الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N° Réf :……

Centre Universitaire
AbdElhafid Boussouf Mila

Institut des Sciences et Technologie **Département des Mathématiques et Informatique**

Mémoire Préparé en vue de l'obtention du diplôme de Master

En : Informatique

Spécialité : Sciences et Technologies de l'Information et de la Communication(STIC)

Thème :

Une approche d'apprentissage automatique pour l'estimation de la durée de vie utile, RUL, pour la maintenance conditionnelle

Préparé par :Zemmouri Nassiba

Meriouche Malak

Soutenue devant le jury:

Président	**Talai Meriem**	**Grade** MAA
Examinateur	**Khalfi Souheila**	**Grade** MAA
Encadreur	**Abderrezak Samira**	**Grade** MAA

Année Universitaire : 2019/2020

RÉSUMÉ

La durée de vie utile restante (RUL) d'un composant ou d'un système, est définie comme la durée entre le moment actuel et la fin de la vie utile. Une estimation précise de RUL joue un rôle essentiel dans les pronostics et la maintenance conditionnelle (CBM). Les approches basées sur les données pour l'estimation de RUL utilisent des données de capteur et des données opérationnelles pour estimer RUL. De nos jours, dans cette classe de problèmes, différents types de réseaux de neurones sont utilisés. En particulier,lorsque l'on souhaite modéliser des problèmes avec des données séquentielles, les réseaux neuronaux récurrents (RNN) sont préférés pour leurs capacités à identifier de manière autonome des motifs de séquences intemporelles.

L'objectif général de ce travail est de trouver la meilleure alternative basée sur le réseau de neurones récurrents LSTM (long short term memory) , jugé puissant dans la littérature pour estimer le RUL d'un turbo réacteur à double flux à partir de la série temporelle de l'ensemble de données C-MAPSS de la NASA.Pour cet effet, nous avons proposé cinq variantes et hybridations des sous types de LSTM, nous avons ensuite essayé de trouver les meilleures configurations en essayant de changer plusieurs valeurs de métriques. Puis nous faisons une analyse et comparaison théorique et pratique des résultats afin d'extraire la meilleure variante. Ensuite, nous avons essayé de proposer plusieurs variantes de cette dernière afin d' aboutir à des meilleurs performances et à une meilleure estimation. A la fin, une étude comparative est faite avec les différents travaux de la littérature, afin d'étudier l'efficacité de la méthode et aussi de mieux classer les performances de la méthode par rapport aux autres.

Les résultats montrent après une analyse et comparaison que les approches proposées sont classées parmi les meilleurs travaux de la littérature. Il y a ceux qui ont pu fournir des résultats qui surpassent les autres surtout CNN_LSTM en termes d'erreur quadratique moyenne (MSE) et de valeur de scoring function et CNN_Bidirectional en terme de RMSE. La méthode CNN_BI_LSTM qui n'est pas encore proposée dans la littérature a donnée

elle aussi de meilleurs résultats.
Mots clés : Maintenance, Maintenance conditionnelle, RUL, deep learning, LSTM, CNN, LSTM Bidirectionnel.

ABSTRACT

The remaining useful life (RUL) of a component or system is defined as the time between the current moment and the end of its useful life. Accurate RUL estimation plays a critical role in prognosis and condition-based maintenance (CBM). Data-driven approaches for estimating RUL use sensor data and operational data to estimate RUL. Nowadays, in this class of problems, different types of neural networks are used. In particular, when it is desired to model problems with sequential data, recurrent neural networks (RNNs) are preferred for their ability to independently identify patterns of immortal sequences.

The general purpose of this work is to find the best alternative based on the LSTM (long short term memory) recurrent neural network, considered powerful in the literature for estimating the RUL of a turbofan jet engine from the series time of the NASA C-MAPSS dataset. For this object, we have proposed five variants and hybridizations of the LSTM subtypes, we try then to find the best configurations by trying to change several values of metrics. Then we do theoretical and practical analysis and comparison of the results in order to extract the best variant, and then we tried to come up with several variants of the latter in order to get better performance and better estimation.

In the end, a comparative study is made with the various works of the literature in order to study the effectiveness of the method and also to better classify the performances of the method compared to others.

The results show after analysis and comparison that the proposed approaches are ranked among the best works of literature. There are those who have been able to provide results that outperform others especially CNN_LSTM in terms of mean square error (MSE) and value of the scoring function and CNN_Bidirectional in terms of RMSE. The CNN_BI_LSTM method, which is not yet proposed in the literature, also gave better results..

Key words :Maintenance, condition based maintenance (CBM), RUL, deep learning, LSTM, CNN, LSTM Bidirectional.

تلخيص :

يتم تعريف العمر الإنتاجي المتبقي لمكون أو نظام على أنه الوقت بين اللحظة الحالية و نهاية عمره الإنتاجي .يلعب التقدير الدقيق لـ العمر الإنتاجي المتبقي دورًا أساسيًا في التشخيص والصيانة القائمة على الحالة (صيانة مشروطة).المناهج القائمة على البيانات لتقدير العمر الإنتاجي المتبقي تستخدم بيانات الاستشعار والبيانات التشغيلية لتقدير العمر الإنتاجي المتبقي . في الوقت الحاضر ، وفي هذه الفئة من الإشكاليات ، يتم استخدام أنواع مختلفة من الشبكات العصبية .على وجه الخصوص عندما نرغب في نمذجة الإشكاليات مع البيانات المتسلسلة ، تُفضل الشبكات العصبية المتكررة لقدرتها على تحديد أنماط التسلسلات الغير متعلقة بالوقت بشكل مستقل.

المغزى العام من هذا العمل هو العثور على أفضل بديل يعتمد على الشبكة العصبية المتكررة LSTM (ذاكرة طويلة المدى)، والتي تعتبر قوية في الساحة لأجل تقدير العمر الإنتاجي المتبقي لمفاعل توربوفان انطلاقا من السلسلة الزمنية لمجموعة بيانات وكالة ناسا التجارية المعيارية لنظام الدفع الجوي(C-MAPSS) .

لهذا الغرض ، اقترحنا خمسة متغيرات وتهجين للأنواع الفرعية لذاكرة طويلة المدى ، ثم حاولنا العثور على أفضل التكوينات من خلال محاولة تغيير عدة قيم مترية. بعد ذلك قمنا بإجراء تحليل نظري وعملي ومقارنة النتائج لاستخراج أفضل متغير . حاولنا بعدها تقديم العديد من المتغيرات لهذا الأخير من أجل تحقيق أداء أفضل وتقدير أفضل.

في النهاية ، تم إجراء دراسة مقارنة مع الأعمال الموجودة المختلفة من أجل دراسة فعالية الطريقة وأيضًا لتصنيف أداء الطريقة بشكل أفضل مقارنة بالآخرين.

تظهر النتائج بعد التحليل والمقارنة أن الأساليب المقترحة مصنفة ضمن أفضل الأعمال الموجودة. هناك من تمكن من تقديم نتائج تفوق الآخرين وخاصة CNN_LSTM من حيث متوسط الخطأ التربيعي (MSE) وقيمة وظيفة التسجيل و CNN_Bidirectional من حيث RMSE.

أعطت طريقة CNN_BI_LSTM التي لم يتم اقتراحها بعد من بين الأعمال الموجودة نتائج أفضل أيضًا.

الكلمات المفتاحية :الصيانة، الصيانة المشروطة، العمر الإنتاجي المتبقي ، التعلم العميق، ذاكرة طويلة المدى LSTM، CNN, LSTM Bidirectional

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

Glossaire :

ANN: Artificial Neural Network.

ARIMA: Autoregressive Integrated Moving Average.

ARMA: Autoregressive moving average .

ARMAX: Autoregressive moving average model with exogenous inputs model.

Bidirectional LSTM: bidirectional Long Short-Term Memorys.

CBM: condition based maintenance.

CNN_Bidirectional: Convolutional Neural Network & Bidirectional lstm.

CBLSTM: convolution bidirectional long short term memory.

CNN LSTM: Convolutional Neural Network & Long Short-Term Memorys.

CNN: Convolutional Neural Network.

CNTK : Microsoft Cognitive Toolkit.

C-MAPSS: Commercial Modular Aero-Propulsion System Simulation.

CM: condition monitoring**.**

DNN: Deep Neural Networks.

DOD: department of defense .

Encoder-Decoder LSTM: Encoder-Decoder Long Short-Term Memorys.

FD: Fragment Data.

FD X60 000: la norme AFNOR.

FFNN: Feed-Forward Neural Network.

GRU: Gated Recurrent Unit.

HMM: Hidden Markov Model.

HSMM: hidden semi-Markov model.

IA: Artificial Intelligence.

IBL: instance based learning.

IDE: Integrated Development Environment.

IDE: Integrated Development Environment.

JANET: Just Another NETwork.

LSTM: Long Short-Term Memorys.

LSTM générative: Generative Long Short-Term Memorys.

MAE: mean absolute error.

MILA: Montreal Institute for Learning Algorithms.

MLP: Multi Layer Perceptron.

MSE: mean square error.

NASA: National Aeronautics and Space Administration.

Open CV: Open Computer Vision.

PHM : prognostics and health management .

R ^ 2: regression score.

RàPC: raisonnement à partir de cas.

RBD: réseau bayésien dynamique .

RELU: Rectified Linear Unit.

RNA: Réseaux de Neurones Artificielles.

RMSE: Root mean square error.

RNN: Recurrent Neural Network .

RMSProp: Root Mean Square Propagation.

RUL: remaining useful life.

Seq2Seq: Sequence To Sequence

Stacked LSTM: Stacked Long Short-Term Memorys.

Vanilla LSTM: Vanilla Long Short-Term Memorys.

WS: Windows Size.

Dédicace

Je dédie mon travail à mes parents qui etaient source de mon éducation et etaient mon aide durant mon parcours scolaire .

Je dédie aussi mon travail à mes soeurs "Rihab,kahina,wiam" qui etaient toujours à mes cotés, à mes frères "Ayoub et Lounis", à mes chères amies "Soumia ,Abir,Roufaida,Rahma,Ahlam,Roumaissa,Chaima,Zahia" et surtout mon binome "Malak" et en fin à toutes mes collégues du promo 2020.

Merci à tous ceux qui m'ont soutenu même avec un mot gentil, un petit sourire qui a insufflé de l'espoir dans mon cœur. Salutations et gratitude à vous. Merci beaucoup.

Nassiba

Dédicace

Je dédie mon travail à mes parents qui etaient source de mon éducation et etaient mon aide durant mon parcours scolaire

Je dédie aussi mon travail à mes soeurs "wiam,Aroua" qui etaient toujours à mes cotés, à mes frères "Chihab et Moustafa", à mes chères amies "Samar ,Roufaida,Rahma,Ahlam,Roumaissa,Chaima,Radja" et surtout mon binome "Nassiba" et en fin à toutes mes collégues du promo 2020.

Merci à tous ceux qui m'ont soutenu même avec un mot gentil, un petit sourire qui a insufflé de l'espoir dans mon cœur. Salutations et gratitude à vous. Merci beaucoup.

Malak

Remerciement

Tout d'abord, nous tenons à remercier le bon Dieu pour nous avoir illuminés et menés jusqu'ici.

Un remerciement particulier à notre encadreur M^{me} Abderrezak Samira pour sa présence, son aide et surtout pour ses précieux conseils durant toute la période du travail.

Nous remercions également les membres de jury M^{me} Talai Meriem pour avoir accepté de présider et M^{me} Khalfi Souheila pour avoir accepté d'examiner notre travail

Nous remercions aussi le corps professoral et administratif de l'institut des sciences et de la technologie pour la richesse et la qualité de leur enseignement, qui déploient de grandes efforts pour assurer à leurs étudiants une formation actualisée.

Nous tenons à remercier et à adresser notre reconnaissance à toutes les personnes qui ont participé de prés ou de loin à la réalisation de ce travail.

INTRODUCTION GÉNÉRALE

La maintenance est une étape cruciale dans tout domaine de l'industrie tels que l'aérospatiale, les véhicules, l'industrie lourde, etc. Les stratégies traditionnelles de la maintenance telles que la maintenance corrective en cas de panne et la maintenance préventive systématique programmée sont de moins en moins capables de répondre à la demande industrielle croissante d'efficacité et de fiabilité. Les téchnologies intelligentes de gestion des pronostics et de la santé (PHM), également appelées maintenance conditionnelle (CBM), montrent des capacités d'application prometteuses dans les industries.

Les objectifs de la maintenance conditionnelle comprènnent la maximisation de la disponibilité opérationnelle, la réduction des coûts de maintenance et l'amélioration de la fiabilité et de la sécurité du système en surveillant les conditions de l'installation. La maintenance conditionnelle implique l'évaluation de la dégradation des performances des machines et l'estimation de la durée de vie utile restante (en anglais : Remaining Useful Life, RUL) de la machine.

La durée de vie utile restante (RUL) peut être estimée en fonction des données de trajectoire de l'historique, ce qui est très important pour améliorer les calendriers de maintenance afin d'éviter les pannes catastrophiques d'ingénierie et d'économiser les coûts résultants. L'estimation RUL a un rôle important dans différents domaines, y compris les industries aéronautiques, les équipements médicaux et les centrales électriques, ce qui a inspiré les chercheurs à développer une variété d'approches de prédiction RUL. En général, les méthodes existantes pour la prédiction RUL peuvent être regroupées en trois catégories principales, les approches basées sur les modèles, les approches basées sur les données et les approches hybrides.

Les approches basées sur les modèles ont tendance à être plus précises si la dégradation du système complexe est modélisée avec précision , elles nécessitent des connaissances préalables approfondies sur les systèmes physiques qui ne sont généralement pas disponibles dans la pratique . Les approches basées sur les données sont capables de modéliser

les caractéristiques de dégradation basées sur les données historiques des capteurs. Les corrélations et causalités sous-jacentes dans les données de capteur collectées peuvent être révélées, et les informations du système de correspondances telles que RUL peuvent être déduites. Les approches basées sur les données nécessitent généralement des données historiques suffisantes pour la formation des modèles. Ces dernières années, de nombreux algorithmes basés sur les données ont été proposés, de bons résultats pronostiques ont été obtenus, notamment : les modèles d'apprentissage comme les réseaux de neurones, la machine à vecteurs de support (SVM), les modèles de Markov cachés, etc.

Récemment, une famille de modèles d'apprentissage émergée appelée Deep learning qui vise à apprendre des abstractions de niveau supérieur à partir des données brutes, les modèles d'apprentissage en profondeur ne nécessitent aucune fonctionnalité conçue à la main par les gens, au lieu de cela, ils apprendront automatiquement une représentation hiérarchique d'entités à partir de données brutes. Dans l'apprentissage profond, une architecture profonde avec plusieurs couches est construite pour automatiser la conception des fonctionnalités. Le réseau neuronal récurrent, une classe d'architectures de deep learning : est un modèle plus intuitif pour les données de séries chronologiques, mais il convient à la prédiction de la valeur future des séries chronologiques. Nous traitons le problème d'estimation RUL comme une régression de séries chronologiques multivariées .RNN utilise la mémoire à long court terme LSTM. Un LSTM convient à la classification, au traitement et à la prédiction de séries temporelles avec des délais inconnus de temps et de taille entre des événements importants.

Notre objectif s'inscrit dans le cadre de l'estimation de RUL en utilisant le deep learning. Pour atteindre cet objectif, nous proposons cinq variantes basées sur le modèle LSTM en appliquant plusieurs hybridations. Ensuite, nous faisons varier plusieurs métriques et analyser à chaque fois les résultats,puis faire des comparaisons entre les modèles proposés en appliquant les métriques qui ont donné de meilleurs résultats. A la fin, une comparaison entre ces modèles et ceux de la littérature est accomplie.

Organisation du mémoire

Ce mémoire est structuré de la manière suivante :

chapitre 1 : la maintenance conditionnelle

Ce chapitre est composé en deux parties :

partie1 : Nous définissons dans cette partie la maintenance, les différents types de la maintenance ainsi que leurs avantages et inconvénients.

partie2 : Dans cette partie, nous détaillons la maintenance conditionnelle, son objectif

et ses différentes étapes et téchniques.

chapitre 2 :Le temps restant utile (Remaining useful life RUL) :

Dans ce chapitre,nous présentons le RUL avec la classification des différentes approches de pronostic.

chapitre 3 : Les réseaux de neurones et les deep learning

Dans ce chapitre nous introduirons premièrement les réseaux de neurones en général,ensuite nous nous intéressons au deep learning et ses différents types. Nous définissons plus en détail les sous types de LSTM qu'on va utiliser dans notre approche,et en fin nous citons quelques travaux de RUL avec le deep learning.

chapitre 4 : Contribution

Nous allons faire dans ce chapitre une étude expérimentale sur l'approche proposée avec une discussion de chaque résultat. Nous terminons par une analyse et comparaison des résultats des différentes méthodes utilisées ainsi une comparaison avec les résultats de quelques travaux mentionnés dans la littérature.

chapitre 5 : Implémentation

Dans ce chapitre, nous abordons les outils de développement utilisés dans notre application , ainsi que l'interface de notre application. Nous présentons aussi quelques pseudo codes de l'exécution.

Nous terminons enfin,par une conclusion générale qui permet de récapituler notre travail et de donner quelques perspectives suggérées.

CHAPITRE 1

LA MAINTENANCE CONDITIONNELLE

Introduction

La maintenance est devenue une des fonctions stratégiques de l'entreprise. Les actions de maintenance, en garantissant le bon fonctionnement des outils de production (réparation, dépannage, révision...). Dans l'industrie, le rôle de la sûreté de fonctionnement c'est maitriser les risques qui peuvent mettre en cause la fiabilité, la disponibilité et la sécurité d'un matériel ou d'une installation[1].

Dans ce chapitre, nous détaillerons dans la première partie, la maintenance et ses différents types. Ensuite,dans la deuxième partie, nous définirons la maintenance conditionnelle, son objectif et ses différentes étapes et téchniques.

Partie 01 : La maintenance

1.1 Définition

Selon la (norme FD X60-000) La maintenance est un ensemble de toutes les actions téchniques, administratives et de management durant le cycle de vie d'un bien, déstinées à le maintenir ou à le rétablir dans un état dans lequel il peut accomplir la fonction requise[2].

La maintenance est définie comme la combinaison de tous les aspects téchniques,administratifs et actions au cours du cycle de vie d'un élément déstinées à le conserver ou à le réstaurer dans un état dans lequel il peut exécuter la fonction requise. Les tâches de maintenance varient selon les environnements de travail, qui comprennent par exemple inspection visuelle, test, mesure, changement de consommables (graissage, lubrification, filtres à huile), réglage, réparation,entretien, remplacement de pièces, entretien, échantillonnage d'huile,

lubrification, resserrage des boulons, nettoyage, détection de panne, diagnostic de panne, etc [3].

1.2 Les types de la maintenance

Il existe deux types de maintenance : la maintenance corrective et la maintenance préventive. La différence entre elles réside au moment d'intervention vis-à-vis de la panne. Le premier type de maintenance est appliqué après l'occurrence de la panne, alors que le deuxième type s'applique avant cette dernière[1].

Dans la figure 1.1, nous présentons les différents types de maintenance étudiée.

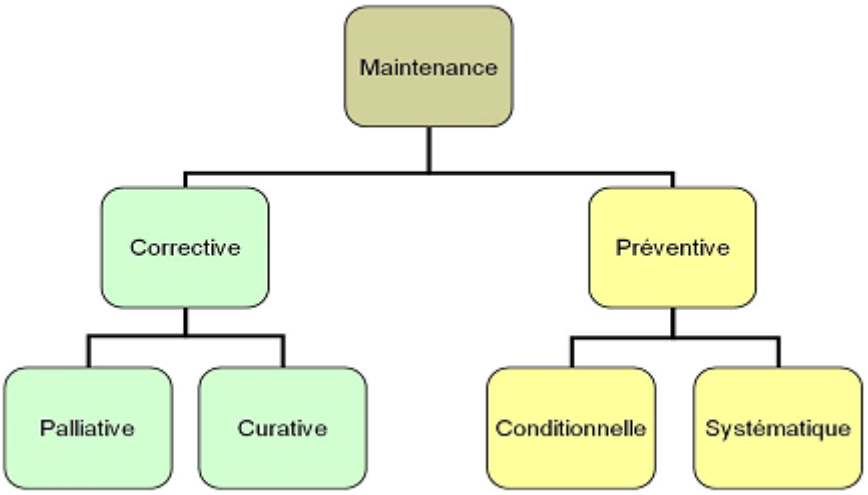


FIGURE 1.1 – Les différents types de maintenance.
[4]

1.2.1 Maintenance corrective

Selon la (norme FD X60-000) La maintenance corrective exécutée après détection d'une panne et destinée à remettre un bien dans un état dans lequel il peut accomplir une fonction requise [2].

La maintenance corrective peut être :

1.2.1.1 Maintenance curative

Selon la (norme FD X60-000) c'est une action de maintenance corrective ayant pour objet de rétablir un bien dans un état spécifié pour lui permettre d'accomplir une fonction requise. Le résultat des actions réalisées doit présenter un caractère permanent. Des modi-

fications et améliorations peuvent être apportées, afin de réduire l'occurrence d'apparition de la défaillance ou d'en limiter l'incidence [2].

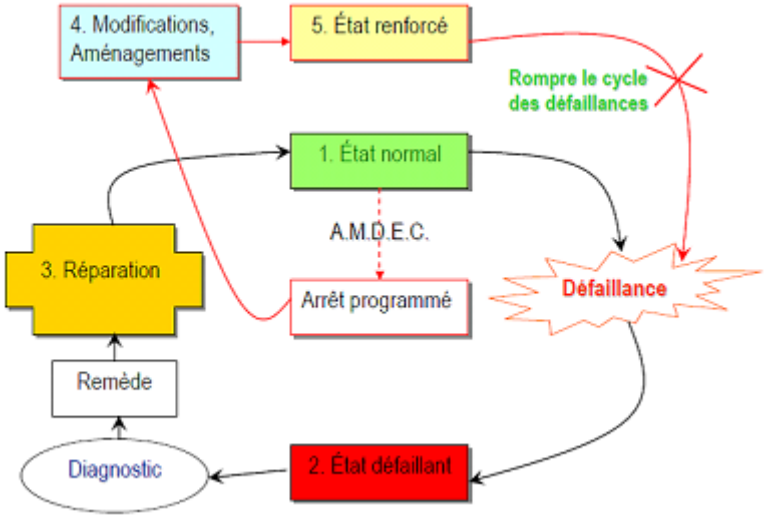


FIGURE 1.2 – Maintenance curative ou réparation .
[55]

1.2.1.2 Maintenance palliative

Selon la (norme FD X60-000) c'est une action de maintenance corrective destinée à permettre à un bien d'accomplir provisoirement tout ou partie d'une fonction requise Appelée couramment «dépannage», la maintenance palliative est principalement constituée d'actions à caractère provisoire qui doivent être suivies d'actions curatives [2].

