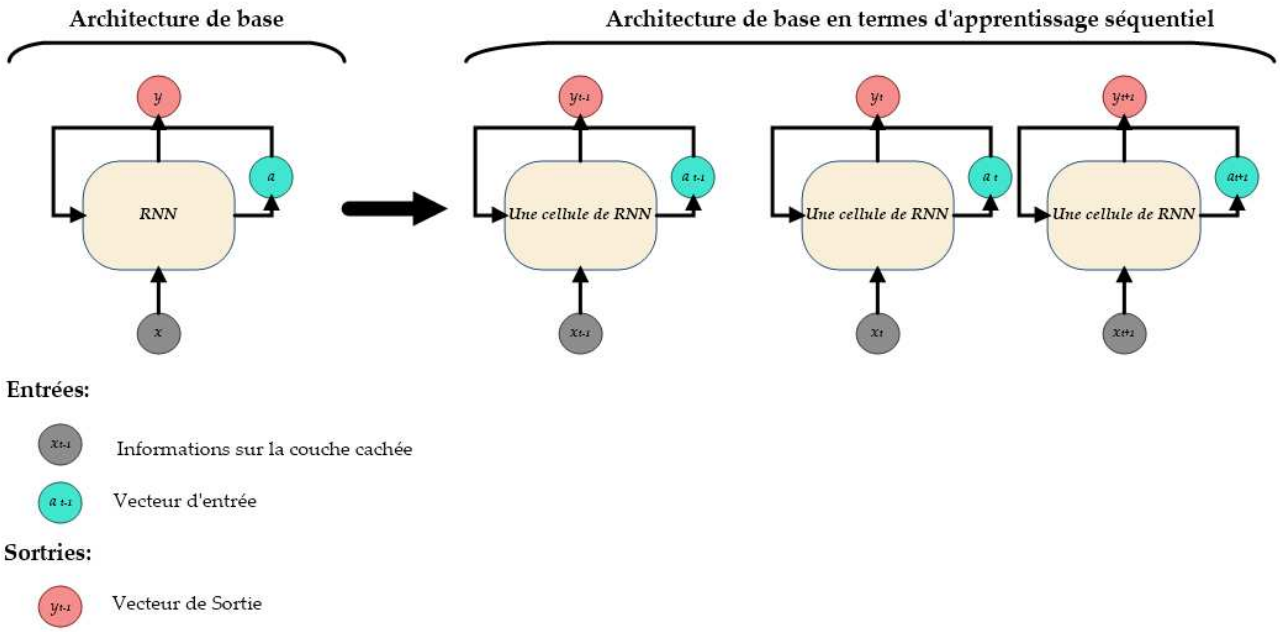

Figure: II.8. Principe de base du fonctionnement des RNN [66]

Néanmoins, afin d'acquérir une meilleure intuition et visualiser correctement leur fonctionnement, nous déroulons généralement le RNN en une chaîne répétitive de cellules ou de pas de temps qui correspondent à la longueur des données de séquence (**Figure II.8**). A titre d'exemple, si la séquence qui nous intéresse est une phrase de 3 mots, le réseau se détroule en un réseau RNN à 3 pas de temps, un pas de temps pour chaque mot.

Dans le domaine du pronostic et la gestion de la santé, les RNN sont très utilisés pour l'analyse des mesures des capteurs pour l'estimation de l'état de santé du système. Les travaux récents tels que [22,68] sont les meilleurs exemples dans ce domaine.

4.3.3. 3. Deep Belief Neural Networks (DBN)

Les réseaux de neurones de type DBN sont des modèles génératifs probabilistes composés de plusieurs couches de variables latentes stochastiques.

Les variables latentes ont généralement des valeurs binaires et sont souvent appelées unités cachées ou détecteurs de caractéristiques. Les deux couches supérieures ont des connexions symétriques non orientés entre elles et forment une mémoire associative. Les couches inférieures reçoivent des connexions dirigées de haut en bas de la couche supérieure. Les états des unités dans la couche la plus basse représentent un vecteur de données [69].

Les deux propriétés les plus importantes des réseaux de DBN sont:

- Existence d'une procédure efficace, couche par couche, pour l'apprentissage à partir des pondérations génératives descendantes qui déterminent comment les variables d'une couche dépendent des variables de la couche ci-dessus.

- Après l'apprentissage, les valeurs des variables latentes dans chacune des couches peuvent être déduites par une seule passe ascendante qui commence par un vecteur de données observé dans la couche inférieure et utilise les poids génératifs dans le sens inverse.

Les réseaux de DBN forment une couche à la fois en traitant les valeurs des variables latentes dans une couche, lorsqu'elles sont déduites des données, comme les données pour l'apprentissage de la couche suivante. Cet apprentissage efficace peut être suivi ou combiné avec d'autres procédures d'apprentissage qui affinent tous les poids pour améliorer les performances génératives ou discriminantes de l'ensemble du réseau.

Un réglage fin (fine tuning) discriminatoire peut être effectué en ajoutant une couche finale de variables qui représentent les sorties souhaitées et des dérives d'erreur rétro-propagation. Lorsque des réseaux avec de nombreuses couches cachées sont appliqués à des données d'entrée hautement structurées, telles que des images, la rétropropagation fonctionne beaucoup mieux si les détecteurs d'entités dans les couches cachées sont initialisés en apprenant un DBN qui modélise la structure des données d'entrée [69]. La **figure II.9** présente un récapitulatif général sur l'apprentissage de la méthode DBN.

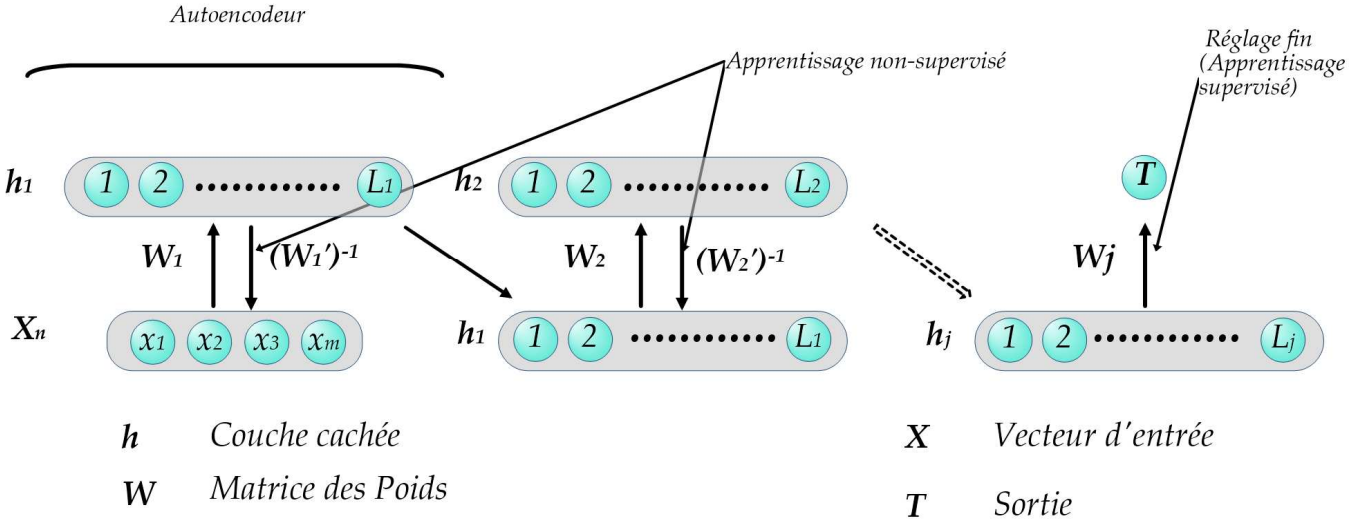

Figure: II.9. Architecture de base d'un DBN

Les DBN permettent de bénéficier de l'utilisation d'un apprentissage non supervisé qui contribue directement à l'extraction appropriée des paramètres et aide le modèle d'apprentissage à apprendre les formes (patterns) importantes à partir des données.

Depuis l'apparition du premier type d'autoencodeurs, machines de Boltzmann par Geoffrey Hinton [70], plusieurs travaux dans la littérature adoptent l'utilisation des DBN en PHM.

D. Feng *et al* [71] ont réalisé l'apprentissage d'un DBN basé sur l'entropie d'informations et utilisent une couche logistique cachée à la dernière étape de réglage fin du modèle d'apprentissage. Ce processus permet la détection des défaillances (classification) dans les moteurs des turbines à gaz ne dépendant que d'un petit ensemble d'apprentissage en raison de la non-disponibilité de l'installation de capteurs dans l'environnement d'une température plus élevée. J. Deutsch *et al*.[72], ont utilisé des DBN formés avec des données collectées à partir des profils de roulement pour prédire la durée de vie restante dans un espace dimensionnel plus élevé. L'avantage de l'utilisation des DBN réside dans le fait qu'il ne dépend que d'un petit ensemble d'entraînement pour obtenir un bon réglage des paramètres non-supervisés et supervisés. J. Ma *et al*.[73], ils utilisé des autoencodeurs basés sur le «Sparse Coding» combiné à une couche cachée de sortie logistique pour l'apprentissage d'une DBN pour la prédiction du RUL des moteurs d'avion. Les algorithmes de recherche de grille sont utilisés pour les hyper-paramètres pendant le processus d'apprentissage.

Dans le cadre de ces travaux, le DBN est considéré comme une méthode de prédiction très intelligente et disponible pour une variété d'applications. L'intégration de ses règles d'apprentissage dans ce domaine de recherche est d'actualité et semble prometteuse. Dans le cadre dans notre travail, nous avons orienté l'utilisation de paradigmes DBN sous l'adaptation dynamique des règles ELM pour effectuer des tâches de prédiction du RUL très rapides et précises (chap V et VI).

4.3.3. 4. Les autoencodeurs

Les autoencodeurs sont des réseaux de neurones d'apprentissage simples qui visent à transformer les entrées en sorties avec le moins de distorsion possible. Bien que conceptuellement simples, ils jouent un rôle important dans le Machine Learning [74].

Les autoencodeurs automatiques ont été introduits pour la première fois dans les années 1980 par Hinton [74] pour résoudre le problème de la «rétropropagation sans étiquette», en utilisant seulement les données d'entrées.

Les autoencodeurs sont beaucoup utilisés dans le Deep Learning dans plusieurs domaines, et vu leur importance dans notre travail en particulier, pour leur application lors de l'apprentissage non supervisé des DBNs. Cette section présente les types d'autoencodeurs les plus utilisés, leurs architecture et leur impact dans la détection des défauts et le diagnostic prédictif.

- Les Machines de Boltzmann restreinte (RBM)

Les RBM sont les premiers types de réseaux de neurones utilisés pour l'apprentissage non supervisé. Il s'agit d'un réseau neuronal profond avec seulement deux couches, la couche visible et la couche cachée [75]. Dans ce réseau, chaque nœud de la couche visible est connecté à chaque nœud de la couche cachée. Un RBM est considérée comme restreint car deux nœuds de la même couche ne partagent pas de connexions. Dans la passe avant, le RBM prend de manière itérative un ensemble d'entrées et les traduit en un ensemble de nombres qui code l'entrée. Dans la passe arrière, il reprend ces nombres pour former l'entrée reconstruite. Les mouvements vers l'avant et vers l'arrière sont connus sous le nom « d'échantillonnage de Gibbs (Gibbs sampling)».

Au niveau de la couche visible et après plusieurs échantillonnages, l'entrée reconstruite est comparée à l'entrée d'origine pour déterminer la qualité des résultats. Un réseau bien formé est en mesure d'effectuer la traduction en arrière avec un haut degré de précision. Dans les deux étapes, les poids et les biais sont ajustés de manière itérative pour permettre au RBM de décider quelles caractéristiques sont les plus importantes lors de la détection des formes (patterns).

La **figure II.10** indique les composants fondamentaux d'une RBM comme on peut trouver une illustration simple sous MATLAB de l'algorithme de divergence contrastive pour le RBM [76][4].

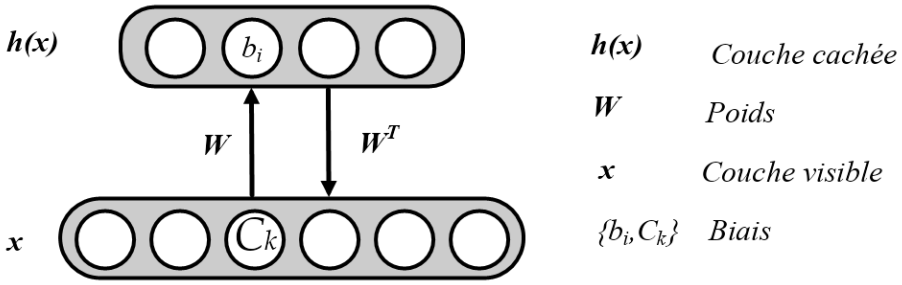

Figure: II.10. Structure d'un RBM

D'une part, dans les domaine de PHM, les RBM sont plus utilisés pour l'apprentissage des DBN tel que les travaux déjà motionnées dans la section DBN [71–73]. Les RBM peuvent être aussi utilisé pour l'extraction des paramètres nécessaires pour l'apprentissage comme présenté dans le RBM Gaussien [77] , le RBM avec régularisation [78] et le RBM convolutionnel [79].

D'autres part, et en général, les RBM sont formés à l'aide d'un algorithme de divergence contrastive (CD) qui consomme souvent plus de coûts de calcul en raison de la nécessité d'un grand *nombre de nœuds cachés* [70]. En outre, les hyper-paramètres tels que le taux d'apprentissage, le nombre d'échantillonnage de Gibbs, le nombre de neurones cachés et le nombre d'itérations varient d'une application à l'autre qui nécessite une intervention plus humaine.

- Les autoencodeurs ordinaires

Contrairement aux RBM qui ne dépendent que de deux couches cachées pour reconstruire l'entrée basée sur l'algorithme de divergence contrastive, l'autoencodeur ordinaire utilise

[4] Cet algorithme est un travail original que nous avons développé en utilisant les règles d'apprentissage inspirées du guide de Hinton [70] .

l'algorithme de backpropagation pour transformer l'entrée passant par trois couches et différentes matrices de poids tels que présentées dans la **figure II.11**.

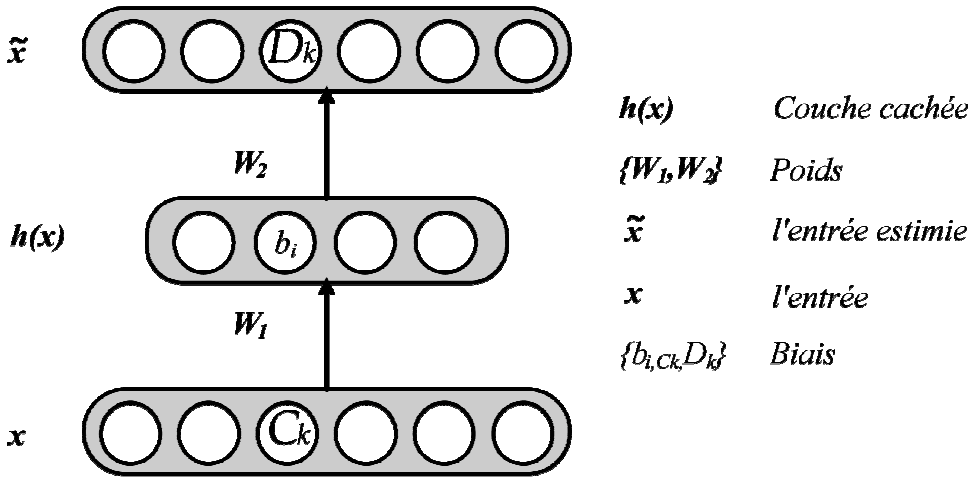

Figure: II.11 . Structure générale d'un autoencodeur ordinaire.

Dans le cas du diagnostic industriel, l'autoencodeur ordinaire peut être utilisé pour la compression des données [80][5], ou bien juste pour l'extraction des paramètres ou le redimensionnement [5][6] de l'espace des données à haute dimension. Une illustration de l'algorithme de l'autoencodeur avec le software MATLAB est valable [81][7].

Parmi les avantages de l'autoencodeur, l'algorithme de rétropropagation est le responsable sur le réglage de paramètres qui peuvent être utilisés pour gagner plus de temps de calcul que la divergence contrastive.

- **Les autoencoders de débruitage**

[5] Cette contribution fait partie de notre travail réalisé dans le cadre d'une conférence internationale de systèmes de défense qui illustre l'utilisation des autoencodeurs.

[6] Ce travail est un exemple d'extraction de paramètres avec des autoencodeurs connectés en série dans le domaine PHM.

[7] Dans ce cas, l'autoencodeur doit être pris en utilisant la technique récente appelé l'ELM.

Un autoencodeur de débruitage est développé pour reconstruire une entrée «réparée» propre à partir d'une version corrompue de son entrée. Pour cela, il faut d'abord corrompre l'entrée initiale en ajoutant du bruit à ses caractéristiques en fonction du bruit spécifié. La sortie estimée de ce type d'autoencodeur doit être comparé avec l'entrée originale et non pas la version corrompue [82].

Dans le travail [80] où nous avons mentionné que le bruit Gaussien à utiliser pour corrompre l'entrée. La reconstruction des entrées est réalisée à partir de sa version compressée de la couche cachée à un taux de compression égal à 50%. Les résultats prouvent que l'autoencodeur débruteur fonctionne mieux que l'ordinaire en termes d'extrusion des paramètres significatifs.

La **figure II.12** permet d'illustrer les différences entre les règles d'apprentissage ordinaire et celles de l'algorithme débruiteur.

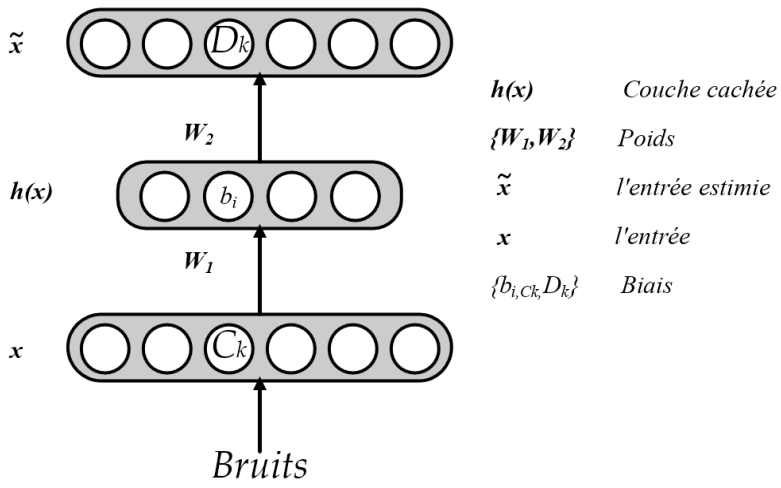

Figure: II.12. Règles de l'apprentissage d'un autoencodeur de débruitage.

- **Les autoencodeurs contractuels**

L'autoencodeur contractuel est un type d'autoencodeur où seuls les éléments les plus significatifs de la couche cachée sont utilisés pour le processus d'apprentissage. Avec une autre manière, la couche cachée ne doit refléter que la variation de l'entrée. Les unités cachées qui ne

présentent aucune variation lors de l'accumulation des séquences de données doivent être éliminées en fixant leurs poids à zéro [34].

Pour encourager la robustesse de la représentation $f(x)$ obtenue pour une entrée d'apprentissage x, il est proposé de pénaliser sa sensibilité à cette entrée, mesurée comme la norme de **Frobenius** de la **Jacobian** $Jf(x)$ d'une transformation non linéaire. Formellement, si l'entrée $x \in IR^{dx}$ est transformé par une fonction de codage f à la couche cachée $h \in IR^{dh}$, ce terme de pénalisation de la sensibilité est la somme des carrés de toutes les dérivées partielles des entités extraites par rapport aux dimensions d'entrée **[II.1].**

$$\left\| Jf(x) \right\|_F^2 = \sum_{ij} \left(\frac{\partial h_j(x)}{\partial (x_i)} \right)^2 \qquad \textbf{[II.1]}$$

Dans le domaine de détection des anomalies, ce type d'autoencodeur est largement utilisé en particulier pour le traitement de données de dimensions supérieures. L'autoencoder aussi peut être aussi utilisé comme un outil qui mesure la corrélation entre la sortie et l'entré car il aide à éliminer les variables indésirables de la représentation cachée.

Changqing Shen *et al.* [83], ont développé une méthode d'apprentissage des caractéristiques automatique et robuste pour le diagnostic des défaillances des machines tournantes basé sur l'autoencodeur contractuel. Edmond Q. Wu *et al.* [84] construisent un outil pour la reconnaissance de l'état de fatigue des pilotes à l'aide d'un réseau d'autoencodeurs contractuels au profond (Deep representation). Un exemple illustratif basé sur MATLAB dans le cadre de notre travail est téléchargeable sur ces liens [85][8].

[8] Dans cette publication, les codes MATLAB d'un nouvel ELM contractuel basé sur un autoencodeur sont développés où la norme **Frobenius** de la matrice **Jacobéenne** est ajoutée à la fonction de reconstruction.

- **L'autoencodeur sparse**

Dans ce type d'autoencodeurs, la couche cachée est transformée en un autre espace où la plupart des éléments réduits à zéro. l'objectif recherché par cette transformation est de forcer la couche cachée à souffler les éléments importants en peu de représentations (c'est surtout comme une compression) [86].

En plus des distributions probabilistes telles que la SVD basée sur la distribution normale, des fonctions de transfert telles que (temps à fréquence) peuvent également être utilisées pour cette mission. Toutefois, les solutions sont différentes d'un outil à l'autre; par exemple, SVD est généralement basé sur la minimisation de la norme L_2 et d'autres domaines fréquentiels tels que les ondelettes, basées sur la minimisation L_1[35][9].

En général, dans le diagnostic ou le diagnostic prédictif, l'autoencodeur de ce type est combiné avec des algorithmes d'apprentissage supervisé pour préformer un niveau plus élevé de précision de détection, le même que les travaux en cours [87,88] dont la pré-représentation des couches cachées utilisées pour l'apprentissage de DBN. Des exemples illustratifs sous MATLAB sont illustré dans [89] utilisant la norme L_1.

5. Les défis de la prédiction

Comme Il a été déjà mentionné précédemment, la prédiction basée sur l'analyse de données avec des outils de Machine Learning et ses différents types et classes peut être une excellente solution sous la disponibilité des données. Cependant, les modèles d'apprentissage peuvent souffrir de certains problèmes dans certaines situations en termes de diagnostic ou de pronostic, En effet,

[9] Ces codes sont une simple contribution développée dans le cadre de nos travaux visant à aborder le «Sparse Coding» lors de l'apprentissage d'un autoencodeur.

- L'apprentissage hors ligne ne sera plus considéré comme une solution, si les données du système étudié sont accumulées séquentiellement.
- Selon la courbe en baignoire, la modélisation statique même si elle est précise en temps réel n'est pas une solution souhaitable pour les systèmes industriels.
- Selon la courbe en baignoire, l'apprentissage en ligne ne satisfera pas la variation des données pendant les périodes de vie des systèmes industriels.
- L'augmentation de la dimensionnalité des données (telles que les images) peut retarder le processus de l'apprentissage et éloigner le modèle de prédiction de l'estimation correcte à un certain niveau d'apprentissage en ligne.
- Les algorithmes d'apprentissage classiques tels que la rétropropagation ou la divergence contrastive ne résisteront plus aux overfitting et au minimum local, en particulier sous un niveau de variation de données plus élevé.
- Les algorithmes d'apprentissage classiques tels que la rétropropagation ou la divergence contrastive consomment énormément beaucoup de coûts de calcul (temps, matériels et logiciels) et nécessitent beaucoup d'intervention humaine lors du réglage des hyper paramètres.
- Le réglage du taux d'apprentissage est un problème courant lorsque la taille des données dans les problèmes réels est inconnue.
- Les additions telles que les algorithmes de réglage communes comme le « gride search », « Swarm Intelligence » et la validation croisée seront un énorme défi lorsque les données sont séquentiellement pilotées dans le temps ainsi que la régularisation itérative.
- La généralisation de l'algorithme de prédiction sur des données multi-variées inconnues sera un problème si l'apprentissage adaptatif n'a pas considéré.

- L'architecture complexe et multicouche des algorithmes Deep Learning, consommera énormément beaucoup de ressources de calcul plus que d'autres techniques mentionnées.

6. Tendances actuelles possibles

A partir de ces brèves analyses basées sur des recherches bibliographiques bien structurées, la complexité de la prédiction peut être réduite si nous considérions ces solutions proposées [33].

- Tenir compte de l'apprentissage adaptatif en ligne pour que le modèle de prédiction puisse interagir avec l'analyse des données en temps réel.
- Utiliser un algorithme de l'apprentissage simple qui permet moins de réglage des paramètres et un apprentissage en ligne rapide.
- Comme les algorithmes de Deep Learning jouent un rôle clé dans la prédiction précise, l'intégration de cet algorithme simple dans l'apprentissage sera une excellente solution.
- Pour réaliser une bonne prédiction, il vaudrait mieux ne pas tomber dans le problème des minimum locaux.

Toutes ces raisons nous incitent à dire que l'ELM, a été largement utilisé dans ces deux dernières décennies en raison de sa prédiction confidente et de son apprentissage rapide. Il nécessite également moins d'ajustement des paramètres en fonction de la meilleure adéquation d'une approximation de fonction linéaire [9].

Zhou *et al.* [90] ont proposé un ELM empilé (Stacked ELM: S-ELM) dans lequel une pile de petits ELM est spécialement conçue pour résoudre des problèmes de données complexes et volumineux. Li et al. [91] a proposé un OS-ELM amélioré qui est l'une des variantes ELM avec un facteur d'oubli adaptatif et une stratégie de sélection de mise à jour pour prédire le taux d'utilisation du gaz à l'aide de données variant dans le temps. Yang et al. [92] ont développé une nouvelle approche de régularisation de l'apprentissage récursif de l'OS-ELM pour réduire

l'overfitting et les risques à la fois empiriques et structurels du modèle de prédiction. Lu et al. [49] ont amélioré l'apprentissage récursif de l'OS-ELM en introduisant la propagation de filtre de Kalman d'ensemble pour le réglage des poids de sortie de réseau de neurone sous la prédiction du RUL des moteurs d'avion.

OS-ELM peut répondre à l'apprentissage en ligne en temps réel tel que le problème de prédiction réel sans réglage itératif du taux d'apprentissage, qui contribue fortement à la réduction des coûts de calcul. Les règles d'apprentissage d'OS-ELM permettent un apprentissage récursif avec n'importe quel bloc de nouvelles données arrivées facilement, même qu'avec des tailles différentes ou fixes.

Selon cette brève analyse, nous pouvons conclure que les règles d'apprentissage ELM peuvent être utilisées comme un remplacement des algorithmes d'apprentissage classiques (Backpropagation, divergence contrastive) et contribuent à la résolution les problèmes mentionnés précédemment.

7. Conclusion

A partir de ce chapitre, nous pouvons conclure que l'apprentissage en ligne est la meilleure solution pour une prédiction fiable sous l'utilisation de données variant dans le temps qui sont récupérées à partir des systèmes industriels réels. Cependant, l'apprentissage en ligne ne sera pas possible s'il ne traite pas avec une programmation dynamique Ceci signifie qu'il doit avoir une interaction entre ses hyper-paramètres et des variables physiques ou environnementales, lors des cycles de vie et les changements des caractéristiques des systèmes ou lors des changements des variables environnementaux.

Ce chapitre a également abordé le fait que les outils de Deep Learning sont les plus appropriés à ce type de prédiction, en raison de la philosophie d'apprentissage bien conçue et permettent aux représentations cachées d'être plus significatives.