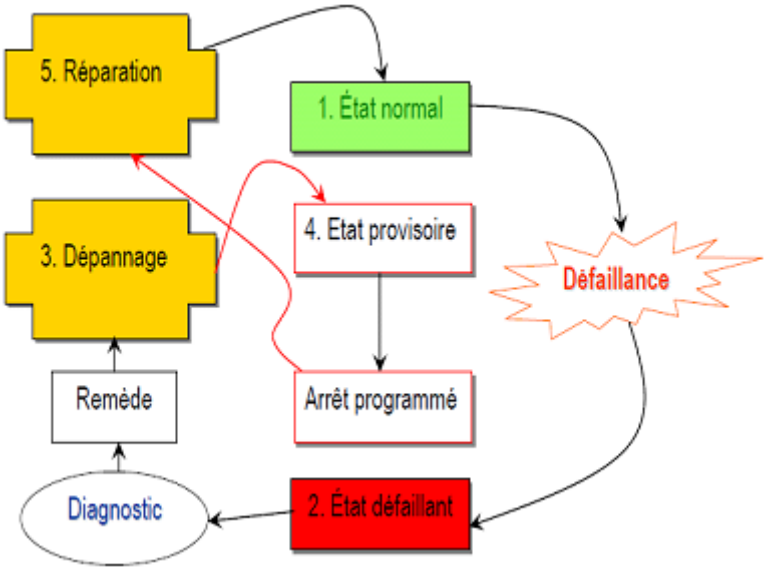


FIGURE 1.3 – Maintenance palliative ou dépannage.
[55]

1.2.2 Maintenance préventive

Selon la (norme FD X60-000) est une maintenance exécutée à des intervalles prédéterminés ou selon des critères prescrits [2]. L'objectif de la maintenance préventive demeure de réduire la probabilité de défaillance ou la dégradation du fonctionnement d'un bien [11].

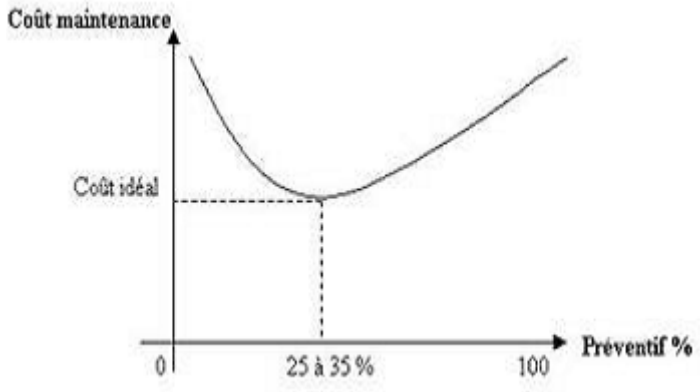


FIGURE 1.4 – Equilibre maintenance corrective (curative)/préventive(Valeurs indicatives pour la mécanique).

[11]

Trop de maintenance préventive n'est pas économiquement viable (le zéro panne n'est à priori recherché que pour des raisons de sécurité) : il faut trouver le bon équilibre maintenance préventive maintenance corrective (ou curative). L'équilibre maintenance corrective / préventive est différent pour chaque site industriel, de process, de manufacture, de service, etc. Il faut donc le déterminer spécifiquement et elle est justifiée quand le coût de défaillance est supérieur au coût d'intervention préventive[11]. La maintenance préventive est pratiquée selon la panne (la figure 1.5) :

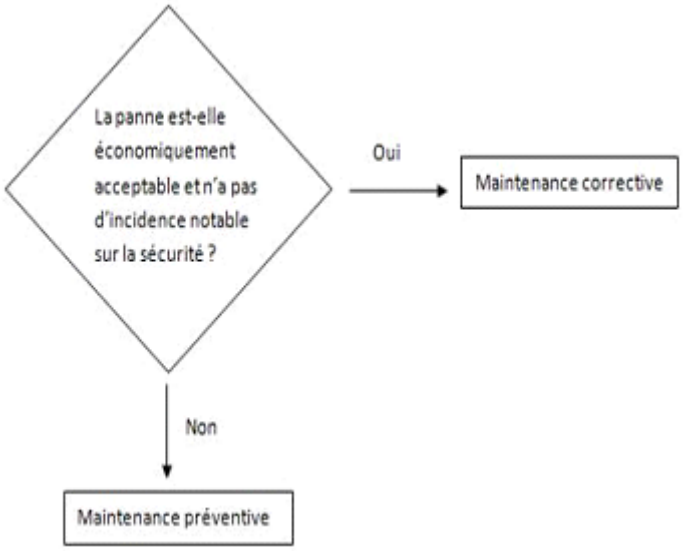


FIGURE 1.5 – Equilibre maintenance corrective (curative)/préventive(Valeurs indicatives pour la mécanique)

[11]

1. **Buts de la maintenance préventive**

   — Augmenter la durée de vie des matériels.

   — Augmenter la fiabilité d'un équipement, donc diminuer la probabilité des défaillances en service .

   — réduction des coûts de défaillance et amélioration de la disponibilité.

   — Améliorer l'ordonnancement des travaux, donc les relations avec la production.

   — Réduire et régulariser la charge de travail.

   — Diminuer les temps d'arrêt en cas de révision ou de panne.

   — Supprimer les causes d'accidents graves par moins d'improvisations dangereuses [55].

Les visites préventives permettent de cumuler des informations relatives au comportement du matériel.

- Si les résultats montrent une loi de dégradation, il sera aisé de connaître l'instant où une action systématique sera possible.
- S'ils montrent l'existence de pannes soudaines, répétitives, se rapportant à un sous-ensemble dit « fragile », une analyse statistique des résultats s'orientera sur une politique de maintenance[55].

elle est subdivisée en :

1.2.2.1 La maintenance conditionnelle

Selon la (norme FD X60-000) est une maintenance préventive basée sur une surveillance du fonctionnement du bien et/ou des paramètres significatifs de ce fonctionnement et intégrant les actions qui en découlent, La surveillance du fonctionnement et des paramètres peut être exécutée selon un calendrier, ou à la demande, ou de façon continue. La maintenance conditionnelle est dite aussi prévisionnelle Selon la (norme FD X60-000) la maintenance prévisionnelle est une maintenance conditionnelle exécutée en suivant les prévisions extrapolées de l'analyse et de l'évaluation de paramètres significatifs de la dégradation du bien [2].

1.2.2.2 La maintenance systématique

Selon la (norme FD X60-000) est une maintenance préventive exécutée à des intervalles de temps préétablis ou selon un nombre défini d'unités d'usage mais sans contrôle préalable de l'état du bien. [2]

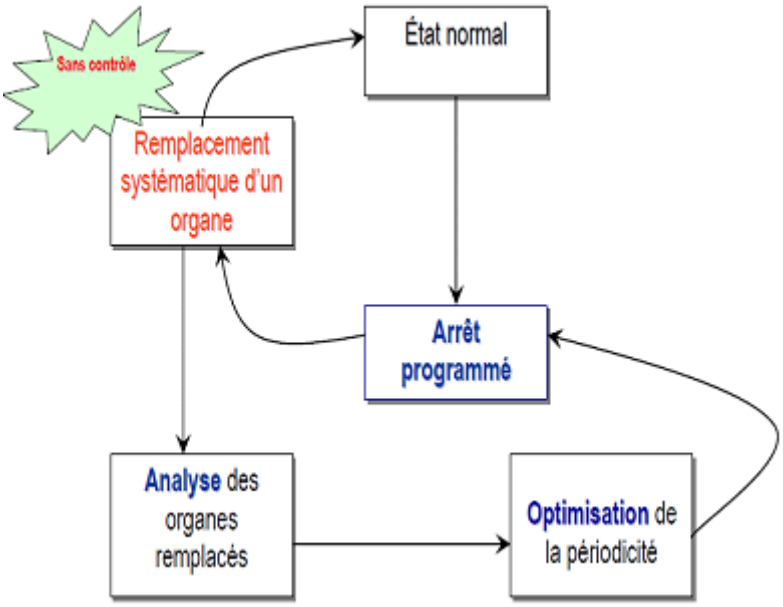


FIGURE 1.6 – Maintenance Systématique.
[51]

L'opération de maintenance est effectuée conformément à un échéancier, un calendrier déterminé à priori. Aucune intervention n'a lieu avant l'échéance prédéterminée. L'optimisation d'une maintenance préventive systématique consiste à déterminer au mieux la périodicité des opérations de maintenance sur la base du temps, du nombre de cycles de fonctionnement, du nombre de pièces produites, etc[5].

1. **Cas d'application**
   - **Equipements soumis à une législation en vigueur (sécurité réglementée) :** appareils de levage, extincteurs, réservoirs sous pression, convoyeurs, ascenseurs, monte-charge, etc.
   - **Equipements dont la panne risque de provoquer des accidents graves :**tous les matériels assurant le transport en commun des personnes, avions, trains, etc.
   - **Equipement ayant un coût de défaillance élevé :**éléments d'une chaîne de production automatisée, processus fonctionnant en continu (industries chimiques ou métallurgiques).
   - **Equipements dont les dépenses de fonctionnement deviennent anormalement élevées au cours de leur temps de service :**consommation excessive d'énergie, éclairage par lampes usagées, allumage et carburation déréglés (moteurs thermiques), etc[51].

1.3 Les avantages et les inconvénients

Type de maintenance	Avantages	Inconvénients
maintenance corrective	Faible coût de maintenance.	Bris inopportun Coût de réparation important Peu de sécurité des travailleurs Stockage important des pièces Temps de réparation élevé Perte de production élevéeé
maintenance conditionnelle	L'accroissement de la durée de vie des pièces par rapport à une politique de changement systématique La suppression des défauts de jeunesse lors de remise en route après un entretien systématique	La nécessité à une équipe de maintenance formée en analyse vibratoire et en essais non destructifs Niveau technologique plus élevé.
maintenance systematique	peu de catastrophes. Bonne planification des opérations et des ressources Contrôle du niveau de stockage des pièces de rechange Sécurité accrue	Remplacement de pièces en bon état Création de défauts lors des remontage (si les procédures ne sont pas claires et contrôlées).

TABLE 1.1 – Avantages et inconvénients des différents types de maintenance. [6]

Partie 02 :La maintenance conditionnelle

1.4 Définition 01

La maintenance conditionnelle, également appelée maintenance prédictive, est la technique de maintenance la plus moderne et la plus populaire discutée dans la littérature. CBM a été introduit en 1975 afin de maximiser l'efficacité de la prise de décision de la maintenance préventive. Selon Jardine, Lin et Banjevic (2006), CBM est un programme de maintenance qui recommande des actions de maintenance (décisions) basées sur les informations collectées à travers le processus de surveillance de l'état. Dans CBM, la

durée de vie (âge) de l'équipement est surveillée par son état de fonctionnement, qui peut être mesuré en fonction de divers paramètres de surveillance, tels que les vibrations, la température, l'huile de graissage, les contaminants et les niveaux de bruit. La motivation de CBM est que 99% des pannes d'équipement sont précédées de certains signes, conditions ou indications qu'une panne va se produire (Bloch et Geitner, 1983). Par conséquent, CBM est nécessaire pour une meilleure gestion de la santé des équipements, un coût du cycle de vie inférieur, une évitement des pannes catastrophiques, etc[7].

1.5 Définition 02

Pour mieux tenir compte de la dégradation réelle du matériel/équipement (par exemple : des conditions réelles d'exploitation), des mesures périodiques ou continues de paramètres observables et significatifs de l'état de dégradation du bien permettent d'espacer ou de supprimer des tâches répétitives, coûteuses et parfois non justifiées. La maintenance préventive conditionnelle suppose l'idée de ne pas réaliser une action de maintenance sur un équipement tant qu'il n'est pas sur le point de ne plus assurer sa fonction requise. Ceci pour tenir compte du fait que la durée de vie de certains équipements peut diminuer si ces derniers sont arrêtés et redémarrés trop fréquemment ou s'ils sont démontés plus que nécessaire. Elle peut aussi permettre de réduire la fréquence de certaines actions de maintenance préventive qui nécessitent l'arrêt ou le démontage des équipements. La maintenance conditionnelle représente une démarche d'optimisation de la maintenance préventive systématique, basée sur la mesure objective de paramètres de la dégradation du bien. Elle repose sur l'extrapolation de mesures et courbes de tendance en fonction de l'usage du bien [2].

1.6 l'objectif de la maintenance conditionnelle

Le cœur du CBM est le processus de surveillance de l'état (CM), où les signaux sont surveillés en permanence en utilisant certains types de capteur ou autres indicateurs appropriés . Ainsi, la maintenance les activités (p. ex. réparations ou remplacements) sont effectuées seulement «en cas de besoin» ou juste avant l'échec .En général, l'objectif principal de CBM est d'effectuer un évaluation de l'état des équipements pour effectuer la maintenance décisions, réduisant ainsi la maintenance inutile et les coûts associés. La figure 1.7 présente deux importants processus de CBM[7].

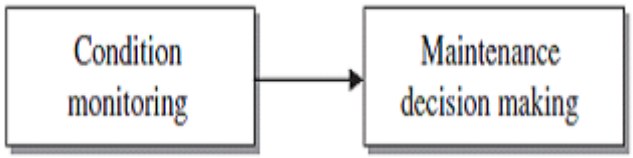


FIGURE 1.7 – processus de CBM en général
[7]

1.6.1 Surveillance de l'état (condition monitoring)

Surveillance de l'état (condition monitoring) est définie comme la collection et l'interprétation des paramètres d'équipement pertinents aux fins de l'identification de l'état des changements d'équipement par rapport à les conditions normale et la tendances de la santé de l'équipement. Ces caractéristiques, telles que les vibrations et températures, restent généralement stables tant que l'équipement est sain. Cependant, une anomalie de ces caractéristiques peut indiquer la survenue d'une défaillance fonctionnelle. Information sur une panne d'équipement naissante peut être obtenue auprès du surveillance de ces paramètres pronostiques [8].

1.6.2 Prise de decision (maintenance decision making)

La prise de décision de maintenance dans le cadre du programme CBM peut être classés en deux : diagnostic et pronostic. Selon Jeong, Leon et Villalobos (2007), le diagnostic est le processus de recherche la source d'un défaut, tandis que le pronostic est le processus d'estimation /prédire quand une panne peut survenir [7].

1.7 Téchniques de la surveillance de l'état (Condition Monitoring)

1.7.1 Surveillance des vibrations

La technique CM la plus populaire utilisée dans le programme CBM, en particulier pour les équipements rotatifs (par exemple, roulement et boîte de vitesses), est la surveillance des vibrations. La technique de surveillance des vibrations fait référence à l'utilisation de la détection non destructive in situ et de l'analyse des caractéristiques des équipements. Cela signifie que la santé de l'équipement est testée ou déterminée in situ (ou en place) à l'aide de dispositifs spéciaux, tels que les capteurs de vibration, pour détecter les changements qui peuvent indiquer des dommages ou dégradation. Surveillance des processus basée sur des mesures de vibration sont effectuées en ligne, soit périodiquement, soit en continu [7].

1.7.2 Surveillance sonore ou acoustique

La surveillance sonore ou acoustique est une autre technique CM fréquemment utilisée dans la CBM ; elle a une forte relation avec la technique de surveillance de vibration. Cependant, il existe également une différence entre les deux. Alors que les capteurs de vibrations sont rigidement montés sur le composant impliqué pour enregistrer les motions locales, des capteurs acoustiques «écoutent» l'équipement. Comme la surveillance des vibrations, la surveillance sonore ou acoustique est exécutée en ligne, soit par des moyens périodiques ou continus[7].

1.7.3 Analyse d'huile ou surveillance de lubrifiant

Une autre technique CM est l'analyse d'huile ou la technique de surveillance des lubrifiants. Dans cette technique, l'état (qualité) de l'huile est évalué pour déterminer si l'huile convient ou non à d'autres utilisations. En même temps, les résultats de l'analyse de l'huile peuvent montrer les conditions d'usure des composants internes en contact avec l'huile, tels que le moteur arbres. Cette technique a deux objectifs généraux : la sauvegarde du pétrole qualité et la sauvegarde des composants impliqués. Une discussion détaillée des tests physiques et des procédures d'identification de la contamination qui constituent un programme normal d'échantillonnage périodique d'huile est donné par Newell (1999)[7].

1.7.4 Autres téchniques de CM

D'autres techniques de CM comprennent l'électricité, la température et la surveillance de l'état physique. La technique de surveillance électrique implique de mesurer les changements dans les propriétés de l'équipement, comme la résistance, la conductivité, la rigidité diélectrique et potentiel. Cette technique peut être utilisée pour détecter la détérioration de l'isolation électrique, barres de rotor de moteur cassées et stratification de stator de moteur court-circuitée. La technique de surveillance de la température est souvent appliquée pour l'identification de la panne et la surveillance des composants électriques et électroniques. La surveillance de l'état physique se concentre sur l'identification des changements physiques des matériaux, tels que les fissures et la corrosion. Cette technique est généralement réalisée hors ligne et elle est surtout populaire dans l'industrie de la construction. Une autre technique CM qui est moins bien connue par rapport à d'autres techniques est la surveillance des performances. Cette technique prédit essentiellement les problèmes en surveillant les changements de certains paramètres, tels que la pression, le débit et la consommation électrique[7].

1.8 Déscription du système CBM

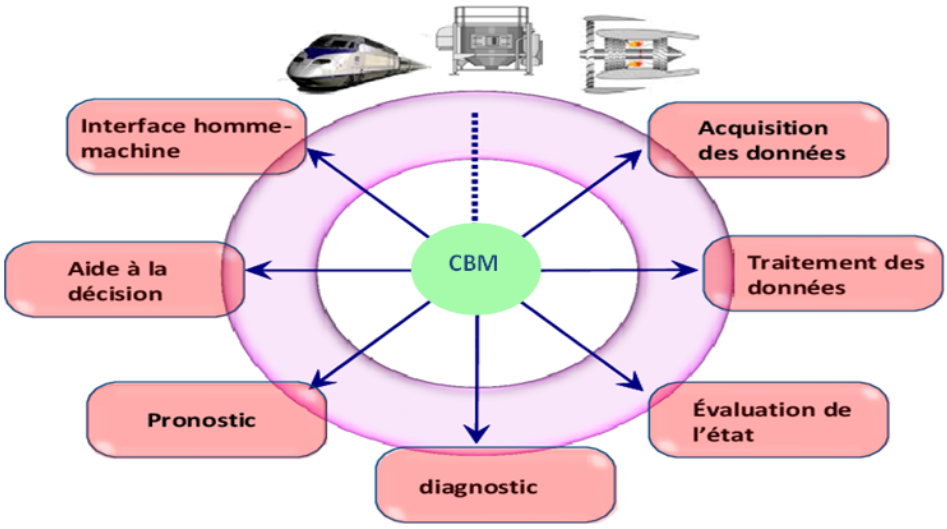


FIGURE 1.8 – procédure d'approche CBM.

1.8.1 L'acquisition des données

C'est le processus de collecte de signaux provenant de sources de mesure, telles que des capteurs connectés aux composants / sous-systèmes critiques, et la numérisation des signaux pour le stockage, l'analyse et la présentation sur les ordinateurs personnels (PC). Ce processus est une étape essentielle pour la mise en œuvre de CBM et peut affecter la qualité des décisions finales. Généralement, les données collectées à partir du composant critique peuvent être classées en deux types principaux : données sur les événements et données de surveillance de l'état[3].

1. **Données sur les événements :**Inclues des informations qualitatives sur la composante surveillée, telles que la description de l'installation, la panne, la révision, les causes etc., et la description de ce qui a été fait pour réparer la défaillance et la gravité de la réparation [3].

2. **Données de surveillance de l'état :**Les mesures sont liées à l'état de santé /l'état de l'actif physique. Les données de surveillance d'état sont très polyvalentes. Ils peuvent être des données de vibration, des données acoustiques, des données d'analyse d'huile, de température, de pression, de l'humidité, des données météorologiques ou environnementales, etc [3].

1.8.2 Le traitement des signaux/données

Analyse et interprète les signaux afin d'extraire des informations caractérisant le comportement du système, soit dans le domaine temporel et/ou fréquentiel [10]. Cette étape consiste à pré-traiter les données reçues afin d'éliminer les valeurs absurdes, les bruits et les vides afin de limiter l'impact des mauvaises mesures sur l'évaluation de l'état de santé du système[9].

1.8.3 Evaluation de l'état :

Sera obtenue à partir de ces caractéristiques et permettra à l'aide de comportement nominal de détecter les différentes anomalies possibles [10]. Parmi les technologies utilisées, la plus simple est la création, pour chaque capteur, de seuils dont le franchissement signale une valeur anormale. Une autre méthode basée sur la Physics of Failure, autrement dit, utiliser des modèles théoriques afin de représenter le système et détecter les dégradations par des techniques statistiques.Une troisième méthode consiste à utiliser les techniques de reconnaissance de modèle basées sur le ML(machine learning).Ce choix nécessite d'alimenter le système de détection avec un jeu de donnée d'apprentissage suffisamment complet et précis pour lui permettre de définir un fonctionnement nominal et de reconnaître les anomalies[9].

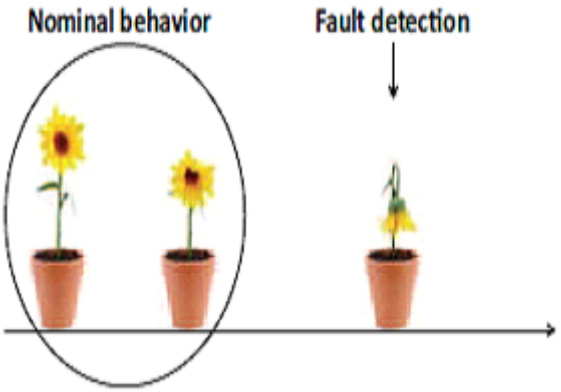


FIGURE 1.9 – Illustration de la détection des défauts [3]

1.8.4 Le diagnostic

Le diagnostic permet de définir la panne rencontrée à partir des anomalies détectées.les technologies les plus utilisées pour le diagnostic dans le CBM sont les réseaux bayésiens et la classification basée sur du ML(machine learning), en particulier les réseaux de neurones et le Support Vector Machine (SVM)[9].

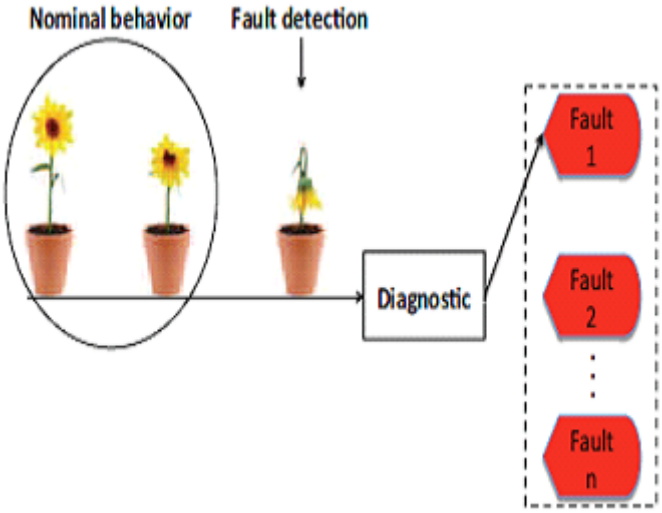


FIGURE 1.10 – Illustration de diagnostic
[3]

1.8.5 Le pronostic

C'est le processus d'estimation du temps restant(en englais remaining useful life RUL) pour un système ou un composant avant échec. Les pronostics sont utilisés par l'industrie pour gérer les risques résultant de panne d'équipement. Jusqu'à présent, il est toujours basé sur l'expérience des ingénieurs de maintenance. Cependant, la prise de décision humaine n'est pas toujours suffisamment fiable lorsqu'il s'agit des équipements complexes. Par conséquent, ces dernières années, un nombre important de recherches été entrepris pour développer des modèles pouvant être utilisés pour réduire la dépendance de l'industrie sur les individus. Cela peut être fait en effectuant une évaluation de la santé et une estimation de RUL pour les composants surveillés afin de planifier à l'avance les actions de maintenance requises [3].

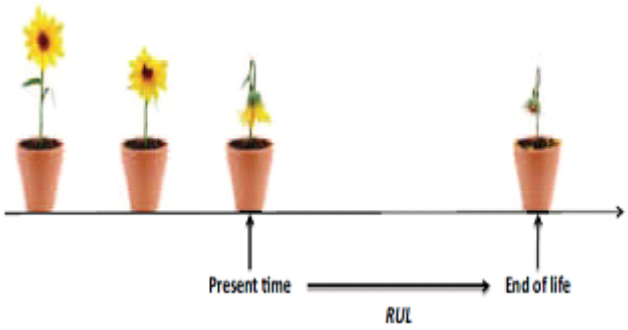


FIGURE 1.11 – Illustration de pronostic
[3]

1.8.6 L'aide à la décision

L'aide à la décision consiste à utiliser toutes les informations recueillies sur l'état du système surveillé pour choisir les actions de maintenance optimales. Il comprend des techniques de planification pour bien planifier les activités de maintenance. Par exemple, les informations sur la dégradation de la machine, c'est-à-dire la RUL prédite, l'état de santé estimé et les incertitudes correspondantes, produites par cette méthode peuvent être utilisées comme entrée pour la routine de prise de décision de maintenance [3].

1.8.7 L'interface homme machine

L'interface homme-machine est un support qui gère l'interaction entre le système PHM et l'utilisateur. Il gère également les interactions entre les différentes couches PHM [3].

1.9 Application de la maintenance conditionnelle

- Utilisée pour détecter le dysfonctionnement.
- Détection de panne de composant.
- Utilisée pour évaluer les techniques formalisées.

Le processus CBM peut être appliqué pour maintenir les activités dans toutes les industries, y compris :

- Systèmes d'armes du DoD.
- Moteurs à réaction.
- Générateurs d'éoliennes.
- Moteurs diesel marins
- Compression de gaz naturel.
- Fabrication de cartes de circuits[55].

Conclusion

Nous venons de présenter dans ce présent chapitre l'essentiel des concepts actuels attachés à la maintenance, en précisant les principaux types associés et on s'intéresse à la maintenance conditionnelle, ses types et ses différentes techniques et étapes. Dans le deuxième chapitre on s'intéressera à l'état de l'art du RUL.

CHAPITRE 2

LA DURÉE DE VIE UTILE RESTANTE

Introduction

La durée de vie utile résiduelle (RUL) peut être simplement définie comme une prédiction du temps restant pendant lequel un système est capable d'exécuter sa fonction prévue ; elle est mesurée à partir de l'instant présent jusqu'à la défaillance finale. En général, le but de la prévision de la RUL est d'influencer la prise de décision pour le système [12].

Dans ce chapitre nous allons commencer par présenter le RUL, ensuite nous étudierons les différentes approches de résolution de problème avec les avantages et les inconvénients de chacune.

2.1 Définition de la durée de vie utile RUL

La durée de vie utile restante (RUL) d'un actif ou d'un système est définie comme le temps restant entre l'heure actuelle et la fin de sa vie utile . Le sens de cette vie utile et sa gestion varie selon le domaine étudié et peut généralement être définie par le concepteur, ingénieurs et utilisateurs de l'actif. Pour illustrer le concept général de RUL, dans la figure 2.1 montre l'évolution de la détérioration d'un système donné. Dans ce cas, le RUL est le temps entre l'heure actuelle, qui correspond à la condition A, et le temps dans lequel une condition maximale acceptable de détérioration B est atteinte. Ce dernier peut correspondre au moment de la panne (c'est-à-dire pour 100 des la détérioration) ou à un seuil prédéfini dans une valeur inférieure. Par conséquent, on peut définir une durée de vie utile depuis le 'début de la vie 'jusqu'à la défaillance ou jusqu'au moment de la condition B. En général, une prédiction RUL appropriée dans A est utile pour effectuer des activités

de maintenance de A à B économiser ressources et améliorer les processus [12].

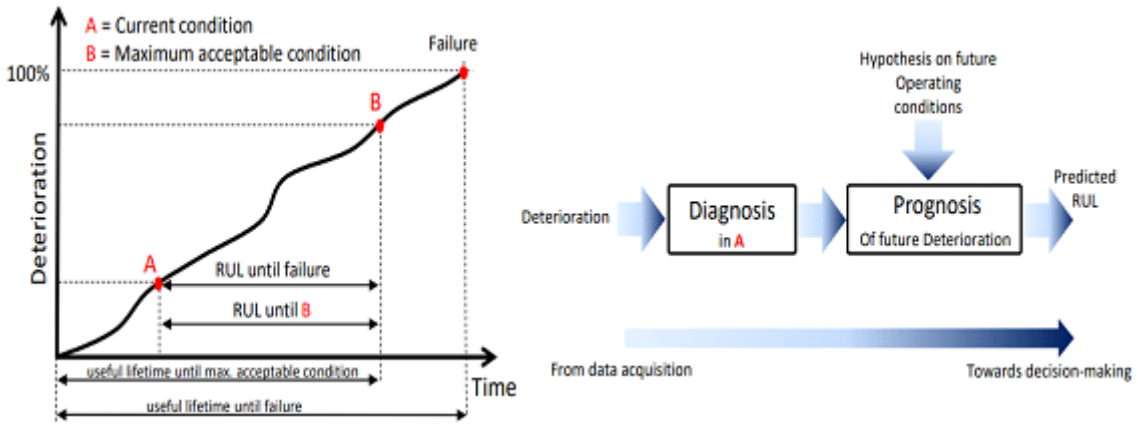


FIGURE 2.1 – Durée de vie utile restante(RUL), son concept et son utilisation [12]

2.2 La prédiction de RUL

La prédiction de la durée de vie utile restante (RUL), également connue sous le nom de pronostic RUL, est sujet étudié en PHM de ressources pour l'automatisation et la mécanique. Il peut être défini comme une estimation du temps restant entre l'instant courant et le temps de panne (ou un seuil d'une condition maximale acceptable) ; cette prédiction est généralement donnée en termes de probabilité. La durée de vie utile restante (RUL) d'un actif ou d'un système est définie comme le temps restant entre le moment actuel et la fin de sa vie utile [12].

Le RUL est défini comme une variable aléatoire conditionnelle dans :

$$RUL = t_f - t | t_f > t; Z(t) \tag{2.1}$$

où t_f est une variable aléatoire représentant le moment de la défaillance qui doit être caractérisée, t est le temps actuel et Z (t) est le profil de la condition passée (y compris l'environnement) données) et / ou les connaissances existantes sur l'utilisation future du système jusqu'à t. [12]

2.3 Classification des approches des pronostics

2.3.1 Selon la littérature

Dans la littérature, de nombreuses méthodes et outils de pronostic de défaillance ont été proposés. La classification des différentes approches permet de cartographier les travaux

dans ce domaine et de mettre en valeur les points forts et les inconvénients de chaque modèle.13 Nous avons présenté les différentes classifications des approches de pronostic dans la figure suivant14 :

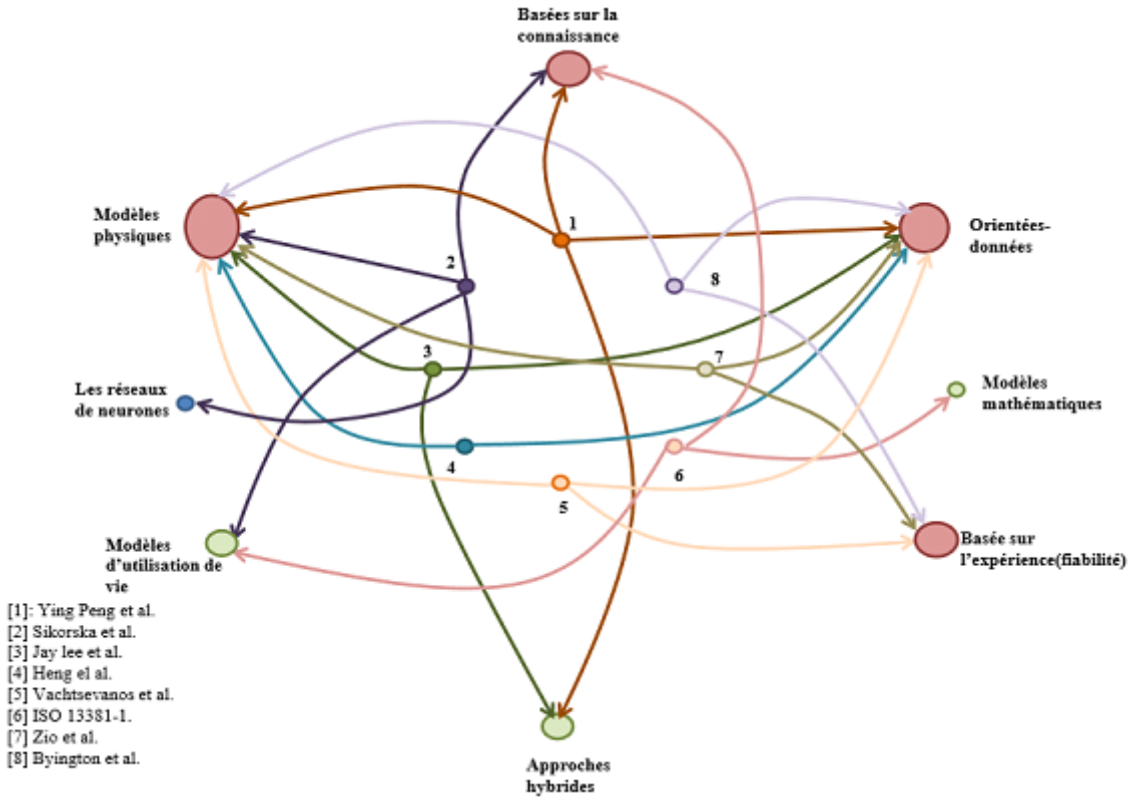


FIGURE 2.2 – classification des approches selon la littérature.
[14]

Les cercles représentant les classes sont d'autant plus grands que les classes sont citées. Nous retiendrons de cette cartographie les classes les plus représentées : modèles physiques, guidées par les données et basées sur la connaissance[14].

2.3.1.1 Pronostic basé sur l'expérience

Les approches de pronostic basées sur l'expérience se fondent sur la formalisation des mécanismes de défaillance des systèmes par modèles probabilistes (loi de durée de vie, processus markoviens ou non-markoviens) construits par connaissance à priori, par retour d'expérience ou par jugement d'expert. Cette méthode est principalement utilisé dans la situation où il n'y a aucune connaissance disponible sur la nature physique du système et où aucun dispositif de suivi de l' état de dégradation n'est opérationnel[13].

2.3.1.2 Pronostic guidé par les données

Les approches basées sur les données s'appuient sur l'évolution d'indicateurs de dégradation d'un système issus de données (entrées/sorties). Ces données peuvent être obtenues en ligne par le système de surveillance ou être issues de campagnes de mesures des périodes d'utilisation précédentes ou sur des équipements similaires. Ce type de pronostic se base sur l'hypothèse que les caractéristiques statistiques des données sont relativement inchangées à moins qu'un défaut de fonctionnement ne se produise dans le système[13].

2.3.1.3 Pronostic basé par les modèles

Les approches basées par les modèles se fondent sur un modèle dynamique représentant le comportement du système et intégrant le mécanisme de dégradation dont l'évolution est modélisée par une loi physique déterministe a laquelle peut s'ajouter un aléa intrinsèque au phénomène ou de type bruit de mesure par exemple [13].

2.3.2 Selon la communauté de CBM

La classification la plus consensuelle dans la communauté CBM est de considérer les trois approches suivantes : les méthodes guidées par les données, les méthodes basées sur des modèles physiques et les méthodes hybrides, comme le montre la figure suivant. Toutefois, on définira dans l'approche orientée données, 3 types de démarches : une statisticienne, une liée aux outils d'intelligence artificielle et plus spécifiquement à l'apprentissage automatique et une orientée connaissance [10].

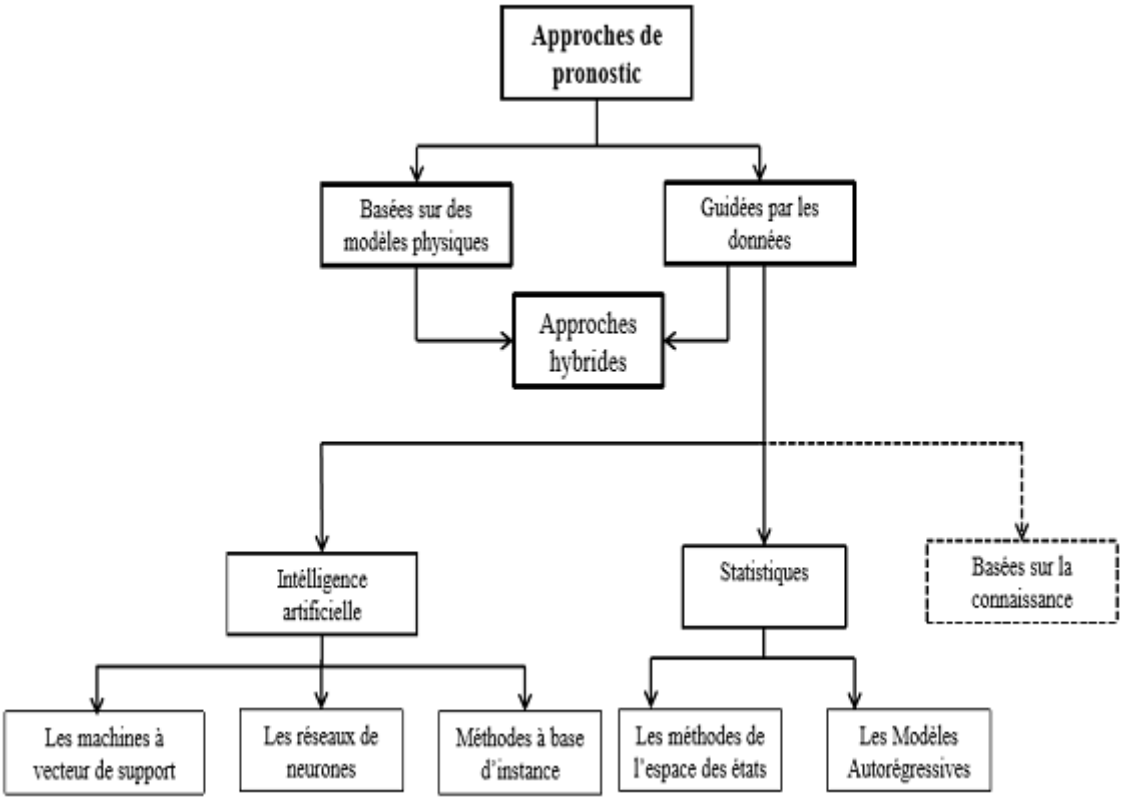


FIGURE 2.3 – classification des approches de pronostic selon la communauté CBM. [10]

2.3.2.1 Les approches basées sur les modèles physiques

Les approches basées sur des modèles physiques ou basées sur la physique de défaillance construisent des modèles analytiques qui sont directement liés aux processus physiques influençant la santé des composants. Ces approches nécessitent des connaissances sur les lois physiques (mécanique, chimie, électricité, hydraulique etc). Le modèle physique peut être décrit par des systèmes dynamiques tels que les équations non-linéaires, les équations différentielles, la représentation d'état, etc.

Le principe du pronostic basé sur un modèle physique est résumé à la figure 3.4. Le comportement du système est représenté par un modèle analytique. Ensuite, un test de cohérence est effectué en comparant les données capteurs issues du système réel et les sorties du modèle analytique. Les résidus générés sont évalués. En présence d'un dysfonctionnement, les résidus dépassent un seuil de détection de défauts [10].

L'estimation du temps restant avant défaillance est basée sur la projection de comportement du système et de sa dégradation dans le future. Les modèles physiques traitent par exemple des problèmes d'usures de matériau, de croissance de fissure, de cassure par fatigue et corrosion.[14]

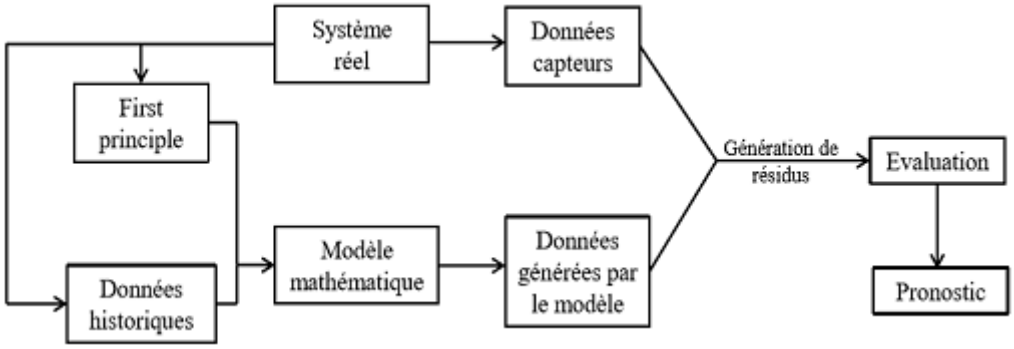


FIGURE 2.4 – organigramme des approches basées sur le modèle physique. [10]

2.3.2.2 Les approches guidées par les données

Les approches guidées par les données cherchent à extraire à partir d'un historique de données de surveillance des modèles d'évolution du fonctionnement du système surveillé allant jusqu'à sa dégradation. Ces approches comportent deux phases. Une première **h**ors ligne est dédiée à la compréhension et l'apprentissage du comportement de la dégradation. Puis, une deuxième phase **e**n ligne estime l'état de santé courant du système et prédit sa durée de fonctionnement avant défaillance. Elles peuvent être classées en deux catégories : les approches statistiques et issues de l'intelligence artificielle [10].

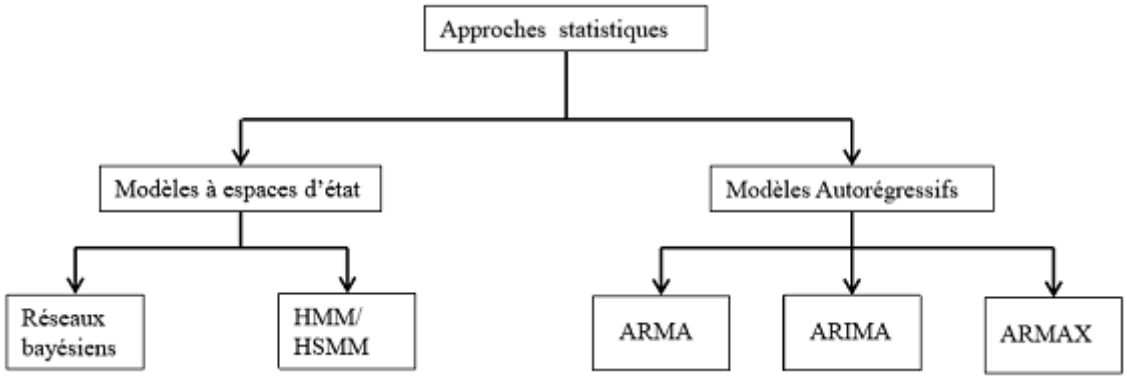


FIGURE 2.5 – La classification des approches statistiques. [10]

1. **Les statistiques multidimensionnelles** Nous passons en revue les méthodes les plus largement utilisées en pronostic, qui sont également illustrées à la figure 2.5

   a. **Les Modèles Auto-régressifs :**Les modèles ARMA (modèles auto-régressifs et moyenne mobile), ARIMA (modèles moyenne mobile auto-régressif intégré) et ARMAX sont les méthodes de référence dans la modélisation des séries temporelles . Les modèles ARMA et ARMAX ne sont utilisés que pour des données stationnaires. Ils sont améliorés par l'application d'une opération d'intégration

dans le modèle ARIMA. Les modèles ARMA ont été utilisés afin d'ajuster les données de vibration des roulements des boites de transmission des hélicoptères, pour prédire les tendances des turbines à flux, afin d'estimer la durée de vie utile restante avant la défaillance d'un système de mouvement de porte d'ascenseur. Les modèles ARIMA sont utilisé pour prédire les caractéristiques vibratoires des machines rotatives [14].

Les modèles régressifs ne nécessitent pas un historique de dégradation et doivent être évalués de manière récursive jusqu'à atteindre un certain seuil, ce qui engendre une accumulation d'erreur systématique et détériore la performance du prédicateur[10].

b. **Les modèles à espace d'état**

i. **Les modèles de Markov cachés :** ou HMM ont été appliqués en pronostic mais ne sont pas adaptés à représenter une structure temporelle. Ils ont été remplacés par les HSMM qui comptent une composante temporelle dans la structure de HMM , ont développé une approche basée sur les HSMM afin de prédire le RUL des actionneurs entraînés (motorisés) par un moteur asynchrone. Une méthode a été proposé de combiner les HSMM avec une méthode Monte Carlo séquentielle. Ces outils calculent les probabilités de transition entre les états de santé et leur durée alors que la méthode Monte Carlo définit la relation probabiliste entre les états de santé et les observations de l'équipement surveillé. Les HSMM ont aussi été utilisés pour le pronostic des défaillances d'hélicoptères. Un autre modèle HSMM a été proposé pour le diagnostic et pronostic de UH-60Blackhawk (un hélicoptère utilitaire moyen de l'armée américaine). Le HSMM formé peut être utilisé pour diagnostiquer et estimer l'état courant de santé de l'équipement, et sera à la base de l'estimation du RUL à l'aide d'une équation récursive composée de la durée dans les états et les paramètres du modèle estimé. ont étudié l'applicabilité des HSMM pour prédire le RUL des arbres à cames utilisés dans les hélicoptères[10].

ii. **Les réseaux bayésiens :** sont des modèles graphiques directs, ils sont issus de la fusion de la théorie des graphes et des probabilités. Le modèle RB a été étendu à un modèle réseau bayésien dynamique appliqué au pronostic grâce à la modélisation des changements de système au fil du temps. Il est en mesure de surveiller, mettre à jour et prédire les états futurs du système étudié. Une approche bayésienne hiérarchique a été proposée afin de construire un modèle probabiliste d'estimation du RUL. D'autre part, une approche bayésienne a été appliqué dans le but de mettre à jour la

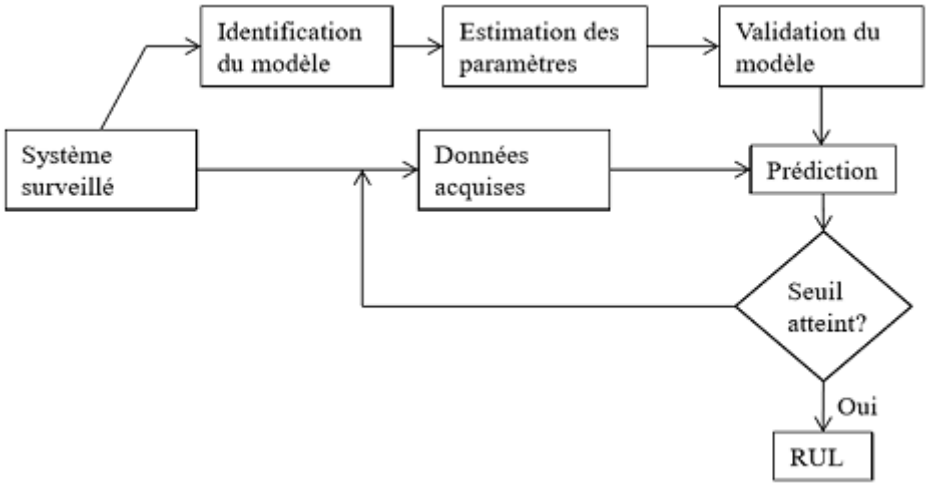


FIGURE 2.6 – Organigramme des approches auto-régressives pour la prédiction du RUL. [10]

distribution des paramètres stochastiques du modèle exponentiel décrivant le processus de dégradation des paliers d'essai. La méthode de pronostic basée sur les RBD à été utilsé et étudier afin de prédire la durée de vie utile restante des trépans d'une machine de forage vertical. Le RUL est considéré comme un état caché et est déduit des données invisibles en utilisant le modèle de prédiction ainsi que des règles d'inférence[14].