Si en remplaçant les anciens algorithmes d'apprentissage tels que la rétropropagation et la divergence contrastive par de nouveaux algorithmes rapides tels que l'ELM et en conservant en même temps un apprentissage en ligne adaptatif, les résultats d'estimation peuvent être améliorés en termes de précision et de coûts de calcul. Dans la littérature, et parce que les RNAs sont des outils d'estimation très performants en termes de précision, l'intégration d'ELM sera un pas géant pour les RNAs.

Par conséquent, l'objectif du chapitre suivant est une étude sur les règles d'apprentissage ELM et ses variantes.

Extreme learning machine: nouveau schéma d'apprentissage pour les réseaux de neurones artificiels

Résumé:

Pour pouvoir comprendre les règles d'apprentissage de base de l'ELM et de ses variantes, ce chapitre est conçu pour informer le lecteur sur les principes fondamentaux de l'ELM et son évolution, en commençant par l'ELM hors ligne de base vers l'apprentissage en ligne, l'apprentissage multicouches et le Deep Learning et les autres paradigmes de l'apprentissage automatique.

1. Introduction

Dans les paradigmes d'apprentissage des réseaux de neurones artificiels, les algorithmes classiques tels que la rétropropagation visent à aborder les théories d'apprentissage biologique à travers un réglage itératif de tous les hyper-paramètres des couches cachées pour chaque séquence de données. Par conséquent, sans considérer le réapprentissage, dans les problèmes réel, le processus de réglage pourrait prendre des jours ou des mois sous l'utilisation des ordinateurs ordinaires disponibles pour pouvoir généraliser le réseau à l'ensemble des échantillons définis. Mais, le fait qu'un microprocesseur soit plus rapide qu'un cerveau d'environ douze millions de fois, nie que ces algorithmes soient capables de répondre à la pensée du cerveau humain qui prend moins de secondes pour classer ou restaurer de nouvelles images ou sensations.

En 2004, ELM donne la naissance à de nouvelles règles d'apprentissage pour les réseaux de neurones artificiels. Les règles d'apprentissage de l'ELM éliminent les barrières entre la pensée biologique humaine et les réseaux de neurones artificiels en abordant le fait que: les paramètres de la couche cachée n'ont pas besoin d'être ajustés, et le seul élément responsable de l'approximation universelle et de la généralisation sont les poids de sortie [93]. En conséquence, ELM a été étudié à travers plusieurs applications et s'étend à une multitude de paradigmes tels que l'ensemble, l'hybride et le Deep Learning et atteint une excellente réputation.

Par conséquent, l'objectif de ce chapitre est la représentation des règles d'apprentissage de base de l'ELM et de ses principales variantes qui ont été développées et étudiées au cours de ces deux dernières décennies.

2. Les variantes de l'ELM

2.1. Extreme Learning Machine

L'ELM a été d'abord proposé pour l'apprentissage d'un réseau neuronal à couche cachée unique. Ensuite, il s'étend pour s'adapter à une variété d'architectures pour répondre à plusieurs types d'applications et de paradigmes d'apprentissage. Les règles d'apprentissage de base de l'ELM étaient basées sur les méthodes des moindres carrés qui utilisent « Moore-Penrose" le pseudo-inverse de la matrice pour le réglage des hyper-paramètres.

Contrairement aux anciens algorithmes complexes tels que la rétro-propagation, la divergence contrastive, le Swarm intellegence, les premières règles ont donné à l'algorithme de l'ELM, utilise un paradigme d'apprentissage hors ligne qui peut être simplifié en seulement trois étapes.

Pour un ensemble d'apprentissage donné $S=\{X,T\}$ tels que **X** et **T** sont ses entrées et sorties respectivement, les étapes, sont:

Etape 01 : La matrice des poids d'entrées W et le vecteur des biais b sont générés aléatoirement à partir de toute distribution stochastique disponible, indépendamment des données d'apprentissage.

Etape 02 : La couche cachée H de l'ensemble de données est calculée et activée à l'aide de toute fonction d'activation limitée continue G comme indiqué dans l'équation **[III.1]**.

$$H = G(WX + b) \qquad \textbf{[III.1]}$$

Etape 03 : les poids de sortie β doivent être déterminés analytiquement en utilisant le pseudo-inverse de la couche cachée **[III.2]**.

$$\beta = H^{-1}T \quad \textbf{[III.2]}$$

Les règles de l'ELM tentent de résoudre le problème de minimisation présenté **[III.3]**

$$e = \min(H\beta - T) \quad \textbf{[III.3]}$$

La **figure III.1** ci-dessous représente l'architecture des réseaux ELM avec ses notations de base des paramètres.

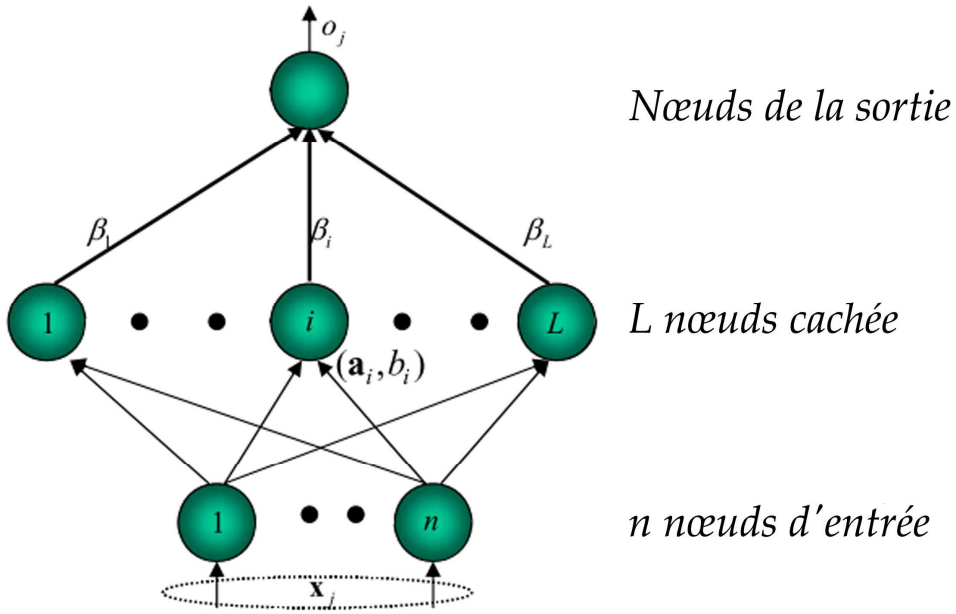

Figure: III.1. Architecture de base d'un réseau ELM [94].

Le nombre de nœuds cachés l dans ELM est déterminé expérimentalement en fonction de chaque type d'applications. Généralement, il y a plus de nœuds cachés, plus la précision ne peut être atteinte.

L'algorithme de L'ELM ne présente aucun réglage du taux d'apprentissage ou des fonctions de tolérance, ce qui simplifie les paradigmes d'apprentissage et réduit l'intervention humaine. En

plus de cela, les algorithmes ELM ont été testés sur des ordinateurs personnels de faibles capacités et prouvent qu'il n'a pas besoin de ressources de calcul coûteuses [9].

2.1.1. Régularisation de l'ELM

L'ELM montre qu'il s'agit d'une vitesse d'apprentissage extrêmement rapide et de bonnes performances de généralisation. Mais, il peut toujours être considéré comme un thème de minimisation de risque qui tend à générer un modèle qui peut être sujet à un problème d'overfitting.

Afin de remédier à ces inconvénients, le R-ELM a été proposé [95] pour résoudre ces problèmes sans affecter le temps d'apprentissage.

Selon la théorie de l'apprentissage statistique, le risque réel de l'apprentissage est composé du risque empirique et du risque structurel. Un modèle avec une bonne capacité de généralisation devrait avoir le meilleur compromis entre les deux risques. Par conséquent, le risque réel peut être représenté par la somme pondérée de ces deux types de risque. Leur proportion peut être régularisée en introduisant un facteur de pondération pour le risque empirique. Le risque empirique est représenté par un carré de la somme d'erreur et le risque structurel peut être représenté avec la somme des poids au carré de la couche de sortie [95]. De plus, afin d'obtenir une estimation robuste, on peut pondérer la variable d'erreur par des facteurs de pondération comme cela est expliqué dans la fonction de minimisation [**III.4**].

$$\min\left(\frac{1}{2}\gamma\left\|De\right\|^2+\frac{1}{2}\left\|\beta\right\|^2\right) \qquad \textbf{[III.4]}$$

où:

γ : est un hyper-paramètre de régularisation réglés après le processus d'apprentissage.

D : la matrice des hyper-paramètres de pondération de l'erreur.

En réglant γ, on peut ajuster la proportion de risque empirique et de risque structurel. Le compromis optimal entre ces deux risques permettra au modèle d'obtenir les meilleures performances de généralisation.

selon R-ELM [95], la méthode de Lagrange présente le problème d'optimisation actuel [**III.5**]:

$$L(\beta, e, \alpha) = \frac{\gamma}{2}\|De\|^2 + \frac{1}{2}\|De\|^2 - \alpha(H\beta - T - e) \qquad \textbf{[III.5]}$$

où

α : est un vecteur des multiplicateurs de Lagrange. En fixant les gradients de ce lagrangien par rapport à ***(β, e, α)*** égal à zéro, il donne les conditions d'optimalité suivantes :

$$\begin{cases} \dfrac{\partial L}{\partial \beta} \rightarrow \beta^T = \alpha H \\ \dfrac{\partial L}{\partial e} \rightarrow \gamma e^T D^2 + \alpha = 0 \\ \dfrac{\partial L}{\partial \alpha} \rightarrow H\beta - T - e = 0 \end{cases} \qquad \textbf{[III.6]}$$

La substitution des dernières expressions dans **[III.6]** dans la deuxième expression donne une expression explicite de ***α***:

$$\alpha = -\gamma(H\beta - T)^T \qquad \textbf{[III.7]}$$

Finalement, la démonstration mathématique de la fonction objective basée sur la méthode de Lagrange peut conduire à une nouvelle formule de réglage des poids de sortie [**III.8**].

$$\beta = (\frac{I}{\gamma} H^T D^2 H)^{-1} H^T D^2 T \qquad \textbf{[III.8]}$$

Lorsque la matrice de pondération est devenue une matrice unitaire, la formule de réglage des poids devient [**III.9]**:

$$\beta = (\frac{I}{\gamma} H^T H)^{-1} H^T T \qquad \textbf{[III.9]}$$

Nous appelons [**III.9]** comme ELM régularisé non pondéré. Il n'est pas si difficile de trouver que l'ELM est juste le cas particulier de l'ELM régularisé non pondéré lorsque $\gamma \to \infty$.

Il existe de nombreux types de méthodes pour calculer les poids de la matrice ***D***. Par exemple, supposant que $D = \{d_1, d_2, d_3, \ldots, d_N\}$:

$$d_i = \begin{cases} 10^{-4} \\ 1, |e_i / s| \\ (c_2 - |e_i / s|)/(c_2 - c_1), c_1 \le e_i / s \le c_2 \end{cases} \qquad \textbf{[III.10]}$$

où

s : est une estimation robuste de l'écart type des variables d'erreur de l'ELM régularisées non pondérées e. s est calculé selon cette équation :

$$s = \frac{IQR}{2 \cdot 0.6745} \qquad \textbf{[III.11]}$$

L'intervalle interquartile IQR[10] est la différence entre le 75eme centile[11] et le 25eme centile. Dans l'estimation de S on tient compte de l'écart entre la distribution d'erreur estimée et la distribution gaussienne. C_1 et C_2 sont généralement définis comme $C_1 = 2{,}5$ et $C_2 = 3$.

[10] Les quartiles désignent des valeurs qui divisent une table de données (ou seulement une partie) en quatre groupes contenant un nombre approximativement égal d'observations.

Par conséquent, l'algorithme de l'ELM régularisé peut être résumé dans les six étapes suivantes:

Etape 01: les poids d'entrée et les biais sont généralement générés de la même manière que l'ELM de base.

Etape 02: la couche cachée est également déterminée selon l'équation [**III.1**].

Etape 03: les poids initiaux de la couche de sortie sont calculés selon l'équation [**III.9**].

Etape 04: le multiplicateur de Lagrange est déterminé utilisant la formule [**III.7**].

Etape 05: la matrice de pondération est calculée sur la base des équations [**III.10**] et [**III.11**].

Etape 06: la version finale de la matrice des poids de la couche de sotie est déterminée selon la formule [**III.8**].

2.2. Online Sequential Extreme Learning Machine

Comme ELM apprend une grande quantité de données à la fois, les poids de sortie de la couche cachée sont également réglés pour une fois et ne peut pas être mis à jour. Par conséquent, un nouveau problème de mise à jour des poids apparaît lorsque les données changent avec le temps. Ainsi, l'ELM libère l'OS-ELM comme l'une de ses variantes qui dépendent d'une attitude d'apprentissage en ligne avec des séquences de données de taille variable ou fixe, car il pourrait également apprendre des données qui viennent consécutivement dans une instance [23].

[11] En statistique descriptive, un **centile** est chacune des 99 valeurs qui divisent les données triées en 100 parts égales, de sorte que chaque partie représente 1/100 de l'échantillon de population

Expérimentalement et selon la méthode récursive des moindres carrés, OS-ELM a donné les règles de l'apprentissage qui se sont divisées en deux phases essentielles: la phase initiale et la phase séquentielle :

Pour un petit ensemble de données donné $D= (X_k, T_k)$ ou k est le nombre des séquences de données :

- **La phase initiale :**

- Tous les poids d'entrée et les biais de la couche cachés sont générés aléatoirement. Les paramètres des nœuds cachés (poids et biais) doivent être normalisés entre {-1,1} [96].
- La couche cachée initiale H_k $(k = 0)$ doive être calculé selon les mêmes théories de l'ELM de base que celles présentées dans [**III.1],** où G peut être généré indépendamment des données d'apprentissage selon une fonction d'activation bornée continue.
- La détermination de la matrice des covariances initiales P_k $(k = 0)$ de la couche cachée est basée sur l'équation [**III.12]**.
- La matrice de poids de sortie initiales β_k $(k = 0)$ a été aussi déterminé comme indiqué dans l'équation **[III.2]**.

- **La phase séquentielle :**

- Calcul de la nouvelle couche cachée H_{k+1} ; Équation **[III.1]**.
- Mise à jour de la matrice de poids de sortie β_{k+1} utilisant l'équation **[III.13]** qui dépend sur le formules [**III.14**], [**III.15**], et [**III.16**].
- Incrémentation de k, $(k = k + 1)$.

$$P_0 = (H_0^T H_0)^{-1} \quad \textbf{[III.12]}$$

$$\beta_{k+1} = \beta_k - P_{k+1} H_{k+1}^T e_{k+1} \quad \textbf{[III.13]}$$

$$e_{k+1} = T_{k+1} - H_{k+1}^T \beta_k \quad \textbf{[III.14]}$$

$$P_{k+1} = P_k - K_{k+1} H_{k+1}^T P_k \quad \textbf{[III.15]}$$

$$K_{k+1} = \frac{P_k H_{k+1}}{H_{k+1}^T P_k H_{k+1}^T} \quad \textbf{[III.16]}$$

2.2.1. Régularisation pour OS-ELM

Bien que l'algorithme OS-ELM semble parfait en théorie, il présente encore des lacunes lorsqu'il est directement appliqué à des applications du monde réel. Comme nous pouvons le voir dans les équation [**III.12**] et [**III.13**], la dérivation de OS-ELM est basée sur l'hypothèse que la matrice des covariances $(H^T H)$ est *non singulière* (les éléments des colonnes et des lignes linéairement indépendant[97]). Une fois le problème singulier ou mal posé, les performances de généralisation de l'OS-ELM se détériorent considérablement. Afin de surmonter ce problème, Huynh et Won [98] ont proposé un OS-ELM régularisé (R-OS-ELM) en utilisant la régularisation de Tikhonov (minimisation de la norme L_2 des poids).

La procédure d'apprentissage du R-OS-ELM est presque la même que celle de l'OS-ELM, il suffit d'ajouter un terme de régularisation à la matrice des covariances pour éviter le problème de singularité [99]. Donc l'ancienne équation **[III.12]** devient :

$$P_0 = (H_0^T H_0 + \delta I)^{-1} \quad \textbf{[III.17]}$$

où $\boldsymbol{\delta}$ est une valeur réelle positive appelée paramètre de régularisation et $\boldsymbol{I}$ est une matrice d'identité avec les mêmes dimensions que $(H^T H)$.

Selon [98,99] , et pour la même base de données de l'OS-ELM, est si nous supposons que, $H_k = [h_1\, h_2\, h_3 \ldots\ldots\ldots, h_n]$, alors les poids initiaux peuvent être déterminés comme l'OS-ELM original.

Ensuite dans le la phase séquentiel, chaque fois qu'un nouveau échantillon (X_k, T_k) arrive, puis

$H_{k-1}^T = [H_{k-1}^T \; h_k^T]^T$ et $T_{k-1}^T = [T_{k-1}^T \; t_k]^T$, et les poids de sortie actuels sont récursivement calculé :

$$\begin{aligned} P_k &= (H_k^T H_k + \delta I)^{-1} = (H_{k-1}^T H_{k-1} + h_k^t h_k + \delta I) \\ &= (P_{k-1}^{-1} + h_k^T h_k)^{-1} = P_{k-1} - \frac{P_{k-1} h_k^T h_k P_{k-1}}{1 + h_k P_{k-1} h_k^T} \end{aligned} \quad \textbf{[III.18]}$$

$$\begin{aligned} \beta_k &= P_k H_k^T T_k = P_k (H_{k-1}^T T_{k-1} + h_k^T t_k) \\ &= P_k (P_{k-1}^{-1} P_{k-1} H_{k-1}^T T_{k-1} + h_k^T t_k) \\ &= P_k ((P_k^{-1} - h_k^T t_k)\beta_{k-1} + h_k^T t_k) \\ &= \beta_{k-1} + P_k h_k^T (t - h_k^T \beta_{k-1}) \end{aligned} \quad \textbf{[III.19]}$$

La démonstration mathématique de la substitution de la nouvelle matrice de covariance régularisée dans la phase séquentielle, prouve que les autres termes de mise à jour resteront les mêmes que l'OS-ELM de base. Cependant nous devons garder que dans la procédure d'apprentissage séquentiel de R-OS-ELM, l'élément de régularisation $\boldsymbol{\delta I}$ est intégré dans les formules récursives. En outre, si le paramètre de régularisation $\boldsymbol{\delta}$ est égal à zéro, puis le R-OS-ELM dégénère en l'original OS-ELM.

2.2.2. Le paramètre d'oubli pour OS-ELM

Pour un système qui varie dans le temps en termes de caractéristiques physiques, le comportement du système change souvent avec le temps. En conséquence, les nouveaux exemples sont plus efficaces pour refléter les tendances de changement du système par rapport aux anciens. Pour mieux exprimer l'opportunité des échantillons et améliorer encore les performances de prédiction pour le système variant dans le temps, la méthode de la fonction d'oubli ainsi que la technique de régularisation sont introduites dans l'OS-ELM, et l'algorithme FR-OS-ELM est proposé [100]. Différent de l'OS-ELM et du R-OS-ELM qui traitent tous les échantillons d'entraînement de manière égale, le FR-OS-ELM attribue un poids différent à

chaque échantillon, respectivement, selon leur ordre de temps. C'est-à-dire que les échantillons récents se voient attribuer des poids plus élevés tandis que les anciens échantillons se voient attribuer des poids inférieurs afin de représenter leurs différentes contributions au modèle d'apprentissage.

L'implémentation de FR-OS-ELM peut être brièvement décrite comme suit:

Supposons que nous ayons des poids de sortie initiaux $\boldsymbol{\beta_{k+1}}$.

Maintenant un nouvel échantillon d'apprentissage (X_k, T_k) arrive; alors le correspondant $\boldsymbol{\beta_{k+1}}$ est à l'origine exprimé comme:

$$\beta_k = (H_{k-1}^T H_{k-1} + \delta I + h_k^T h_k)(H_{k-1}^T T_{k-1} + h_k^T t_k) \quad \textbf{[III.20]}$$

Compte tenu de l'efficacité différente de l'échantillon nouvellement arrivé et des échantillons du modèle [**III.20**] peut être écrit comme suit:

$$\beta_k = (\lambda H_{k-1}^T H_{k-1} + \lambda\delta I + h_k^T h_k)(\lambda H_{k-1}^T T_{k-1} + h_k^T t_k) \quad \textbf{[III.21]}$$

Où $\boldsymbol{\lambda \in (0, 1]}$ est appelé le facteur d'oubli, qui est utilisé pour affaiblir les influences des anciens échantillons et renforcer indirectement les effets du dernier sur le modèle afin de mieux représenter l'état actuel du système variant dans le temps.

Considerons:

$$P_k = (\lambda H_{k-1}^T H_{k-1} + \lambda\delta I + h_k^T h_k)^{-1} \quad \textbf{[III.22]}$$

En inversant les deux côtés de l'équation [**III.22**]:

$$P_{k-1}^{-1} = \lambda P_{k-1}^{-1} + h_k^T h_k \quad \textbf{[III.23]}$$

Application de la formule Sherman-Morrison aux [**III.23]**:

$$P_k = \frac{P_{k-1}}{\lambda} - \frac{P_{k-1} h_k^T h_k P_{k-1}}{\lambda(\lambda + h_k P_{k-1} h_k^T)} \qquad \textbf{[III.24]}$$

En substituant [**III.22**] à [**III.21**], on obtient :

$$\begin{aligned} \beta_k &= P_k(\lambda H_{k-1}^T T_{k-1} + h_k^T t_k) \\ &= P_k(\lambda P_{k-1}^{-1} P_{k-1} H_{k-1}^T T_{k-1} + h_k^T t_k) \\ &= P_k(\lambda P_{k-1}^{-1} \beta_{k-1} + h_k^T t_k) \\ &= P_k((P_k^{-1} - h_k^T t_k)\beta_{k-1} + h_k^T t_k) \\ &= \beta_{k-1} + P_k h_k^T (t - h_k^T \beta_{k-1}) \end{aligned} \qquad \textbf{[III.25]}$$

Les équations [**III.24**] et [**III.25**] donnent les formules récursives pour la mise à jour des poids de sortie de FR-OS-ELM. Lorsque $\boldsymbol{\delta = 1}$, le FR-OS-ELM dégénère en R-OS-ELM général. Lorsque $\boldsymbol{\lambda = 1}$ ***et*** $\boldsymbol{\delta = 0}$, le FR-OS-ELM dégénère en OS-ELM d'origine.

3. Les principales améliorations pour l'ELM

3.1. ELM multicouches

Les théories de l'ELM montrent également que l'architecture de base ELM peut être étendue pour s'adapter à une architecture multicouches [94]. Contrairement aux anciens paradigmes de l'apprentissage, dans ELM, l'architecture multicouche ne permet aucun réglage des paramètres des couches cachées. En fait, les poids de sortie porteront le coefficient important qui amène la dernière couche cachée à la sortie.

La **figure III.2** donne l'architecture multicouche de l'ELM.

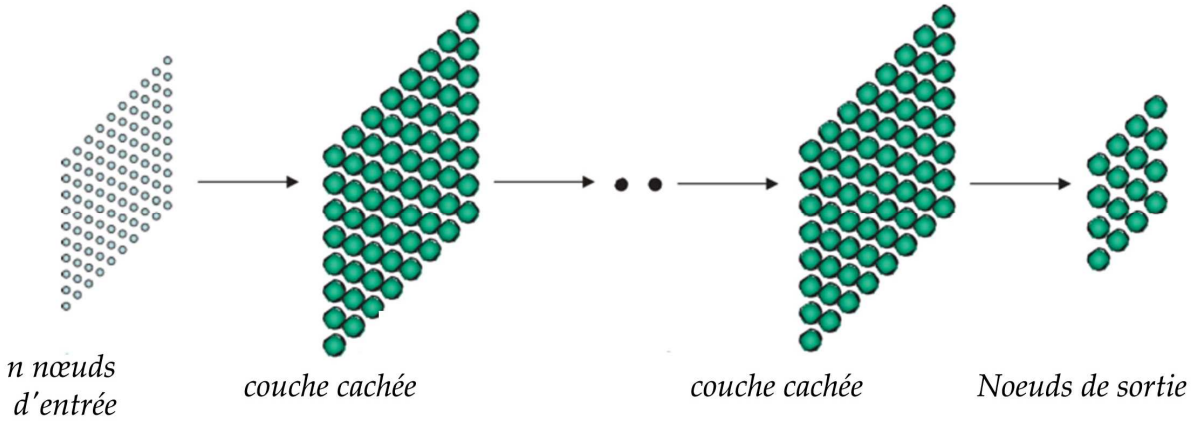

Figure: III.2. Architecture multicouches ordinaire de l'ELM

Comme cela est présenté dans la **figure III.3**, et en tenant compte du fait que le nombre et les connexions des neurones biologiques humains sont encore inconnus, les théories de l'ELM montrent également que les nœuds cachés peuvent sous forme de sous-réseaux à condition que tous les paramètres des couches internes soient générés aléatoirement [94].

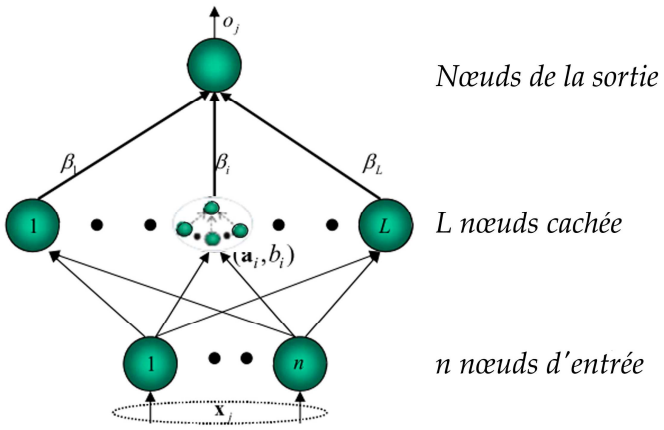

Figure: III.3. Réseau ELM avec un nœud combinatoire

Dans l'ELM, les nœuds combinatoires pouvaient être entièrement connectés ou bien localement connectés, en fonction du type d'apprentissage comme était illustré sur la **figure III.4.**

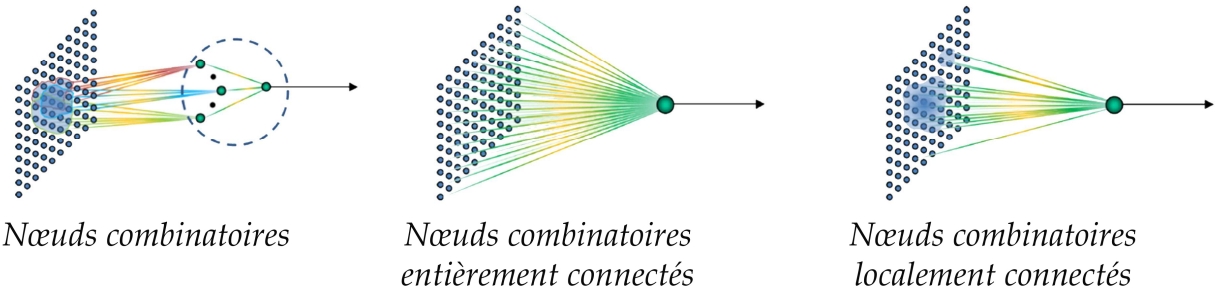

Figure: III.4. Différents types possibles de nœuds dans l'architecture multicouches

Une illustration avec MATLAB de la structure multicouche avec l'algorithme ELM peut être trouvé dans le lien [101][12] avec des exemples des nœuds combinatoire entièrement connecté dans[102][13].

3.2. L'ELM et le Deep Learning

L'ELM est différent du Deep Learning dans le sens où les neurones cachés de l'ELM entier n'ont pas besoin d'être réglés. En raison des différents rôles d'ELM d'apprentissage, ELM peut être utilisé comme réseau multicouches dans lequel les couches tardives sont entraînées par d'autres méthodes telles que le Deep Learning.

Parmi les variantes les plus populaires de l'ELM qui ont été développées jusqu'à présent et utilisées dans la littérature nous citons: l'ELM-ELR (ELM with Local Receptive Fields) et le Stacked ELM.

[12] Cet algorithme illustre l'architecture multicouches des théories d'apprentissage hors ligne de l'ELM basiques sans aucune fonctionnalité additive ni réglage des couches cachées.

[13] L'algorithme est un exemple MATLAB pour créer des sous-réseaux dans une simple couche cachée.

3.2.1. Extreme Learning Machine avec champs réceptifs locaux

Extreme Learning Machine avec champs réceptifs locaux (Extreme Learning Machine with Local Receptive Fields:ELM-LRF) est la version CNN de l'ELM. Il permet un processus de prédiction en passant par différentes étapes de projection: convolution et sous-échantillonnage, comme les CNN. cependant, ELM-LRF est différent des CNN en termes de réglage dans lequel aucun réglage de nœuds cachés n'est nécessaire. Seuls les poids de sortie doivent être déterminés analytiquement selon la méthode Pseudo-inverse [103].

Bien que ELM-LRF et contrairement aux CNN, il supporte un large type de champs récepteurs locaux tant qu'ils sont continus non linéaires, un nœud convolutionnels aléatoire peut être considéré comme un nœud combinatoire spécifique de ELM-LRF comme illustré dans la **figure III.5**.

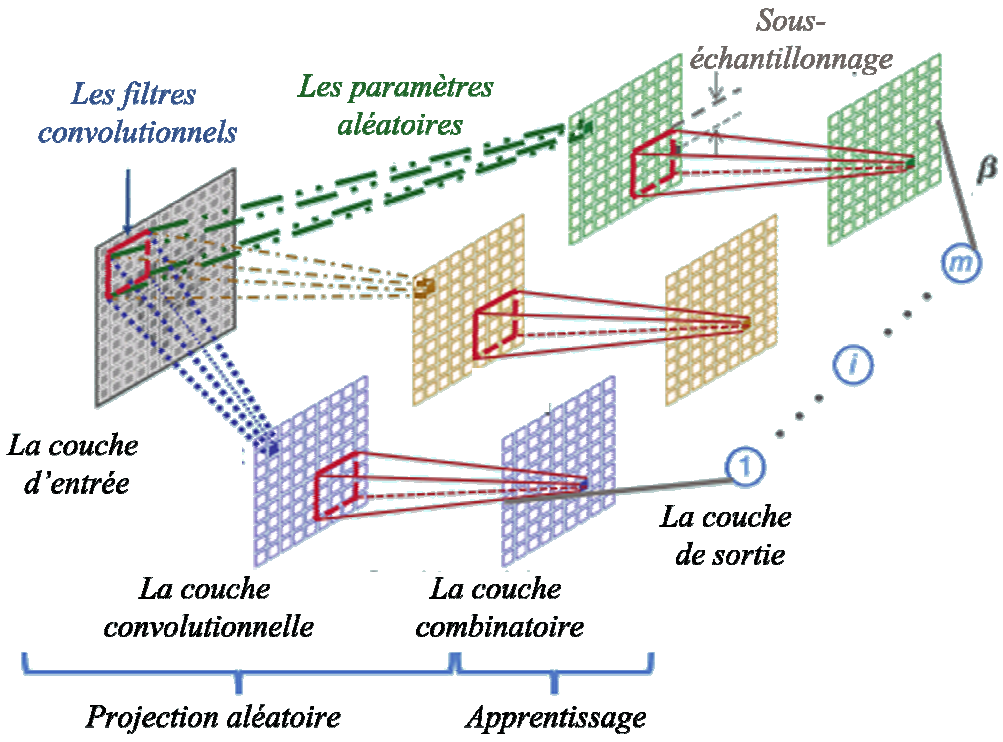

Figure: III.5. Architecture de base d'un réseau hiérarchique de type ELM-LRF

3.2.2. Stacked ELM

Stacked ELM est une variante de l'ELM basée sur des paradigmes d'apprentissage similaires au DBN ordinaire. Stacked ELM utilise des autoencodeurs connectés en série pour effectuer une représentation de données de plus grande précision sur la base d'un petit ensemble de données d'apprentissage. Après cela, une seule opération de réglage fin concentre la couche de sortie pour un apprentissage supervisé. Dans les anciens DBN, et après un processus d'apprentissage non supervisé, le réglage fin est également pris en considération pour réajuster tous les poids de l'apprentissage supervisées et non supervisée contrairement à Stacked ELM qui ne règle que les poids de sortie [103]. La **figure III.6** présente quelques indications sur les règles d'apprentissage de l'architecture Stacked-ELM

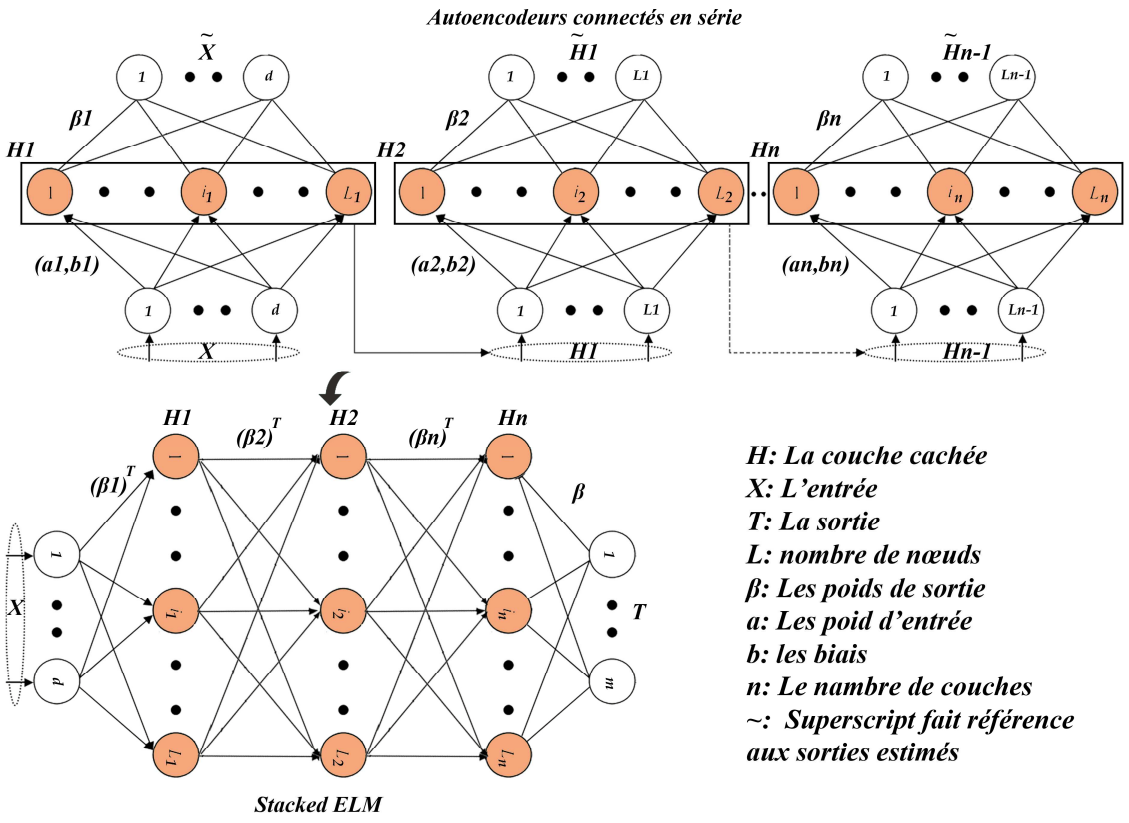

Figure: III.6 . Architecture générale Stacked ELM

4. Conclusion

ELM est un algorithme très rapide pour l'apprentissage de réseaux de neurones avec différentes architectures. Il possède une flexibilité pour s'adapter à tous les paradigmes de Machine Learning tels que les multicouches ou le Deep Learning.

L'apprentissage en ligne de l'ELM a l'avantage de ne pas nécessiter de réglage du taux d'apprentissage ou de problèmes de minimum local en raison de l'utilisation de la formule de SMW. Ainsi qu'il a la possibilité d'aborder l'apprentissage adaptatif qui permet le suivi des données variant dans le temps.

Le seul problème des règles de l'algorithme ELM est le risque structurel produit par la méthode des moindres (*obtenir une couche cachée linéairement indépendante*) carrés qui reste jusqu'à présent comme un problème ouvert pour la recherche et l'amélioration.

Maintenant, Après avoir étudié l'ELM sur plusieurs points, nous concluons qu'il est capable de gérer la détection des défauts dans le cadre de l'évaluation de la santé des systèmes. Dans la mesure où la plupart des règles et variantes de l'ELM importantes dans notre travail ont été abordées, la prochaine partie de ce travail (partie 02) s'attachera à expliquer son utilisation dans différentes applications du domaine du pronostic et de la gestion de la santé. Elle impliquera l'utilisation de l'ELM non seulement dans l'apprentissage non supervisé, mais également dans l'apprentissage supervisé avec des architectures de Deep Learning. En étudiant l'ELM pour des utilisations différentes, nous allons essayer de prouver sa flexibilité et surtout promouvoir son utilisation comme source alternative de prédiction pour l'évaluation de la santé. La seconde partie est entièrement consacrée à nos principales contributions dans le domaine du pronostic et de l'évaluation de la santé.

Partie 02:

Applications de l'Extreme Learning Machine dans le domaine du pronostic et de l'évaluation de la santé

Extreme Learning Machine pour la compression des données de pronostic

Résumé:

Dans ce chapitre, une nouvelle approche de la compression avec perte développée est présentée. L'algorithme de compression conçu est un autoencodeur basé sur OS-ELM. Tout d'abord, une adaptation dynamique de l'algorithme aux séquences de données est obtenue en intégrant une stratégie de sélection mise à jour (USS: Updated Selection Strategy) et une fonction d'oubli dynamique (DFF: Dynamic Forgetting Factor) pour rendre l'algorithme d'apprentissage capable de résister à la variation en échantillons. Deuxièmement, le (SVD: Singular Value Decompostion) est impliqué pour améliorer la représentation des couches cachées via un Sparse Coding. Ce nouvel autoencodeur développé est comparé à l'autoencodeur ordinaire a prouvé sa précision sur une base de données C-MAPSS.

1. Introduction

Il est connu que les réseaux de neurones avec différentes architectures et paradigmes d'apprentissage ont obtenu des succès dans la représentation des données, l'extraction des paramètres et la compression des données avec perte, en particulier pour les images [104–106].

Pendant ce temps, les autoencodeurs basés sur des algorithmes de l'apprentissage classique comme la rétropropagation souffrent de nombreux problèmes tels que: l'overfitting, critères d'arrêt des minima locaux, temps et intervention humaine [9]. D'autres parts, l'ELM a été introduit pour l'apprentissage d'un réseau neuronal à une seule couche cachée qui tente de résoudre ces problèmes. ELM est un outil très important dans une variété d'applications dans différents domaines [107].

L'ELM a d'abord été présenté comme une méthode d'apprentissage hors ligne puis étendu pour s'adapter à l'apprentissage en ligne et à tout autre paradigme d'apprentissage tel que l'apprentissage adaptatif et l'apprentissage par renforcement [9,107].

Le « Sparse Coding » est un domaine bien connu de l'extraction des caractéristiques, de l'analyse ou de la reconstruction de données, spécialement pour le traitement d'images [108]. Des représentations 'sparse' peuvent être réalisées en utilisant plusieurs transformations telles que Fourier, les ondelettes, le domaine discret et SVD (Singular Value Decomposition) [109]. La minimisation des normes L_1 et L_2 est souvent utilisée pour la reconstruction des caractéristiques dans les problèmes de sparse coding [110]. La SVD basée sur l'exploration des données est largement utilisée dans l'ELM [37,109,110], en raison de sa rapidité, de traitement non itérative et paradigme de reconstruction.

Contrairement à d'autres techniques de représentation de Sparse Coding, la reconstruction des caractéristiques SVD peut facilement interagir avec les outils de minimisation de la norme L_2 et produire des résultats précis. D'autres techniques peuvent donner des résultats précis en

utilisant la minimisation de la norme L_1, mais c'est un algorithme itératif [35], et nécessite plus de temps et de coûts de calcul.

Sur la base de la brève analyse précédente, dans cette section, un nouvel autoencodeur baptisé «adaptive sparse online sequential autoencodeur» et basé sur OS-ELM pour la compression avec perte de données variant dans le temps est développé.

Les performances de la nouvelle approche de compression sont étudiées sous l'utilisation de l'ensemble de données C-MAPSS et comparées à un autoencodeur ordinaire ont prouvé leur précision.

Les algorithmes de l'apprentissage aussi sont conçus pour confirmer que l'ELM n'est pas en mesure de traiter uniquement l'apprentissage supervisée. Mais en fait, il est capable de traiter la conception de l'ensemble de tout le réseau de prédiction en l'impliquant dans la préparation des données et la prédiction en même temps.

Concernant la distribution de nos algorithmes sur le web, nous utilisons deux principaux serveurs web qui permettent le montage gratuit d'algorithmes de Machine Learning: « Google Colab[14]» et «Octave Online[15]». Ces serveurs peuvent gérer à la fois des codes basés sur Python et MATLAB

[14] Colaboratory, ou «Colab» en abrégé, est un produit de Google Research. Colab permet d'écrire et d'exécuter du code Python arbitraire via le navigateur, et est particulièrement bien adapté au Machine Learning, à l'analyse de données et à l'éducation. Plus techniquement, Colab est un service de notebook Jupyter hébergé qui ne nécessite aucune configuration à utiliser, tout en offrant un accès gratuit aux ressources informatiques, y compris les GPU.

[15] GNU Octave (General Public License) est un logiciel doté d'un langage de programmation de haut niveau, principalement destiné aux calculs numériques. Octave aide à résoudre numériquement des problèmes linéaires et non linéaires et à réaliser d'autres expériences numériques en utilisant un langage principalement compatible avec MATLAB. Puisqu'il fait partie du projet GNU, il s'agit d'un logiciel libre sous les termes de la GNU.

Le chapitre vise à introduire une brève description de la base de données du système étudié. Des informations générales sur l'algorithme de base utilisé sont également présentées. Enfin, ce chapitre se termine par la présentation de l'approche développée.

2. Les données C-MAPSS

Dans le logiciel C-MAPSS, le moteur simulé, est un turboréacteur à double flux et double bobinage avec une poussée élevée jusqu'à 4.082.328 kilogrammes qui apparaît dans le diagramme de la **figure IV.1**. Le moteur fonctionne dans des conditions d'altitude inférieures, au niveau de la mer allant jusqu'à 12.192 mètres et une température ambiante varie de -51 ° C à 39 ° C. Dans ce type de moteurs, l'air contaminé du ventilateur est comprimé en deux étapes en passant par des compresseurs basses et hautes pressions (HPC et LPC). Après cela, dans la chambre de combustion, l'air comprimé est chauffé pour produire suffisamment de poussée pour entraîner les turbines basse et haute pression (LPT et HPT). La puissance de poussée sera produite à la fois par l'air produit par le ventilateur dans le flux de dérivation et par l'air entrant dans le cœur du moteur.

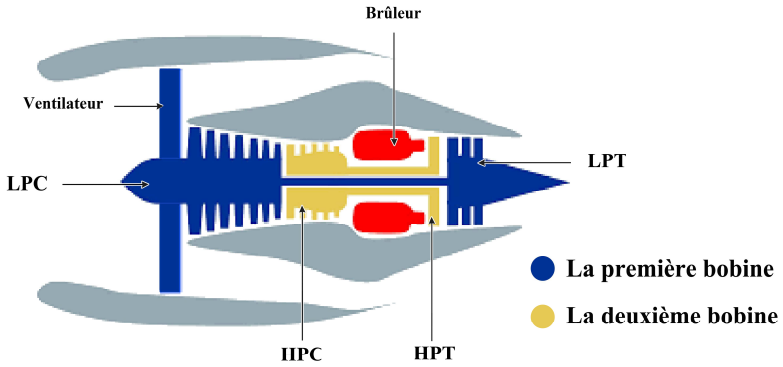

Figure: IV.1. Schéma du type de turboréacteur étudié [111].

Les données retirées du logiciel C-MAPSS fournies au public comme référence pour comparer et étudier les modèles de prédiction du RUL des moteurs d'avion « basés sur les données

(data-driven) » [5]. L'ensemble de données décrit différents modes de défaillance dans différentes conditions de fonctionnement du moteur et se compose de quatre sous-ensembles différents (FD001, FD002, FD003, FD004) où chaque sous-ensemble est divisé en ensembles d'apprentissage et de test. Le **tableau IV.1** indique les principales caractéristique de chaque sous ensemble.

Tableau: IV.1. Caractéristiques générales de la base de données C MAPPS [60].

Base de données	Cycles l'apprentissage	Cycles de test	Conditions de fonctionnement	Modes de défaillance
FD001	100	100	1	1
FD002	260	259	6	1
FD003	100	100	1	2
FD004	249	248	6	2

Selon la fiche technique de l'ensemble de données C-MAPSS [112], les modes de défaillance des cycles de vie présentés respectivement dans les quatre sous-ensembles sont: i) Dégradation HPC, ii) Dégradation HPC, iii) Dégradation HPC & Dégradation des ventilateurs, iv) Dégradation HPC & Dégradation des ventilateurs. Concernant les sous-ensembles qui ont une seule condition de fonctionnement, cela signifie que le moteur fonctionne au niveau de la mer. Par contre, les six conditions de fonctionnement des autres sous-ensembles ne sont pas révélées dans la description de l'ensemble de données.

L'ensemble de données contient des série temporelle multi-variées (cycles de vie) de 26 caractéristiques (numéro de moteur, cycle de temps, conditions de fonctionnement, mesures de capteurs) de 100 profils de dégradation différents de moteurs similaires. **Le défi le plus** intéressant de l'ensemble de données est que les mesures des capteurs sont contaminées par du *bruit* provenant de différentes sources.

Les paramètres de chaque sous-ensemble de l'ensemble de données C-MAPSS sont aussi organisés comme indiqué sur le **tableau IV.2**.

Tableau: IV.2. Statistiques descriptives de toutes les variables de la base de données [44].

Indice	Paramètres	Caractéristiques statistiques	
		La moyenne	La variance
1	Numéro d'unité	/	/
2	Durée de vie restante	108.808	68.881
3	Paramètres opérationnel (1)	−8.870*10−6	0.003
4	Paramètres opérationnel (2)	2.350*10−6	0.003
5	Paramètres opérationnel (3)	100.002	10−6
6	Température totale à l'entrée du ventilateur (°R)	518.670	0
7	Température totale à la sortie du LPC (°R)	642.681	0.500
8	Température totale à la sortie HPC (°R)	1590.523	6.131
9	Température totale à la sortie LPT (°R)	1408.934	9.000
10	Pression à l'entrée du ventilateur (psia)	14.620	3.390×10−6
11	Pression totale dans le conduit de dérivation (psia)	21.609	0.001
12	Pression totale à la sortie HPC (psia)	553.368	0.885
13	Vitesse physique du ventilateur (tr / min)	2388.097	0.070
14	Vitesse de corps des moteur (tr / min)	9065.243	22.082
15	Rapport de pression moteur (P50 / P2)	1.300	4.66×10−13
16	Pression statique à la sortie HPC (psia)	47.5412	0.267
17	Rapport du débit de carburant à Ps30 (pps / psi)	521.414	0.738
18	Vitesse du ventilateur corrigée (tr / min)	2388.096	0.719
19	Vitesse de base corrigée (tr / min)	8143.753	19.076
20	Rapport de dérivation	8.442146	0.038
21	Rapport d'air combustible du brûleur	0.0300	1.56×10−14
22	Enthalpie de saignement	393.212	1.549
23	Vitesse du ventilateur demandée (tr / min)	2388.000	0
24	Vitesse de ventilation corrigée demandée (tr / min)	100.000	0
25	Purge du liquide de refroidissement HPT (lbm / s)	38.8163	0.181
26	Purge du liquide de refroidissement LPT (lbm / s)	23.279	0.108

Au début de chaque profil de dégradation, le moteur est considéré comme fonctionnant normalement dans les conditions de fonctionnement initiales. A un certain niveau de cycles de fonctionnement, le moteur commence à perdre ses performances et tend vers le mode de défaillance.

Selon le «PHM data challenge 2008» [113], qui est un concours ouvert à toutes les participations aux conférences, la fonction de dégradation dans un cycle de vie peut être définie comme une fonction linéaire à un seuil pour présenter une détérioration non cumulative.

La **figure IV.2** est un exemple du comportement des mesures des capteurs dans un cycle de vie avec sa fonction de détérioration. Les différentes couleurs représentent différents types de mesures de capteurs au cours du cycle de vie.

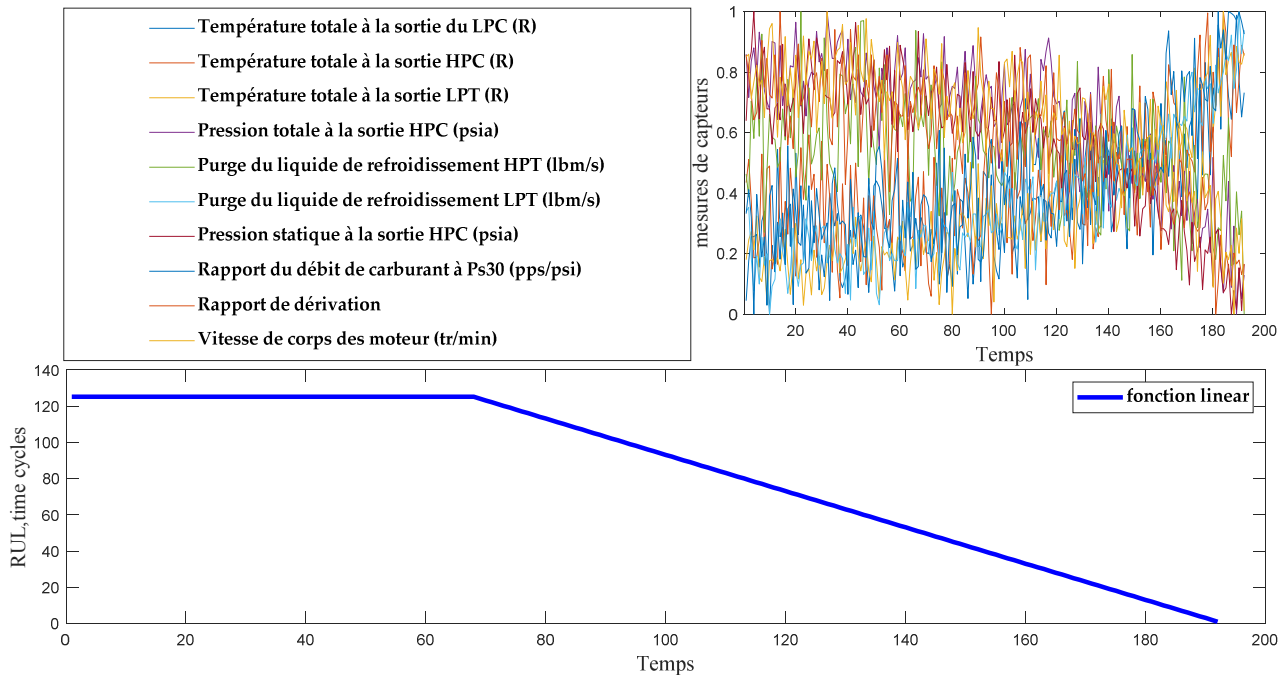

Figure: IV.2. Comportement des mesures des capteurs et de leurs étiquettes en un seul cycle de vie (FD001_train, Cycle 1).

3. Bref rappel sur l'algorithme de l'OS-ELM de base

Les règles de l'apprentissage de l'OS-ELM sont:

- Générer les paramètres de la couche cachée de la même manière que les théories de l'ELM de base.
- Calculer la couche cachée en fonction de toute fonction d'activation en utilisant l'équation **[IV.1].**
- Déterminer les poids de sortie initiaux de la même manière que celle utilisée dans **[IV.3]** à l'aide de la matrice de covariance déterminée par l'équation [**IV.2]**.
- Définir et mettez à jour les poids de sortie pour les nouveaux blocs à venir selon l'équation **[IV.4]** en utilisant les équations **[VI.5],** [**IV.6]** et [**IV.7**].

$$H_{k+1} = G(a, b, X_{k+1}) . \quad \textbf{[IV.1]}$$

$$P_0 = pinv(H_0^T H_0) \ . \quad \textbf{[IV.2]]}$$

$$\beta_0 = P_0 H_0^T T_0 . \quad \textbf{[IV.3]}$$

$$\beta_{k+1} = \beta_k - P_{k+1} H_{k+1}^T e_{k+1} . \quad \textbf{[IV.4]}$$

$$e_{k+1} = T_{k+1} - H_{k+1} \beta_k . \quad \textbf{[IV.5]}$$

$$P_{k+1} = P_k - K_{k+1} H_{k+1}^T P_k . \quad \textbf{[IV.6]}$$

$$K_{k+1} = \frac{P_k H_{k+1}}{H_{k+1}^T P_k H_{k+1}^T} . \quad \textbf{[IV.7]}$$

L'algorithme d'OS-ELM est bien illustré dans **Algorithme IV.1**.

Algorithme IV.1. l'OS-ELM de base.

Début
Les entrées : $\{X, T, n, G, l\}$
Les sorties : $\{\beta\}$

k=1 :
Générer les paramètres de la couche cachée $(a,b) = P(X,l)$;
Calculer la couche cachée en fonction de toute fonction d'activation $H_{k+1} = G(a,b,X_{k+1})$. ;
Déterminer les poids de sortie initiaux $\{\beta_0 = P_0 H_0^T T_0 ; \quad P_0 = pinv(H_0^T H_0)\}$. ;
Pour *k= 1 : n :*
Mettez à jour les poids de sortie:
$\{e_{k+1} = T_{k+1} - H_{k+1}\beta_k ; \quad P_{k+1} = P_k - K_{k+1} H_{k+1}^T P_k ; \quad K_{k+1} = (P_k H_{k+1})(H_{k+1}^T P_k H_{k+1}^T)^{-1} ; \beta_{k+1} = \beta_k - P_{k+1} H_{k+1}^T e_{k+1} .\}$;
fin
fin

4. Approche développée pour la compression des données dynamiques

Les autoencodeurs utilisés comme des extracteurs de paramètres pour donner aux données une représentation plus significative via une représentation spéciale [114]. Les autoencodeurs sont un type de réseaux de neurones qui se sont entraînés en définissant les sorties des réseaux de neurones que la même manière que les échantillons d'entrée.

Ce type d'apprentissage est un apprentissage non supervisé qui n'a pas besoin d'étiquettes de l'apprentissage [82]. Les autoencodeurs font généralement un apprentissage à l'aide d'un algorithme de rétropropagation pour le réglage des poids ou des biais [115].

Dans le nouvel algorithme proposé et baptisé **SA-OS-AE** «Sparse Adaptive On-line Séquentiel AutoEncoder» un OS-ELM amélioré est utilisé pour régler les paramètres d'apprentissage de l'autoencodeur de compression. Nos contributions dans le cadre de l'amélioration du processus de l'apprentissage de l'OS-ELM sont:

- Contrairement aux paramètres d'oubli statique (voir chapitre 3, § 2.2.2), l'intégration d'une fonction d'oubli dynamique permet de mieux aborder l'apprentissage adaptatif. Un paramètre dynamique qui change avec le temps peut ajuster les poids

d'apprentissage de manière séquentielle pour s'adapter à nouvelles formes (patterns) de données.