2. **L'intelligence artificielle ou apprentissage automatique :** Nous recensons les trois types de méthodes les plus utilisées en pronostic : les réseaux de neurones artificiels (RNA), les méthodes de régression à vecteurs de support (SVR), et les méthodes à base d'instances (IBL)[10].

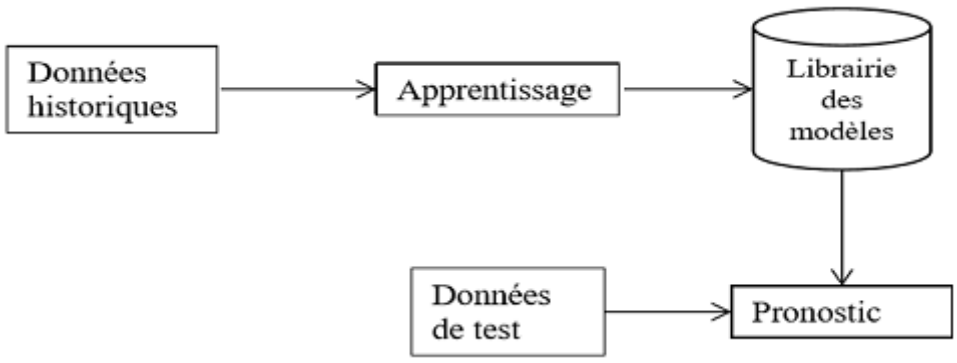


FIGURE 2.7 – Approche d'apprentissage automatique. [10]

a. **Les Réseaux de neurones :** la plupart des approches d'apprentissage automatique développées en pronostic sont les réseaux de neurones artificiels et leurs variantes. En fait, les RNA sont connus pour leur capacité à modéliser

des systèmes complexes, multidimensionnels et non-linéaires sans la nécessité d'une compréhension physique du comportement du système. Un RNA est une fonction d'approximation non linéaire à multiples entrées et sorties et peut être utilisé à différentes fins parmi les quelles le pronostic. Selon les entrées du modèle RNA, le dernier a la souplesse nécessaire pour exécuter différentes tâches, comme la prédiction du temps restant avant la défaillance par exemple [10].

b. **Les machines a vecteurs de support :**les machines à vecteurs de support (SVM) ont été développées par Vapnick et ses collègues. Les SVM sont à la base d'un système d'apprentissage qui représente un espace de caractéristiques d'entrée dans un espace de plus grande dimension. Cela est appelé l'astuce de noyaux (kernel trick) et donne aux SVM la capacité à traiter des données non-linéaire. Ces méthodes ont une bonne généralisation et ont deux types d'application : la classification à vecteurs de support, utilisée pour résoudre le problème de diagnostic, et la régression à vecteurs de support, utilisée pour résoudre le problème de pronostic[10].

c. **Méthodes à base d'instances :**les approches à base d'instances prennent appui sur un ensemble de données, sur lequel des techniques d'apprentissage sont appliquées. Elles se basent sur le principe : les expériences acquises lors de la résolution d'un problème peuvent être utilisées pour résoudre des cas similaires". Un algorithme souvent utilisé pour ce type d'approches est les k plus proches voisins. Une instance est généralement représentée par un vecteur dans un espace euclidien de dimension n et l'objectif est de trouver les instances similaires. L'avantage de ces méthodes est l'apprentissage incrémental. Quand un problème est résolu avec succès, l'expérience est conservée afin de résoudre de nouveaux problèmes similaires. Les méthodes à base d'instances font partie du raisonnement à partir de cas (RàPC) qui est un paradigme de raisonnement par analogie. Les nouveaux problèmes sont résolus en se basant sur la connaissance spécifique acquise lors de la résolution d'anciens problèmes [10].

2.3.2.3 Approches hybrides

Les approches physiques sont généralement accompagnées d'un module en ligne qui permet d'estimer et mettre à jour les paramètres du modèle à partir des données de surveillance acquises. Ces approches combinant les modèles physiques, hors ligne et les modèles guidées par les données, en ligne sont dites hybrides. Par exemple, l'estimateur bayésien récursif est une approche importante pour intégrer des modèles du système avec les données de capteurs et activer les modèles dits hybrides. Les estimateurs bayésiens récursifs

ont deux procédures communes à chaque itération : prédire et actualiser. Deux variantes importantes de l'estimateur bayésien sont le filtre de Kalman et le filtre particulaire. Le filtre de Kalman est approprié pour les modèles espace-état linéaires avec un bruit gaussien tandis que le filtre particulaire peut être utilisé pour les modèles non linéaires avec du bruit non gaussien[10].

Conclusion

Dans ce chapitre nous avons présenté le RUL avec une étude de ses différentes approches qui existent dans la littérature, ainsi que leurs avantages et leurs inconvénients.

Un état de l'art sur les deep learning et quelques travaux qui existent dans le domaine en utilisant l'apprentissage profond seront présenté dans le chapitre suivant.

CHAPITRE 3

LES RÉSEAUX DE NEURONES ET LES DEEP LEARNING

Introduction

Les réseaux de neurones en générale et les réseaux de neurones profond (en anglais deep learning) en particulier constituent un outil de l'intelligence artificielle. Ce sont un ensemble de méthodes d'apprentissage automatique tentant de modéliser avec un haut niveau d'abstraction des données grâce à des architectures articulées de différentes transformations non linéaires. Ils offrent des solutions très performantes pour le diagnostic et le pronostic des procédés industriels. Ces réseaux se manifestent sous forme de plusieurs architectures. L'application de chacune de ces architectures présente des points forts et des points faibles[17]

Ce chapitre présente ces deux concepts, il contient trois parties. Dans la première partie nous allons introduire les réseaux de neurones en générale et dans la deuxième partie nous aborderons les deep learning. Nous concentrons sur le LSTM et ses différents modèles, sujet de notre approche. Nous terminerons dans la troisième partie par une revue de quelques travaux de RUL avec les deep learning.

3.1 Les réseaux de neurones

3.1.1 Définition d'un réseau de neurone

Le réseau de neurones artificiels (Artificial Neural Network - ANN) fut introduit comme un modèle rudimentaire du traitement de l'information dans le cerveau humain. Ainsi, la structure élémentaire d'un ANN est un réseau de petits nœuds de calcul reliés entre eux par des liens dirigés et pondérés (Fig. 2.1). Les nœuds représentent les neurones et les liens pondérés représentent la force des connections synaptiques reliant les neurones entre

eux. Dans cette représentation, le neurone peut alors être un sommateur des potentiels des signaux synaptiques qui lui parviennent, et qui transmet à son tour une information basée sur cette somme via une fonction de transfert de préférence non linéaire. Un ANN est activé en injectant des données au niveau de tout ou partie des nœuds puis en propageant l'information en suivant les liens pondérés. Une fois l'information propagée, on peut collecter les niveaux d'activation de tout ou partie des nœuds et les utiliser comme commande d'un système, comme prédiction ou encore comme classification [18].

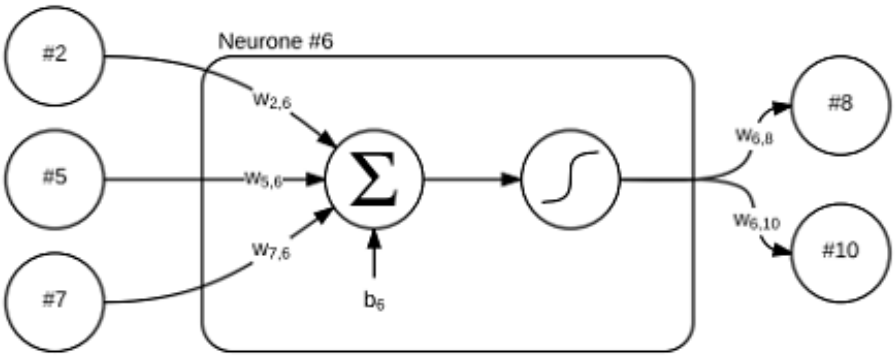


FIGURE 3.1 – Structure interne d'un nœud de calcul dans le cas d'un neurone artificiel sommateur.Tout d'abord, la somme pondérée des entrées du nœud est calculée puis on applique à cette somme une fonction de transfert non-linéaire. La sortie est alors communiquée aux autres nœuds de calculs
[18].

3.1.2 La fonction d'activation

La fonction d'activation est la fonction de transfert qui relie la sommation pondéré au signal de sortie. Il y'a plusieurs types de fonction d'activation, parmi lesquelles nous trouvons [17] :

3.1.2.1 La fonction sigmoïde

Cette fonction est l'une des plus couramment utilisées. Il est borné entre 0 et 1, et il peut être interprété stochastiquement comme la probabilité que le neurone s'active, et il est généralement appelé la fonction logistique ou le sigmoïde logistique. [19]

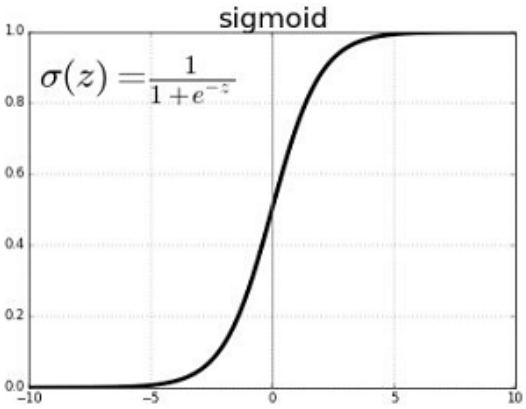


FIGURE 3.2 – représentation graphique de la fonction sigmoïde
[19]

3.1.2.2 La fonction RELU

La fonction RELU est probablement la plus proche de sa correspondante biologique. Cette fonction est récemment devenue le choix de nombreuses tâches (notamment en computer vision). Comme dans la formule ci-dessus, cette fonction renvoie 0 si l'entrée z est inférieure à 0 et retourne z lui-même si il est plus grande que 0[19].

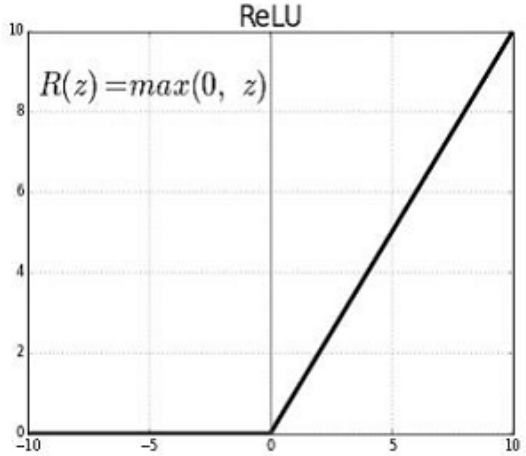


FIGURE 3.3 – représentation graphique de la fonction Relu
[19]

3.1.2.3 La fonction linear

Comme on l'a vu, la fonction est une ligne ou linéaire. Par conséquent, la sortie des fonctions ne sera confinée entre aucune plage [20].

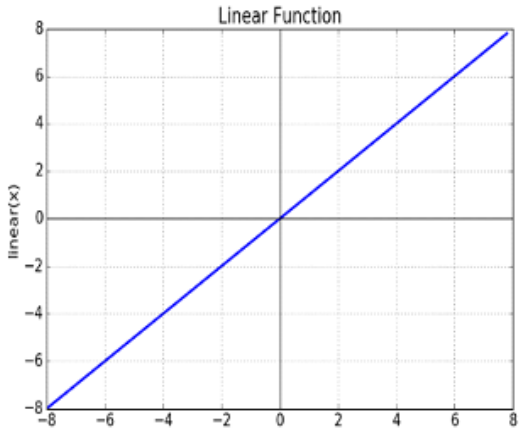


FIGURE 3.4 – représentation graphique de la fonction linear
[20]

3.1.3 L'architecture des réseaux de neurones

3.1.3.1 Perceptron simple

Le réseau le plus simple (Rosenblatt, 1958,Minsky et Papert, 1969) est constitué d'un seul neurone, avec n entrées et une seule sortie. L'algorithme d'apprentissage de base du perceptron analyse la configuration d'entrée (pattern) et les variables de pondération à travers le synapses - décide à quelle sortie est associée la configuration. Ce type d'architecture présente la limitation majeure de pouvoir résoudre uniquement des problèmes séparables linéairement [21].

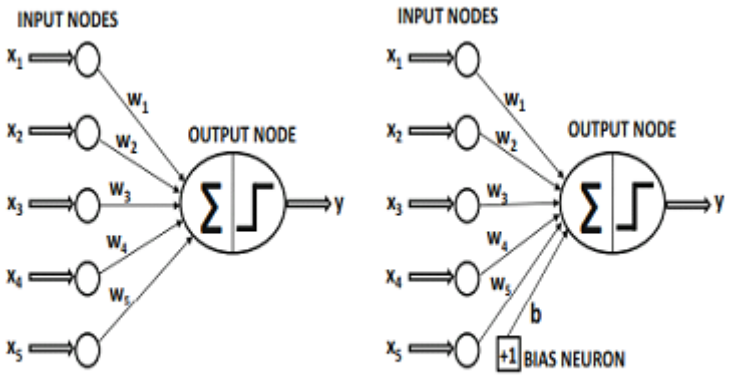


FIGURE 3.5 – perceptron simple avec et sans le bias.
[22]

3.1.3.2 Perceptron multicouches

Le modèle du Perceptron Multi-Couches (MLP)a été proposé en 1986 par Rumelhart et al. Le neurone est un sommateur pondéré suivi d'une non-linéarité de type tangente

hyperbolique . Les neurones sont organisés en couches . Il n'y a pas de connexions à l'intérieur d'une même couche. Chaque neurone reçoit ses entrées de la couche Directement inférieure. Un neurone seuil (neurone sans entrées et dont la sortie vaut toujours 1), est souvent disposé sur chaque couche (sauf en sortie), afin d'améliorer les capacités du réseau. En effet, en leur absence, la présentation d'un vecteur nul à l'entrée du réseau conduirait systématiquement à des sorties nulles, quelles que soient les valeurs des poids [23].

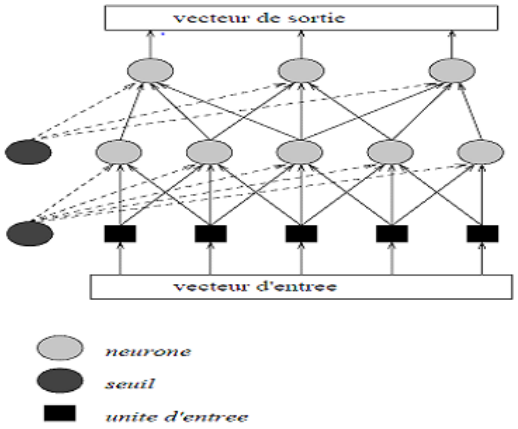


FIGURE 3.6 – perceptron multi couches
[23]

3.1.4 Apprentissage d'un réseau de neurone

L'apprentissage est une étape très importante du développement d'un réseau de neurone durant laquelle le comportement du réseau est modifié itérativement jusqu'à l'obtention du comportement désiré, et ce par l'ajustement des poids (connexion ou synapse) des neurones à une source d'information bien définie . L'apprentissage consiste également à extraire des conformités parmi les données utilisées pour 1' entrainement du réseau mais l'objectif essentiel de l'apprentissage est la résolution du problème par la prévision, la classification ,la regression etc . Pour un RNA, l'apprentissage peut être regardé également comme étant le processus de la mise à jour des poids (connexion) au sein du réseau dans le but d'ajuster la réponse du réseau à l'expérience et aux exemples. Il existe plusieurs types de règles d'apprentissage qui peuvent être rassemblées en deux catégories qui sont l'apprentissage supervisé et l'apprentissage non supervisé [24].

3.1.4.1 Apprentissage supervisé

Dans ce type d'apprentissage, on cherche à imposer au réseau un fonctionnement donné en forçant les sorties des réseaux à prendre des valeurs bien données (choisie par l'opérateur) et ce en modifiant les poids synaptiques. Le réseau se comporte alors comme

un filtre dont les paramètres de transfert sont ajustés à partir des couples entrée-sortie présentés . L'adaptation des paramètres du réseau s'effectue à partir d'un algorithme d'optimisation, l'initiation des poids synaptiques étant le plus souvent aléatoire [24].

En effet, grâce à l'apprentissage supervisé, il est possible de résoudre deux grands types de problèmes, les problèmes de classification et les problèmes de régression.

1. **régression("Regression") :**permet de prédire la valeur de la variable dépendante pour une valeur donnée de la variable indépendante. elle est utilisée lorsque la valeur cible à prédire est continue. Il existe 2 types :
   - **R**égression linéaire simple :permettant de modéliser la relation linéaire entre 2 variables quantitatives. Le modèle de régression simple est :
     $y_i = ax_i + b + \varepsilon_i$
   - **R**égression linéaire multiple : Utilisée chaque fois qu'une variable observée,dite variable dépendante, doit être exprimée en fonction de 2 ou plusieurs autres variables observées, dites indépendantes ou mieux explicatives. Le cas le plus simple est celui où les variables explicatives sont des variables non aléatoires, leurs valeurs étant toutes choisies a priori de façon arbitraire (dose d'un médicament...). Le modèle est de la forme :
     $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \varepsilon$
     où p est le nombre de variables indépendantes.
2. **Classement ,classification ou catégorisation ("Classification") :** lorsque la valeur cible à prédire est discrète.

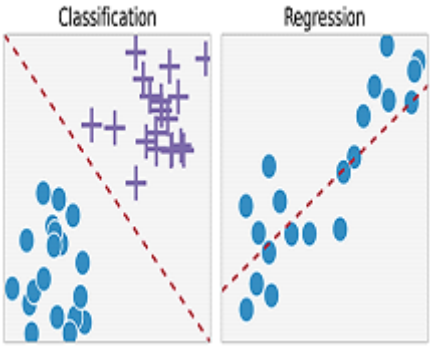


FIGURE 3.7 – Illustration de la différence entre classification linéaire et régression linéaire.

3.1.4.2 Apprentissage non supervisé

Les algorithmes de ML non supervisé n'ont pas besoin de données étiquetées. Ils passent au crible les données sélectionnées par une équipe de data science pour rechercher des schémas pouvant servir à regrouper des points de données en sous-ensembles. Les algorithmes d'apprentissage non supervisé conviennent aux tâches suivantes :

- **Regroupement ou « clustering » :**division de l'ensemble de données en groupes en fonction de la similarité.
- **Détection d'anomalies :** identification de points de données inhabituels dans un ensemble de données.
- **Recherche d'associations :**identification dans un ensemble de données de séries d'éléments souvent associées.
- **Réduction de la dimensionnalité :**réduction du nombre de variables dans un ensemble de données [55].

3.2 Le deep learning

3.2.1 Définition

Les réseaux de neurones dits "profonds" (Deep Neural Networks en anglais) sont des MLP avec un nombre de couches supérieur à trois. Pendant longtemps les méthodes d'apprentissage pour ce type de réseaux de neurones acycliques ne permettaient pas de converger vers un réseau de neurones performant. Des avancées majeures sur les méthodes d'entraînement et le choix de la fonction de transfert Rectified Linear Unit (ReLU) qui minimise l'impact de la dilution du gradient dans les couches basses du réseaux ont permis d'utiliser des réseaux de neurones de plus en plus gros [18].

Il exist deux types de réseaux de neurones profonds :

— Les réseaux de neurones Feed forward.

— Les réseaux de neurones récurrents, utilisés pour les données séquentielles telles que le texte ou séries chronologiques(time series)[25].

3.2.2 la profondeur de l'apprentissage profond(the deep in deep learning)

L'apprentissage profond est un sous-domaine spécifique de l'apprentissage automatique : une nouvelle approche de l'apprentissage des représentations à partir de données qui met l'accent sur l'apprentissage de couches successives de représentations de plus en plus significatives. L'apprentissage approfondi n'est pas une référence à une quelconque compréhension plus approfondie obtenue par l'approche ; il représente plutôt cette idée de couches successives de représentations. Le nombre de couches qui contribuent à un modèle de données est appelé la profondeur du modèle. D'autres noms appropriés pour le champ auraient pu être l'apprentissage des représentations en couches et l'apprentissage des représentations hiérarchiques[26].

3.2.3 Domaine d'application

L'apprentissage profond en particulier a gagné en popularité ces derniers temps, inspiré par des réalisations notables dans la classification d'images , le traitement du langage naturel et l'apprentissage par renforcement . En incorporant des hypothèses architecturales sur mesure - ou des biais inductifs qui reflètent les nuances des ensembles de données sous-jacents, les réseaux de neurones profonds sont capables d'apprendre des représentations de données complexes , ce qui réduit le besoin d'ingénierie manuelle des caractéristiques et de conception de modèles. La disponibilité de frameworks de rétropropagation open-source a également simplifié la formation réseau, permettant la personnalisation des composants de réseau et des fonctions de perte Compte tenu de la diversité des problèmes de séries chronologiques dans divers domaines, de nombreux réseaux de neurones des choix de conception ont émergé [27].

3.2.4 Les types de réseaux de neurones profonds

3.2.4.1 Résaux de neurone FeedForward

Ce réseau de neurones est l'une des formes les plus simples d'ANN, où les données ou l'entrée se déplacent dans une direction. Les données passent par les nœuds d'entrée et sort sur les nœuds de sortie. Ce réseau neuronal peut avoir ou non les couches cachées. Tous simplement, il a une onde propagée en façade et aucune rétropropagation en utilisant généralement une fonction d'activation [28].

i. **Résaux de neurones convolutifs CNN** Les réseaux de neurones convolutifs sont similaires aux réseaux de neurones feed forward, où les neurones ont des poids apprenants et biases. Son application a été dans le signal et traitement d'image qui reprend Open CV dans le domaine de la vision par ordinateur. ConvNet sont appliqués dans des techniques comme le traitement du signal et techniques de classification d'images [29]. Les CNN sont très efficaces pour le traitement d'image, car une caractéristique peut se produire n'importe où dans l'image. Au fur et à mesure qu'une couche alimente sa sortie dans la couche suivante, les entités extraites peuvent devenir hiérarchiquement et progressivement plus complexes. Le processus d'optimisation des paramètres tels que les noyaux est appelé apprentissage, qui est effectué de manière à minimiser la différence entre les sorties et les ground truth labels grâce à un algorithme d'optimisation appelé rétropropagation(back propagation) et descente de gradient, entre autres [29].