- Une stratégie de sélection pour la meilleure adaptation de la compression aux données nouvellement arrivées et pour la sélection de séquences importantes. Le processus de sélection des meilleurs échantillons peut également impliquer une minimisation empirique des risques, ce qui réduit le overfitting (voire chapitre 2, § 2).
- Intégration de la programmation «Sparse Coding» pour obtenir une représentation des données plus significative que la représentation ordinaire des couches cachées.

La nouveauté de l'algorithme proposé réside dans la combinaison des trois principales caractéristiques à savoir le facteur d'oubli dynamique, la stratégie de sélection mise à jour et le Sparse Coding. Et ce dans le cadre d'un apprentissage non supervisé.

4.1. Facteur d'oubli dynamique

Selon [116], et contrairement aux anciennes utilisations des paramètres d'oubli statique (voir Chap 3, § 2.2.2), la mise à jour dynamique des facteurs d'oubli peut éliminer progressivement les anciennes données d'entraînement sans affecter les poids d'apprentissage du réseau de neurones. La fonction d'oubli proposée a été inspiré de [116] est illustré dans [**IV.8**].

$$\lambda_{k+1} = \lambda_{\min} + (1-\lambda_{\min})e^{-\mu\left\|e_{k+1}\right\|}. \qquad \textbf{[IV.8]}$$

où : λ représente le facteur d'oubli, λ_{min} la valeur minimale où λ_{min} = (0,1] et μ est un facteur de sensibilité qui contrôle la vitesse de convergence vers la valeur 1. Des paramètres de contraintes doivent être ajoutés pour contrôler la convergence [**IV.9**].

$$\begin{cases} \lambda_{k+1} = 1, \lambda_{k+1} \geq 1 \\ \lambda_{k+1} = \lambda_{\min}, \lambda_{k+1} \leq 0 \end{cases} \quad \textbf{[IV.9]}$$

Le facteur d'oubli dynamique a été intégré dans les formules de mise à jour des matrices de covariance et de gain présentées par les équations [**IV.10]** et [**IV.11]**

$$P_{k+1} = \frac{1}{\lambda_{k+1}}(P_k - K_{k+1} H_{k+1}^T P_k). \quad \textbf{[IV.10]}$$

$$K_{k+1} = \frac{P_k H_{k+1}}{\lambda_{k+1} + H_{k+1}^T P_k H_{k+1}^T}. \quad \textbf{[IV.11]}$$

Si nous comparons notre mécanisme d'oubli de contradiction avec celui présenté dans [116]. Dans [116] les auteurs intègrent cette fonction dans l'apprentissage supervisé où la formule d'erreur présentée par l'équation **[IV.12]** est utilisée pour estimer l'erreur de prédiction. Dans le cadre de notre approche, nous avons utilisé l'erreur de reconstruction de l'apprentissage non supervisé $\{T = X\}$ pour contrôler les poids d'apprentissage.

4.2. Stratégie de sélection mise à jour

La stratégie de sélection mise à jour est introduite pour accélérer le processus de l'apprentissage en utilisant la condition présentée dans [**IV.12**] [91] au lieu de au lieu de celle présentée dans [**IV.4**], pour permettre uniquement aux séquences importantes de données de passer par le processus de mise à jour et de rejeter les indésirables. Si la condition $\|e_{k+1}\| \geq \|e_k\|$ est vérifiée, la séquence passera au processus d'apprentissage. Sinon, l'algorithme d'apprentissage passera à la séquence de données suivante et effectuera le même test dessus jusqu'à ce que les séquences d'apprentissage soient terminées.

$$\beta_{k+1} = \begin{cases} \beta_k - P_{k+1} H_{k+1}^T e_{k+1}, & \left\| e_{k+1} \right\| \geq \left\| e_k \right\| \\ \beta_k . \end{cases} \quad \textbf{[IV.12]}$$

$$e_{k+1} = T_{k+1} - H_{k+1} \beta_k . \quad \textbf{[IV.13]}$$

4.3. Codage clairsemé (Sparse Coding)

Les équations [**IV.14]**, [**IV.15**] et [**IV.16**] illustrent les nouvelles formules utilisées pour déterminer la matrice de poids de sortie initiale dans le domaine sparse comme a été démontré mathématiquement dans [37], l représente le nombre de nœuds cachés et N le nombre introduit d'échantillons d'apprentissage. Dans notre cas, le problème traité est un problème de compression que nous rencontrons habituellement. Par conséquent, seule la première partie de [**IV.16]** est utilisée. Dans les règles d'apprentissage des méthodes des moindres carrés (sur lesquelles OS-ELM est basé), le calcul des poids d'apprentissage dépend de la condition présentée dans l'équation **[IV.15]** en fonction du nombre de nœuds cachés. En cas de compression ($l \leq N$), donc seule la première partie sera utilisée.

$$(V, U, D^2) = SVD(H_0). \quad \textbf{[IV.14]}$$

$$P_0 = pinv((D^2)^T D^2). \quad \textbf{[IV.15]}$$

$$\beta_0 = \begin{cases} Vpinv(D^2) H_0 T, & l \leq N . \\ H_0^T Upinv(D^2) U^T T, & l \geq N . \end{cases} \quad \textbf{[IV.16]}$$

Selon les règles d'apprentissage des autoencodeurs (Chap III, § 4.3.3.d, ¶ 4),une note importante que nous devons prendre en considération, c'est-à-dire lors de la mise à jour de l'autoencodeur dans la phase séquentielle, nous devons remplacer les poids d'entrée aléatoires par la transposition des poids de sortie du dernier mini-lot (mini-batch) de l'apprentissage

comme illustré dans **[IV.17]** [80], pour garantir la représentation la couche cachée dans la phase séquentielle. De plus, pendant l'entraînement de l'autoencodeur, nous devons définir la cible égale à l'entrée.

$$H_0 = G(\beta_0^T, b, X_{k+1}). \qquad \textbf{[IV.17]}$$

La convergence en ELM avec SVD à représentation sparse est déjà prouvée dans de nombreuses applications telles que [37], [108] et [117].

Le pseudo code de l'algorithme IV.2 récapitule les étapes d'apprentissage de l'algorithme proposé.

Algorithme IV.2. l'OS-ELM proposée.

Début

Les entrées : $\{X, T, n, G, l\}$

Les sorties : $\{\beta\}$

k=1 ;

Générer les paramètres de la couche cachée $(a,b) = P(X,l)$ *;*

Calculer la couche cachée en fonction de toute fonction d'activation $H_{k+1} = G(a,b,X_{k+1})$ *. ;*

Déterminer les poids de sortie initiaux $\{\beta_0 = Vpinv(D^2)H_0T;\ P_0 = pinv((D^2)^T D^2);\ (V,U,D^2) = SVD(H_0).\}$ *. ;* % *Sparse coding*

Pour *k= 2 : n :*

Calculer la couche cachée utilisant les nouveaux poids d'entrées $H_0 = G(\beta_0^T, b, X_{k+1})$ *.* **%** *Pour l'autoencodeur*

Mettez à jour les poids de sortie:

$\{e_{k+1} = T_{k+1} - H_{k+1}\beta_k;\quad P_{k+1} = \frac{1}{\lambda_{k+1}}(P_k - K_{k+1}H_{k+1}^T P_k).;\quad K_{k+1} = (P_k H_{k+1})(\lambda_{k+1} + H_{k+1}^T P_k H_{k+1}^T)^{-1};\ \beta_{k+1} = \beta_k - P_{k+1}H_{k+1}^T e_{k+1}.\}$

$\lambda_{k+1} = \lambda_{min} + (1 - \lambda_{min})e^{-\mu\|e_{k+1}\|}.;$ % *mécanisme d'oubli*

fin

fin

5. Résultats obtenus et discussion

Pendant ce temps, dans notre problème basé sur les données (data-driven), une préparation adéquate des données est un processus bien nécessaire avant toute application. Dans cette

expérience, les chemins de préparation des ensembles de l'apprentissage et de test sont présentés dans les étapes suivantes:

- Seul un ensemble de capteurs a été sélectionné pour l'ajustement du modèle de l'apprentissage. Les paradigmes de sélection dépendent de la variation des mesures des capteurs selon les cycles de temps. Si une mesure ne représente aucune variation, elle sera éliminée [23]. Parmi les 26 caractéristiques présentées de l'ensemble de données, Celles choisies dans notre expérience sont des capteurs dont les indices sont: {7, 8, 9, 12, 14, 16, 17, 20, 25, 26}. Ce choix dépend de la corrélation entre les paramètres de la base de données. la non corrélation conduira le modèle de l'apprentissage à des risques empiriques et à d'autres problèmes mal posés. La matrice de corrélation de Ali, A. [118] utilisant la base de données C-MAPSS et explique bien cette situation de sélection d'entités. En effet,

- Chaque colonne (type de mesures des capteurs) de l'ensemble de données est mise à l'échelle séparément dans l'intervalle [0,1] en utilisant la normalisation min-max.

- Les données de l'apprentissage et de test sont préparées dans des lots de tailles différentes selon chaque cycle de vie du moteur.

- La moyenne, l'écart type et nombreuses statistiques les valeurs de chaque observation peuvent être ajoutées à chaque observation pour réduire les informations manquantes lors de la compression. Les paramètres additifs peuvent être supprimés de la reconstruction ultérieurement pour récupérer la version originale d'entrées (décompression).

L'étude comparative entre l'approche proposée **SA-OS-AE**-et l'OS-ELM à base d'autoencoder ordinaire est réalisée dans les mêmes conditions. Après plusieurs apprentissages, les résultats nous amènent à affiner les hyper-paramètres des algorithmes comme le montre le **tableau IV.3**.

Tableau: IV.3. Tableau des hyper-paramètres.

λ_{min}	μ	Taille des Lots	Nombre de neurones	Fonction d'activation	Taux de compression
0.78	0.2	variée	5	ReLU	50%

Nous pouvons remarquer que la taille des séquences de données est variée, car le choix des échantillons dans notre cas dépend de la taille du cycle de vie du moteur, pour garantir un apprentissage itératif avec une attitude de détérioration à chaque étape.

Lors de l'entraînement du réseau neuronal basé sur l'algorithme proposé, un ensemble de séquences a été ignoré grâce à la stratégie de mise à jour comme illustré dans la **figure IV.3**. Cette stratégie peut impliquer dans la réduction des coûts de calcul en gardant l'adaptation du modèle d'apprentissage vers les données d'entraînement.

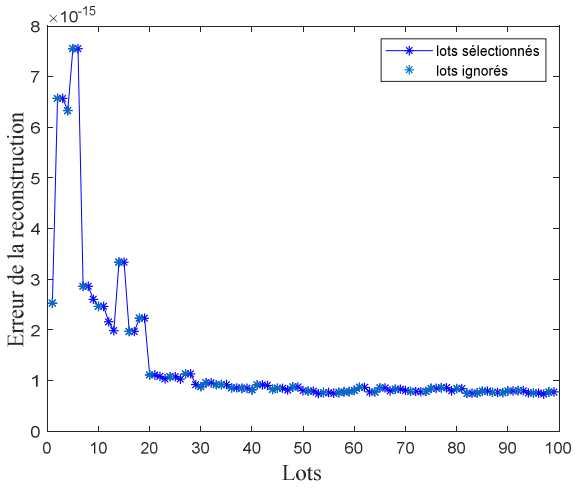

Figure: IV.3. Comportement de la fonction de mise à jour aux séquences de données.

Concernant le mécanisme d'oubli, plus le facteur de sensibilité est grand, plus la vitesse de divergence de la fonction d'oubli est faible comme l'indique la **figure IV.4**. Si le modèle de l'apprentissage est décidé à être utilisé pour une formation à long terme, alors il doit être plus petit.

De nombreuses mesures ont été utilisées pour évaluer l'algorithme proposé sur l'ensemble de données C-MAPSS. Nous pouvons citer l'erreur quadratique moyenne (RMSE), le rapport signal/bruit (PSNR) et l'erreur absolue normalisée (NAE). Ces métriques sont appliquées uniquement sur les entrées reconstruites des encodeurs automatiques et les entrées originales des ensembles d'apprentissage et de test.

Le taux de compression souhaité peut être choisi en fonction de la décision de l'utilisateur. Mais, puisque l'autoencodeur basé sur la compression est un type de compression de données avec perte, plus le taux de compression est faible, plus les informations peuvent être perdues. Ensuite, l'efficacité de la compression, dépendra du type d'application [119].

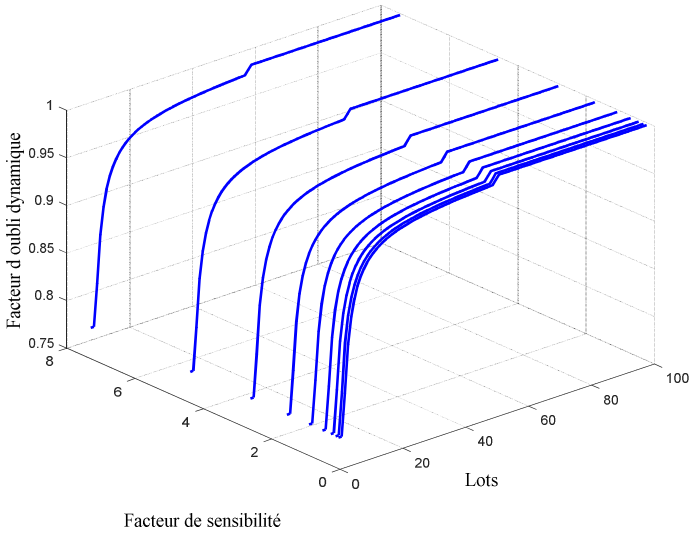

Figure: IV.4. Variation du facteur d'oubli en fonction de l'erreur de variable de sensibilité.

Les résultats du **tableau IV.4** prouvent que l'algorithme proposé (**AS-OS-AE** : Adaptive Sparse Online Sequantial AutoEncoder) surpasse l'autoencodeur ordinaire (OS-AE : online sequantial autoencoder) dans de nombreuses évaluations de métriques avec moins d'échantillons d'apprentissage.

Tableau: IV.4. Résultats de reconstruction.

Sous-ensemble	Méthodes	Lots	Apprentissage				Lots	Test			
			Temps	**RMSE**	**PSNR**	**NAE**		**Temps**	**RMSE**	**PSNR**	**NAE**
FD001	**OS-AE**	100	1.7940	0.1308	65.9343	0.2077	100	0.4992	0.1312	65.917	0.2076
	AS-OS-AE	**48**	**0.4212**	0.1315	**65.8719**	0.2094	100	**0.4212**	0.1313	**65.8960**	0.2085
FD002	**OS-AE**	260	5.1792	0.0082	89.8619	0.0137	260	1.6848	0.0077	90.4210	0.0129
	AS-OS-AE	**130**	7.6596	**0.0073**	90.9815	**0.0126**	260	**1.1076**	**0.0063**	92.2516	**0.0109**
FD003	**OS-AE**	100	1.7940	0.0886	69.2188	0.1435	100	0.4524	0.1314	65.8905	0.2079
	AS-OS-AE	**50**	3.0732	**0.0861**	69.4706	**0.1377**	100	**0.4212**	**0.1296**	66.0180	**0.2010**
FD004	**OS-AE**	248	5.5692	0.0085	89.5757	0.0132	248	1.2636	0.0081	89.9852	0.0123
	AS-OS-AE	**123**	8.4397	**0.0080**	90.1589	0.0139	248	1.0452	**0.0078**	90.3271	0.0133

Pour des raisons illustratives de l'effet de la représentation 'Sparse' basée sur SVD sur les valeurs de la couche cachée, nous considérons la couche cachée comme une image. En effet, la couche cachée est la version compressée des données d'entrée, elle est très petite pour l'afficher en utilisant des dimensions standard. Par conséquent, nous avons redimensionné la couche cachée et nous l'avons représentée à l'aide d'une mise à l'échelle, comme illustré dans la **figure IV.5**. Les régions plus colorées avec le rouge de la couche cachée représentent les valeurs rétrécies vers zéro. Nous pouvons conclure du **Tableau IV.3** et de la **Figure IV.5** qu'une représentation sparse peut améliorer la précision de reconstruction avec moins de nombre d'éléments expressifs.

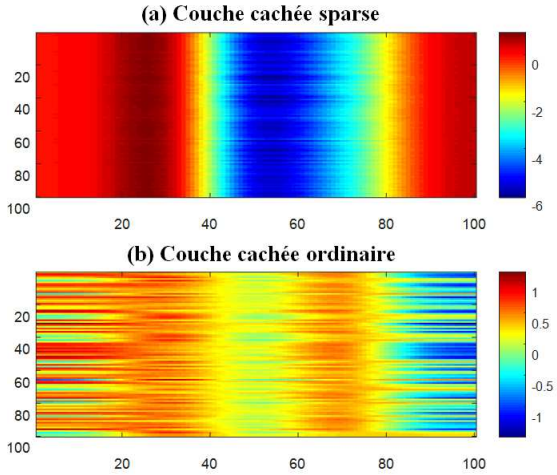

Figure: IV.5. Différence entre la couche cachée et la couche sparse

Un exemple de reconstruction de mesures de capteurs utilisant **SA-OS-AE** et **OS-AE** est présenté dans la **figure IV.6**.

La **Figure IV.6** explique clairement la différence de reconstruction entre l'autoencodeur proposé et ordinaire. Cela conduit à conclure que l'adaptation dynamique et le codage 'Sparse' peuvent impliquer une représentation plus significative que la representation basée sur l'autoencodeur ordinaire.

Les résultats indiquent que la meilleure représentation des paramètres dans la couche cachée sous les données dynamiques peut par conséquent conduire la fonction d'approximation à la meilleure généralisation vers les nouvelles données souhaitées.

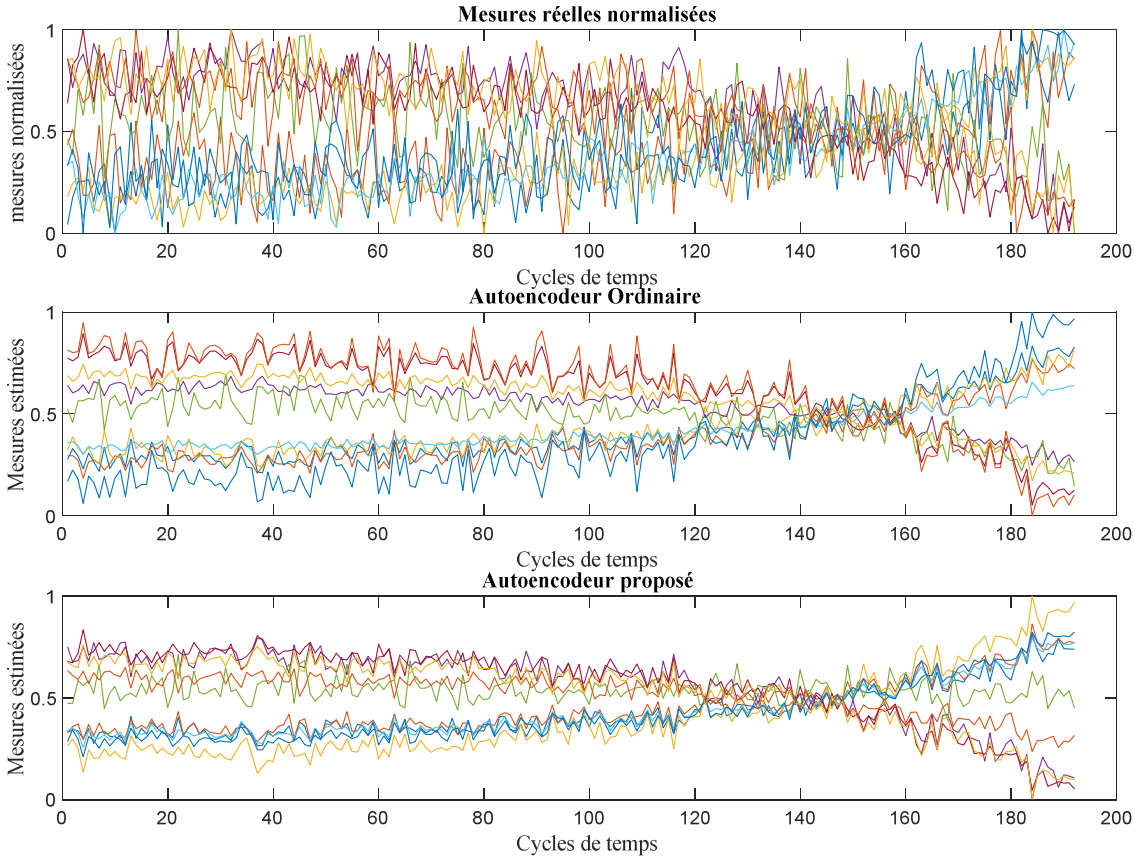

Figure: IV.6. Résultats de la reconstruction des mesures avec taux de compression égal à 50%.

6. Conclusion

Un nouvel outil de compression adaptive est proposé dans ce travail. L'approche proposée permet au réseau neuronal d'adapter dynamiquement les poids d'apprentissage aux données nouvellement arrivées.

Cette adaptation dépend de la suppression dynamique des anciennes données d'apprentissage à l'aide de la fonction d'oubli dynamique. Une stratégie de sélection mise à jour est intégrée pour réduire les coûts de calcul et également pour éliminer les échantillons d'apprentissage indésirables afin que la fonction de perte (loss function) converge toujours vers la valeur souhaitée.

La représentation SVD basée sur le paradigme de sparse coding proposé et la reconstruction de la norme L_2 peuvent améliorer la précision de reconstruction du réseau neuronal de type ELM.

L'autoencodeur proposé est comparé à sa version ordinaire dans les mêmes conditions et options d'apprentissage. L'évaluation est réalisée à l'aide de données dynamiques riches extraites de C-MAPSS et a prouvé ses capacités.

Les résultats de ce travail pourraient être une étape de reconstruction d'une approche d'apprentissage supervisé basée sur les données pour la prédiction de l'état de santé des moteurs d'avion dans les travaux futurs.

Cet outil est très important surtout de nos jours, où les technologies de capteurs se répandent largement, et comme des données dynamiques sont disponibles, la compression est une étape nécessaire pour la préparation des données. Dans ce contexte, la réduction de la dimensionnalité jouera un rôle clé dans l'accélération du processus de l'apprentissage ainsi que l'extraction de modèles de signification pour garantir une source d'information fiable avec moins de paramètres.

Maintenant, lorsque nous apprenons à utiliser l'ELM pour un apprentissage non supervisé dans l'évaluation de la santé, l'objectif des prochains chapitres sera basé sur l'utilisation de l'ELM pour un apprentissage supervisé et plus particulièrement les réseaux de neurones adaptatifs et adaptatifs de débruitage de type DBN (Deep Belief Neural network) à base de l'ELM pour l'estimation du RUL.

Les réseaux de neurones adaptatifs de type DBN à base d'ELM pour l'estimation du RUL

Résumé:

Ce chapitre présente une nouvelle approche d'évaluation de la santé des systèmes pour l'apprentissage supervisé. L'approche proposée est un DBN (Deep Belief Neural network) basé sur les règles de l'apprentissage d'ELM. Une nouvelle théorie de l'apprentissage adaptatif a été intégrée pour rendre l'algorithme capable d'aborder la programmation dynamique et d'interagir avec les variables environnementales. L'évaluation des algorithmes a été effectuée à l'aide des données d'apprentissage obtenues auprès de C-MAPSS. Les résultats obtenus encouragent fortement l'utilisation d'ELM dans le domaine du pronostic.

1. Introduction

L'investigation efficace des données pour la prévision rapide et précise de la durée de vie restante des systèmes peut être considérée comme une tâche très importante pour les opérations de maintenance.

Dans ce contexte, *la question clé est de savoir comment mener une analyse appropriée pour l'extraction d'informations importantes à partir de séquences de données dans un espace de grande dimension afin de garantir une conclusion fiable sure l'état de système.*

Dans ce chapitre, un nouveau schéma d'apprentissage basé sur un algorithme de l'OS-ELM est proposé pour la prévision du RUL en considérant les contributions suivantes:

- Proposition d'une nouvelle technique d'extraction des paramètres basée sur des autoencodeurs empilés (Stacked autoencouders) modifiés et non-ordinaires, pour améliorer les représentations des données par une reconstruction précise.
- Développement d'une nouvelle fonction d'oubli dynamique basée sur la différence temporelle de l'apprentissage récursif, pour améliorer la capacité de suivi dynamique des nouvelles données Pour tenter d'aborder la programmation dynamique basée sur les variables environnementales.
- Proposition d'une nouvelle stratégie de sélection par mise à jour basé sure la différence temporelle, afin d'éliminer les séquences de données indésirables et d'assurer la convergence des paramètres du modèle d'apprentissage vers leurs valeurs appropriées.

L'approche proposée baptisée cette fois ci TD-OSELM (Temporal Difference Online Sequential Extreme learning machine) est validée sur l'ensemble de données C-MAPSS où les résultats expérimentaux confirment qu'elle fournit une précision et une efficacité très satisfaisante du modèle de prédiction par rapport à d'autres méthodes existantes dans la littérature.

2. L'algorithme proposé de la nouvelle TD-OS-ELM

Dans l'algorithme de TD-OS-ELM[16] proposé, nous avons ajouté un paramètre de régularisation C pour réduire les risques empiriques et structurels [92], [120][17], [121] pendant la phase d'initialisation. Une fois le modèle d'apprentissage initialisé, la fonction objectif de l'erreur temporelle δ dans l'équation [**V.5**] basée sur les conditions de mise à jour dans [**V.4**] est proposée pour adopter les poids aux nouvelles variations de données [42]. La différence temporelle δ permet de déterminer la quantité de changement entre la couche cachée actuelle et la dernière en considérant le facteur d'actualisation γ. Il s'agit d'une mémoire temporelle qui *conserve des informations* sur les dernières caractéristiques. Des implémentations similaires de la mémoire temporelle ont été utilisées dans d'autres modèles de «Deep Learning» tels que *long-short terme memory* (LSTM) (Chapitre II, § 4.3.3.b)

La formule d'oubli dynamique λ est proposée dans l'équation [**V.8**] et intégrée dans la phase récursive dans la formule de calcul de la matrice des covariances [**V.6**] et la matrice de gain [**V.7**].

Par conséquent, l'algorithme d'OS-ELM de base doit être changé pour s'adapter a nouvelles formules présentées dans les équations [**V.1**] à [**V.8**].

$$H_k = G(aX_k + b) \qquad \textbf{[V.1]}$$

[16] La version MATLAB de cet algorithme est valable gratuitement dans les liens [127],[128]. Les liens vers l'application **Octave** à base de cloud computing sont valable aussi sur [129].

[17] Un algorithme développé basé sur python est distribué sur **Google Colab** permet de voir l'implémentation de la régularisation dans ELM et son effet sur l'ajustement de courbe de la fonction de dégradation à l'aide du sous-ensemble FD001. Les utilisateurs peuvent modifier la valeur du paramètre de régularisation et déduire les résultats de la prédiction.

$$P_0 = \begin{cases} (\frac{1}{C}+H_0^T H)^{-1}, N \le l \\ (\frac{1}{C}+H_0^T H)^{-1} H^T, N > l \end{cases} \quad \textbf{[V.2]}$$

$$\beta_0 = P_0 H_0 T_0 \quad \textbf{[V.3]}$$

$$\beta_{k+1} = \begin{cases} \beta_k + M_{k+1}\delta_{k+1}, \delta_{k+1} > \delta_k \\ \beta_k, \delta_{k+1} \le \delta_k \end{cases} \quad \textbf{[V.4]}$$

$$\delta_{k+1} = T_{k+1} - (H_{k+1} - \gamma H_{k+2})^T \beta_k \quad \textbf{[V.5]}$$

$$P_{k+1} = \frac{1}{\lambda}(P_k - k_{k+1}(H_{k+1} - \gamma H_{k+2})^T P_k) \quad \textbf{[V.6]}$$

$$K_{k+1} = \frac{P_k H_k}{\lambda + 1 + (H_{k+1} - \gamma H_{k+2})^T P_k H_k} \quad \textbf{[V.7]}$$

$$\lambda_{k+1} = \lambda_{\min} + (1 - \lambda_{\min}) e^{\mu \| \delta_{k+1} \|} \quad \textbf{[V.8]}$$

Le facteur d'oubli dynamique change selon la valeur de facteur de sensibilité $\boldsymbol{\mu}$ dans la formule proposée en [**V.8**] où $\{\lambda_{\min}, \mu\} \in (0,1]$ et $\lambda_{\min} \le \lambda \le 1$.