   (a) **Les couches CNN** CNN est une construction mathématique généralement composée de trois types de couches (ou blocs de construction) : la convolution, le pooling et les couches fully connected[29].

— Fully connected : Couche classique de perceptron et dernière couche d'un réseau profond qui opère la discrimination finale entre par exemple des images à reconnaître. Les couches précédentes construisant, extrayant, des caractéristiques (features) de celles-ci.

— Convolution :opère une convolution sur le signal d'entrée en associant une réduction de dimension.

— Pooling :réduction de dimension en remplaçant un sous-ensemble des entrées (sous-image) par une valeur, généralement le max.

[30].

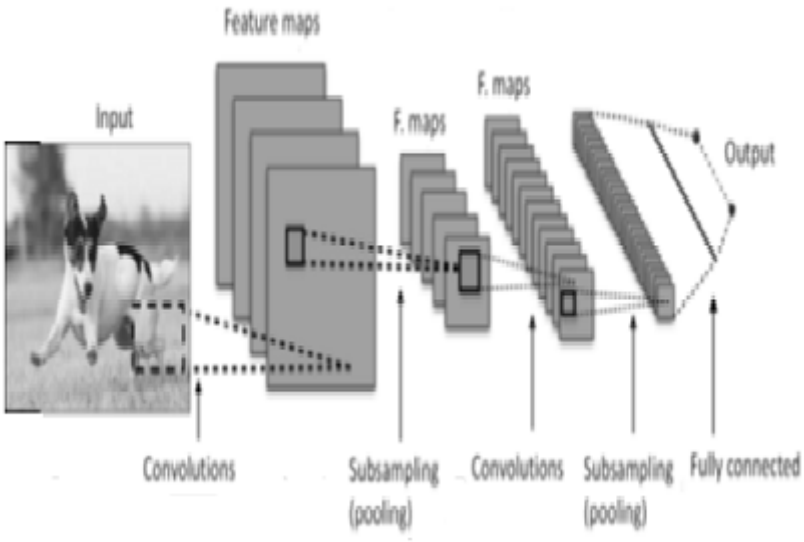


FIGURE 3.8 – Le schéma de CNN
[31]

(b) **L'algorithme de retropropagation :** L'algorithme d'apprentissage de rétro-propagation consiste dans un premier temps à circuler vers l'avant les données d'entrées jusqu'à l'obtention d'une entrée calculée par le réseau, puis la seconde étape est de comparer la sortie calculée à la sortie réelle connue (Rumelhart et al 1986). Les poids sont modifiés de telle sorte qu'à la prochaine itération, l'erreur commise entre la sortie calculée est minimisée, en prenant en considération la présence des couches cachées, l'erreur est rétro-propagé vers l'arrière jusqu'à la couche d'entrée tout en modifiant la pondération. Le processus est répété sur tous les exemples jusqu'au temps où l'on obtienne une erreur de sortie considérée comme négligeable [24].

3.2.4.2 Les réseaux de neurones Réccurents RNN

Contrairement à un réseau de neurone acyclique (Feed-Forward Neural Network - FFNN), un réseau de neurone récurrent (Recurrent Neural Network - RNN) est un réseau de neurones dont le graphe de connexion contient au moins un cycle. Un réseau neuronal récurrent est un algorithme d'apprentissage en profondeur conçu pour traiter une variété de tâches informatiques complexes telles que la classification d'objets, la détection de la parole et tout les domaines du traitement du langage naturel et les séries chronologiques. Les RNN sont conçus pour gérer une séquence d'événements qui se succèdent, la compréhension de chaque événement étant basée sur les informations des événements précédents [28].

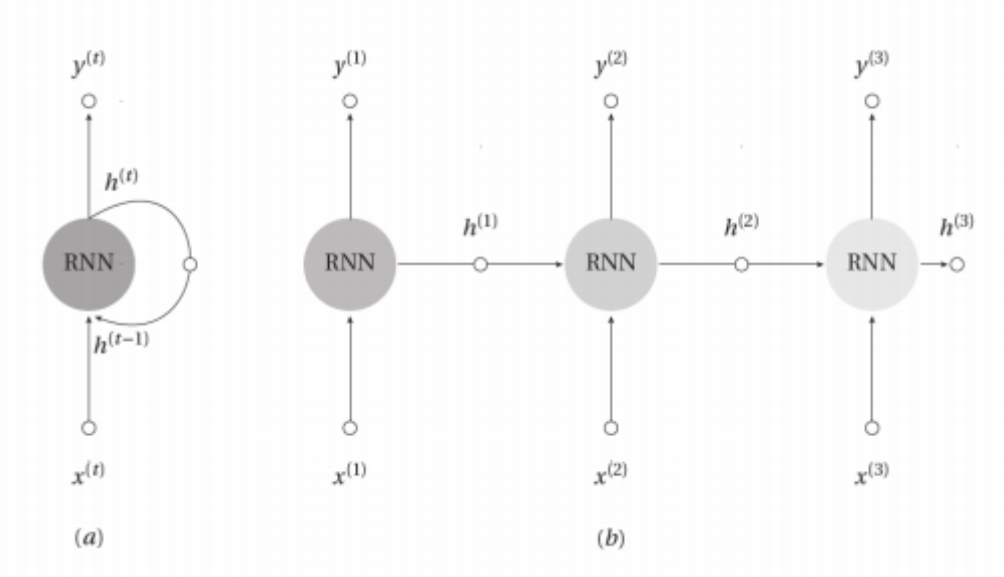


FIGURE 3.9 – Structure de RNN
[34].

i. **Les types des résaux RNN** Comme pour les FFNN, de nombreux types de RNN ont été développés au cours des 30 dernières années tels que les réseaux d'Elman , les réseaux de Jordan et les Echo State Networks . Au cours des dernières années, un type de RNN est devenu la norme grâce à ses excellentes performances sur des tâches aussi nombreuses que variées : les réseaux de neurones à base de cellules Long Short-Term Memory (LSTM). Parmi ces types de RNN nous citons quelques uns :

(a) **Les réseaux de Hopfeild :** Ce type de réseaux utilise un apprentissage non-supervisé, il est particulièrement utilisé dans la résolution de problèmes d'optimisation. Les réseaux de Hopfield sont considérés comme des réseaux totalement connectés et il n'y a aucune différenciation entre les neurones d'entrée et de sortie . Ce genre de réseaux opère comme une mémoire associative non-linéaire qui a la capacité de discerner un objet stocké dans un espace de données) [32].

(b) **LSTM :** Les cellules de mémoire à long terme court (LSTM) ont été introduites par Hochreiter et Schmidhuber (1997) et ont été créées afin de pouvoir apprendre les dépendances à long terme [25]. LSTM (de l'anglais : Long Short Term Memory) est l'une des topologies récurrentes de réseaux de neurones artificiels. Comme dans le cas des réseaux récurrents, un LSTM est composé de couches de neurones avec une connexion récurrente de sorte que l'état précédent du neurone dans un pas de temps précédent soit utilisé comme contexte pour formuler une sortie. Les unités de calcul de LSTM sont appelées cellules ou blocs de mémoire. Contrairement à d'autres RNNs, LSTM est une architecture spécialement conçue pour résoudre le problème du gradient.

Pour un neurone, le gradient est l'erreur relative à ce neurone. Il peut en quelque sorte être vu comme la contribution du neurone à l'erreur globale.Le principal défi auquel sont confrontés les RNNs est de savoir comment les former efficacement.

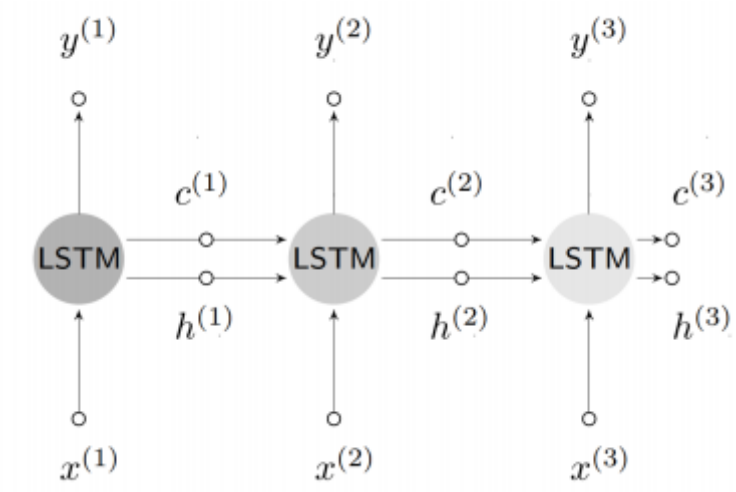


FIGURE 3.10 – Structure de LSTM
[34]

(c) **GRU :** L'unité récurrente fermée (GRU) peut être considérée comme une simplification du LSTM, qui n'utilise pas d'états de cellule explicites. Une autre différence est que le LSTM contrôle directement la quantité d'informations modifiées dans l'état caché en utilisant des portes d'oubli et de sortie séparées. D'autre part, un GRU utilise une seule porte de réinitialisation pour atteindre le même objectif [33].

(d) **JANET :** Ce type de réseaux semble répondre aux questions de savoir si toutes les portes d'un LSTM sont strictement nécessaire, fait déjà réfuté avec la proposition du Gated Recurrent Unit (GRU). De cette manière, un nouveau réseau appelé JANET qui ne garde que la porte de l'oubli et la cellule mémoire obtenant un réseau qui nécessite moins de puissance de calcul et est un modèle plus général [34].

3.2.5 Long short term memory

3.2.5.1 Définition

Une couche LSTM consiste en un ensemble de blocs connectés de manière récurrente, appelés blocs mémoire. Ces blocs peuvent être considérés comme une version différenciée des puces de mémoire dans un ordinateur numérique. Chacun contient une ou plusieurs cellules mémoire connectées de manière récurrente et trois unités multiplicatives - les portes d'entrée, de sortie et d'oubli - qui fournissent des analogues continus des opérations d'écriture, de lecture et de réinitialisation pour les cellules [35].

1. **poids de LSTM (weights) :** Une cellule de mémoire a des paramètres de poids pour l'entrée, la sortie, ainsi qu'un état interne qui est construit par l'exposition aux pas de temps d'entrée.
   - Poids d'entrée : Utilisé pour pondérer l'entrée pour le pas de temps actuel.
   - Poids de sortie : Utilisé pour pondérer la sortie du dernier pas de temps.
   - État interne : État interne utilisé dans le calcul de la sortie pour ce pas de temps.

   [35].
2. **portes de LSTM (Gates) :** Les portes sont la clé de la cellule mémoire. Ce sont également des fonctions pondérées qui régissent davantage le flux d'informations dans la cellule. Il y a trois portes :
   - Forget Gate : décide des informations à supprimer de la cellule.
   - Input Gate : décide des valeurs de l'entrée pour mettre à jour l'état de la mémoire.

— Output Gate : Décide ce qu'il faut sortir en fonction de l'entrée et de la mémoire de la cellule.

[35]

3. **Structure du modèle LSTM :**Le modèle consiste séquentiellement en une couche d'entrée, suivie d'une couche LSTM, d'une couche dropout et de la couche finale, la couche dense[34].

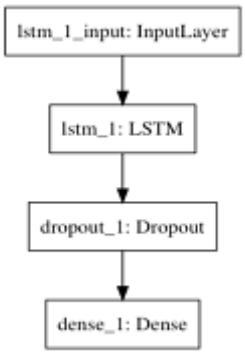


FIGURE 3.11 – La structure du modèle LSTM
[35]

— Couche lstm :Les couches Keras LSTM attendent une entrée sous la forme d'un tableau numérique de 3 dimensions (samples, time steps, features) où les samples sont le nombre de séquences d'entraînement, les time steps est la fenêtre de retour en arrière ou la longueur de la séquence et les features est le nombre d'entités de chaque séquence à chaque pas de temps.

— Dropout : Généralement, la formation des réseaux de neurones prend plus de temps si bien que le sur ajustement (overfitting) devient un problème sérieux avec l'augmentation du nombre de couches réseau. Le sur ajustement (overfitting) donne au modèle de prédiction d'excellentes performances dans les données d'entraînement, mais pas dans les données de test. Pour surmonter ce problème, dropout est adopté dans le LSTM pour empêcher la capture répétée des mêmes fonctionnalités (features).

[36]

— Dense :Une couche entièrement connectée qui suit souvent les couches LSTM et est utilisée pour générer une prédiction est appelée Dense ().

3.2.5.2 Les modèles du LSTM

Les différentes architectures de LSTM sont :

1. **Vanilla LSTM**Une configuration LSTM simple est le Vanilla LSTM. qui a une seule couche cachée d'unités LSTM. l'architecture qui donnera de bons résultats

sur la plupart des petits problèmes de prédiction de séquence. Le Vanilla LSTM est défini comme :

— Couche d'entrée.

— Couche cachée LSTM entièrement connectée.

— Couche de sortie entièrement connectée.

[35]

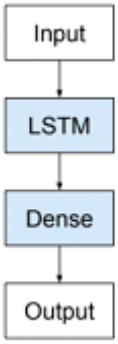


FIGURE 3.12 – Structure de Vanilla LSTM
[35]

2. **Stacked LSTM**est un modèle qui a plusieurs couches LSTM cachées où chaque couche contient plusieurs cellules de mémoire. Nous l'appellerons ici un LSTM empilé pour le différencier du LSTM non empilé (Vanilla LSTM) et d'une variété d'autres extensions du modèle LSTM de base[35].

FIGURE 3.13 – Structure de stacked LSTM
[35]

3. **CNN LSTM**L'architecture CNN LSTM implique l'utilisation de couches CNN (Convolutional Neural Network) pour l'extraction de caractéristiques sur les données d'entrée combinées avec des LSTM pour prendre en charge la prédiction de séquence.

Les LSTM CNN ont été développés pour les problèmes de prédiction de séries chronologiques visuelles et l'application de la génération de descriptions textuelles à partir de séquences d'images (par exemple des vidéos)[35].

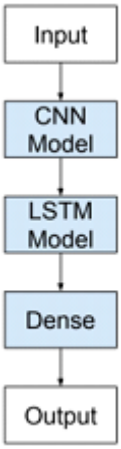


FIGURE 3.14 – Structure de CNN_ LSTM [35]

— Structure CNN LSTM : Le modèle cnn-lstm consiste séquentiellement des couches convolutionnelle 1D chaque couche suivie d'une couche de max pooling, la sortie est ensuite aplatie pour alimenter les couches LSTM (où la couche Flatten relier entre cnn et lstm).

Les couches LSTM cachées suivies d'une couche dense pour fournir la sortie [35].

4. **Encodeur-décodeur** Une approche des problèmes de prédiction seq2seq qui s'est avérée très efficace est appelée Encoder-Decoder LSTM. Cette architecture est composée de deux modèles : un pour lire la séquence d'entrée et la coder dans un vecteur de longueur fixe, et un second pour décoder le vecteur de longueur fixe et sortir la séquence prédite. L'utilisation des modèles de concert donne à l'architecture son nom d'encodeur-décodeur LSTM conçu spécifiquement pour les problèmes seq2seq[35].
5. **LSTM bidirectionnel** Nous avons vu l'intérêt d'inverser l'ordre des séquences d'entrée pour les LSTM discuté dans l'introduction des LSTM Encoder-Decoder. Les LSTM bidirectionnels se concentrent sur le problème de tirer le meilleur parti de la séquence d'entrée en parcourant les pas de temps d'entrée dans les directions avant et arrière. En pratique, cette architecture consiste à dupliquer la première couche récurrente dans le réseau afin qu'il y ait maintenant deux couches côte à côte, puis à fournir la séquence d'entrée telle quelle en entrée de la première couche et à

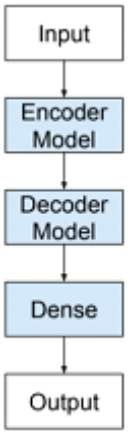


FIGURE 3.15 – Structure de encoder-decoder

fournir une copie inversée de la séquence d'entrée. à la seconde. Cette approche a été développée il y a quelque temps comme approche générale pour améliorer les performances des réseaux neuronaux récurrents (RNN) [35].

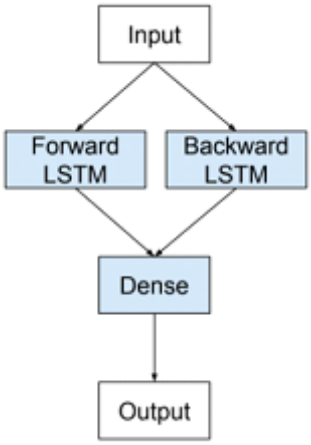


FIGURE 3.16 – Structure de LSTM bidirectionnel [35]

6. **LSTM générative** Un LSTM génératif n'est pas vraiment une architecture, c'est plutôt un changement de perspective sur ce qu'un modèle prédictif LSTM apprend et comment le modèle est utilisé. On pourrait imaginer utiliser n'importe quelle architecture LSTM comme modèle génératif. Dans ce cas, nous utiliserons un simple LSTM Vanilla[35].

3.3 Le RUL et le deep learning

3.3.1 Revue de quelques travaux existants de RUL avec le deep learning

Récemment, différents chercheurs ont démontré un grand succès dans l'utilisation de modèles d'apprentissage profond dans l'application de pronostics et la surveillance de l'état de la machine . Plusieurs techniques d'apprentissage en profondeur ont été développées pour apprendre la cartographie des données collectées à l'RUL associé. On peut citer le travail de :

Xiang Li et al.[36], ont proposé une nouvelle approche basée sur les données pour les pronostiques utilisant des réseaux de neurones à convolution profonde (DCNN). L'approche de la fenêtre temporelle est utilisée pour la préparation des échantillons afin d'obtenir une meilleure extraction des caractéristiques par DCNN. Les données brutes collectées avec normalisation sont directement utilisées comme entrées réseau proposé, et aucune expertise préalable sur les pronostiques et le traitement du signal n'est nécessaire, qui facilite l'application de la méthode proposée.après avoir testé la méthode proposée en utilisant la métrique RMSE, les résultats de test ont montré un RMSE avec 18,45 de DCNN. La majorité des approches existantes pour estimer RUL n'intègrent qu'un seul modèle.

Une approche hybride, d'autre part, a pour objectif de combiner les avantages de différentes techniques grâce à leur intégration de sorte que les résultats puissent être agrégés pour améliorer les performances de prédiction. Dans cette catégorie on peut citer les travaux suivants qu'on trouve plus proches de notre travail et de qui on a inspiré beaucoup de choses :

Ali et al.[14], ont proposé une nouvelle approche hybride de réseau de neurones profonds pour l'estimation RUL qui intègre deux architectures d'apprentissage profond (cnn et LSTM) .La méthode proposée consiste en deux chemins parallèles chemin cnn et chemin lstm, le LSTM est utilisé pour extraire des caractéristiques temporelles tout en simultanément, le CNN est utilisé pour extraire des caractéristiques spatiales .les performances de la méthode proposée sont mesurées par le terme de métrique RMSE. Les résultats de test ont donné RMSE avec 13.07, ce qui est considéré comme l'un des meilleurs des résultats existants.

Zhang et al.[15], ont construit un réseau neuronal récurrent LSTM (LSTM-RNN) pour capturer et apprendre les dépendances à long terme parmi les capacités dégradées des batteries lithium-ion. La technique de rétroprojection quadratique moyenne résiliente (RMSprop) a été utilisée pour entraîner le réseau neuronal construit, et la technique d'abandon a été utilisée pour éviter le problème de surajustement (overfitting)et améliorer

la capacité de prédiction du LSTM-RNN.Les résultats ont montré que la valeur d'erreur est de 40, ce qui améliore l'efficacité du LSTM par rapport au reste des méthodes proposées.

Un autre modèle hybride est introduit par **Z**hao et al.[16] où une structure de réseau neuronal profond (dénommé réseaux de mémoire à long court terme bidirectionnels à convolution (CBLSTM)) a été construite pour traiter les tâches de prédiction d'usure des outils. Le modèle utilise d'abord un CNN pour extraire des caractéristiques robustes locales à partir de données sensorielles brutes. Ensuite, un LSTM bidirectionnel implémenté en série par rapport au CNN (pour capturer les dépendances à long terme de manière directe et descendante) est utilisé pour coder les informations temporelles et apprendre les représentations. Au-dessus des LSTM bidirectionnels, des couches empilées entièrement connectées et une couche de régression linéaire sont construites pour prédire la valeur cible .

Ces travaux cités , ont donné de bons résultats en terme d' accuracy ce qui prouve que le modèle LSTM appliqué seul ou hybridé avec d'autre méthodes est puissant dans ce domaine. Donc c'est pour cela on a essayer dans notre travail d'appliquer le LSTM et ses différents types ou hybrider avec d'autres méthodes pour étudier son efficacité. De ces travaux cités aussi, on a inspiré plusieurs choses à appliquer dans notre travail, telles que les paramètres à faire varier comme les optimizers etc... ainsi que le window size qui n'est pas utilisé pour plusieurs valeurs ...etc.

Conclusion

Dans ce chapitre nous avons présenté les notions importantes qui sont en relation avec les réseaux de neurones et ainsi une vision générale sur l'apprentissage profond, toute en expliquant en détail le LSTM, la méthode choisie dans notre travail de recherche. Ainsi,nous avons terminé par une exploration de quelques travaux de RUL avec le deep learning.

Le prochain chapitre, s'intéresse à présenter notre contribution dans le domaine, les modèles que nous allons proposer ainsi que les différents résultats et comparaisons.

CHAPITRE 4

CONTRIBUTION

Introduction

Dans ce chapitre, nous allons utiliser le deep learning pour résoudre le problème de l'estimation de RUL pour la maintenance conditionnelle. En se basant sur les travaux connexes décrits dans le chapitre 3, nous essayerons dans ce chapitre de proposer de nouvelles variantes de LSTM et ses sous types, ainsi que des hybridations entre eux et avec un autre type de deep learning qui est le CNN avec une hybridation qui n'a pas encore été proposé dans la littérature.

L'objectif principal de ces modèles proposés est de chercher la meilleure configuration qui donne de bons résultats pour une meilleure prédiction. En effet, nous montrons les différentes expérimentations menées et nous discutons les résultats obtenus.

En premier lieu, nous allons décrire la problématique et les motivations de notre contribution, puis nous présenterons l'architecture des différents modèles proposés et le dataset que nous allons utiliser. Ensuite, nous expliquerons en détail les cinq variantes de LSTM proposés pour faire la prédiction de RUL en comparant et analysant les différents résultats dans chaque cas. A la fin, une analyse et comparaison et faite entre les variantes proposées et quelques travaux de la littérature.

4.1 Problématique et Motivation

L'estimation de la durée de vie utile restante (en anglais : Remaining Useful Life, RUL) et l'évaluation de la dégradation des performances des machines ont un rôle crucial dans la maintenance conditionnelle. Elles aident à réduire les coûts de maintenance, à améliorer la fiabilité et ainsi elles influencent la prise de décision dans le système[12]. L'estimation

de RUL a été le sujet de plusieurs approches (décrites en chapitre 2) qui ont données des résultats proches de la réalité. L'intelligence artificielle et notamment les deep learning ont joué un rôle important dans la réalisation de ces résultats. Les problèmes résolus par les deep learning sont classés en deux types ; classification et régression. La prédiction de RUL peut être vue comme un problème de régression linéaire. Les données décrivant les paramètres de la CBM sont classées de types séries chronologiques (time series). Ainsi, les problèmes de prédiction de séries chronologiques (time series) sont un type difficile de problème de modélisation prédictive. Contrairement à la modélisation prédictive de régression, les séries chronologiques ajoutent également la complexité d'une dépendance de séquence entre les variables d'entrée.Un type puissant de réseau neuronal conçu pour gérer la dépendance de séquence est appelé réseaux neuronaux récurrents. Le réseau de mémoire à long court terme ou réseau LSTM est un type de réseau neuronal récurrent utilisé dans l'apprentissage profond, car de très grandes architectures peuvent être entraînées avec succès [37].

Des travaux récents ont utilisé LSTM et ont donné de bons résultats (voir chapitre 2), pour cette raison et pour leurs capacité d'apprendre les caractéristiques temporelles des donnéespour la prédiction de séries chronologiques, nous avons choisi d'utiliser le LSTM dans notre travail afin d'estimer le RUL.

4.2 Architecture de Deep Learning proposée

Notre projet de recherche porte sur le problème de prédiction de séries chronologiques (time series) qui est l'estimation de RUL sur les données de la NASA ; C-MAPSS (section 2.1). L'objectif de ce travail est d'implémenter plusieurs variantes des modèles proposés en faisons varier plusieurs métriques, analyser les résultats et faire des comparaisons entre eux pour chaque métrique puis à la fin faire une comparaison entre ces modèles et ceux de la littérature.

Au cours de nos expériences, nous avons créé cinq modèles avec des architectures différentes, pour atteindre l'objectif et pour obtenir la meilleure performance possible. Le premier est un modèle LSTM (Long Short Term Memory), le deuxième est un modèle hybride CNN_LSTM (Convolutional Neural Networks and Long Short Term Memory), le troisième est un modèle bidirectionnel LSTM, le quatrième est un modèle hybride CNN_Bi_LSTM (Convolutional bidirectional LSTM) et le cinquième est un modèle hybride CNN_Bidirectional (Convolutional bidirectional).

1. **Méthode 1 :**Estimation de RUL en utilisant le model LSTM simple. Un LSTM peut être défini en ajoutant des couches LSTM avec une couche dense en sortie. La figure suivante représente les différentes couches que contient notre modèle :

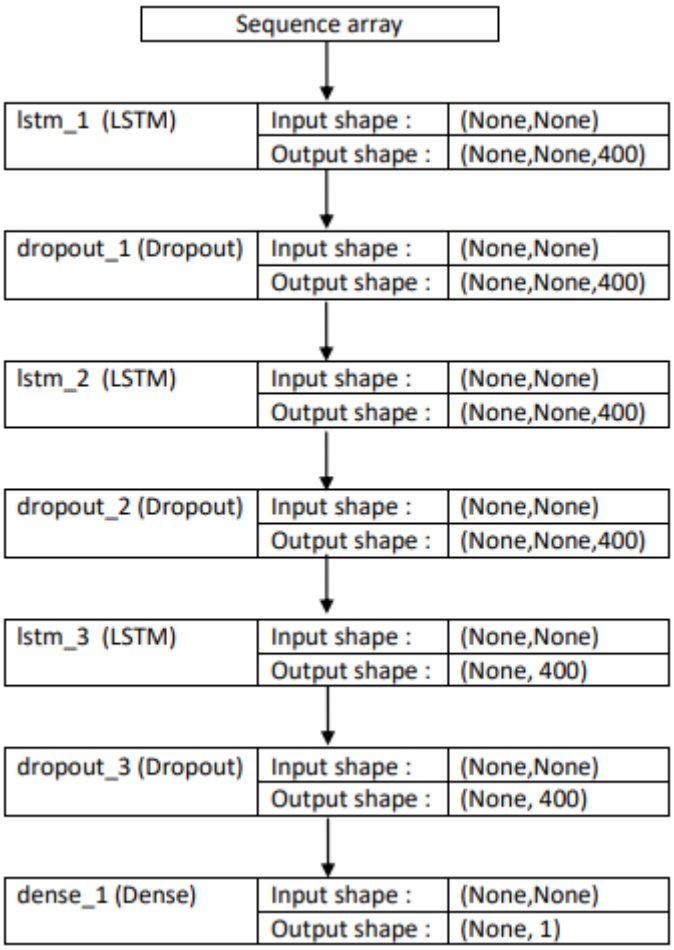


FIGURE 4.1 – L'architecture du modèle LSTM.

- **La couche LSTM :**nous ajoutons trois couches de LSTM avec 400 unités cachées pour chaque couche.
- **La couche Dropout :** nous avons appliqué trois couches Dropout suivi de chaque couche LSTM pour évite le problème de overfiting.
- **La couche Dense :**nous ajoutons un couches entièrement connectées qui mappe la sortie de LSTM à la taille de sortie désirée qui est égale à 1 et prévue par la fonction d'activation Relu.

2. **Méthode 2 :**Estimation de RUL en utilisant le model hybride CNN-LSTM. Un CNN LSTM peut être défini en ajoutant des couches CNN en premier, suivies de couches LSTM avec une couche dense en sortie. La figure suivante représente les différentes couches que contient notre modèle :
   - **La couche Conv1D :**nous ajoutons 2 couches convolutives d'une dimension avec 400 unités cachées pour chaque couche.
   - **La Couches MaxPooling1D :**nous utilisons deux couches de maxpooling suivie de chaque couche convolutives d'une dimension .

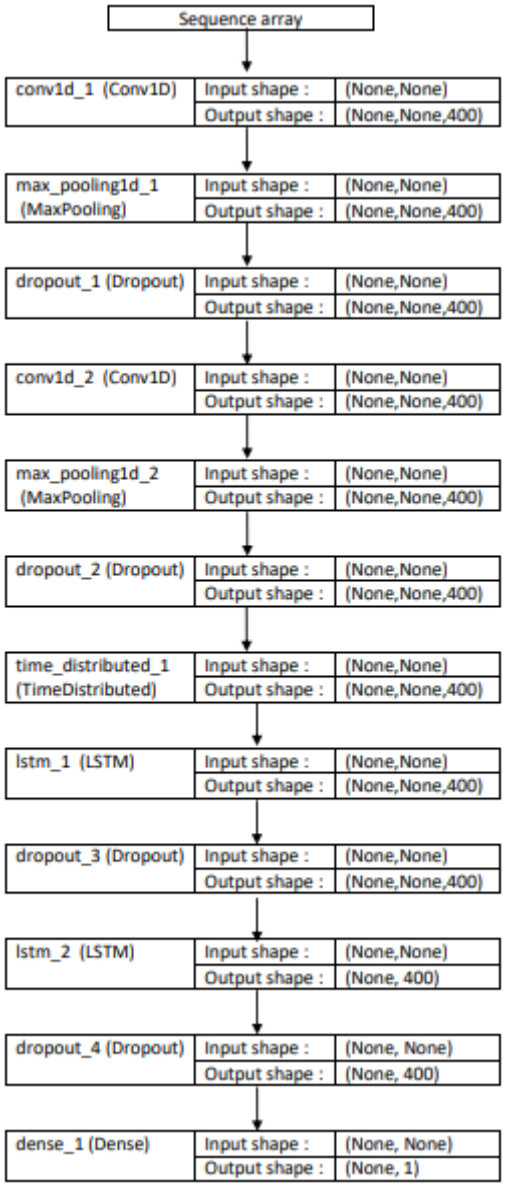


FIGURE 4.2 – L'architecture du modèle CNN- LSTM.

- **La couche TimeDistributed :**nous ajoutons une couche time distributed pour relier le modèle CNN avec le modèle LSTM.
- **La couche LSTM :**nous ajoutons 2 couches de LSTM avec 400 unités cachées pour chaque couche.
- **La couche Dropout :**nous avons appliqué quatre couches Dropout suivi de chaque couche MaxPooling et LSTM pour évite le problème de overfitting.
- **La couche Dense :**nous ajoutons un couches entièrement connectées avec qui mappe la sortie de LSTM à la taille de sortie désirée qui est égale à 1 et prévue par la fonction d'activation Relu.

3. **Méthode 3 :** Estimation de RUL en utilisant le modèle bidirectionnel LSTM. Dans les problèmes où tous les pas de temps de la séquence d'entrée sont disponibles, les

LSTM bidirectionnels entraînent deux LSTM au lieu d'un sur la séquence d'entrée. Le premier sur la séquence d'entrée en l'état et le second sur une copie inversée de la séquence d'entrée. Cela peut fournir un contexte supplémentaire au réseau et entraîner un apprentissage plus rapide et encore plus complet du problème. Un LSTM bidirectionnel peut être défini en ajoutant des couches Bidirectionnel LSTM avec une couche dense en sortie. La figure suivante représente les différentes couches de notre modèle :

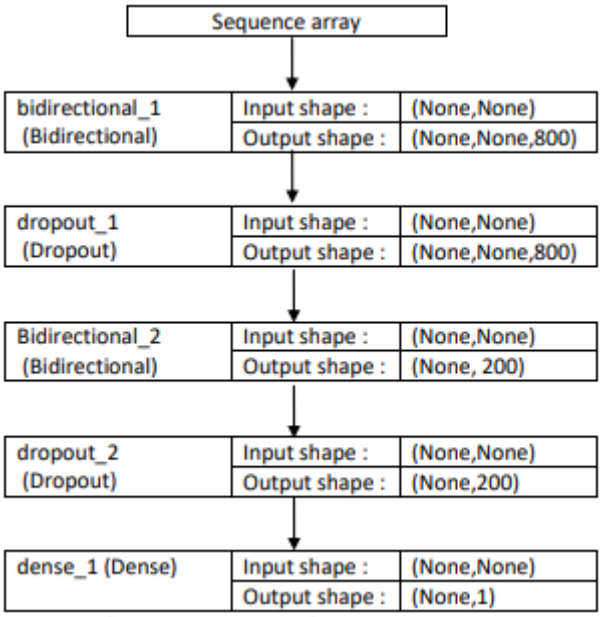


FIGURE 4.3 – L'architecture du modèle Bidirectionnel.

- **La couche Bidirectionnelle :**nous ajoutons 2 couches de bidirectionnelles lstm avec 800 unités pour la première couche et 200 unités pour la deuxième couche.
- **La couche Dropout :**nous avons appliqué deux couches Dropout suivi de chaque couche Bidirectional pour évite le problème de overfitting.
- **La couche Dense :** nous ajoutons une couche entièrement connectée qui mappe la sortie de Bidirectional à la taille de sortie désirée qui est égale à 1 et prévue par la fonction d'activation Relu.

4. **Méthode 4 :** Estimation RUL en utilisant le modèle CNN_BI_ LSTM. Un CNN_BI_ LSTM peut être défini en ajoutant des couches CNN en premier, suivies de couche Bidirectionnel, ensuite de couche LSTM avec une couche dense en sortie. La figure suivante représente les différentes couches que contient notre modèle :
   - **La couche Conv1D :**nous ajoutons 2 couches convolutives d'une dimension avec 400 unités cachées pour chaque couche.

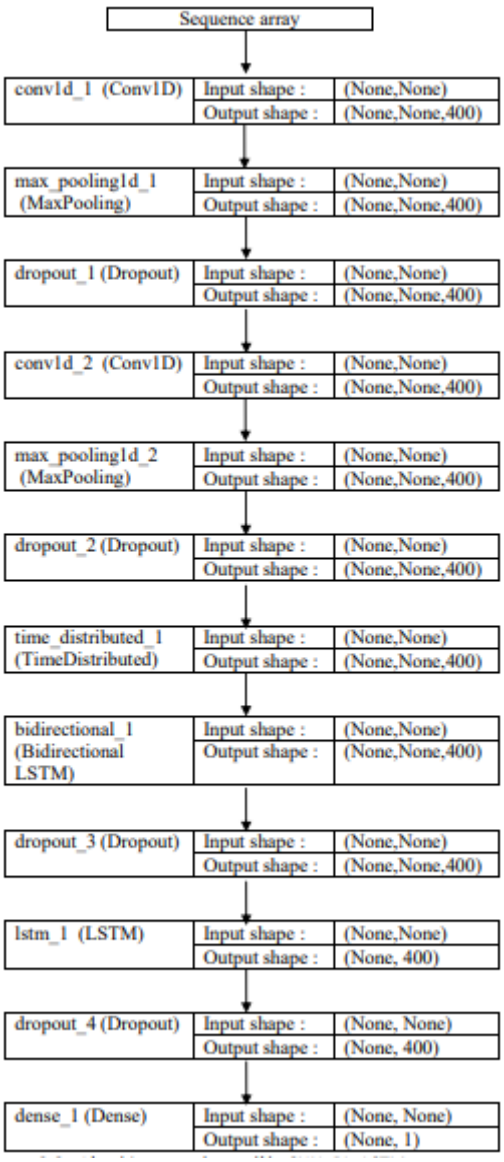


FIGURE 4.4 – L'architecture du modèle CNN_BI_LSTM

- **La Couches MaxPooling1D :**nous utilisons deux couches de maxpooling suivie de chaque couche convolutives d'une dimension .
- **La couche TimeDistributed :**nous ajoutons une couche time distributed pour relier le modèle CNN avec le modèle LSTM.
- **La couche bidirectional LSTM :**nous ajoutons deux couches de Bidirectionnelles LSTM avec 200 unités cachées pour chaque couche.
- **La couche LSTM :**nous ajoutons deux couches de LSTM avec 400 unités cachées pour chaque couche.
- **La couche Dropout :**nous avons appliqué quatre couches Dropout suivi de chaque couche MaxPooling et LSTM pour évite le problème de overfitting.
- **La couche Dense :**nous ajoutons un couches entièrement connectées avec qui

mappe la sortie de LSTM à la taille de sortie désirée qui est égale à 1 et prévue par la fonction d'activation Relu.

5. **Méthode 5 :** Estimation RUL en utilisant le modèle CNN_Bidirectional. Un CNN_Bidirectional peut être défini en ajoutant des couches CNN en premier, suivies de couches Bidirectionnelles avec une couche dense en sortie. La figure suivante représente les différentes couches que contient notre modèle :

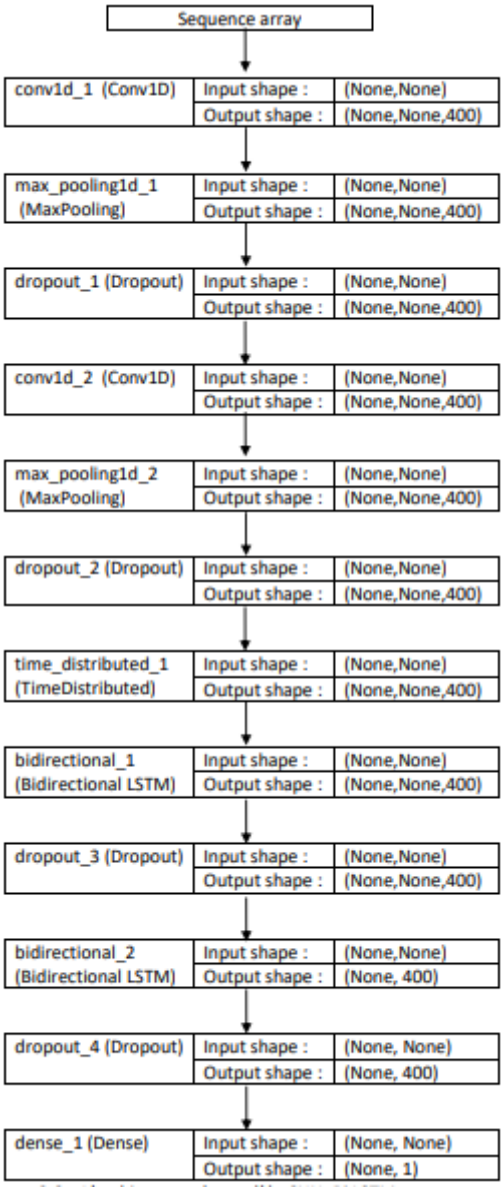


FIGURE 4.5 – L'architecture du modèle CNN-Bilstm

- **La couche Conv1D :**nous ajoutons 2 couches convolutives d'une dimension avec 400 unités cachées pour chaque couche.
- **La Couches MaxPooling1D :**nous utilisons deux couches de maxpooling suivie de chaque couche convolutives d'une dimension .

- **La couche TimeDistributed :**nous ajoutons une couche time distributed pour relier le modèle CNN avec le modèle LSTM.
- **La couche bidirectional LSTM :**nous ajoutons deux couches de Bidirectional LSTM avec 200 unités cachées pour chaque couche.
- **La couche Dropout :**nous avons appliqué quatre couches Dropout suivi de chaque couche MaxPooling et LSTM pour évite le problème de overfitting.
- **La couche Dense :**nous ajoutons un couches entièrement connectées avec qui mappe la sortie de LSTM à la taille de sortie désirée qui est égale à 1 et prévue par la fonction d'activation Relu.

4.3 Etude éxpérimentale et validation

Nous évaluons les performances du LSTM proposé et leur types puis le compare le meilleur résultat. Tout d'abord, nous avons presenté les outils de developement dans la sous section 2.1, les descriptions des ensembles de données sont fournies dans la sous-section 2.2, suivies les métriques d'évaluations dans la sous-section 2.4. Enfin, les étapes de notre travail dans la sous-section 2.6.

4.3.1 Dataset utilisé

La méthode proposée est évaluée sur un problème de benchmarking pronostique, c'est-à-dire. Problème de dégradation des turbosoufflantes de la NASA. Cet ensemble de données populaire contient des données simulées produites par un programme de simulation basé sur un modèle, à savoir la simulation de système aéropropulseur modulaire commercial (C-MAPSS), qui a été développé par la NASA[14].

4.3.1.1 NASA C-MAPSS DATASET

Les données NASA C-MAPSS est un ensemble de données de référence largement utilisé, généré à l'aide des logiciels de programme de simulation basé sur un modèle propriétaire (nommé C-MAPSS). Le C-MAPSS est un logiciel pour simule les effets des défauts et de la détérioration a différentes conditions de fonctionnement dans les cinq principaux composants rotatifs (ventilateur, compresseur basse pression, Compresseur haute pression (HPC), turbine haute pression et basse Turbine à pression) trouvée dans un gros turboréacteur commercial. L'ensemble de données est divisé en quatre sous-ensembles de données (étiquetés à partir de FD001 à FD004) avec un nombre différent de conditions de fonctionnement et de modes de défaut[14]. Chaque sous-ensemble de données est ensuite divisé en sous-ensembles d'apprentissage et de test . Les ensembles de données décrits dans

le tableau 1 : Les ensembles de données sont organisés dans une matrice N par 26, où N

Dataset	C-MAPSS			
	FD001	FD002	FD003	FD004
train trajectories	100	260	100	249
test trajrctories	100	259	100	248
condition	1	6	1	6
fault modes	1	1	2	2

TABLE 4.1 – Les détails de dataset (simulée à partir de C-MAPSS).
[14]

correspond au nombre de points de données dans chaque ensemble de données. Chaque ligne est un instantané des données prises au cours d'un cycle de temps de fonctionnement unique, qui comprend 26 colonnes et chaque colonne représente une variable différente. Les 26 colonnes de données se composent de deux valeurs d'index représentant le numéro du moteur et le numéro du cycle opérationnel actuel, trois paramètres opérationnels qui ont un effet substantiel sur les performances du moteur, ainsi que 21 valeurs de capteur. Chaque trajectoire dans les ensembles de données simule la durée de vie d'un moteur. Alors que chaque moteur est simulé avec des conditions initiales différentes, l'état de fonctionnement de chaque moteur est sain au début et commence à se dégrader au fil du temps jusqu'à ce qu'une panne se produise. Pour chaque trajectoire du moteur dans les ensembles d'entraînement, la dernière entrée de données correspond au moment où le moteur est déclaré malsain. Bien que les trajectoires dans les ensembles de test se terminent à un certain moment avant la défaillance et l'objectif est de prédire le nombre de cycles jusqu'à la fin de la durée de vie du produit pour chaque moteur communément appelé RUL. La valeur RUL réelle des trajectoires d'essai pour l'ensemble de données C-MAPSS a été rendue publique[14].

4.3.1.2 Conditions du fonctionnement et modes de défaut

Les points de données sont classés en différents clusters distincts en utilisant deux facteurs, à savoir les conditions de fonctionnement et les modes de défaut associés. Quant aux conditions de fonctionnement, FD001 et FD003 sont simulées en un seul point (niveau de la mer), tandis que FD002 et FD004 sont simulées à six conditions de fonctionnement différentes. Pour les modes de défaut, FD001 et FD002 sont simulés avec uniquement une dégradation HPC, tandis que FD003 et FD004 sont simulés avec deux dégradation HPC et dégradation du ventilateur, ce qui entraîne des prédictions RUL plus complexes et difficiles [14].

4.3.2 Normalisation des données

Les données temporelles multi-variées de l'ensemble de données C-MAPSS contiennent des mesures d'unité moteur de 21 capteurs . Cependant, certaines lectures de capteur ont des sorties constantes pendant la durée de vie du moteur et elles ne fournissent pas d'informations précieuses pour l'estimation de la RUL. Pour chacun des 4 sous-ensembles de données dans C-MAPSS, les données de mesure collectées de chaque capteur sont normalisées pour être dans la plage de [-1, 1] en utilisant la méthode de normalisation min-max [36].

$$x_{norm}^{j,j} = \frac{2(x^{i,j} - x_{min}^{i,j})}{x_{max}^{j} - x_{min}^{j}} - 1, \forall i, j \tag{4.1}$$

où $x^{i,j}$ désigne le i-ème point de données d'origine du j-ème capteur, et $x_{norm}^{i,j}$ est la valeur normalisée de $x^{i,j}$. x_{max}^{j} et x_{min}^{j} désignent respectivement les valeurs maximale et minimale des données de mesure originales du j-ème capteur. [36] Cette normalisation garantira une contribution égale de toutes les fonctionnalités dans toutes les conditions de fonctionnement [14].

4.3.3 Métriques d'evaluation

Dans cette étude, les paramètres ont été utilisés pour évaluer les performances de la méthode pronostique proposée. Fonction de score de régression(R^2) , mean squared error (MSE),mean absolute error(MAE)et root mean squared error(RMSE).