Dans ce cas, et contrairement à l'OS-ELM de base qui dépend de la formule d'erreur ordinaire (**[V.9]**) pour mettre à jour ses poids, la nouvelle différence temporelle $\boldsymbol{\delta}$ permet de considérer "Le montant de la corrélation" entre les séquences arrivées et les anciennes.

$$e_{k+1} = T_{k+1} - (H_{k+1})^T \beta_k \quad \textbf{[V.9]}$$

Théoriquement, le $\boldsymbol{\delta}$ sera responsable de mesurer à la fois la précision de la prédiction pour l'apprentissage supervisé et la précision de la reconstruction pour l'apprentissage non supervisé.

De plus, contrairement à tout mécanisme d'oubli, le renforcement de l'oubli dynamique par la mémoire temporelle permettra un meilleur apprentissage adaptatif.

Les instructions de l'algorithme V.1 donnent un aspect plus évident à l'algorithme proposé.

Algorithme V.1. TD-OS-ELM proposé

Début

Les entrées : $\{X, T, n, G, l\}$

Lés sortie : $\{\beta\}$

k=1 ;

Générer les paramètres de la couche cachée $(a,b) = P(X,l)$ *;*

Calculer la couche cachée en fonction de toute fonction d'activation $H_{k+1} = G(a,b,X_{k+1})$ *. ;*

Déterminer les poids de sortie initiaux $\{\beta_0 = Vpinv(D^2)H_0T;\ P_0 = pinv((D^2)^T D^2);\ (V,U,D^2) = SVD(H_0).\}$ *. ;*

Pour *k= 2:n:*

Mettez à jour les poids de sortie:

$\{\lambda_{k+1} = \lambda_{\min} + (1-\lambda_{\min})e^{\mu\|\delta_{k+1}\|},$ *% mécanisme d'oubli*

$\delta_{k+1} = T_{k+1} - (H_{k+1} - \gamma H_{k+2})^T \beta_k;\quad P_{k+1} = \frac{1}{\lambda}(P_k - k_{k+1}(H_{k+1} - \gamma H_{k+2})^T P_k);$

$K_{k+1} = P_k H_k(\lambda + 1 + (H_{k+1} - \gamma H_{k+2})^T P_k H_k)^{-1}; \beta_{k+1} = \begin{cases} \beta_k + M_{k+1}\delta_{k+1}, \delta_{k+1} > \delta_k \\ \beta_k,\ \delta_{k+1} \le \delta_k \end{cases}\}$ *% Stratégie de mise à jour*

fin

fin

3. Le «Deep architecture» de TD-OS-ELM proposé

Les autoencodeurs empilés (Stacked autoencoders) sont des autoencodeurs connectés en série. Les représentations encodées d'un autoencodeur sont considérées comme les entrées du suivant (**Figure V.1**).

L'apprentissage non supervisé et supervisé dans l'approche proposée a été effectué avec des tailles fixes de lots pour vérifier la fonction objective de l'erreur temporelle dans l'équation [**V.5**] et pour éviter une erreur d'inadéquation de dimensions pour des raisons de programmation. Les résultats de l'extraction des autoencodeurs sont directement utilisés pour le réglage fin (fine tuning) du réseau de prédiction du RUL, la mise à jour des modèles d'apprentissage supervisés et non supervisés est réalisée simultanément.

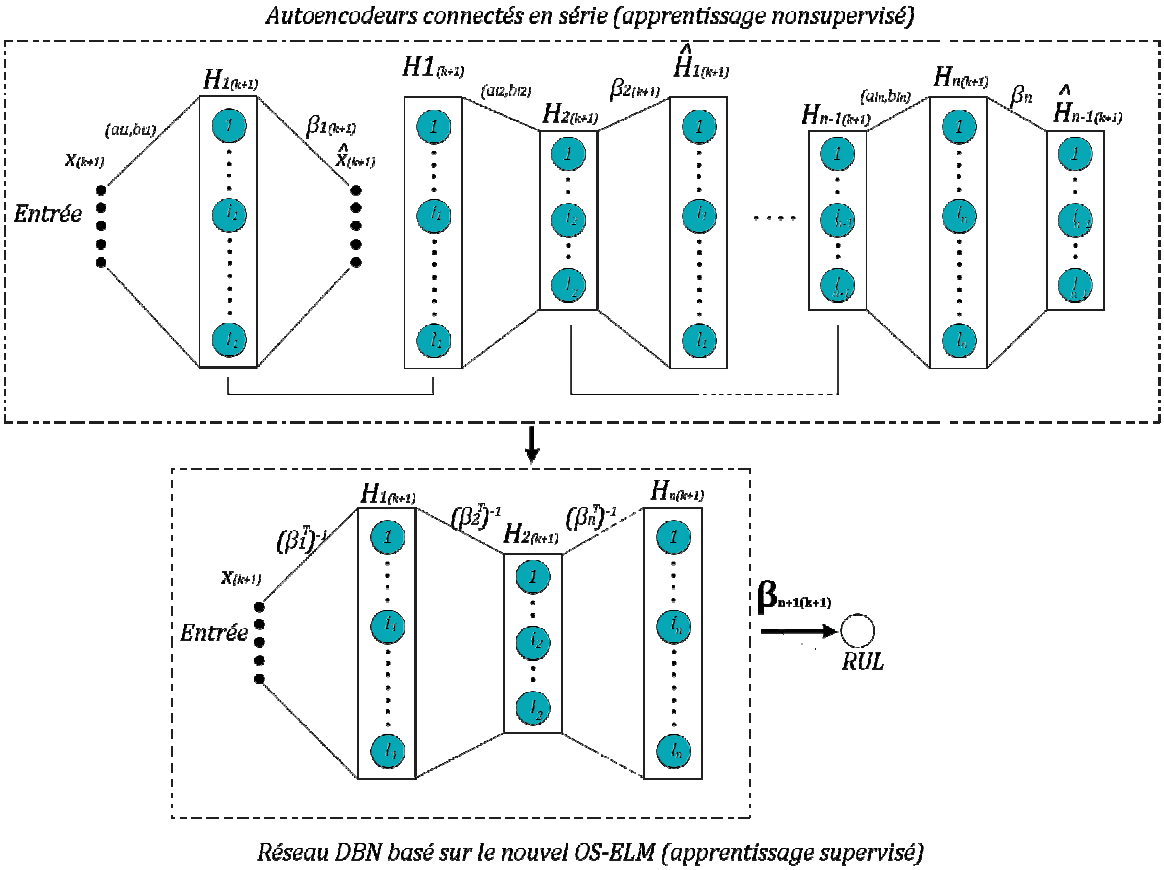

Figure: V.1. Architecture proposée de DBN.

Pendant l'apprentissage des autoencodeurs, nous devons changer les étiquètes dans les équations proposées ci-dessus pour qu'elles soient identiques à l'entrée [115]. Par conséquent, les équations [**V.3**] et [**V.5**], doivent être modifiées pour s'adapter au processus de l'apprentissage des autoencodeurs.

Notre principale contribution à la modification de l'autoencodeur dans ce travail est que les théories de l'ELM de base pour l'apprentissage de l'autoencodeur montrent que: «*Après la processus d' apprentissage de l'autoencodeur, nous pouvons encoder des entités en utilisant la matrice de transposition des poids de sortie* [115], *équation [V.10]*». Cependant, mathématiquement, il est prouvé que le meilleur codage peut être obtenu en utilisant la matrice inverse de la

transposition des poids de sortie comme indiqué dans l'équation [**V.11**]. La formule proposée est testée expérimentalement et a prouvé sa précision [5].

$$X = H\beta \Rightarrow H = X\beta^T, \quad \textbf{[V.10]}$$

$$H = (\beta^T)^{-1} X, \quad \textbf{[V.11]}$$

Une note très importante que nous devons prendre en compte lors de l'apprentissage des autoencodeurs est qu'après la phase d'initialisation (k=k+1), nous n'avons plus besoin des poids d'entrée générés aléatoirement et nous utiliserons la formule de l'équation [**V.12**] au lieu de celle de l'équation [**V.1**].

$$H_{k+1} = G((\beta_k^T)^{-1}, BI, X_{k+1}) \quad \textbf{[V.12]}$$

Où X_{k+1} est le $(k+1)^{eme}$ lot d'entrées de l'ensemble d'apprentissage.

4. Application, résultats et discussion

Maintenant, et comme le problème que nous voulons résoudre est le problème d'apprentissage supervisé pour prédire la RUL, les entrées du modèle d'apprentissage ont été sélectionnés et préparés de la même manière qu'au chapitre IV. Pendant ce temps, les sorties ont été définies selon une fonction linéaire pour satisfaire le comportement de détérioration de la santé du moteur.

Les paramètres d'apprentissage de l'algorithme proposé sont ajustés après avoir répété l'expérience pour tenter d'obtenir la meilleure valeur de score (exactitude) dans l'équation [**V.13**] et le minimum RMSE en utilisant l'équation [**V.14**] pour l'ensemble de données de test.

$$S = \begin{cases} \sum_{i=1}^{Z} e^{-d_i/13} - 1, d_i < 0 \\ \sum_{i=1}^{Z} e^{-d_i/10} - 1, d_i \geq 0 \end{cases} \quad \textbf{[V.13]}$$

$$RMSE = \sqrt{\frac{1}{Z}\sum_{i=1}^{Z} d^2} \quad \textbf{[V.14]}$$

Les résultats expérimentaux ont conduit aux paramètres indiqués dans le **tableau V.1**.

Tableau: V.1.Paramètres d'apprentissage de l'algorithme proposé.

C	λ_{min}	λ_{max}	γ	μ	l	n	G
100	0.98	1	10e-6	0.98	100	205	ReLU

Selon le PHM 2008 data challenge [22], la fonction de score pénalise la prédiction latente $d_i \geq 0$ plus que les estimations précoces $d_i < 0$ pour des raisons de maintenance. Une prédiction trop tardive peut retarder les opérations de maintenance, et une prédiction trop précoce pourrait ne pas être dangereuse, mais consomme plus de ressources de maintenance. La **figure V.2** explique que la propagation de la fonction RMSE et Score vers des valeurs plus élevées est inférieure à celle des autres vers zéro. Cela confirme la crédibilité de nos résultats. Plus le score du modèle d'apprentissage est proche de zéro, plus le modèle d'apprentissage est précis et les résultats RUL sont plus proches des valeurs réelles.

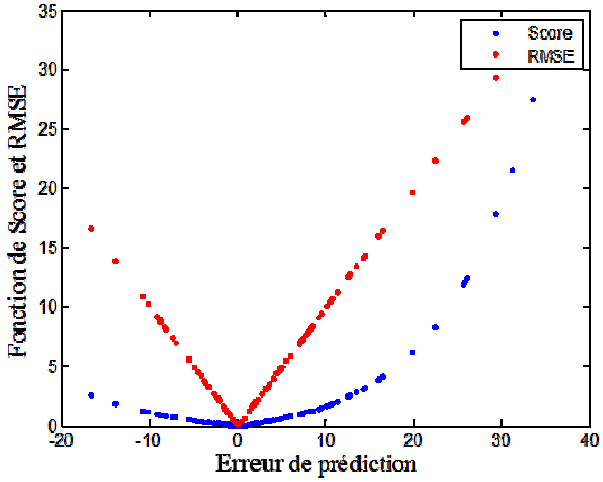

Figure: V.2. Fonctions d'évaluation des résultats de l'ensemble de test.

Un ensemble de lots de donnés a été ignoré lors de l'apprentissage des modèles de prédiction du RUL avec la contribution de la stratégie de mise à jour. Cela peut contribuer à réduire le temps et à éviter la divergence des poids.

La **figure V.3** illustre la réaction de la stratégie de mise à jour envers les données inutiles en montrant les indices des séquences ignorées et sélectionnées. Il est évident que les conditions présentées précédemment dans l'équation [**V.9**] sont confirmées sur la **figure V.3**. L'erreur RMSE de chaque bloc de données sélectionné est plus grande que celle qui le précède.

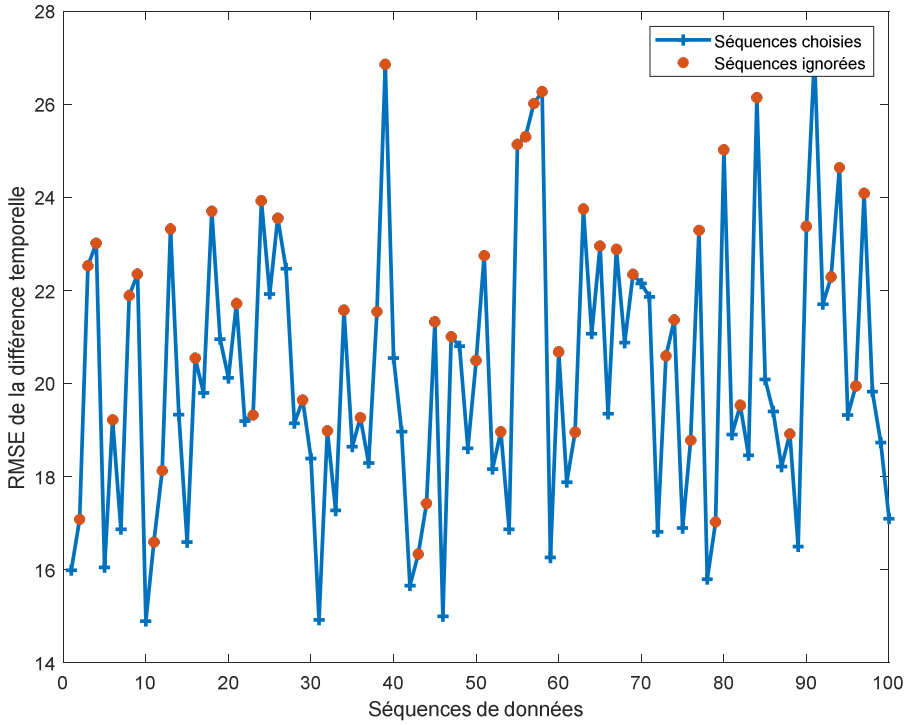

Figure: V.3.Comportement de la stratégie de sélection mise à jour vers les séquences inutilisées.

Pendant l'apprentissage, le facteur d'oubli dynamique par l'intervention de la fonction de minimisation d'erreur temporelle basée sur l'apprentissage récursif peut contribuer à l'adaptation de l'algorithme de prédiction vers aux données nouvellement arrivées.

En adaptant progressivement les poids pour améliorer la fonction d'approximation, la précision de la prédiction restera stable et convergera toujours vers la valeur souhaitée. La **figure V.4** montre la variation du paramètre d'oubli pendant l'apprentissage. Si l'erreur temporelle change trop, cela signifie que les nouvelles données à venir ont également des caractéristiques différentes des premières; ce qui oblige par conséquent les poids d'apprentissage $\boldsymbol{\beta}$ à s'adapter à ces nouveaux changements. L'adaptation de $\boldsymbol{\beta}$ est le résultat de la mise à jour du facteur d'oubli λ, et la quantité de changement est contrôlée par les hyper-paramètres utilisateur exprimés dans le **tableau V.1**.

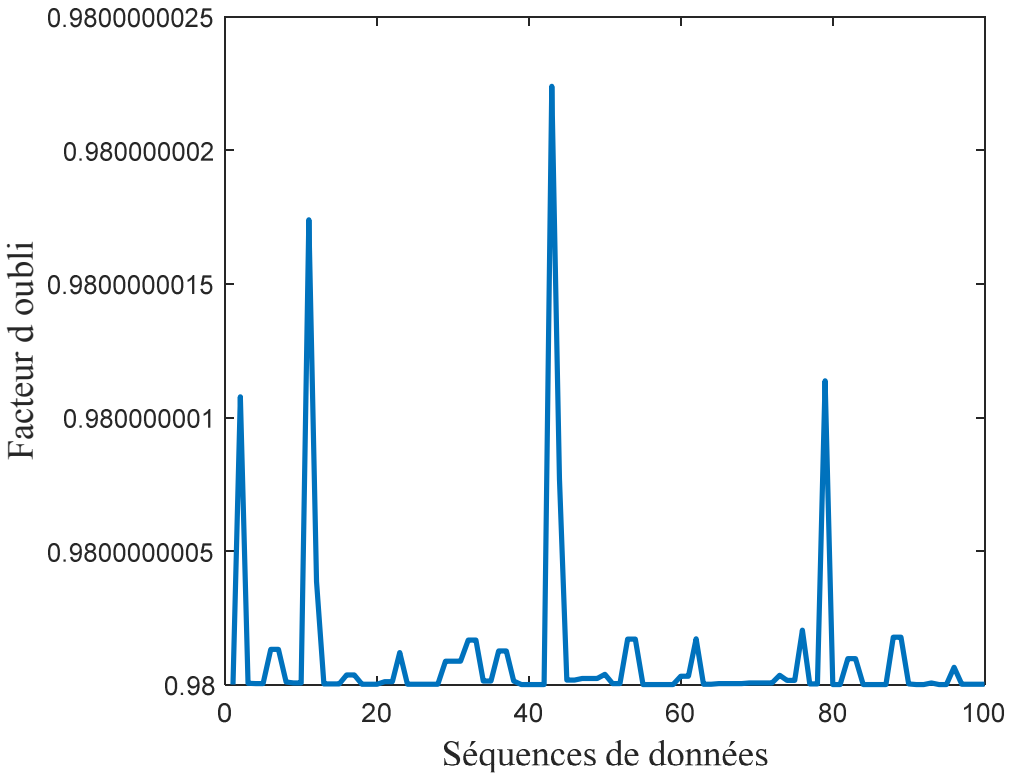

Figure: V.4. Comportement de la fonction d'oubli dynamique pendant l'apprentissage.

Le modèle d'apprentissage a été testé à l'aide de nouvelles données (inconnues du modèle) à partir de l'ensemble de test de FD001. L'exemple d'ajustement de courbe de la fonction cible RUL sur la **figure V.5** prouve que l'approche proposée a une performance acceptable.

La réponse du réseau conçu sur la **figure V.7** est tracée en utilisant un ensemble de cycles de vie différents de différents moteurs de l'ensemble de test.

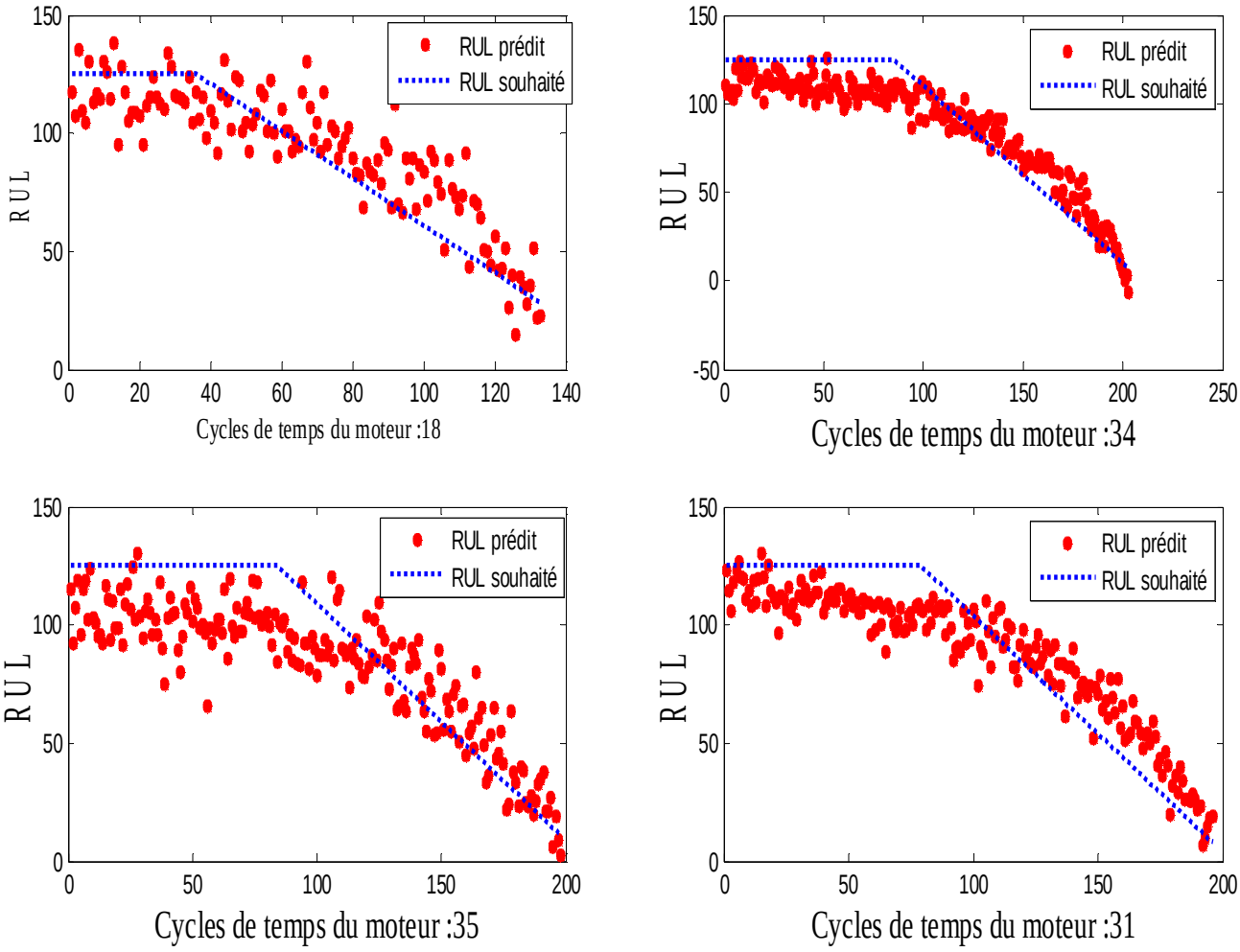

Figure: V.5. Fonction de prédiction RUL par rapport à la fonction souhaitée.

Les performances des approches proposées sont comparées à un ensemble d'autres approches récentes dans la littérature. Les résultats du **tableau V.2** indiquent que l'algorithme proposé a la capacité d'atteindre une valeur de score faible en fonction de moins d'échantillons d'apprentissage et de moins de temps.

Tableau: V.2. Résultats de comparaison entre les algorithmes de prédiction.

Méthodes	RMSE	Score	Echantillons d'apprentissage	Temps d'apprentissage [s]
ARIMA SVM [44]	39.6843	-	20631	-
DCNN [65]	18.4480	1286.7	20631	-
LSTM [56]	16.17	338	20631	714.53
DCNN [122]	12.61	273.7	**17731**	-
WELM [56]	13.78	267.31	20631	5.04
HDNN [59]	13.017	245	20631	-
OS-ELM	15.08	221.72	20631	-
ResCNN [60]	**12.16**	212.48	20631	-
Approche proposée (Deep TD-OS-ELM)	13.74	**199.1712**	**10250**	**5.85**

5. Conclusion

Un nouvel algorithme d'apprentissage séquentiel en ligne TD-OS-ELM est présenté dans ce travail pour la prédiction du RUL des moteurs d'avion. Le modèle proposé a expliqué qu'en acceptant toutes séquences de données variant dans le temps dans chaque mise à jour sans tenir compte de la divergence des poids à un certain niveau, le modèle ne se comportera plus positivement pour atteindre l'approximation et la généralisation nécessaires pour les données nouvellement arrivées.

Jusqu'à présent, la stratégie de mise à jour proposée implique une réduction d'overfitting et une sélection précise des paramètres. L'estimation de l'erreur temporelle basée sur la fonction d'oubli dynamique joue un rôle clé dans l'adaptation des poids entre les séquences d'apprentissage actuels et suivantes. La représentation précise des paramètres basée sur un apprentissage non supervisé robuste peut résoudre les problèmes de données de dimension supérieure.

Les résultats expérimentaux prouvent que le nouveau schéma d'apprentissage pour les données dynamiques de dimension supérieure est rapide et précise, ce qui en fait un candidat compétitif pour les problèmes de prédiction de données variant dans le temps.

Dans la littérature, l'ensemble de données C-MAPSS a été étudié avec différents modèles d'apprentissage automatique. L'une des caractéristiques importantes qui n'ont pas encore été discutées est que les mesures des capteurs sont contaminées par du bruit dont la source et le comportement sont inconnus. Par conséquent, le chapitre suivant est entièrement consacré à l'amélioration de l'approche proposée.

Les réseaux de neurones adaptatifs de débruitage de type DBN a base de l'ELM pour l'estimation du RUL

Résumé:

Dans ce chapitre, une nouvelle approche de prédiction du RUL est proposée. Tout d'abord, une extraction robuste utilisant un autoencodeur de débruitage modifié est introduite pour apprendre des représentations les plus importantes. Une stratégie de sélection est alors proposée pour garantir que seules les séquences de données de l'apprentissage utiles passent par le processus d'apprentissage. Enfin, l'algorithme OS-ELM est utilisé pour adapter la fonction de dégradation du moteur et aborder la programmation dynamique basée sur un nouveau mécanisme d'oubli. L'approche proposée est testée sur l'ensemble de données C-MAPSS et comparée à l'OS-ELM avec un autoencodeur ordinaire et un OS-ELM de base et des travaux dans la littérature et a prouvé ça précision et de ça généralisation. En plus de cela, la souplesse de la fonction d'ajustement de la courbe et la capacité de filtrage des échantillons, recommandent fortement son utilisation pour la prévision du RUL.

1. Introduction

De nos jours, et en raison de l'évolution remarquable du volume, de la variété et de la vélocité des données à cause de l'évolution de la technologie des capteurs, l'estimation du RUL des composants ou sous-systèmes sur la base des données historiques disponibles devient fortement recommandée et mérite de motiver les chercheurs vers de nouvelles approches de prédiction [63], [123], [68].

L'estimation du RUL pour l'évaluation de la santé est une tâche très cruciale dans les opérations de maintenance conditionnelle [44]. Ces opérations sont liées à l'état de santé réel de l'équipement ou des sous-systèmes dans des conditions de fonctionnement, où la prévision précoce et précise des défaillances implique une prise de décision de maintenance, pour des catastrophes probables [65].

La complexité de l'estimation du RUL en termes d'approches basées sur les données réside dans les multiples séquences de données accumulées de dimension supérieure qui varient dans le temps. Ce flux de données accumulées peut facilement empêcher la fonction de perte du modèle d'apprentissage de converger vers l'erreur souhaitée.

La meilleure solution pour ce problème ou les problèmes de données variant dans le temps, en général, est d'intégrer une stratégie de suivi et de sélection dynamique (**Chapitre V**) dans les modèles d'apprentissage pour résister à toute variation dans les nouvelles séquences à venir [36], [42], [91] et [124]. Cependant, une telle solution est réalisable sous la contrainte que les données de l'apprentissage ne soient pas **contaminées par du bruit** de source ou de comportement **inconnu**[116], [125].