4.3.3.1 Fonction de score de régression (R^2)

La fonction r2-score calcule le coefficient de détermination, généralement noté R^2. Il représente la variance de y (où Y est la droite de la régression) qui a été expliquée par les variables indépendantes du modèle. Il fournit une indication de la qualité de l'ajustement et, par conséquent, une mesure de la façon dont les échantillons invisibles sont susceptibles d'être prédits par le modèle, grâce à la proportion de variance expliquée. Comme une telle variance dépend de l'ensemble de données, R^2 peut ne pas être véritablement comparable entre différents ensembles de données. Le meilleur score possible est de 1,0 et il peut être négatif (car le modèle peut être arbitrairement pire). Un modèle constant qui prédit toujours la valeur attendue de y, sans tenir compte des caractéristiques d'entrée, obtiendrait un score R^2 de 0,0. Si est la valeur estimée de l' -ème échantillon et est la valeur vraie correspondante pour les échantillons totaux, le R^2 estimé est défini comme[38] :

$$R^2(y, \bar{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.2}$$

Où :

$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \qquad \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} e_i^2$

4.3.3.2 Mean Squared Error (MSE)

Pour les tâches de prédiction numérique, la «Mean Squared Error »(MSE) est une méthode courante. MSE quantifie la différence entre un estimateur et la valeur ciblée. Le MSE d'un ensemble de données est la moyenne de la somme de toutes les erreurs carrées de chaque modèle. Étant donné n modèles de l'ensemble de données, pour le ième exemple, soit p_i la valeur prédite et a_i la valeur réelle. Le MSE de l'ensemble de données testé est [39] :

$$MSE = \frac{\sum_{i=0}^{n}(a_i - p_i)^2}{n} \tag{4.3}$$

4.3.3.3 Root Mean Squared Error (RMSE)

Le RMSE est couramment utilisé comme mesure de la performance car il donne des poids égaux pour les prévisions précoces et tardives. La formulation du RMSE est donnée par [14] :

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} h_i^2} \tag{4.4}$$

4.3.3.4 Mean Absolute Error (MAE)

La fonction mean-absolute-error calcule l'erreur absolue moyenne, une mesure de risque correspondant à la valeur attendue de la perte d'erreur absolue ou de la perte normale[39]. Si est la valeur prédite du -ème échantillon, et est la valeur vraie correspondante, alors l'erreur absolue moyenne (MAE) estimée sur est définie comme :

$$MAE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples-1}} |yi - \hat{y}_i| \tag{4.5}$$

4.3.4 La taille de la fenêtre(window size)

Lors de l'utilisation de LSTM dans le domaine des séries temporelles (time series), un paramètre important à choisir est la longueur de séquence qui est la fenêtre pour les LSTM de regarder en arrière. Cela peut être considéré comme similaire à la sélection de window-size = 70 cycles pour calculer les caractéristiques de roulement dans le modèle de maintenance prédictive qui sont une moyenne mobile et un écart-type mobile pour 21 valeurs de capteur. L'idée d'utiliser les LSTM est de laisser le modèle extraire des caractéristiques abstraites de la séquence de valeurs de capteur dans la fenêtre plutôt que de les concevoir manuellement. On s'attend à ce que s'il y a un modèle dans ces valeurs de

capteur dans la fenêtre avant l'échec, le modèle doit être codé par le LSTM[40].
L'un des principaux avantages des LSTM est leur capacité à se souvenir des séquences à long terme (tailles de fenêtre), ce qui est difficile à obtenir par l'ingénierie des fonctionnalités traditionnelle. Par exemple, le calcul de moyennes mobiles sur une taille de fenêtre de 50 cycles peut conduire à une perte d'informations due au lissage et à l'extraction des valeurs sur une période aussi longue, au lieu de cela, l'utilisation des 50 valeurs en entrée peut fournir de meilleurs résultats. Bien que l'ingénierie des fonctionnalités sur de grandes tailles de fenêtre puisse ne pas avoir de sens, les LSTM peuvent utiliser des tailles de fenêtre plus grandes et utiliser toutes les informations de la fenêtre comme entrée[40].

4.3.5 Résultats et discussions

Dans cette section, nous présentons divers résultats expérimentaux pour évaluer les performances des modèles LSTMs proposées pour l'estimation RUL. Tout d'abord , nous avons testé l' approches proposés (LSTM, CNN_LSTM, BidirectionalLstm, CNN_Bi_LSTM et CNN_Bidirectional). Nous obtenons les résultats suivants :

4.3.5.1 Les résultats des différents modèles (pour FD001)

Pour FD001 (Epoch=10, window size =50, Activation=linear, Optimizer =rmsprop) D'après le tableau précédent, nous observons les résultats de métriques d'évaluation après

Algorithms	train metric		Test metric	
	MSE	R^2	MSE	R^2
LSTM	703.99	0.63	803.27	0.50
CNN	1214.15	0.46	498.77	0.70
CNN_LSTM	**353.98**	**0.81**	825.56	0.50
B_LSTM	1620.58	0.26	947.91	0.43
CNN_BI_LSTM	566.75	0.78	441.54	0.74
CNN_BiLstm	523.45	0.79	421.23	0.75

TABLE 4.2 – les résultats des modèles (sous-données FD001).

l'entraînement des modèles LSTM, CNN_LSTM, bidirectionnel LSTM et CNN pendant 10 époques pour l'ensemble de données FD001. Nous remarquons que le modèle LSTM a donné de bons résultats où la performance est de 0.63, mais le modèle CNN_LSTM a donné mieux que le LSTM avec une performance égale à 0.81. Ce résultat est expliqué par l'architecture de CNN_LSTM qui implique l'utilisation de couches CNN (Convolutional Neural Network) qui servent à l'extraction de caractéristiques des données d'entrée, combinées avec des couches LSTM pour prendre en charge la prédiction de séquence.Une couche 1D de CNN est utilisée pour capturer des représentations de bas niveau des lectures de capteur. Ces

représentations sont ensuite envoyées à une couche LSTM afin de capturer les dépendances à long terme. Ensuite, la couche réseau entièrement connectée mappe le RUL aux entrées de la couche précédente.Ainsi les CNN_LSTM ont été développés pour les problèmes de prédiction de séries chronologiques et spécialement pour les problèmes de prédiction de séquence[41]. Par contre le modèle LSTM bidirectionnel a donné des résultats plus faibles où la performance est 0.26. Le motif de ce résultat est que le modèle est utilisé généralement et d'après ce qu'on a vu dans la littérature, pour améliorer les performances des problèmes de classification de séquences [42], et notre problème à résoudre est un problème de régression. Mais afin de profiter des caractéristiques de ce modèle, nous l'avons hybridé avec le CNN et le LSTM et nous avons trouvé que l'hybridation donne de bons résultats. Avec CNN_Bi_LSTM la performance est 0.78 et avec CNN_Bidirectional la performance est égale à 0.79.

Comme nous avons trouvé que le CNN_LSTM donne de meilleurs résultats, dans le tableau 3 nous essayons d'appliquer le modèle avec tous les sous ensembles de données (FD001, FD002, FD003 et FD004) avec 10 époques, afin de suivre les résultats pour une future amélioration. Les résultats sont les suivants : (Epoch=10, window size =50, Activation=linear, Optimizer =rmsprop)

	Métriques	**FD001**	**FD002**	**FD003**	**FD004**
Train	MSE	353.98	202.77	1036.08	395.10
	R^2	0.81	0.93	0.64	0.79
Test	MSE	925.56	1467.11	952.05	1660.47
	R^2	0.45	0.43	0.25	0.20

TABLE 4.3 – les résultats de metriques de régression pour CNN_ LSTM (de FD001 à FD004)

D'aprés cette discussion,on a décidé de tester l'évaluation et la performance de ces sinq modèles, afin d'atteindre les meilleurs résultats et choisir le modèle approprié de notre travail.On commence par l'amélioration des paramètres, ensuite les résultats finales de chaque modéle :

4.3.5.2 Les résultats avec différents paramètres

Comme on peut le voir dans les tableaux 4, Nous avons comparé les différentes fonctions d'activation avec différents optimizer et nous avons entraîné le modèle CNN_LSTM pendant 10 époques pour l'ensemble de donnée FD001. Nous avons pu voir que l'activation=relu avec optimizer=rmsprop et l'activation=linear avec optimizer=rmsprop donnent de bons résultats où la valeur de performance est égale à 0.85 et 0.81 respectivement, mais les résultats de Relu sont meilleurs que linear. Car Relu est la fonction d'activation la plus

largement utilisée pour les applications d'apprentissage en profondeur avec des résultats à la pointe de la technologie à ce jour. Elle offre de meilleures performances et une meilleure généralisation en apprentissage profond par rapport aux fonctions d'activation Sigmoïdee [43].Comme il s'agit de l'une des fonctions les plus puissantes, elle est rapide et simple à calculer, elle casse la linéarité des valeurs négatives et désactive certains neurones[44].

Nous avons utilisé les optimizer Rmsprop et Adam dans notre travail et les résultats indiquent que Rmsprop donne de bons résultats qu' Adam. Malgré que nous avons trouvé dans la littérature qu'Adam est le seul qui a surpassé Rmsprop, mais ce dernier correspond plus étroitement à notre travail qu'Adam. L'explication de ces résultats est dans «A Peek at Trends in Machine Learning» [45] d' Andrej karpathy où il a montré que Rmsprop est un bon optimizer, rapide et aussi l'un des algorithmes d'optimisation les plus populaires utilisé dans l'apprentissage en profondeur, spécialement les réseaux de neurones RNN ,sa popularité n'est dépassée que par Adam [46].

Pour FD001 (Epoch=10, window size =50)

Algorithms	train		Test	
	MSE	R^2	MSE	R^2
CNN_LSTM (activation=linear ,optmizer=rmsprop	353.98	0.81	925.56	0.50
CNN_LSTM (activation=linear ,optmizer=Adam	1149.03	0.57	524.49	0.68
CNN_LSTM (activation=Relu ,optmizer=Adam	918.88	0.69	439.83	0.73
CNN_LSTM (activation=Relu ,optmizer=rmsprop	**322.77**	**0.85**	679.36	0.60

TABLE 4.4 – les résultats des métriques de régression pour différents paramètres (de FD001).

4.3.5.3 Les résultats avec différentes tailles de la fenêtre de temp (WS)

Pour montrer l'avantage de l'efficacité de notre cadre CNN_LSTM proposé, nous examinons les effets de la taille de la fenêtre sur les résultats. Le tableau 5 et 6 montrent les résultats obtenus à partir du cadre CNN_LSTM proposé en utilisant des tailles de fenêtre de SW=30, SW=50 et SW=70 pour FD001 et FD002 après 10 époques, Les valeurs de MSE et de R^2pour SW=70 sont nettement meilleures que les valeurs des autres SW. En tant que, tel les ensembles de données FD002 et FD004, qui sont de nature complexe avec plus de conditions de fonctionnement par rapport à FD001 et FD003. On observe que les résultats obtenus à partir d'une taille de fenêtre plus grande étaient plus efficaces, puisqu' il est clairement montré qu'une plus grande taille de fenêtre temporelle donne

de meilleures estimations RUL. Des informations plus brutes peuvent être couvertes par une plus grande fenêtre de temps, qui est la base pour une extraction supplémentaire des fonctionnalités[36].Contrairement à ce que nous avons trouvé dans l'article[14] car la difference entre notre travail et leur travail est que notre modèle est sequenciel et leur modele est parallèl. Les résultats sont également présentés sous forme de tableau ci-dessous : Pour FD001 (Epoch=10, activation=relu, optimizer= rmsprop) Pour FD002

Algorithm	train		Test	
	MSE	R^2	MSE	R^2
CNN_LSTM WS=30	495.68	0.83	813.04	0.58
CNN_LSTM WS=50	322.77	0.85	679.36	0.60
CNN_LSTM WS=70	**255.37**	**0.89**	**497.44**	**0.70**

TABLE 4.5 – les résultats des métriques de régression pour différentes tailles de fenêtre de temp(WS)de FD001.

(Epoch=10, activation=relu, optimizer= rmsprop)

Algorithm	train		Test	
	MSE	R^2	MSE	R^2
CNN_LSTM WS=30	362.44	0.91	1149.0836	0.53
CNN_LSTM WS=50	299.76	0.88	1421.14	0.44
CNN_LSTM WS=70	**220.31**	**0.93**	**41129.94**	**0.54**

TABLE 4.6 – les résultats des métriques de régression pour différentes tailles de fenêtre de temp (WS) de FD002.

4.3.5.4 Les résultats avec différentes couches

1. **Avec CNN_LSTM :**Le tableau 7 montre les résultats obtenus à partir du cadre CNN_LSTM proposé en utilisant des différents nombres de couches pour FD001 après 10 époques, généralement, 2 couches se sont révélées suffisantes pour détecter des entités plus complexes. Plus de couches peuvent être plus difficiles à entraîner. En règle générale, une couche cachée fonctionne avec des problèmes simples, et deux suffisent pour trouver des fonctionnalités raisonnablement complexes [47]. Dans notre cas, l'ajout d'une troisième couche n'améliore que les métriques de test par rapport au résultats de deux couches (des résultats convergent) après 10 époques. Mais l'ajout de quatrième ,cinquième et sixième couche donne de faibles performances soit de l'entrainement ou du test. Pour FD001 (Epoch=10, activation=relu, optimizer= rmsprop, WS=70)

Algorithm	train		Test		temp d'execution
	MSE	R^2	MSE	R^2	
CNN_LSTM (2 couches cnn et 2 lstm)	**255.37**	**0.89**	497.44	0.70	12612.98
CNN_LSTM (3 couches cnn et 3 lstm)	539.73	0.78	354.90	0.78	16928.14

TABLE 4.7 – les résultats des métriques de régression pour CNN_LSTM.

2. **Avec LSTM :**D'après le tableau 8, nous observons les résultats des métriques d'évaluation après l'entraînement de modèles LSTM pendant 100 époques pour l'ensemble de donnée FD001. Nous testons le modèle avec différentes couches, Afin de savoir combien de couches conviennent à notre modèle, nous observons que l'utilisation de trois couches a donné les meilleurs résultats qui sont la valeur de performance est égale à 0.76 et la valeur d'erreur égale à 644.17 pour l'entraînement, donc les résultats final son affiché avec trois couches dans les tableaux 11 et 12 pour FD001 et FD004 : Pour FD001 (Epoch=10, activation=relu, optimizer= rmsprop, WS=70)

Algorithm	train		Test		temp d'execution
	MSE	R^2	MSE	R^2	
Regression LSTM (2 couches lstm)	727.02	0.75	393.06	0.76	7656.1385
Regression LSTM (3 couches lstm)	644.17	0.76	302.52	0.81	5733.32

TABLE 4.8 – les résultats des métriques de régression pour LSTM.

4.3.5.5 Les résultats finals des cinq modèles proposés

Les résultats finaux ont été obtenus après l'évaluation des cinq modèles proposés de sous data sets (FD001) après 100 époques avec l'utilisation des paramètres que nous avons choisi au dessus ce qui rend le modèle plus performant. Nous avons testé le modèle avec trois fonctions d'erreur MSE, RMSE et MAE, Afin de comparer et savoir quelle métrique correspond à quel modèle. Pendant la recherche, nous avons trouvé que RMSE est la métrique la plus utilisée pour les tâches de régression et correspond à la racine carrée de la différence quadratique moyenne entre la valeur cible et la valeur prédite par le modèle. Il est préférable dans certains cas car les erreurs sont d'abord mises au carré avant le calcul de la moyenne, ce qui entraîne une forte pénalité sur les erreurs importantes. Cela implique que RMSE est utile lorsque des erreurs importantes ne sont pas souhaitées[48].Et MAE est plus robuste aux valeurs aberrantes et ne pénalise pas les erreurs aussi fortement que MSE. Et que MSE est l'une des mesures les plus préférées pour les tâches de régression. En corrigeant les différences, cela pénalise même une petite erreur qui conduit à une surestimation de la gravité du modèle. Il est préférable aux autres métriques car il est différenciable et peut donc être mieux optimisé[48].Et nos résultats prouvent cette affirmation avec l'utilisation des CNN_LSTM et LSTM qui dans Le tableau 9, 10 et 11 nous observons quand on utilise CNN_LSTM la valeur de performance pour MSE égale à 0.98 alors qu'il est égal à 0.95 et 0.96 pour RMSE et MAE, et quand on utilise LSTM la valeur de performance pour MSE, RMSE et MAE égale à 0.98, -1.39 et 0.92 . Donc ces deux modèles conviennent avec la métrique MSE.

Comme Nous avons vu , le modèle Bidirectionnel donne des résultats plus faibles, Et la raison comme on montre au début dans la section 3.1 qu'il est peut améliorer les performances du modèle sur les problèmes de classification de séquence[49].Et notre problème est un problème de régression, donc ce modèle n'est pas efficace pour notre travail. Mais l'hybridation avec les modèles CNN et LSTM (comme notre proposition CNN_BI_LSTM et CNN_Bidirectional) donne de bons résultats, qui dans CNN_BI_LSTM la valeur de performance égale à 0.97 avec l'utilisation des métriques MSE, MAE et 0.99 avec RMSE. Comme nous n' ;avions pas trouvé ce modèle auparavant dans la littérature, c'est nous qui l'avons proposé, et nous obtenons des bons résultats.

Le CNN_Bidirectional aussi efficace pour notre travail, Il est également peu utilisé dans la littérature, Mais nous voulions l'essayer dans notre travail, et nous avons obtenu de bons résultats, qui la valeur de performance égale à 0.96 , 0.97 et 0.99 avec l'utilisation des trois métriques MSE, MAE et RMSE. Avec ce modèle, nous avons obtenu les meilleures performances par rapport à tous nos modèles, où la valeur de performance atteignant à 0.99 en utilisant la métrique RMSE. Donc le modèle CNN_Bidirectional convient à la métrique RMSE. (Epoch=100, Activation=Relu, Optimizer= Rmsprop, WS=70)

(Epoch=100, Activation=Relu, Optimizer= Rmsprop, WS=70)

Algorithm	train		Test		temp d'execution
	MSE	R^2	MSE	R^2	
CNN_LSTM **FD001**	**27.68**	**0.98**	736.53	0.60	73554.83
LSTM **FD001**	**27.43**	**0.98**	846.46	0.53	49327.57
Biderectional **FD001**	375.08	0.83	1048.33	0.37	11357.41
CNN_BI_LSTM **FD001**	**45.81**	**0.97**	**463.07**	**0.73**	117060.54
CNN_Bidirectional **FD001**	**37.45**	**0.96**	**368.43**	**0.79**	41864.96

TABLE 4.9 – les résultats finals des modèles proposés avec métrique MSE.

Algorithm	train		Test		temp d'execution
	RMSE	R^2	RMSE	R^2	
CNN_LSTM **FD001**	4.65	0.95	14.45	0.64	95857.56
LSTM **FD001**	7.83	0.93	21.23	0.73	43068.02
Biderectional **FD001**	10.14	0.93	23.13	0.65	11524.32
CNN_BI_LSTM **FD001**	**6.63**	**0.95**	**16.12**	**0.85**	148546.073
CNN_Bidirectional **FD001**	2.81	**0.99**	24.28	**0.64**	39269.83

TABLE 4.10 – les résultats finals des modèles proposés avec métrique RMSE.

Algorithm	train		Test		temp d'execution
	MAE	R^2	MAE	R^2	
CNN_LSTM **FD001**	3.10	0.96	13.50	0.65	105857.56
LSTM **FD001**	5.13	0.92	15.67	0.51	47866.90
Biderectional **FD001**	10.6	0.87	12.82	0.72	10800.34
CNN_BI_LSTM **FD001**	**3.97**	**0.97**	**14.55**	**0.73**	117060.54
CNN_Bidirectional **FD001**	**2.24**	**0.97**	**11.45**	**0.81**	43142.04

TABLE 4.11 – les résultats finals des modèlesproposés avec métrique MAE.

4.3.5.6 Les plots des résultats finals

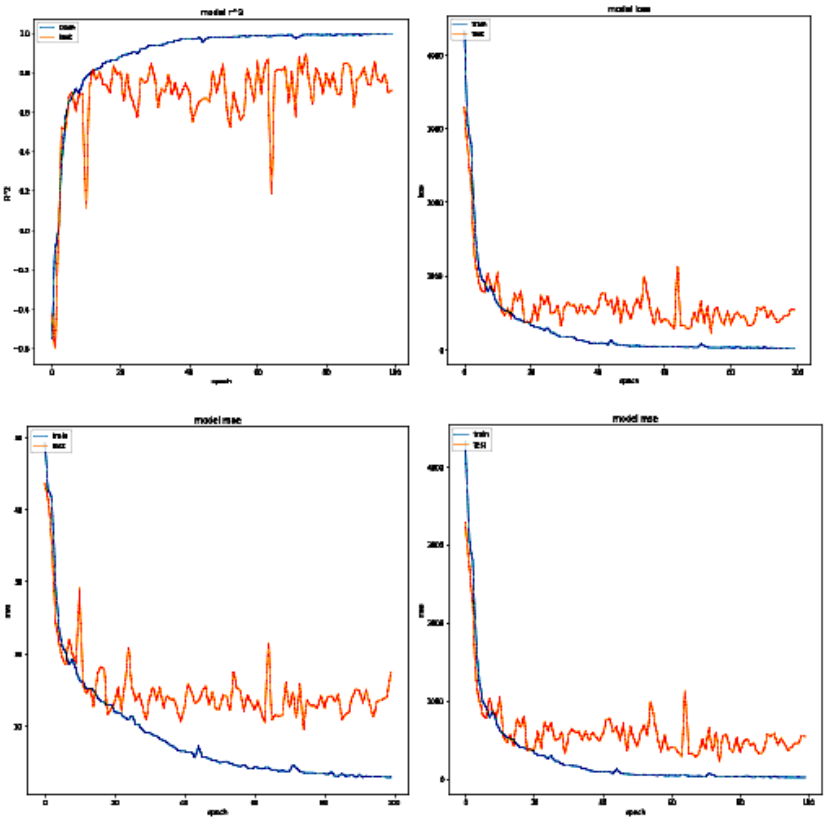


FIGURE 4.6 – Les plots des métriques MSE, MAE, RMSE et R^2 avec le modèle (CNN_BI_LSTM) pour la sous donnée FD001 de CMAPSS.

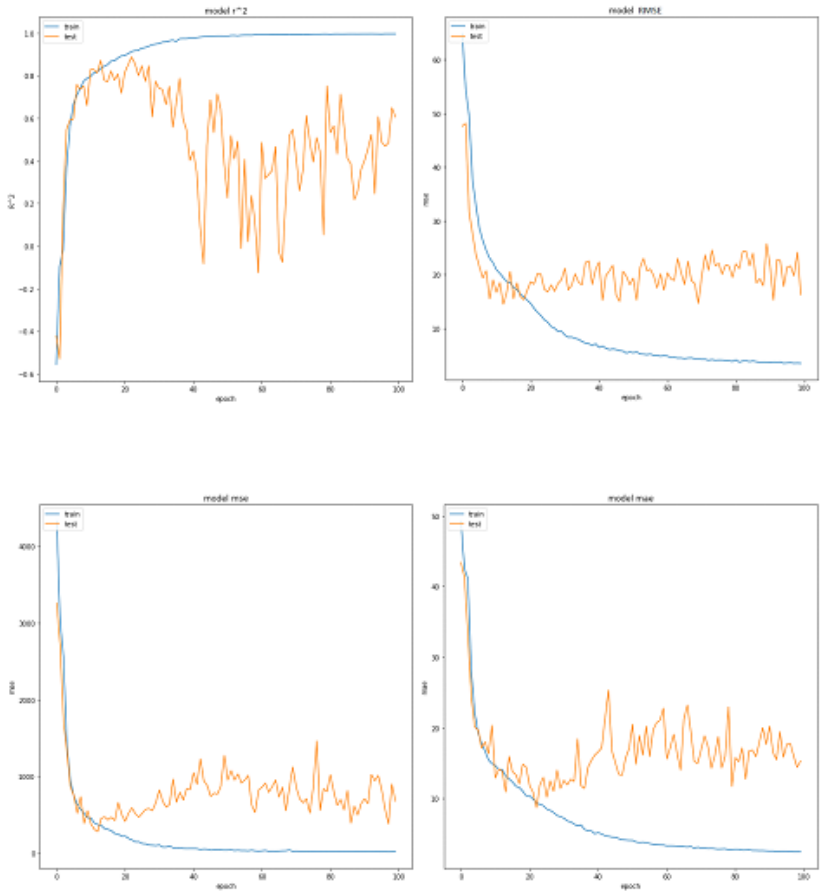


FIGURE 4.7 – Les plots des métriques MSE, MAE, RMSE et R^2 avec le modèle (CNN_Bidirectional) pour la sous donnée FD001 de CMAPSS.