Par conséquent, dans ce chapitre, une nouvelle approche basée sur l'OS-ELM et une technique de filtrage avec double fonctions d'oubli dynamique et une stratégie de sélection de mise à jour

est introduite. L'algorithme proposé vise à obtenir une estimation précise du RUL en proposant les solutions suivantes:

- Développement d'un autoencodeur de débruitage (DAE) à base de l'OS-ELM pour obtenir des meilleures représentations des paramètres et pour la réduction du bruit sous l'utilisation des données dynamiques.
- Intégration des règles d'apprentissage de l'OS-ELM lors de la conception du modèle d'estimation du RUL pour assurer un apprentissage adaptif en ligne.
- Intégration d'une capacité de suivi dynamique dans l'autoencodeur de débruitage et dans l'étape de fine-tuning à l'aide d'un mécanisme d'oubli est proposée.
- Utilisation d'une sstratégie de mise à jour pour sélectionner uniquement les nouveaux échantillons expressifs pour le processus de l'apprentissage.

L'approche proposée DOS-ELM (Denoising Online Sequential Extreme Learning Machine) est validée sur la base de données publiques C-MAPPS des turboréacteurs à double flux [111]. Les résultats sont comparés à l'OS-ELM avec un autoencodeur ordinaire et l'OS-ELM de base, et des travaux dans la littérature où des performances plus élevées sont enregistrées.

Par ailleurs et toujours dans le cadre de ce chapitre, nous présentons l'algorithme développé et les résultats expérimentaux. Les performances du processus d'apprentissage sont également présentées et comparés à d'autres méthodes basées sur des données. Enfin ce chapitre se termine par une discussion générale des résultats obtenus.

2. OS-ELM proposé et son schéma de débruitage

Le schéma de l'apprentissage proposé pour la prédiction du RUL est illustré dans la **figure VI.1**. Deux types de réseaux de neurones sont utilisés pour l'apprentissage d'une seule couche cachée trop complète (an over-complete single hidden layer). L'autoencodeur de débruitage est utilisé pour apprendre une représentation significative des entrées en réduisant le bruit qui les

accompagne [23]. La fonction de reconstruction tente de reconstruire les entrées d'origine à partir de celles corrompues.

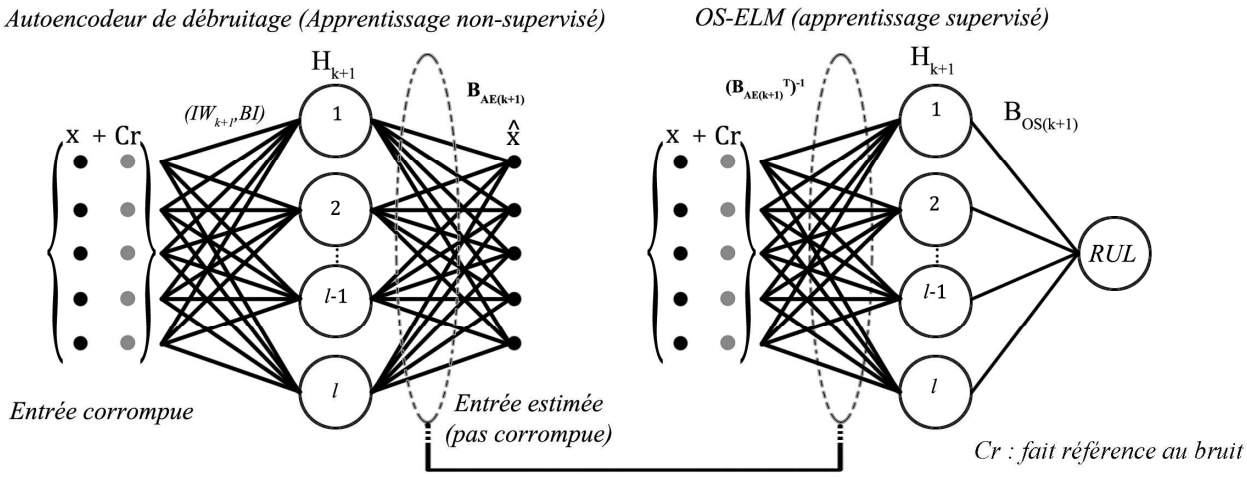

Figure: VI.1. Architecture de l'algorithme proposé pour l'estimation du RUL.

Le mécanisme d'oubli est proposé et intégré dans les phases de mise à jour pour l'apprentissage supervisé et non supervisé pour que le modèle de prédiction rejette dynamiquement les anciennes données en accordant plus d'attention uniquement aux nouvelles données [6]. Les fonctions d'oubli sont illustrées dans les équations [**VI.13**] et [**VI.14**] pour l'autoencodeur et l'OS-ELM. La stratégie de mise à jour est proposée sur la base de [6] pour que le modèle accepte uniquement les données utiles pendant l'apprentissage.

Pour une base de séquences de données $\{X_k T_k\}_{k=1}^{m}$ Les étapes de l'apprentissage proposé pour la nouvelle DOS-ELM sont les suivantes:

- **Phase initiale (k = 0):**

- Générer aléatoirement des poids et des biais d'entrée identiques à ceux de l'OS-ELM de base pour l nombre de neurones.

- Générer un bruit aléatoire Cr_{k+1} à partir d'une distribution stochastique normalisée entre 0 et 1.
- Corrompre les données initiales X_{k+1} en ajoutant le bruit avec la magnitude γ souhaitée par l'utilisateur comme indiqué dans l'équation [**VI.1**]
- calculer la couche cachée H_{k+1} selon la condition présentée dans [**VI.1].**

$$H_{k+1} = G(a_{k+1}(X_{k+1} + \gamma Cr_{k+1}) + b) \quad \textbf{[VI.2]}$$

$$a_{k+1} = \begin{cases} a, k=0 \\ (\beta_{AE(k+1)}^{T})^{-1}, k>0 \end{cases} \quad \textbf{[VI.3]}$$

- Calculer les poids de sortie initiaux $\beta_{AE\,(k+1)}$ et $\beta_{os\,(k+1)}$ de l'autoencodeur et de l'OS-ELM comme indiqué dans l'équation [**VI.3**] et [**VI.4**] respectivement.

$$\beta_{AE(0)} = P_0 H_0^T X_0 \quad \textbf{[VI.4]}$$

$$\beta_{OS(0)} = P_0 H_0^T T_0 \quad \textbf{[VI.5]}$$

- Calculer l'erreur de reconstruction initiale $e_{AE\,(k+1)}$ et l'erreur de prédiction $e_{os\,(k+1)}$ à partir des équations [**VI.7**] et [**VI.8**].

$$e_{AE(k+1)} = X_{k+1} - H_{k+1}\beta_{AE(k)} \quad \textbf{[VI.6]}$$

$$e_{OS(k+1)} = T_{k+1} - H_{k+1}\beta_{OS(k)} \quad \textbf{[VI.7]}$$

➢ **Phase de mise à jour ($k = k + 1$):**

- Remplacer les poids d'entrée du codeur comme illustré dans l'équation [**VI.2**]
- Calculer les nouvelles erreurs de reconstruction et de prédiction $e_{AE\,(k+1)}$ *et* $e_{os\,(k+1)}$;
- Mettre à jour à la fois $\beta_{AE\,(k+1)}$ *et* $\beta_{os\,(k+1)}$ selon la condition de stratégie de sélection mise à jour présentée dans l'équation **[VI.5]** et **[VI.6]** sur la base des nouvelles formules mises à

jour d'erreur d'estimation, de covariance et de gain où le mécanisme d'oubli est intégré à tous.

$$\beta_{AE(k+1)} = \begin{cases} \beta_{AE(k)} - P_{AE(k+1)} H_{k+1}^{T} e_{AE(k+1)}, e_{OS(k+1)} \leq e_{OS(k)} \\ \beta_{AE(k)}, e_{OS(k+1)} \geq e_{OS(k)} \end{cases} \quad \textbf{[VI.8]}$$

$$\beta_{OS(k+1)} = \begin{cases} \beta_{OS(k)} - P_{OS(k+1)} H_{k+1}^{T} e_{OS(k+1)}, e_{OS(k+1)} \leq e_{OS(k)} \\ \beta_{OS(k)}, e_{OS(k+1)} > e_{os(k)} \end{cases} \quad \textbf{[VI.9]}$$

$$P_{AE(k+1)} = \frac{1}{\lambda_{AE}} (P_{AE(k)} - K_{AE(k+1)} H_{k+1}^{T} P_{AE(k)}) \quad \textbf{[VI.10]}$$

$$P_{OS(k+1)} = \frac{1}{\lambda_{os}} (P_{OS(k)} - K_{OS(k+1)} H_{k+1}^{T} P_{OS(k)}) \quad \textbf{[VI.11]}$$

$$K_{AE(k+1)} = \frac{P_{AE(k)} H_{k+1}}{\lambda_{AE} + H_{k+1}^{T} P_{AE(k)} H_{k+1}^{T}} \quad \textbf{[VI.12]}$$

$$K_{OS(k+1)} = \frac{P_{OS(k)} H_{k+1}}{\lambda_{OS} + H_{k+1}^{T} P_{OS(k)} H_{k+1}^{T}} \quad \textbf{[VI.13]}$$

$$\lambda_{AE(k+1)} = \lambda_{min} + (1 - \lambda_{min}) e^{-\mu \left\| e_{AE(k+1)} \right\|} \quad \textbf{[VI.14]}$$

$$\lambda_{os(k+1)} = \lambda_{min} + (1 - \lambda_{min}) e^{-\mu \left\| e_{os(k+1)} \right\|} \quad \textbf{[VI.15]}$$

- Enfin, le facteur d'oubli $\boldsymbol{\lambda}$ et le facteur de sensibilité $\boldsymbol{\mu}$ doivent toujours être ajustés en fonction des contraintes des bandes inférieures et supérieures qui sont respectivement égales à [$\boldsymbol{\lambda_{min}}$,1] [$\boldsymbol{\mu_{min}}$,1].

Il est mentionné dans les règles de l'apprentissage de l'autoencoder à base de l'ELM dans [21] que le processus de codage peut être réalisé en utilisant l'équation [**VI.15**], mais, en fonction de la même formule qui est mentionnée dans [21] et montrée dans l'équation **[VI.16]**, il est

mathématiquement prouvé que l'équation **[VI.16]** donnera une meilleure représentation, cette formule est déjà testée et a prouvé sa précision [5,7] (déjà mentionné dans chapitre IV).

$$H = XB^T \qquad \textbf{[VI.16]}$$

$$X = H\beta \Rightarrow H = X(\beta^T)^{-1} \qquad \textbf{[VI.17]}$$

Le pseudo-code de **l'Algorithme VI.1** récapitule les règles d'apprentissage de l'approche proposée.

Algorithme VI.1. Algorithme de l'approche proposée (DOS-ELM)

Début
entrées: $X,T,l,G,\lambda_{\min},\gamma,m$
sorties: β_{AE},β_{os}
%% La phase initiale (k=0)
Générer des poids et des biais d'entrées $\{a,b\}$;
Ajouter du bruit et calculer le couche caché: H_{k+1} $H_{k+1} = G(a_{k+1}(X_{k+1} + \gamma Cr_{k+1}) + b)$;
Calculez la matrice de covariance initiale de la couche cachée: $P_0 = (H_0^T H_0)^{-1}$;
Déterminez les poids de sortie initiaux du DAE: $\beta_{AE(0)} = P_0 H_0^T X_0$;
Déterminez les poids de sortie initiaux du DOS-ELM: $\beta_{OS(0)} = P_0 H_0^T T_0$;
Calculer l'erreur de reconstruction initiale: $e_{AE(k+1)} = X_{k+1} - H_{k+1}\beta_{AE(k)}$;
Calculer l'erreur de prédiction initiale: $e_{OS(k+1)} = T_{k+1} - H_{k+1}\beta_{OS(k)}$;
%% la phase récursive (k=k+1)
pour *k=1:m;*
Mettez à jour les poids d'entrée: $a_{k+1} = \begin{cases} a & k=0 \\ (\beta_{AE(k+1)})^{-1}, & k>0 \end{cases}$;
Calculer la nouvelle erreur de reconstruction du DAE: $e_{AE(k+1)} = X_{k+1} - H_{k+1}\beta_{AE(k)}$;
si $e_{AE(k+1)} > e_{AE(k)}$ ***; %% Stratégie mise à jour***
$\lambda_{AE(k+1)} = \lambda_{min} + (1 - \lambda_{min})e^{-\mu\left\|e_{AE(k+1)}\right\|}$; *%% Mise à jour du paramètre d'oubli.*
si $\lambda_{AE(k+1)} > 1$ *%% Contraintes du paramètre d'oubli.*
$\lambda_{AE(k+1)} = 1$;
sinon si $\lambda_{AE(k+1)} < 0$
$\lambda_{AE(k+1)} = 0$;
fin
$K_{AE(k+1)} = \dfrac{P_{AE(k)}H_{k+1}}{\lambda_{AE} + H_{k+1}^T P_{AE(k)} H_{k+1}^T}$; *% Mise à jour de la matrice de gain.*

$P_{AE(k+1)} = \frac{1}{\lambda_{AE}}(P_{AE(k)} - K_{AE(k+1)}H_{k+1}^{T}P_{AE(k)})$; *% Mise à jour de la matrice de covariance.*

$\beta_{AE(k+1)} = \beta_{AE(k)} - P_{AE(k+1)}H_{k+1}^{T}e_{AE(k+1),}e_{OS(k+1)} \leq e_{OS(k)}$; *% Mise à jour de la matrice des poids.*

fin

Calculer la nouvelle erreur de prédiction: $e_{OS(k+1)} = T_{k+1} - H_{k+1}\beta_{OS(k)}$;

si $e_{os(k+1)} > e_{os(k)}$ *% Stratégie mise à jour*

$\lambda_{os(k+1)} = \lambda_{min} + (1 - \lambda_{min})e^{-\mu\left\|e_{os(k+1)}\right\|}$; *% Mise à jour du paramètre d'oubli.*

si $\lambda_{os(k+1)} > 1$ *%% Contraintes du paramètre d'oubli.*

$\lambda_{os(k+1)} = 1$;

sinon si $\lambda_{os(k+1)} < 0$;

$\lambda_{os(k+1)} = 0$;

fin

$K_{OS(k+1)} = \frac{P_{OS(k)}H_{k+1}}{\lambda_{OS} + H_{k+1}^{T}P_{os(k)}H_{k+1}^{T}}$; *% Mise à jour de la matrice de gain.*

$P_{OS(k+1)} = \frac{1}{\lambda_{os}}(P_{OS(k)} - K_{OS(k+1)}H_{k+1}^{T}P_{OS(k)})$; *% Mise à jour de la matrice de covariance.*

$\beta_{OS(k+1)} = \beta_{OS(k)} - P_{OS(k+1)}H_{k+1}^{T}e_{OS(k+1)}, e_{OS(k+1)} \leq e_{OS(k)}$; *% Mise à jour de la matrice des poids*

fin

fin

fin

3. Simulation numérique et discussion

Avant d'aller plus loin pour toute explication, nous devons mentionner que l'analyse comparative est effectuée en utilisant les hyper-paramètres du **tableau VI.1**, où ces valeurs de paramètres sont affectées expérimentalement en répétant le processus d'apprentissage plusieurs fois. Ces paramètres ont été obtenus simplement par une recherche exhaustive basée sur une *recherche de grille* à travers un ensemble d'hyper-paramètres spécifié manuellement.

Tableau: VI.1. Tableau des hyper-paramètres

λ_{min}	μ	l	G	lots
0.98	0.001	100	ReLU	Varié

L'algorithme proposé est appliqué sur l'ensemble de données FD001 obtenues à partir de C-MAPSS, les performances des algorithmes sont évaluées selon deux fonctions métriques en plus du temps de l'apprentissage. La fonction de score et l'erreur quadratique moyenne (RMSE) sont exprimées respectivement dans les équations [**VI.18**] et [**VI.19**] qui ont été proposées dans [21] (chap IV, § 4), ou ***N*** représente le nombre d'échantillons et *d* c'est la différence entre la valeur estime et la valeur désiré du RUL.

$$s = \begin{cases} \sum_{i=1}^{N} e^{-d/13} - 1, d_i \geq 0 \\ \sum_{i=1}^{N} e^{-d/10} - 1, d_i < 0 \end{cases} \quad \textbf{[VI.18]}$$

$$R^{MSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} d^2} \quad \textbf{[VI.19]}$$

Avant d'illustrer les résultats finaux de cette expérience, il est préférable d'expliquer l'effet de chaque contribution séparément sur l'apprentissage de l'algorithme proposé passant par la stratégie le débruitage, le mécanisme d'oubli et la stratégie de sélection.

3.1. Processus de débruitage

Dans cette expérience, nous avons utilisé un seul type de bruit. Le bruit est généré à partir de la distribution normale Gaussienne selon le rapport et l'amplitude souhaités par l'utilisateur.

Pour rendre les valeurs des échantillons d'entrée encore normalisées entre 0 et 1 pour satisfaire les contraintes d'apprentissage ELM même après corruption, le bruit est normalisé entre 0 et 1 avant d'être multiplié par le facteur de grandeur souhaité par l'utilisateur γ et mélangé avec des mini-lots d'entrée. Après cela, ce mélange est normalisé à nouveau avec une normalisation min-max entre 0 et 1. La **figure VI.2** traite clairement du processus de corruption d'entrée avec **γ = 0,25** et un taux de corruption égal à **0,60** pour une série temporelle choisie de l'ensemble d'apprentissage.

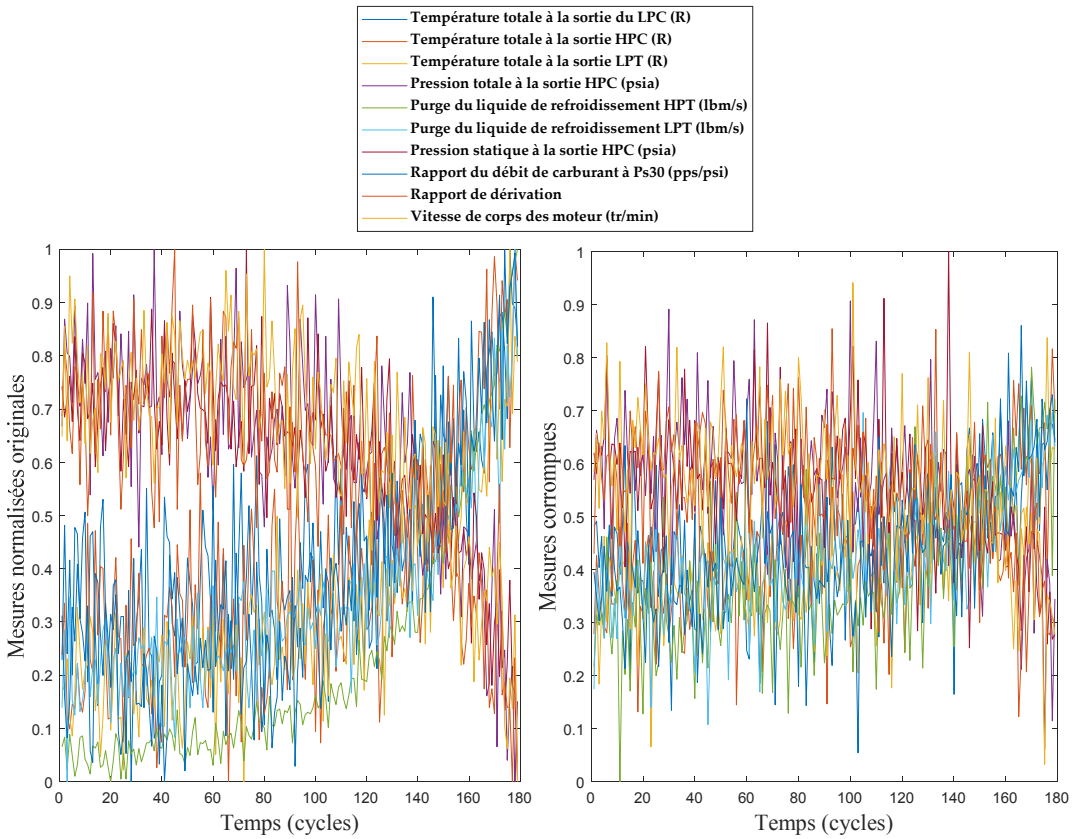

Figure: VI.2. Données d'entrées corrompues par le bruit

Avant de comparer l'algorithme avec ceux mentionnés précédemment, nous avons testé l'effet du taux de corruption et de l'ampleur sur la précision de prédiction du modèle d'apprentissage et les résultats sont présentés dans le **tableau VI.2**.

Tableau: VI.2. Effet du bruit sur la prédiction du RUL

Taux de bruit	Amplitude (γ)	RMSE	Score
0.01	**0.01**	**12.3273**	**198.7692**
0.05	0.01	12.6805	204.5811
0.40	0.09	13.6003	295.7252
0.60	0.09	14.3125	319.5467
0.80	0.09	14.1594	383.9508
0.80	0.15	15.5208	719.5608
0.95	0.25	16.1717	798.3142
1.00	0.60	28.9643	975.1546

Les résultats du **tableau.VI.2** montrent qu'en augmentant progressivement l'amplitude du bruit ou le taux de corruption, le modèle de prédiction commence à perdre son aptitude à une approximation précise. Cependant, en comparant ces résultats plus tard à la version originale de l'algorithme sans schéma de débruitage, les résultats expliqueront que le bruit additif aiderait à améliorer la précision de la prédiction à un certain seuil, et après cela, les échantillons d'apprentissage seront déformés et les formes importantes (important patterns) commenceront à disparaître.

3.2. Mécanisme d'oubli

Dans cette expérience, deux fonctions d'oubli dynamique sont intégrées dans l'autouencodeur et l'OS-ELM pour contrôler la fonction d'approximation de l'estimation du RUL. Les facteurs d'oubli sont modifiés dynamiquement pour s'adapter aux nouvelles données à venir. La **figure VI.3** illustre le changement des deux facteurs d'oubli dynamiques en fonction des mini-lots variant dans le temps.

Nous pouvons voir d'après les comportements illustrés des fonctions d'oubli de la figure VI.3 pour l'autoencodeur et l'OS-ELM que chacun des facteurs d'oubli augmente progressivement de **λmin** vers **λmax (Tableau V.1).** Cette vitesse d'oubli peut être contrôlée en utilisant le facteur de sensibilité μ qui est lié au nombre de séquences décidées par l'utilisateur (plus le nombre de séquences élevé, plus le facteur de sensibilité est petit)

Dans les deux courbes qui se comportent différemment, et contrairement à l'ancien mécanisme d'oubli OS-ELM qui dépend généralement d'un seul hyper-paramètre statique, le nouveau mécanisme d'oubli dynamique utilise un facteur d'oubli qui change en forme de fonction exponentielle, permet l'apprentissage les poids évoluent progressivement pour adapter le modèle d'apprentissage aux variations les plus significatives des échantillons concernés. Et si l'on compare ce mécanisme d'oubli à l'autre dans (chapitre 3), celui-ci dépend de l'erreur d'estimation et non de la formule de différence temporelle (δ).

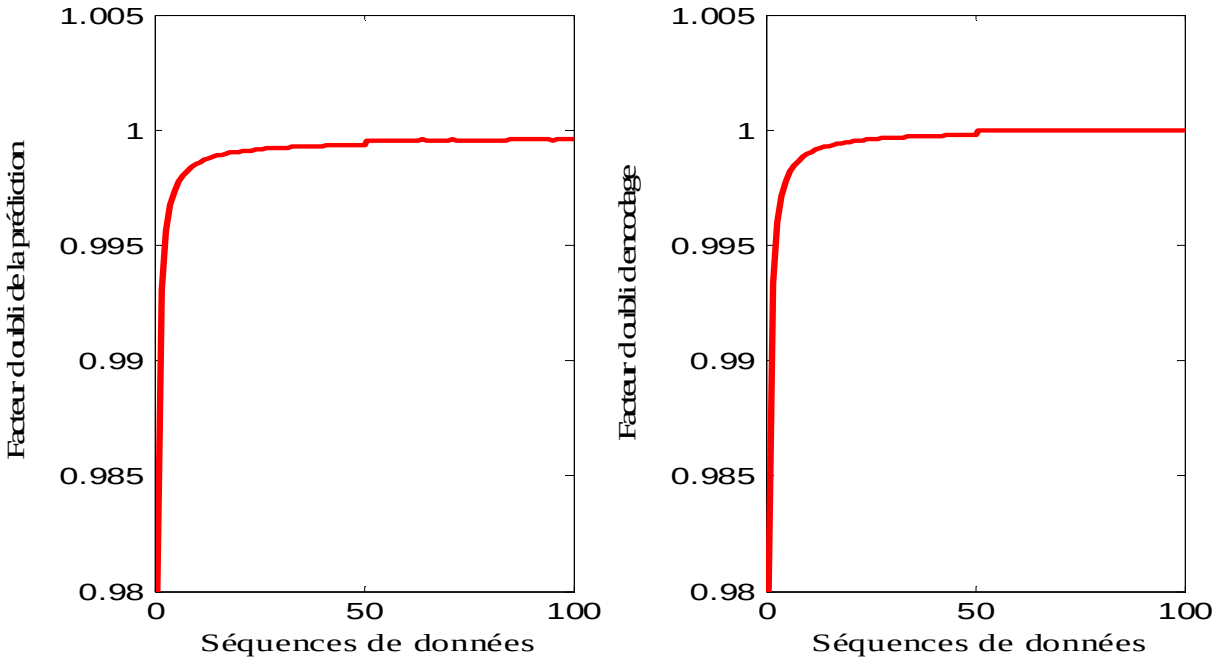

Figure: VI.3. Comportement des facteurs d'oubli lors de l'apprentissage de l'autoencodeur et de l'OS-ELM

3.3. Stratégie de sélection mise à jour

La stratégie proposée est inspiré de [6] où il a été utilisé uniquement pour l'apprentissage supervisé afin de réduire le nombre de mini-lots accumulés pendant la formation en ignorant les non désirés. Dans notre travail, la stratégie mise à jour est intégrée pour l'apprentissage supervisé et non supervisés. La décision de mettre à jour à la fois β_{AE} et β_{OS} dépend du RMSE de l'erreur du RUL montrée précédemment dans l'équation [**VIII.8**]. Si nous voulions le comparer à celui mentionné au chapitre (V) qui dépend du RMSE de la valeur de différence temporelle et non du RMSE d'estimation.

La **figure VI.4** prouve que l'algorithme de la stratégie de sélection fonctionne comme il est suggéré dans les équations [**VI.5**] et [**VI.6**]. Les points rouges représentent le RMSE de l'erreur du RUL pour chaque lot sélectionné. Elles sont supérieures au RMSE de la précédente qui satisfait aux contraintes de la stratégie proposé.

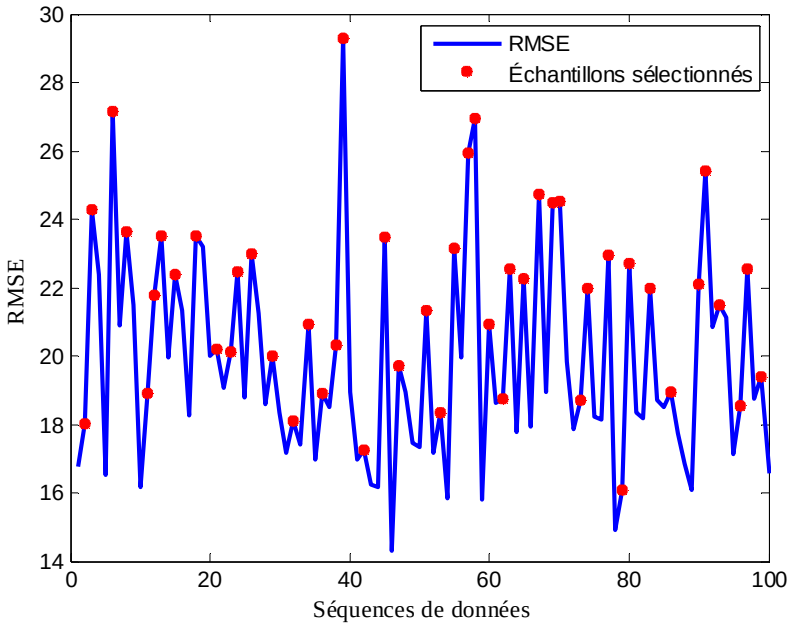

Figure: VI.4. Réponse de la stratégie de sélection aux séquences de données indésirables

L'objectif principal de ces échantillons sélectionnés est de réduire le risque de l'overfitting ainsi que le temps de calcul lors du traitement de ces échantillons en gardant la même précision qui sera expliquée dans les sections suivantes.

3.4. Comparaison des résultats

Pour s'assurer que les nouvelles règles d'apprentissage d'adaptation (mécanisme d'oubli et stratégie de mise à jour) données pour une seule couche cachée et l'algorithme de débruitage (DAE) sont capables d'améliorer à la fois l'apprentissage en ligne OS-ELM et les prédictions d'apprentissage adaptatif. Les performances de l'OS-ELM proposé sont comparées à celles de l'OS-ELM de base et OS-ELM avec un autoencodeur ordinaire de la même manière que la méthodologie proposée et uniquement sans stratégie de sélection et sans mécanisme d'oubli.

Les résultats du tableau VI.3 montrent que DOS-ELM permet une prédiction plus précise que les autres variantes malgré le fait que cette variante utilise tous les échantillons d'apprentissage et que DOS-ELM ne lit que 47 mini-lots grâce à la stratégie USS.

Tableau: VI.3. Comparaison des résultats de Score

Méthodes	RMSE	Score	Temps d'apprentissage (s)	Nombre de séquences
OS-ELM	13.1792	230.9170	3.6972	100
OS-ELM avec autoencodeur ordinaire	12.8721	213.6580	2.1216	100
OS-ELM proposé	12.2919	189.7695	7.4568	47

Dans le travail précédent (Chapitre V), les résultats ont été obtenus à base d'une représentation de caractéristiques spécifiques réalisées via un ensemble d'autoencodeurs ordinaires. Les règles d'apprentissage ont été renforcées par une mémoire temporelle appelée « différence de temps ». Cependant, compte tenu du nouveau système de débruitage, de nouvelles règles d'apprentissage améliorent encore les résultats en raison du bruit réduit des échantillons d'apprentissage. Par conséquent, cela confirme que le nouveau paradigme d'apprentissage peut être une approche intéressante à recommander pour la future prédiction basée sur les données.

Dans la **figure VI.5** et concernant l'OS-ELM ordinaire et la nouvelle variante, le comportement des fonctions d'évaluation a plus de clarté vers des valeurs plus élevées que les autres dans le l'algorithme proposé.

Cela confirme la robustesse des modèles d'apprentissage de la nouvelle approche de représentation des paramètres de l'autoencodeur de débruitage avec le paradigme d'apprentissage adaptatif. Pendant le processus d'apprentissage, l'autoencodeur de débruitage essaie de pousser les couches cachées pour apprendre les formes (patterns) importants à partir d'échantillons corrompus, ce qui augmente par conséquent la qualité des représentations des caractéristiques conduisant à garantir une estimation précise. L'apprentissage adaptatif avec

un mécanisme d'oubli et une stratégie de sélection mise à jour permet un apprentissage dynamique et une réduction des données de formation, ce qui contribue à la minimisation des risques structurels et du sur-ajustement

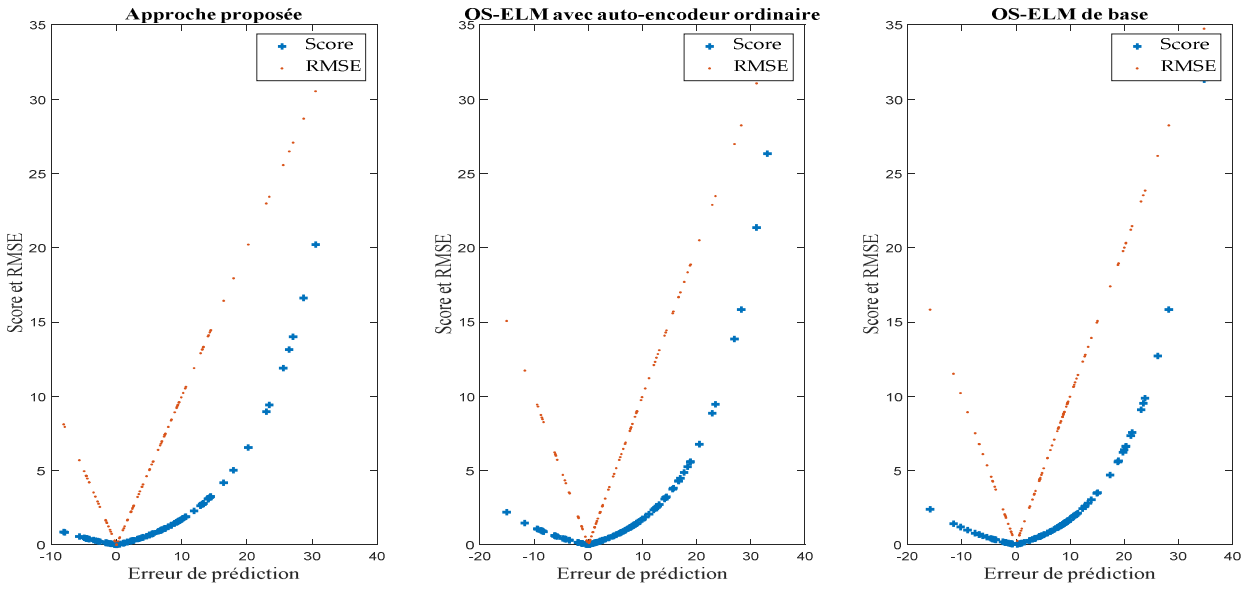

Figure: VI.5 Fonctions de score des approches étudiées

Dans ce cas également, et en comparant les valeurs d'erreur des prédictions tardives et précoces, la **figure VI.5** explique que le RUL estimé sous prédiction précoce est plus précis que celui des prédictions tardives, ce qui clarifie la spécification des prédictions lors de l'utilisation de l'apprentissage récursif des moindres carrés.

Pour des illustrations graphiques du comportement des réponses du réseau aux mesures des capteurs dans chaque profil de dégradation, un ensemble de cycles de vie du turboréacteur à double flux est choisi au hasard dans l'ensemble de test du FD001. La **figure VI.6** donne un exemple comparatif d'ajustement de courbe de la fonction cible du RUL avec les algorithmes étudiés.

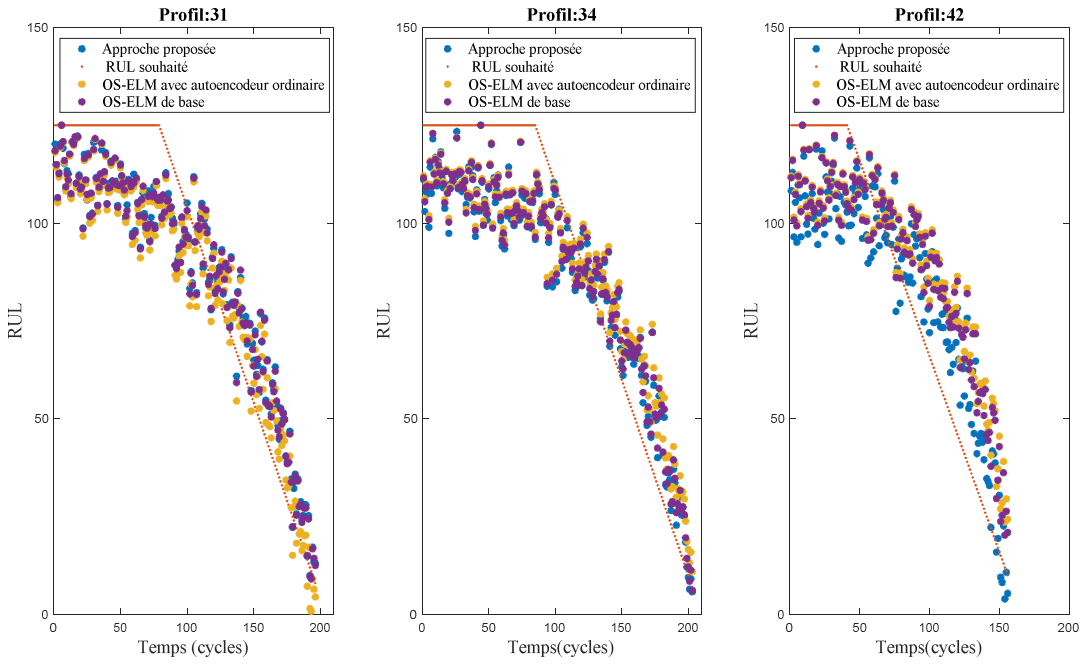

Figure: VI.6 Réponses des réseaux étudiés aux mesures de fonctionnement jusqu'à-défaillance pour un ensemble de moteurs de l'ensemble de test

Nous observons que la nouvelle OS-ELM a une distance de prédiction moins tardive du RUL souhaité et donnent un meilleur ajustement de la courbe pour la fonction RUL que les autres algorithmes. Ces erreurs de prédiction moins tardives expliquent l'importance du processus d'apprentissage adaptatif et de filtrage pour réduire les risques associés à la politique de maintenance mal planifiée.

Dans les théories ELM, et indépendamment de l'overfitting et d'autres problèmes mal posés, plus il y a de nœuds cachés, plus la précision peut être obtenue. Par conséquent, afin d'expliquer les résultats de la **figure VI.6** et d'étudier la précision de l'algorithme proposé pour confirmer sa crédibilité dans différentes circonstances et paramètres aléatoires. La **figure VI.7** élucide les performances des modèles étudiés lors de l'incrémentation du nombre de nœuds cachés en suivant à la fois la validation et les chemins de formation.

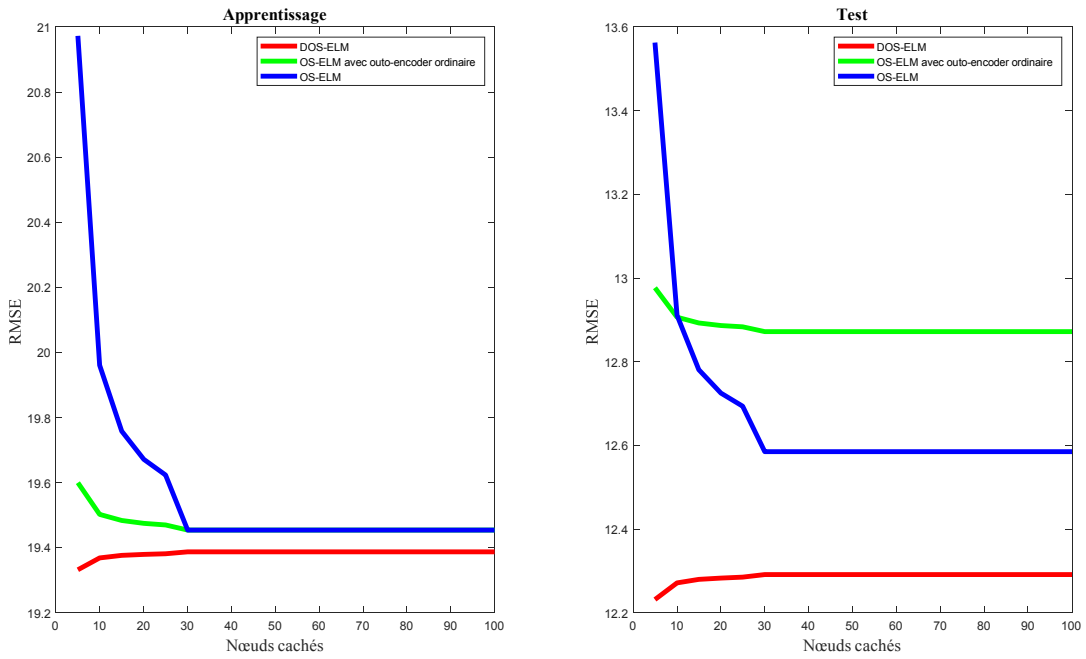

Figure: VI.7. RMSE de l'apprentissage et de test pendant l'incrémentation du nombre de neurones.

Les résultats de la **figureVI.7** confirment que le processus de débruitage adopté force les couches cachées du réseau neuronal à extraire des caractéristiques robustes permettant des représentations plus significatives tout en conduisant à une estimation plus précise. Sur la **figure VI.7**, et en montrant que l'OS-ELM avec un autoencodeur surpasse l'OS-ELM de base pendant l'entraînement et les tests lorsque les nœuds cachés sont inférieurs à 10, cela prouve que l'apprentissage avec des autoencodeurs nécessite moins de nœuds cachés et devient bientôt plus de risque structurel en raison des données bruyantes. Le nouveau système de débruitage montre plus de résistance au risque structurel et permet une précision maximale avec seulement quelques nœuds cachés, ce qui représente moins de coûts de calcul.

4. Conclusion

Une nouvelle approche basée sur les données a été présentée dans ce travail. L'expérience de ces contributions a été réalisée avec un seul sous-ensemble de données du turboréacteur C-

MAPPS, où une seule condition de fonctionnement et un seul mode de défaillance ont été considérés.

Un seul type de bruit a été utilisé pour l'apprentissage de l'autoencodeur de débruitage et pour étudier son effet sur le modèle de prédiction basé sur les données. Les résultats de l'approche proposée expliquent qu'il existe un certain niveau de bruit que le modèle peut gérer. Au-dessus de cette valeur seuil, la fonction d'approximation commencera à s'écarter de la réponse souhaitée. Mais si nous le comparons avec l'algorithme d'origine sans schéma de débruitage, il montrera plus de résistance contre le bruit contaminé.

La stratégie de sélection explique que le modèle basé sur les données n'a pas besoin de toutes les données de l'apprentissage pour obtenir une meilleure précision de prédiction. Le mécanisme d'oubli montre qu'il joue un rôle clé dans l'adaptation des poids de l'OS-ELM pour les nouvelles données variant dans le temps à venir.

La robustesse de l'extracteur de nouvelles fonctionnalités et du schéma d'apprentissage adaptatif associé explique l'importance du processus d'apprentissage adaptatif et de filtrage pour réduire les risques associés à une politique de maintenance mal planifiée.

Conclusion générale et Travaux futurs

Dans les travaux en cours, de nouvelles approches pour l'optimisation des processus d'évaluation de la santé sont proposées. L'objectif principal est de centraliser les systèmes de surveillance des équipements d'exploitation en les projetant sur le web à l'aide de services web et Cloud Computing. De plus, cette projection doit se traduire par des outils d'apprentissage automatique très rapides et précis, capables d'apprendre des informations à partir des données et de s'adapter à l'état physique des systèmes d'exploitation à tout moment. En outre, celui-ci doit prendre en compte le temps d'apprentissage et la réponse pour chaque nouvel échantillon entrant.

D'une part, pour cette mission, nous avons utilisé des théories de l'ELM qui sont basées sur des règles simples d'algèbre linéaire pour satisfaire la généralisation du système de diagnostic et de pronostic ainsi que l'approximation universelle. D'autres parts, nous avons utilisé les services Web pour pouvoir projeter nos systèmes conçus sur le Web en considérant, le temps de calcul et le coût des services.

Ces services web permettent de projeter gratuitement des algorithmes qui dépendent des calculs mathématiques de base tels que l'ELM ainsi que d'un niveau plus élevé d'interaction et de calcul ainsi que de la simplicité d'utilisation.

a. Contributions

Nos contributions dans ce travail est de simplifier la construction de modèles de l'apprentissage précis pour toute étape (nettoyage des données, explorations, apprentissage non supervisé, apprentissage supervisé, post-traitement des résultats) et de minimiser l'intervention humaine pendant le réglages des hyper-paramètres en considérant: a) la réduction de l'overfitting, b) la réduction du temps de calcul durant l'apprentissage, c) la

réduction du nombre des hyper-paramètres, d) l'introduction de l'apprentissage adaptatif et la programmation dynamique.

Tentant de réduire les risques empiriques et structurels pendant le processus d'apprentissage en réduisant l'overfitting, la régularisation de Tikhonov a été adoptée pour réduire la norme L_2 des poids de la couche de sortie de la même manière que celle discutée dans R-ELM au chapitre III.

D'autres parts, l'utilisation des théories d'apprentissage en ligne OS-ELM qui sont basées sur la formule SMW, permet de réduire facilement la consommation de temps en n'introduisant aucun taux d'apprentissage ou réglage itératif. Même si sous «Deep architecture» du réseau de neurones comme le prouvent dans les chapitres V et VI.

Contrairement aux algorithmes de «based gradient descent» tels que la rétropropagation ou la divergence contrastive qui souffrent de nombreux paramètres de réglage tels que (nombre de couches cachées, nombre de nœuds cachés, taille des lots de données , dropout[18], régularisation L_1, régularisation L_2, taux d'apprentissage, erreur de tolérance attendue , taux de convergence, etc.), l'ELM avec une seule couche cachée sans paramètres supplémentaires en plus de la régularisation peut suffire pour satisfaire la généralisation et l'approximation (chap VI).

De plus, OS-ELM permet l'interaction avec les variables environnementales et aborde facilement l'apprentissage adaptatif en introduisant un mécanisme d'oubli tel que discuté au chapitre III et permet d'aborder la programmation dynamique en introduisant d'autres alternatives telles que la fonction d'oubli dynamique proposée dans les chapitres VI, VII et VIII. Ainsi que l'erreur temporelle telle que proposée au chapitre VII.

---

[18] Ce terme fait référence à l'abandon d'unités (à la fois cachées et visibles) dans un réseau neuronal. En termes simples, il fait référence à l'ignorance des unités (c'est-à-dire les neurones) pendant la phase d'apprentissage d'un certain ensemble de neurones qui est choisi au hasard. Par «ignorer», il dit que ces unités ne sont pas prises en compte lors d'une passe avant ou arrière particulière[130].

Les performances des caractéristiques additives de ces contributions sont disponibles sur le web pour téléchargement et exécution sur desktop MATLAB ou pour une utilisation directe sur Octave en ligne ou bien Google Colab.

Théoriquement et mathématiquement, les contributions proposées dans ce travail permettent d'interagir facilement avec la variation de la santé des systèmes étudiés et de leurs conditions environnementales.

Dans un autre travail effectué après la préparation de cette thèse et qui a fait l'objet d'un publication où l'ELM a également été testé pour évaluer la santé d'un autre système qui est un navire de guerre [126]. Les résultats de simulation obtenus ont prouvé encore une fois l'efficacité de l'utilisation de « deep architectures of ELM » en montrant des prédictions plus précises.

Dans ce cas, nous pourrons généraliser l'utilisation des nouvelles approches ELM sur d'autres systèmes de surveillance de diagnostic ou de pronostic autre que le moteur d'avion.

b. Travaux futurs

Le but des travaux futurs sera l'utilisation de ces outils d'apprentissage automatiques proposés sur les systèmes d'exploitation sévères aux caractéristiques et conditions de fonctionnement différentes et d'utiliser différents types de paradigmes pour pouvoir donner une conclusion plus générale et expliquer la limitation de ces contributions.

D'autres parts, cette étude était également limitée sur les environnements Big-data où de longs profils de détérioration avec leurs étiquettes spécifiques sont disponibles. Cependant, les chemins de détérioration de certains systèmes (comme le roulement) sont très coûteux à enregistrer en terme de temps et de mémoire, ce qui oblige les chercheurs à imposer des processus de dégradation accélérés. Ce faisant, les profils de dégradation seront très discriminants et souffriront du manque d'étiquettes. Par conséquent, l'un des principaux

travaux futurs consistera à étudier des problèmes de discrimination similaires à l'aide de l'ELM.

En outre, ces études sont limitées sur l'utilisation de données (dimension 1D) obtenues par des mesures de capteurs uniquement. L'un des principaux défis sera d'étudier l'ELM en utilisant des paramètres de plus grande dimension (dimension 2D) telle que des images identiques à celles du domaine du pronostic médical et de la gestion de la santé. Ces images de plus grande dimension contiennent des indicateurs de santé d'une manière plus compliquée que les observations de 1D. Par conséquent, une segmentation en profondeur et plus d'efforts sont nécessaires pour localiser des formes (patterns) significatifs.

Références bibliographiques

[1]. Domingo, R.; Blanco-Fernández, J.; García-Alcaraz, J.L.; Rivera, L. Complexity in Manufacturing Processes and Systems. *Complexity* **2018**, *2018*, 1–3, doi:10.1155/2018/8738764.

[2]. Efthymiou, K.; Mourtzis, D.; Pagoropoulos, A.; Papakostas, N.; Chryssolouris, G. Manufacturing systems complexity analysis methods review. *Int. J. Comput. Integr. Manuf.* **2016**, *29*, 1025–1044, doi:10.1080/0951192X.2015.1130245.

[3]. Jiang, Q.; Yan, S.; Cheng, H.; Yan, X. Local-Global Modeling and Distributed Computing Framework for Nonlinear Plant-Wide Process Monitoring With Industrial Big Data. *IEEE Trans. Neural Networks Learn. Syst.* **2020**, 1–11, doi:10.1109/tnnls.2020.2985223.

[4]. *Soft Computing in Condition Monitoring and Diagnostics of Electrical and Mechanical Systems*; Malik, H., Iqbal, A., Yadav, A.K., Eds.; Advances in Intelligent Systems and Computing; Springer Singapore: Singapore, 2020; Vol. 1096; ISBN 978-981-15-1531-6.

[5]. Berghout, T.; Mouss, L.; Kadri, O.; Saïdi, L.; Benbouzid, M. Aircraft Engines Remaining Useful Life Prediction with an Improved Online Sequential Extreme Learning Machine. *Appl. Sci.* **2020**, *10*, 1062, doi:10.3390/app10031062.

[6]. Berghout, T.; Benbouzid, M.; Mouss, L.-H. Leveraging Label Information in a Knowledge-Driven Approach for Rolling-Element Bearings Remaining Useful Life Prediction. *Energies* **2021**, *14*, 2163, doi:10.3390/en14082163.

[7]. Berghout, T.; Mouss, L.-H.; Kadri, O.; Saïdi, L.; Benbouzid, M. Aircraft engines Remaining Useful Life prediction with an adaptive denoising online sequential Extreme Learning Machine. *Eng. Appl. Artif. Intell.* **2020**, *96*, 103936, doi:10.1016/j.engappai.2020.103936.

[8]. Willard, J.; Jia, X.; Xu, S.; Steinbach, M.; Kumar, V. Integrating Physics-Based Modeling with Machine Learning: A Survey. **2020**, doi:arXiv:2003.04919v4.

[9]. Huang, G. An Insight into Extreme Learning Machines®: Random Neurons , Random Features and Kernels. *Springer New York LLC* **2014**, *6*, 376–390, doi:10.1007/s12559-014-9255-2.

[10]. Aravanis, T.-C.I.; Sakellariou, J.S.; Fassois, S.D. A stochastic Functional Model based method for random vibration based robust fault detection under variable non–measurable operating conditions with application to railway vehicle suspensions. *J. Sound Vib.* **2020**, *466*, 115006, doi:10.1016/j.jsv.2019.115006.

[11]. Lyonnet, P. *Fiabilité, diagnostic et maintenance prédictive des systèmes*; ed,TICDOC.; livre préparé par Atelier SMB: Paris,France, 2012; ISBN 978-2-7430-1385-1.

[12]. Rezgui, W. Système intégré pour la supervision et le diagnostic des défauts dans les systèmes de production d ' énergies®: les installations photovoltaïque, Thése de doctorat 3eme cycle,Département Génie Industriell, Université de Batna-2, 2015.

[13]. Zwingelstien, G. *Diagnostic des défaillances (théorie et pratique)*; ed,Hermès.; Livre imprimé par EMD SAS: Lantiez,Paris, France, 2012; ISBN 2-86601-463-4.

[14]. Chakour, C. Diagnostic et surveillance des procédés industriels et de leur environnement sur la base de l ' analyse de données, Thèse de Doctorat 3eme cycle, Département d'Electronique,Université Badji Mokhtar, 2016.

[15]. Mahdaoui, R. Contribution à la surveillance dynamique des dynamique des systèmes de production évolutifs par les systèmes de production évolutifs par les systèmes Neuro-Flous Temporels, Thése de doctorat en sciences, Département Génie Iindustriel, Université Batna-2, 2013.

[16]. Kadri, O. L'application des algorithmes de colonies de fourmis pour le diagnostic des systèmes dynamiques et complexes, Thése de doctorat en sciences,Département Génie Industriel,Université Batna-2, 2013.

[17]. Mouss, D. Diagnostic et conduite des systémes de production par approache a base de connaissances, Thése de doctorat en sciences, Déparetemnt Génie Iindusteriel, Université Hadj Lakhdar-Batna, 2005.

[18]. William, K.C. Dignostic des dysfonctionnements d'un Systéme de production, Mémoire d'ingéniorat, Département de Génie Industriell, Université de Batna-2, 2008.

[19]. Waghmode, L. Y.; Patil, R.B. an Overview of Fault Tree Analysis (Fta) for Reliability Analysis. *J. Eng. Res. Stud.* **2016**, *IV*, 06–08.

[20]. Joseph, K. L ' Amdec (analyse des modes de défaillance de leurs effets et de leur criticité), Support de cours,École des Hautes Études Commerciales,universié de Montreal, 1994.

[21]. HSSINA, B.; MERBOUHA, A.; EZZIKOURI, H.; ERRITALI, M. A comparative study of decision tree ID3 and C4.5. *Int. J. Adv. Comput. Sci. Appl.* **2014**, *4*, 13–19, doi:10.14569/specialissue.2014.040203.

[22]. Heimes, F.O. Recurrent neural networks for remaining useful life estimation. *2008 Int. Conf. Progn. Heal. Manag.* **2008**, 1–6, doi:10.1109/PHM.2008.4711422.

[23]. Gouriveau, R.; Medjaher, K.; Ramasso, E.; Zerhouni, N. PHM – Prognostics and health management De la surveillance au pronostic de défaillances de systèmes complexes. *Tech. l'ingénieur Fonct. Strat. la Maint.* **2013**.

[24]. Mathworks. Predictive Maintenance◎: Estimating Remaining Useful Life with MATLAB Available online: https://www.mathworks.com/campaigns/offers/estimating-remaining-useful-life-with-matlab.html?s_iid=fx_74186-aircraft-engines-remaining-useful-life-prediction_rcspot (accessed on Oct 12, 2020).

[25]. Weber, F. Machine Learning Available online: https://encyclopedia.pub/118 (accessed on Oct 12, 2020).

[26]. De Bie, T.; Maia, T.T.; Braga, A.P. Machine learning with labeled and unlabeled data. *ESANN 2009 Proceedings, 17th Eur. Symp. Artif. Neural Networks - Adv. Comput. Intell. Learn.* **2009**, 1–10.

[27]. Sathya, R.; Abraham, A. Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification. *Int. J. Adv. Res. Artif. Intell.* **2013**, *2*, doi:10.14569/ijarai.2013.020206.

[28]. Feurer, M.; Hutter, F. Hyperparameter Optimization. In; 2019; pp. 3–33.

[29]. Blockeel, H.; Webb, G.I.; Auer, P.; Webb, G.I. Overfitting. In *Encyclopedia of Machine Learning*; Springer US: Boston, MA, 2011; pp. 744–744.

[30]. KENTON, W. Overfitting Available online: https://www.investopedia.com/terms/o/overfitting.asp (accessed on Oct 12, 2020).

[31]. Martin, E.; Kaski, S.; Zheng, F.; Webb, G.I.; Zhu, X.; Muslea, I.; Ting, K.M.; Vlachos, M.; Miikkulainen, R.; Fern, A.; et al. Structural Risk Minimization. In *Encyclopedia of Machine Learning*; Springer US: Boston, MA, 2011; pp. 929–930.

[32]. Shawe-Taylor, J.; Bartlett, P.L.; Williamson, R.C.; Anthony, M. Structural risk minimization over data-

dependent hierarchies. *IEEE Trans. Inf. Theory* **1998**, *44*, 1926–1940, doi:10.1109/18.705570.

[33]. Berghout, T.; Mouss, L.H.; Ouahab, K. Remaining Useful Life Prediction for aircraft engines with a new Denoising On-Line Sequential Extreme Learning Machine with Double Dynamic Forgetting Factors and Update Selection Strategy. In Proceedings of the Mechanical Engineering Conferance; Springer, Ed.; Algiers, Algeria, 2020.

[34]. Rifai, S.; Muller, X. Contractive Auto-Encoders Explicit Invariance During Feature Extraction. In Proceedings of the Proceedings of the 28th International Con- ference on Machine Learning, Bellevue, WA, USA; 2011; Vol. 85, pp. 833–840.

[35]. Donoho, D.L. Compressed sensing. *IEEE Trans. Inf. Theory* **2006**, *52*, 1289–1306, doi:10.1109/TIT.2006.871582.

[36]. Matias, T.; Souza, F.; Araújo, R.; Gonçalves, N.; Barreto, J.P. On-line sequential extreme learning machine based on recursive partial least squares. *J. Process Control* **2015**, *27*, 15–21, doi:10.1016/j.jprocont.2015.01.004.

[37]. Cao, J.; Zhang, K.; Luo, M.; Yin, C.; Lai, X. Extreme learning machine and adaptive sparse representation for image classification. *Neural Networks* **2016**, *81*, 91–102, doi:10.1016/j.neunet.2016.06.001.

[38]. Brandimarte, P. Modeling for Dynamic Programming. In; 2021; pp. 67–97.

[39]. Li, X.; Yang, Y.; Pan, H.; Cheng, J.; Cheng, J. Non-parallel least squares support matrix machine for rolling bearing fault diagnosis. *Mech. Mach. Theory* **2020**, *145*, 103676, doi:10.1016/j.mechmachtheory.2019.103676.

[40]. L.H. Chiang, E.L.R. and R.D.B. *Fault Detection and Diagnosis in industerial Systems*; Athenaeum.; Springer, 2001; ISBN 1-85233-327-8.

[41]. Bellaouar, P.A. Fiabilite Maintenabilite Disponibilite, support de cours,faculté des sciences de la technologie, département génie des transports, 2014.

[42]. Yin, B.; Dridi, M.; Moudni, A. El Recursive least-squares temporal difference learning for adaptive traffic signal control at intersection. *Neural Comput. Appl.* **2019**, *31*, 1013–1028, doi:10.1007/s00521-017-3066-9.

[43]. Ben-David, S.; Kushilevitz, E.; Mansour, Y. Online Learning versus Offline Learning. *Mach. Learn.* **1997**, *29*, 45–63, doi:10.1023/A:1007465907571.

[44]. Ordóñez, C.; Sánchez Lasheras, F.; Roca-Pardiñas, J.; Juez, F.J. de C. A hybrid ARIMA–SVM model for the study of the remaining useful life of aircraft engines. *J. Comput. Appl. Math.* **2019**, *346*, 184–191, doi:10.1016/j.cam.2018.07.008.

[45]. Li, Y.; Zhang, W.; Xiong, Q.; Luo, D.; Mei, G.; Zhang, T. A rolling bearing fault diagnosis strategy based on improved multiscale permutation entropy and least squares SVM. *J. Mech. Sci. Technol.* **2017**, *31*, 2711–2722, doi:10.1007/s12206-017-0514-5.

[46]. Yang, Z.X.; Zhang, P.B. ELM Meets RAE-ELM: A hybrid intelligent model for multiple fault diagnosis and remaining useful life predication of rotating machinery. *Proc. Int. Jt. Conf. Neural Networks* **2016**, *2016-Octob*, 2321–2328, doi:10.1109/IJCNN.2016.7727487.

[47]. Zhang, C.; Zhu, Y.; Dong, G.; Wei, J. Data-driven lithium-ion battery states estimation using neural networks and particle filtering. *Int. J. Energy Res.* **2019**, 1–12, doi:10.1002/er.4820.

[48]. Djeziri, M.A.; Benmoussa, S.; Benbouzid, M.E. Data-driven approach augmented in simulation for robust fault prognosis. *Eng. Appl. Artif. Intell.* **2019**, *86*, 154–164, doi:10.1016/j.engappai.2019.09.002.

[49]. Lu, F.; Wu, J.; Huang, J.; Qiu, X. Aircraft engine degradation prognostics based on logistic regression and

novel OS-ELM algorithm. *Aerosp. Sci. Technol.* **2019**, *84*, 661–671, doi:10.1016/j.ast.2018.09.044.

[50]. Alam, M.M.; Bodruzzaman, M.; Zein-Sabatto, M.S. Online prognostics of aircraft turbine engine component's remaining useful life (RUL). *Conf. Proc. - IEEE SOUTHEASTCON* **2014**, 0–5, doi:10.1109/SECON.2014.6950685.

[51]. Ardabili, S.; Mosavi, A.; Várkonyi-Kóczy, A.R. Advances in Machine Learning Modeling Reviewing Hybrid and Ensemble Methods. *Lect. Notes Networks Syst.* **2020**, *101*, 215–227, doi:10.1007/978-3-030-36841-8_21.

[52]. Banihashemi, S.; Ding, G.; Wang, J. Developing a Hybrid Model of Prediction and Classification Algorithms for Building Energy Consumption. *Energy Procedia* **2017**, *110*, 371–376, doi:10.1016/j.egypro.2017.03.155.

[53]. Xiong, X.; Li, Q.; Cheng, N. Remaining useful life prognostics of aircraft engine based on fusion algorithm. *CGNCC 2016 - 2016 IEEE Chinese Guid. Navig. Control Conf.* **2017**, 628–633, doi:10.1109/CGNCC.2016.7828859.

[54]. García Nieto, P.J.; García-Gonzalo, E.; Sánchez Lasheras, F.; De Cos Juez, F.J. Hybrid PSO-SVM-based method for forecasting of the remaining useful life for aircraft engines and evaluation of its reliability. *Reliab. Eng. Syst. Saf.* **2015**, *138*, 219–231, doi:10.1016/j.ress.2015.02.001.

[55]. Saidi, L.; Ben Ali, J.; Bechhoefer, E.; Benbouzid, M. Wind turbine high-speed shaft bearings health prognosis through a spectral Kurtosis-derived indices and SVR. *Appl. Acoust.* **2017**, *120*, 1–8, doi:10.1016/j.apacoust.2017.01.005.

[56]. Zheng, C.; Liu, W.; Chen, B.; Gao, D.; Cheng, Y.; Yang, Y.; Zhang, X.; Li, S.; Huang, Z.; Peng, J. A Data-driven Approach for Remaining Useful Life Prediction of Aircraft Engines. *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC* **2018**, *2018-Novem*, 184–189, doi:10.1109/ITSC.2018.8569915.

[57]. Chen, Z.; Cao, S.; Mao, Z. Remaining useful life estimation of aircraft engines using a modified similarity and supporting vector machine (SVM) approach. *Energies* **2018**, *11*, doi:10.3390/en11010028.

[58]. Ben, J.; Chebel-morello, B.; Saidi, L.; Malinowski, S. Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network. *Mech. Syst. Signal Process.* **2014**, 1–23, doi:10.1016/j.ymssp.2014.10.014.

[59]. Al-Dulaimi, A.; Zabihi, S.; Asif, A.; Mohammadi, A. A multimodal and hybrid deep neural network model for Remaining Useful Life estimation. *Comput. Ind.* **2019**, *108*, 186–196, doi:10.1016/j.compind.2019.02.004.

[60]. Wen, L.; Dong, Y.; Gao, L. A new ensemble residual convolutional neural network for remaining useful life estimation. *Math. Biosci. Eng.* **2019**, *16*, 862–880, doi:10.3934/mbe.2019040.

[61]. Xiong, X.; Yang, H.; Cheng, N.; Li, Q. Remaining Useful Life Prognostics of Aircraft Engines Based on Damage Propagation Modeling and Data Analysis. *Proc. - 2015 8th Int. Symp. Comput. Intell. Des. Isc. 2015* **2016**, *2*, 143–147, doi:10.1109/ISCID.2015.206.

[62]. Mosavi, A.; Ardabili, S.; Várkonyi-Kóczy, A.R. List of Deep Learning Models. *Lect. Notes Networks Syst.* **2020**, *101*, 202–214.

[63]. Zhu, J.; Chen, N.; Peng, W. Estimation of Bearing Remaining Useful Life Based on Multiscale Convolutional Neural Network. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3208–3216, doi:10.1109/TIE.2018.2844856.

[64]. Yang, H.; Zhao, F.; Jiang, G.; Sun, Z.; Mei, X. A novel deep learning approach for machinery prognostics

based on time windows. *Appl. Sci.* **2019**, *9*, doi:10.3390/app9224813.

[65]. Giduthuri Sateesh Babu Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **2016**, *9642*, 214–228, doi:10.1007/978-3-319-32025-0.

[66]. Gupta, D. Introduction to Recurrent Neural Networks Available online: https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks/.

[67]. Zhou, V. towardsdatascience Available online: https://towardsdatascience.com/an-introduction-to-recurrent-neural-networks-for-beginners-664d717adbd.

[68]. Xiang, S.; Qin, Y.; Zhu, C.; Wang, Y.; Chen, H. Long short-term memory neural network with weight amplification and its application into gear remaining useful life prediction. *Eng. Appl. Artif. Intell.* **2020**, *91*, 103587, doi:10.1016/j.engappai.2020.103587.

[69]. Hinton, G. Deep belief networks. *Scholarpedia* **2009**, *4*, 5947, doi:10.4249/scholarpedia.5947.

[70]. Hinton, G.E. A practical guide to training restricted boltzmann machines. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **2012**, *7700 LECTU*, 599–619, doi:10.1007/978-3-642-35289-8-32.

[71]. Feng, D.; Xiao, M.; Liu, Y.; Song, H.; Yang, Z.; Hu, Z. Finite-sensor fault-diagnosis simulation study of gas turbine engine using information entropy and deep belief networks. *Front. Inf. Technol. Electron. Eng.* **2016**, *17*, 1287–1304, doi:10.1631/FITEE.1601365.

[72]. Deutsch, J.; He, D. Using deep learning based approaches for bearing remaining useful life prediction. *Proc. Annu. Conf. Progn. Heal. Manag. Soc. PHM* **2016**, *2016-Octob*, 1–7.

[73]. Ma, J.; Su, H.; Zhao, W.L.; Liu, B. Predicting the remaining useful life of an aircraft engine using a stacked sparse autoencoder with multilayer self-learning. *Complexity* **2018**, *2018*, doi:10.1155/2018/3813029.

[74]. Baldi, P. Autoencoders, Unsupervised Learning, and Deep Architectures. *ICML Unsupervised Transf. Learn.* **2012**, 37–50, doi:10.1561/2200000006.

[75]. Berghout, T. Online Sequential Extreme Learning Machine: A New Training Scheme for Restricted Boltzmann Machines. **2020**, doi:10.20944/PREPRINTS202005.0444.V1.

[76]. Berghout, T. Ristricted Boltzmann Machine Available online: https://www.mathworks.com/matlabcentral/fileexchange/71212-restricted-boltzmann-machine?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[77]. He, X.H.; Wang, D.; Li, Y.F.; Zhou, C.H. A Novel Bearing Fault Diagnosis Method Based on Gaussian Restricted Boltzmann Machine. *Math. Probl. Eng.* **2016**, *2016*, doi:10.1155/2016/2957083.

[78]. Liao, L.; Jin, W.; Pavel, R. Enhanced Restricted Boltzmann Machine with Prognosability Regularization for Prognostics and Health Assessment. *IEEE Trans. Ind. Electron.* **2016**, *63*, 7076–7083, doi:10.1109/TIE.2016.2586442.

[79]. Shao, H.; Jiang, H.; Zhang, H.; Liang, T. Electric Locomotive Bearing Fault Diagnosis Using a Novel Convolutional Deep Belief Network. *IEEE Trans. Ind. Electron.* **2018**, *65*, 2727–2736, doi:10.1109/TIE.2017.2745473.

[80]. Berghout, T.; Mouss, L.H.; Kadri, O.; Hadjidj, N. Adaptive Sparse On-line Sequential Autoencoder for Sensors Measurements Compression Applied to Military Aircraft Engines. In Proceedings of the DAT

2020; IEEE, Ed.; IEEE: Algiers, Algeria, 2020; pp. 12–17.

[81]. Berghout, T. Autoencoders(MATLAB_Codes) Available online: https://www.mathworks.com/matlabcentral/fileexchange/66080-autoencoders-ordinary-type?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[82]. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.-A. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *J. Mach. Learn. Res.* **2010**, *11*, 3371–3408, doi:10.1111/1467-8535.00290.

[83]. Shen, C.; Qi, Y.; Wang, J.; Cai, G.; Zhu, Z. An automatic and robust features learning method for rotating machinery fault diagnosis based on contractive autoencoder. *Eng. Appl. Artif. Intell.* **2018**, *76*, 170–184, doi:10.1016/j.engappai.2018.09.010.

[84]. Wu, E.Q.; Peng, X.Y.; Zhang, C.Z.; Lin, J.X.; Sheng, R.S.F. Pilots' fatigue status recognition using deep contractive autoencoder network. *IEEE Trans. Instrum. Meas.* **2019**, *68*, 3907–3919, doi:10.1109/TIM.2018.2885608.

[85]. Berghout, T. Contractive autoencoder Available online: https://www.mathworks.com/matlabcentral/fileexchange/71257-contractive-autoencoders?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[86]. Andrew, N. Sparse autoencoder: Deep Learning and Unsupervised Feature Learning, CS294A Lecture notes,stanford university, 2011.

[87]. Qi, Y.; Shen, C.; Wang, D.; Shi, J.; Jiang, X.; Zhu, Z. Stacked Sparse Autoencoder-Based Deep Network for Fault Diagnosis of Rotating Machinery. *IEEE Access* **2017**, *5*, 15066–15079, doi:10.1109/ACCESS.2017.2728010.

[88]. Chen, Z.; Li, W. Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network. *IEEE Trans. Instrum. Meas.* **2017**, *66*, 1693–1702, doi:10.1109/TIM.2017.2669947.

[89]. Berghout, T. Sparse autoencoders Available online: https://www.mathworks.com/matlabcentral/fileexchange/72102-sparse-autoencoder?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[90]. Hongming Zhou; Guang-Bin Huang; Zhiping Lin; Han Wang; Yeng Chai Soh Stacked Extreme Learning Machines. *IEEE Trans. Cybern.* **2015**, *45*, 2013–2025, doi:10.1109/TCYB.2014.2363492.

[91]. Li, Y.; Zhang, S.; Yin, Y.; Xiao, W.; Zhang, J. A novel online sequential extreme learning machine for gas utilization ratio prediction in blast furnaces. *Sensors (Switzerland)* **2017**, *17*, doi:10.3390/s17081847.

[92]. Yang, L.; Yang, S.; Li, S.; Liu, Z.; Jiao, L. Incremental laplacian regularization extreme learning machine for online learning. *Appl. Soft Comput. J.* **2017**, *59*, 546–555, doi:10.1016/j.asoc.2017.05.051.

[93]. Huang, G. Bin; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: A new learning scheme of feedforward neural networks. In Proceedings of the IEEE International Conference on Neural Networks - Conference Proceedings; 2004; Vol. 2, pp. 985–990.

[94]. Huang, G. Bin What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt's Dream and John von Neumann's Puzzle. *Cognit. Comput.* **2015**, *7*, 263–278, doi:10.1007/s12559-015-9333-0.

[95]. Deng, W.; Zheng, Q.; Chen, L. Regularized extreme learning machine. *2009 IEEE Symp. Comput. Intell. Data Mining, CIDM 2009 - Proc.* **2009**, 389–395, doi:10.1109/CIDM.2009.4938676.

[96]. Huang, G.; Liang, N.; Rong, H.; Saratchandran, P.; Sundararajan, N. On-Line Sequential Extreme Learning Machine Review of Extreme Learning Ma- chine (ELM). In Proceedings of the INternational Conference on Computational Intelligence; 2005.

[97]. Glossaire STATISTICA. Singularité de la Matrice Available online: https://www.statsoft.fr/concepts-statistiques/glossaire/s/singularite.html (accessed on Oct 12, 2020).

[98]. Huynh, H.T.; Won, Y. Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks. *Pattern Recognit. Lett.* **2011**, *32*, 1930–1935, doi:10.1016/j.patrec.2011.07.016.

[99]. Guo, W.; Xu, T.; Tang, K.; Yu, J.; Chen, S. Online Sequential Extreme Learning Machine with Generalized Regularization and Adaptive Forgetting Factor for Time-Varying System Prediction. *Math. Probl. Eng.* **2018**, *2018*, doi:10.1155/2018/6195387.

[100]. Du, Z.; Li, X.; Zheng, Z.; Zhang, G.; Mao, Q. Extreme learning machine based on regularization and forgetting factor and its application in fault prediction. *Yi Qi Yi Biao Xue Bao/Chinese J. Sci. Instrum.* **2015**, *36*, 1546–1553.

[101]. Berghout, T. Multilayers perceptron based Extreme Learning Machine Available online: https://www.mathworks.com/matlabcentral/fileexchange/69813-multilayers-perceptron-based-extreme-learning-machine?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[102]. Berghout, T. Extreme learning machine with Sub-nets (classify or predict) Available online: https://www.mathworks.com/matlabcentral/fileexchange/70175-extreme-learning-machine-with-sub-nets-classify-or-predict?s_tid=prof_contriblnk (accessed on Oct 12, 2020).

[103]. Huang, G. Bin; Bai, Z.; Kasun, L.L.C.; Vong, C.M. Local receptive fields based extreme learning machine. *IEEE Comput. Intell. Mag.* **2015**, *10*, 18–29, doi:10.1109/MCI.2015.2405316.

[104]. Liu, H.; Chen, T.; Shen, Q.; Yue, T.; Ma, Z. Deep Image Compression via End-to-End Learning. *Electr. Eng. Syst. Sci.* **2018**, 1–5, doi:https://arxiv.org/abs/1806.01496.

[105]. Campos, J.; Meierhans, S.; Djelouah, A.; Schroers, C. Content Adaptive Optimization for Neural Image Compression. *Comput. Vis. Pattern Recognit.* **2019**, doi:1906.01223.

[106]. Cheng, Z.; Sun, H.; Takeuchi, M.; Katto, J. Performance Comparison of Convolutional AutoEncoders, Generative Adversarial Networks and Super-Resolution for Image Compression. *Electr. Eng. Syst. Sci.* **2018**, 2613–2616, doi:1807.00270.

[107]. Zhang, S.; Tan, W.; Li, Y. A Survey of Online Sequential Extreme Learning Machine. *2018 5th Int. Conf. Control. Decis. Inf. Technol. CoDIT 2018* **2018**, 45–50, doi:10.1109/CoDIT.2018.8394791.

[108]. Yu, Y.; Sun, Z. Sparse coding extreme learning machine for classification. *Neurocomputing* **2017**, doi:10.1016/j.neucom.2016.06.078.

[109]. Shi, W.; Jiang, F.; Liu, S.; Zhao, D. Image Compressed Sensing using Convolutional Neural Network. *IEEE Trans. Image Process.* **2019**, *PP*, 1–1, doi:10.1109/tip.2019.2928136.

[110]. Cao, L. le; Huang, W. bing; Sun, F. chun Building feature space of extreme learning machine with sparse denoising stacked-autoencoder. *Neurocomputing* **2016**, *174*, 60–71, doi:10.1016/j.neucom.2015.02.096.

[111]. Saxena, A.; Ieee, M.; Goebel, K.; Simon, D.; Eklund, N. Damage Propagation Modeling for Aircraft Engine Prognostics. In Proceedings of the International conference on prognostics and health management; 2008.

[112]. Prognostics Center of Excellence - Turbofan Engine Degradation Simulation Data Set Available online:

https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/ (accessed on Nov 7, 2020).

[113]. NASA. DASHlink - Prognostics Data Challenge Available online: https://c3.nasa.gov/dashlink/projects/15/#:~:text=The PHM Data Challenge was,of one or more researchers. (accessed on Jul 10, 2020).

[114]. Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In Proceedings of the Proceedings of the 25th international conference on Machine learning - ICML '08; ACM Press: New York, New York, USA, 2008; pp. 1096–1103.

[115]. Zhou, H.; Huang, G.-B.; Lin, Z.; Wang, H.; Soh, Y.C. Stacked Extreme Learning Machines. *IEEE Trans. Cybern.* **2014**, *PP*, 1, doi:10.1109/TCYB.2014.2363492.

[116]. Bai, J.-M.; Zhao, G.-S.; Rong, H.-J. Novel direct remaining useful life estimation of aero-engines with randomly assigned hidden nodes. *Neural Comput. Appl.* **2019**, *1*, doi:10.1007/s00521-019-04478-1.

[117]. Cheng, X.; Liu, H.; Xu, X.; Sun, F. Denoising deep extreme learning machine for sparse representation. *Memetic Comput.* **2017**, *9*, 199–212, doi:10.1007/s12293-016-0185-2.

[118]. Alhamaly, A. Jet Engine Remaining Useful Life (RUL) Prediction | by Ali Alhamaly | Medium Available online: https://medium.com/@hamalyas_/jet-engine-remaining-useful-life-rul-prediction-a8989d52f194 (accessed on Nov 7, 2020).

[119]. Theis, L.; Shi, W.; Cunningham, A.; Huszár, F. Lossy Image Compression with Compressive Autoencoders. *Stat. Mach. Learn.* **2017**, 1–19, doi:1703.00395.

[120]. Berghout, T. L2 Regularization for Extreme Learning Machine Available online: https://colab.research.google.com/drive/1aufDdSfhTbNVpehFRUFpQI5vpaY-j2KI?usp=sharing (accessed on Oct 11, 2020).

[121]. Gu, Y.; Liu, J.; Chen, Y.; Jiang, X. Constraint Online Sequential Extreme Learning Machine for lifelong indoor localization system. *Proc. Int. Jt. Conf. Neural Networks* **2014**, 732–738, doi:10.1109/IJCNN.2014.6889579.

[122]. 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, doi:10.1016/j.ress.2017.11.021.

[123]. Wang, X.; Han, M. Improved extreme learning machine for multivariate time series online sequential prediction. *Eng. Appl. Artif. Intell.* **2015**, *40*, 28–36, doi:10.1016/j.engappai.2014.12.013.

[124]. Chin, C.S.; Ji, X. Adaptive online sequential extreme learning machine for frequency-dependent noise data on offshore oil rig. *Eng. Appl. Artif. Intell.* **2018**, *74*, 226–241, doi:10.1016/j.engappai.2018.06.010.

[125]. Bektas, O.; Jones, J.A.; Sankararaman, S.; Roychoudhury, I.; Goebel, K. A neural network filtering approach for similarity-based remaining useful life estimation. *Int. J. Adv. Manuf. Technol.* **2019**, *101*, 87–103, doi:10.1007/s00170-018-2874-0.

[126]. Berghout, T.; Mouss, L.H.; Bentrcia, T.; Elbouchikhi, E.; Benbouzid, M. A deep supervised learning approach for condition-based maintenance of naval propulsion systems. *Ocean Eng.* **2021**, *221*, 108525, doi:10.1016/j.oceaneng.2020.108525.

[127]. Berghout, T.; Mouss, H.; Kadri, O.; Saidi, L.; Benbouzid, M. Matlab codes of the proposed OS-ELM. *Researchgate* **2020**, doi:10.13140/RG.2.2.28033.40808.

[128]. Berghout, T. Aircraft Engines Remaining Useful Life Prediction Available online:

https://www.mathworks.com/matlabcentral/fileexchange/74186-aircraft-engines-remaining-useful-life-prediction?s_tid=prof_contriblnk (accessed on May 6, 2020).

[129]. Octave Online · Cloud IDE compatible with MATLAB Available online: https://octave-online.net/bucket~2bw6FdomAeYn8p5e96uYdK (accessed on May 31, 2020).

[130]. Lim, H. A Study on Dropout Techniques to Reduce Overfitting in Deep Neural Networks. In; 2021; pp. 133–139.

Production scientifique

Publications dans les revues de ronommées

1). **Berghout, T**.; Mouss, L.-H.; Kadri, O.; Saïdi, L.; Benbouzid, M. Aircraft engines Remaining Useful Life prediction with an adaptive denoising online sequential Extreme Learning Machine. Eng. Appl. Artif. Intell. 2020, 96, 103936, doi:10.1016/j.engappai.2020.103936. Vol: 96 (103936), ISSN : 0952-1976.

2). **Berghout, T**.; Mouss, L.; Kadri, O.; Saïdi, L.; Benbouzid, M. Aircraft Engines Remaining Useful Life Prediction with an Improved Online Sequential Extreme Learning Machine. Appl. Sci. 2020, 10, 1062, doi:10.3390/app10031062. Vol: 10 (3), ISSN : 2076-3417.

3). **Berghout, T.;** L.-H. Mouss; T. Bentrcia; E. Elbouchikhi; M. Benbouzid A deep supervised learning approach for condition-based maintenance of naval propulsion systems. Ocean Eng. 2021, 221, 108525, doi:10.1016/j.oceaneng.2020.108525. Vol: 221(108525), ISSN : 0029-8018.

4). Berghout, T.; Benbouzid, M.; Mouss, L.-H. Leveraging Label Information in a Knowledge-Driven Approach for Rolling-Element Bearings Remaining Useful Life Prediction. Energies 2021, 14, 2163, doi:10.3390/en14082163.

Conférences internationales

1). Berghout, T.; Mouss, L.H.; Ouahab, K. Remaining Useful Life Prediction for aircraft engines with a new Denoising On-Line Sequential Extreme Learning Machine with Double Dynamic Forgetting Factors and Update Selection Strategy. In Proceedings of the Mechanical Engineering Conferance; Springer, Ed.; Algiers, Algeria, 2020.

2). Berghout, T.; Mouss, L.H.; Ouahab, K. Regularization Based Particle Swarm Optimization for Length Changeable Extreme Learning Machine under Health State

Estimation of Military Aircraft Engines. In Proceedings of the DAT 2020; IEEE: Algiers, Algeria, 2020; pp. 1–6.

3). Berghout, T.; Mouss, L.H.; KadriI, O.; Hadjidj, N. Regularized Length Changeable Extreme Learning Machine with Incremental Learning Enhancements for Remaining Useful Life Prediction of Aircraft Engines. In Proceedings of the 020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP); IEEE: EL-Oued- Algeria, 2020; pp. 358–363.

4). Berghout, T.; Mouss, L.H.; Kadri, O.; Hadjidj, N. Adaptive Sparse On-line Sequential Autoencoder for Sensors Measurements Compression Applied to Military Aircraft Engines. In Proceedings of the DAT 2020; IEEE, Ed.; IEEE: Algiers, Algeria, 2020; pp. 12–17.

5). Berghout, T.; Mouss, L.H.; Kadri, O. Dynamic Adaptation for Length Changeable Weighted Extreme Learning earning Machine. In Proceedings of the intelligent systems; mathworks web site: Notherlands-amesterdam, 2019; pp. 2–7.

Sites Web personnels

1). Berghout, T. File Exchange - MATLAB Central Available online: https://www.mathworks.com/matlabcentral/fileexchange/?q=authorid%3A1475477&sort=date_desc_updated (accessed on Nov 15, 2020)[19].

[19] Ce site Web est relatif à un ensemble d'algorithmes de Machine Learning développé pendant le période de notre thèse de doctorat.