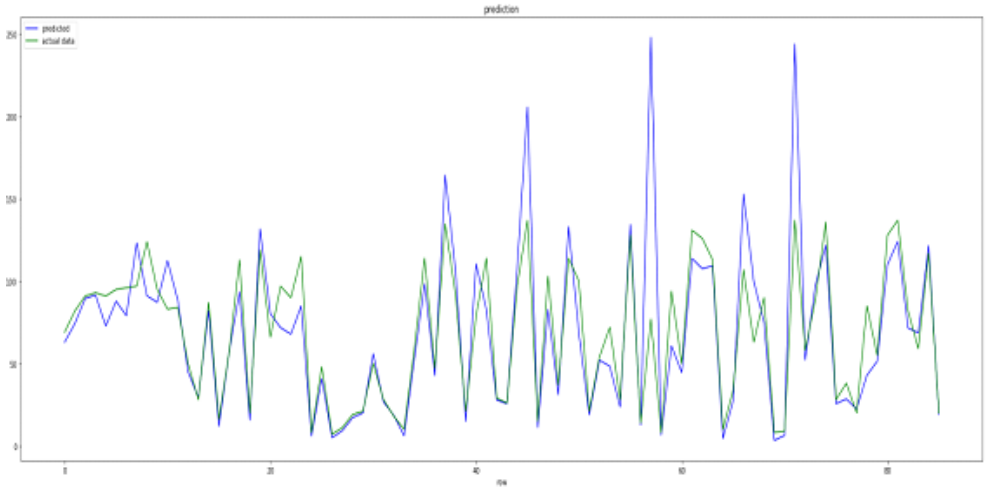


FIGURE 4.8 – Résultat de prédiction de RUL avec le modèle (CNN_LSTM) basé sur la sous données de FD001de C-MAPSS.

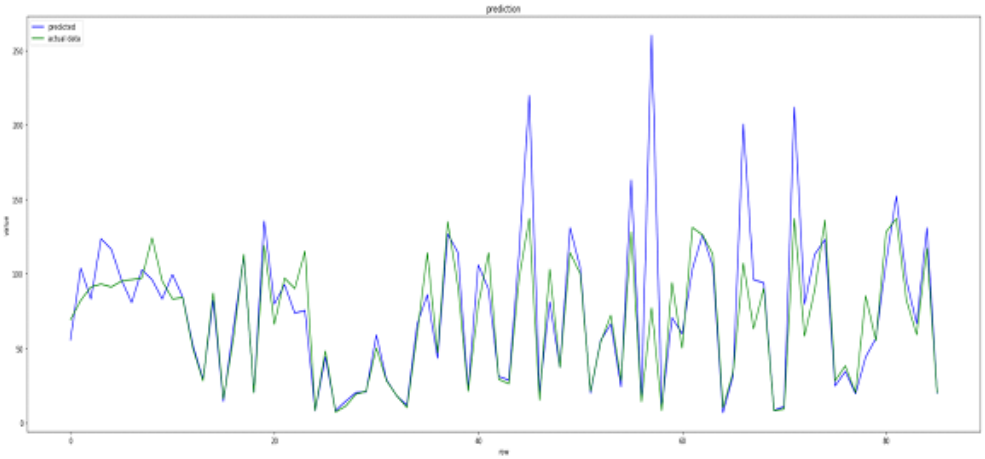


FIGURE 4.9 – Résultat de prédiction de RUL avec le modèle (LSTM) basé sur la sous donnée FD001 de CMAPSS.

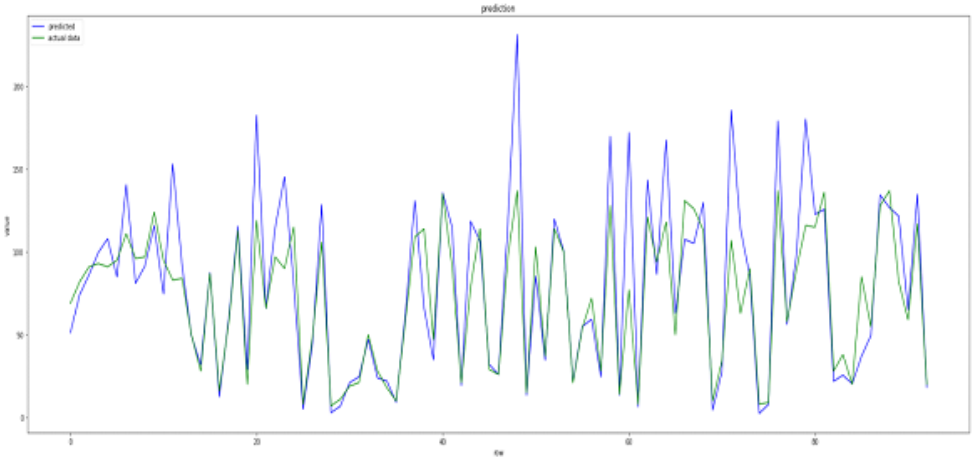


FIGURE 4.10 – Résultat de prédiction de RUL avec le modèle (Bidirectionnel LSTM) basé sur la sous donnée FD001 de CMAPSS.

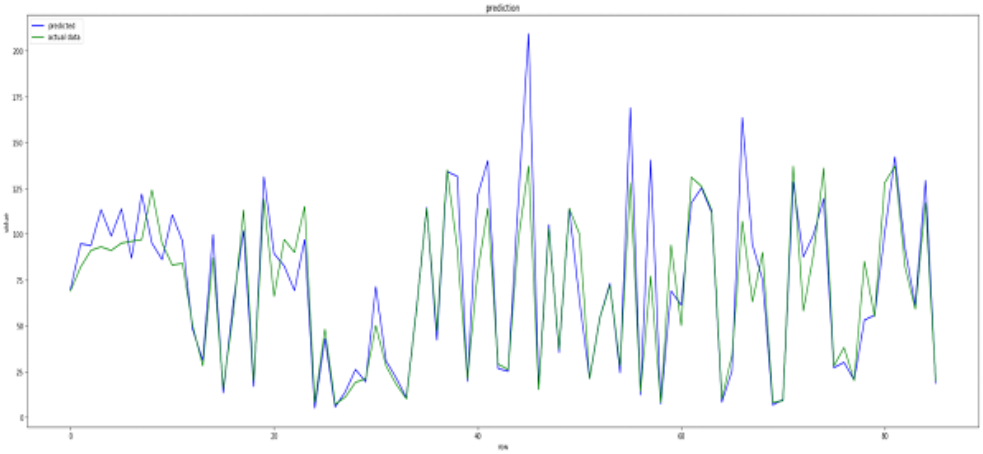


FIGURE 4.11 – Résultat de prédiction de RUL avec le modèle (CNN_BI_LSTM) basée sur la sous donnée FD001 de CMAPSS..

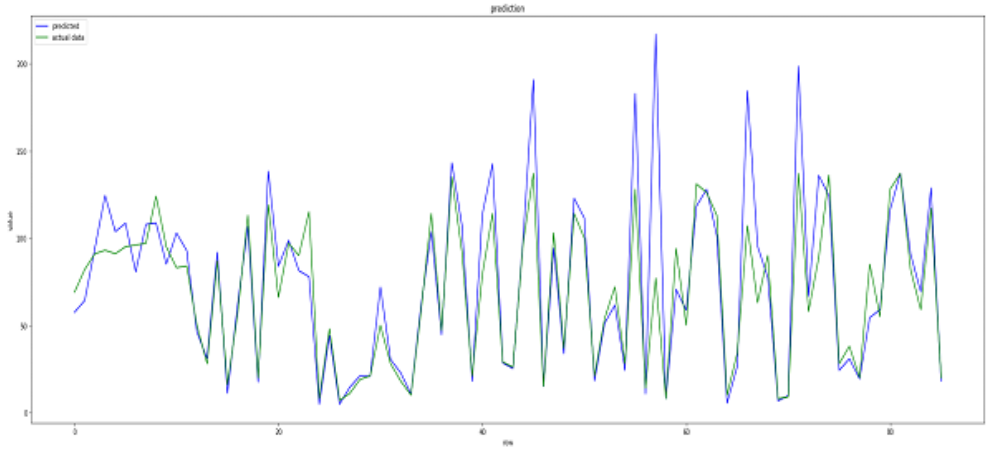


FIGURE 4.12 – Résultat de prédiction de RUL avec le modèle (CNN_Bidirectional) basée sur la sous données FD001de C-MAPSS.

4.3.6 Comparaison entre nos résultats et quelques travaux de la littérature

Comme nous l'avons mentionné précédemment, le remaining useful life est un problème de prédiction de séries chronologiques (time series) qui fait partie d'un projet de régression en utilisant l'ensemble de données NASA CMAPPS. Dans ce projet, nous avons effectué cinq modèles différents de régression qui sont CNN_LSTM, LSTM, CNN_BI_LSTM et CNN_Bidirectional. Dans le tableau suivant nous résumons tous les résultats obtenus de ce projet et les résultats des autres travaux connexes de la littérature utilisant les mêmes données que nous :

les modèles	RMSE
HDNN [14]	13.017
DCNN [14]	12.61
DLSTM [14]	16.14
DLSTM [50]	18.70
CNN_LSTM proposé	14.45
CNN_Bidi_LSTM proposé	16.12
CNN_Bidi proposé	24.28
LSTM proposé	21.23

TABLE 4.12 – Tableau comparatif entre nos résultats et ceux des travaux de la litérature de test en terme de RMSE

Comme nous l'observons dans le tableaux 12, nos modèles sont classés parmi les bons ; il ya quelques modèles qui nous surpassent comme le HDNN [14] et le DCNN [14] mais pour les autres modèles, on les surpasse avec de bonnes valeurs de métriques de régression. Cela nous montre l'efficacité des deep learning LSTM pour ce type de problème, et montre aussi

l'avantage de l'hybridation des modèles de Deep Learning dans ce problème. L'hybridation permet

de profiter des avantages de tous les modèles incorporés. Ainsi les résultats montrent l'importance du choix des paramètres qui composent le réseau pour obtenir des résultats satisfaisants et la nécessité de les prendre en compte pour choisir le meilleur réseau.

Conclusion

Nous avons proposé dans ce chapitre un scénario qui présente quelque modèle de régression pour faire la prédiction des meilleurs RUL, nous avons utilisé un algorithme du deep learning qui est le RNN , ensuite nous avons présenté les résultats avec comparaison pour choisir le meilleur modèle et nous avons vu le changement des résultats après avoir fait varier les quelques paramètres du RNN.

CHAPITRE 5

IMPLÉMENTATION

Introduction

Dans ce chapitre, nous présenterons l'implémentation des approches que nous avons proposées ; les outils utilisés et les pseudo codes implémentés ; ainsi que l'interface que nous avons créé pour lancer les différentes approches et pour l'affichage des différents résultats.

5.1 Les outils de développement

Dans notre travail nous avons utilisé Python comme langage de programmation, Anaconda comme environnement de programmation et Spyder comme outil de programmation.

5.1.1 Le langage Python

Python est un langage de programmation open source créé par le programmeur Guido van Rossum en 1991. Il tire son nom de l'émission Monty Python's Flying Circus. Il s'agit d'un langage de programmation interprété, qui ne nécessite donc pas d'être compilé pour fonctionner. Un programme »interpréteur » permet d'exécuter le code Python sur n'importe quel ordinateur. Ceci permet de voir rapidement les résultats d'un changement dans le code. En revanche, ceci rend ce langage plus lent qu'un langage compilé comme le C . Récemment, Python occupe le monde de l'intelligence artificielle, en particulier l'apprentissage profond[52].

5.1.2 Environnement Anaconda

Anaconda est une distribution open source pour Python et R. Il est utilisé pour la science des données, l'apprentissage automatique, l'apprentissage en profondeur, etc. Avec la disponibilité de plus de 300 bibliothèques pour la science des données, il devient assez optimal pour tout programmeur de travailler sur Anaconda pour science des données. Anaconda aide à simplifier la gestion et le déploiement des packages. Anaconda est livré avec une grande variété d'outils pour collecter facilement des données à partir de diverses sources à l'aide de divers algorithmes d'apprentissage automatique et d'IA[53].

5.1.3 L'outil Spyder

Spyder est un environnement scientifique puissant écrit en Python, pour Python, et conçu par et pour des scientifiques, des ingénieurs et des analystes de données. Il présente une combinaison unique des fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet avec l'exploration de données, l'exécution interactive, l'inspection approfondie et les superbes capacités de visualisation d'un package scientifique. En outre, Spyder offre une intégration intégrée avec de nombreux packages scientifiques populaires, notamment NumPy, SciPy, Pandas, IPython, QtConsole, Matplotlib, SymPy, etc. Au-delà de ses nombreuses fonctionnalités intégrées, Spyder peut être encore étendu via des plugins tiers. Spyder peut également être utilisé comme bibliothèque d'extension PyQt5, vous permettant de tirer parti de ses fonctionnalités et d'intégrer ses composants, tels que la console interactive ou l'éditeur avancé, dans votre propre logiciel [54].

5.1.4 Les bibliothèques Tensorflow et Keras

Le deep learning utilise des divers frameworks ou bibliothèques qui offrent un moyen pratique de définir et de former presque tous les types de ses modèles, les outils les plus utilisées dans le deep learning Keras, TensorFlow, Theano, and CNTK. Dans notre travail, nous avons utilisé Kers et Tonsorflow. Keras est une bibliothèque de niveau modèle, fournissant des éléments de base de haut niveau pour le développement de modèles d'apprentissage en profondeur. Il ne gère pas les opérations de bas niveau telles que la manipulation et la différenciation des tenseurs. Au lieu de cela, il repose sur une bibliothèque de tenseurs spécialisée et bien optimisée pour ce faire, servant de moteur backend de Keras. TensorFlow, CNTK et Theano font aujourd'hui partie des principales plates-formes d'apprentissage en profondeur. Theano est développé par le laboratoire MILA de l'Université de Montréal, TensorFlow est développé par Google et CNTK est développé par Microsoft. Tout morceau de code écrit avec Keras peut être exécuté avec

l'un de ces backends sans avoir à changer quoi que ce soit dans le code [26].

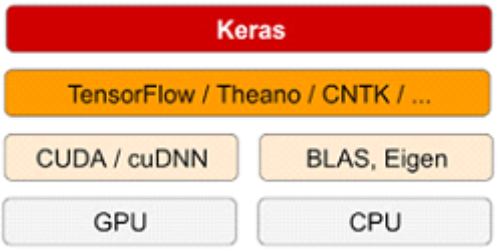


FIGURE 5.1 – pile matérielle et logicielle de deep learning.

5.2 Les etapes de travail

Dans notre travail, nous construisons tous les modèles proposé basé sur le réseau LSTM afin de prédire la durée de vie utile restante (remaining useful life RUL en englais) des moteurs d'avion. Le réseau utilise des valeurs de capteur d'avion simulées pour prédire quand un moteur d'avion tombera en panne dans le futur afin que la maintenance puisse être planifiée à l'avance. Nous pouvons définir le problème comme un problème de régression, car il est donné un nombre en sortie (le nombre de cycles d'un moteur en service durera avant de tomber en panne).

- tout d'abord, charger l'ensemble de données en tant que dataframe Pandas. (nous utiliserons les données C MAPSS dataset de la NASA) et calculer les valeurs de rul comme suit : Selon la description de données fournie au niveau de la source de données, les données d'apprentissage (train-FD00X) sont constituées de multiples séries temporelles multi variées avec cycle comme unité de temps, conjointement avec 21 lectures de capteur pour chaque cycle. Chaque série chronologique peut être supposée être générée à partir d'un moteur différent du même type. Chaque moteur est supposé démarrer avec différents degrés d'usure initiale et de variation de fabrication, et cette information est inconnue de l'utilisateur. Dans ces données simulées, le moteur est supposé fonctionner normalement au début de chaque série temporelle. Il commence à se dégrader à un moment donné pendant la série des cycles de fonctionnement. Cela dégrade les progrès et augmente en ampleur. Lorsqu'un seuil prédéfini est atteint, le moteur est considéré comme dangereux pour un fonctionnement ultérieur. En d'autres termes, le dernier cycle de chaque série temporelle peut être considéré comme le point de défaillance du moteur correspondant. En prenant l'exemple de données d'apprentissage, le moteur avec id=1 échoue au cycle 192, et le moteur avec id=2 échoue au cycle 287.

  Les données de test (test-FD00X) ont le même schéma de données que les données

de traitement. La seule différence est que les données n'indiquent pas quand la panne se produit (en d'autres termes, la dernière période de temps ne représente PAS le point de panne). En prenant les données de test d'échantillon, le moteur avec id=1 fonctionne du cycle 1 au cycle 31. il n'est pas indiqué combien de cycles supplémentaires ce moteur peut durer avant de tomber en panne.

Les données de vérité (RUL-FD00X) fournissent le nombre de cycles de travail restants pour les moteurs dans les données de test. En prenant l'échantillon de données de vérité terrain montré à titre d'exemple, le moteur avec id = 1 dans les données de test peut exécuter un autre cycle 112 avant de tomber en panne. la figure suivante illustre un morcaux de code qui liee l'ensemble des donnees FD001

- Ensuit, Nous pouvons facilement normaliser l'ensemble de données à l'aide de la classe de prétraitement MinMaxScaler de la bibliothèque scikit-learn.

- puisque nous utilisons LSTM pour la modélisation de séries chronologiques (time-series), nous créons une fonction qui générera la séquence à envoyer au LSTM selon la taille de la fenêtre.
- Nous sommes maintenant prêts à concevoir et adapter notre réseaux à ce problèm(les cinq moèles proposés). Une fois le modèle ajusté, nous pouvons estimer les performances du modèle sur les ensembles de données de train et de test. nous utilisons comme métriques d'evaluation de performance R^2 , Mean Squared Error (MSE) , mean absolute error (MAE) et root mean squared error (RMSE).
- Enfin, nous pouvons générer des prédictions à l'aide du modèle pour l'ensemble de données de train et de test afin d'obtenir le vecteur de RUL prédits.

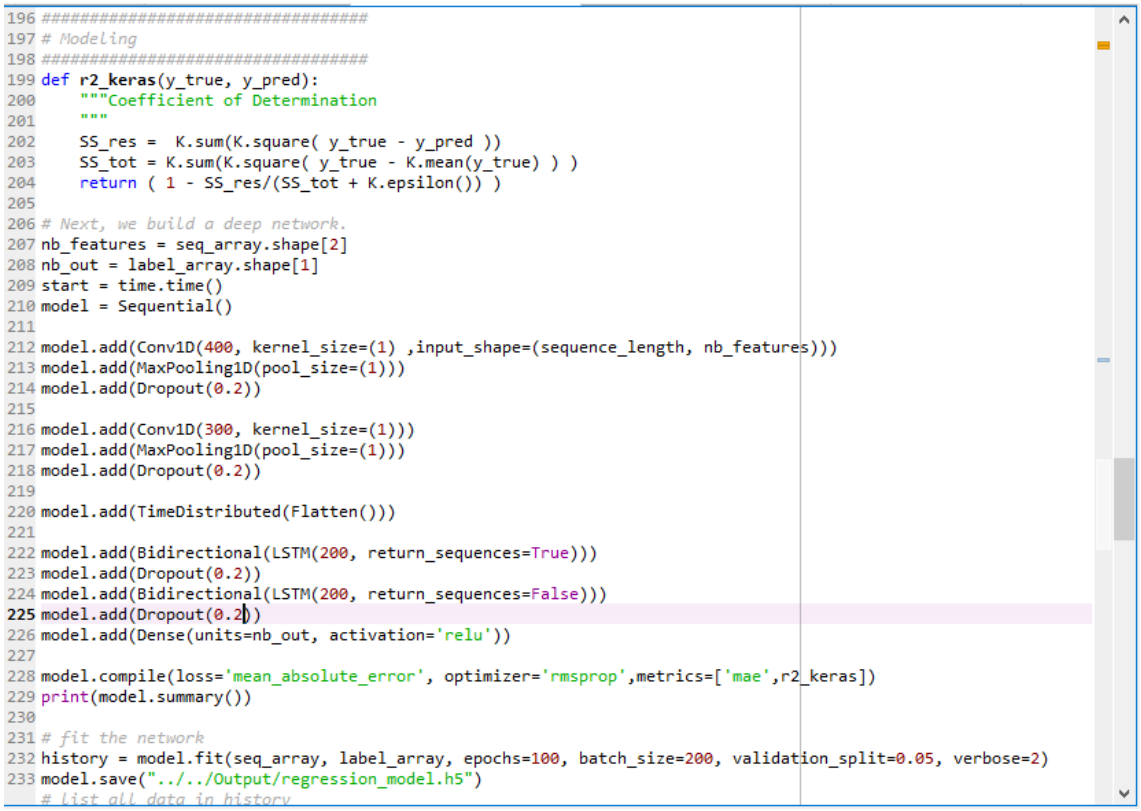
5.3 Fragments de code

```
# TRAIN
rul = pd.DataFrame(train_df.groupby('id')['cycle'].max()).reset_index()
rul.columns = ['id', 'max']
train_df = train_df.merge(rul, on=['id'], how='left')
train_df['RUL'] = train_df['max'] - train_df['cycle']
train_df.drop('max', axis=1, inplace=True)
print(rul)
# MinMax normalization (from 0 to 1)
train_df['cycle_norm'] = train_df['cycle']
cols_normalize = train_df.columns.difference(['id','cycle','RUL'])
min_max_scaler = preprocessing.MinMaxScaler()
norm_train_df = pd.DataFrame(min_max_scaler.fit_transform(train_df[cols_normalize]), columns=cols_normalize,
                             index=train_df.index)
join_df = train_df[train_df.columns.difference(cols_normalize)].join(norm_train_df)
train_df = join_df.reindex(columns = train_df.columns)
print('normalization train')
print(train_df)
#######
# TEST
# MinMax normalization (from 0 to 1)
test_df['cycle_norm'] = test_df['cycle']
norm_test_df = pd.DataFrame(min_max_scaler.transform(test_df[cols_normalize]), columns=cols_normalize,
                            index=test_df.index)
test_join_df = test_df[test_df.columns.difference(cols_normalize)].join(norm_test_df)
test_df = test_join_df.reindex(columns = test_df.columns)
test_df = test_df.reset_index(drop=True)
print('normalization test')
print(test_df)
rul = pd.DataFrame(test_df.groupby('id')['cycle'].max()).reset_index()
rul.columns = ['id', 'max']
print(rul)
truth_df.columns = ['more']
truth_df['id'] = truth_df.index + 1
truth_df['max'] = rul['max'] + truth_df['more']
print(truth_df)
truth_df.drop('more', axis=1, inplace=True)
print(truth_df)
# generate RUL for test data
```

FIGURE 5.2 – Normalisation des données.

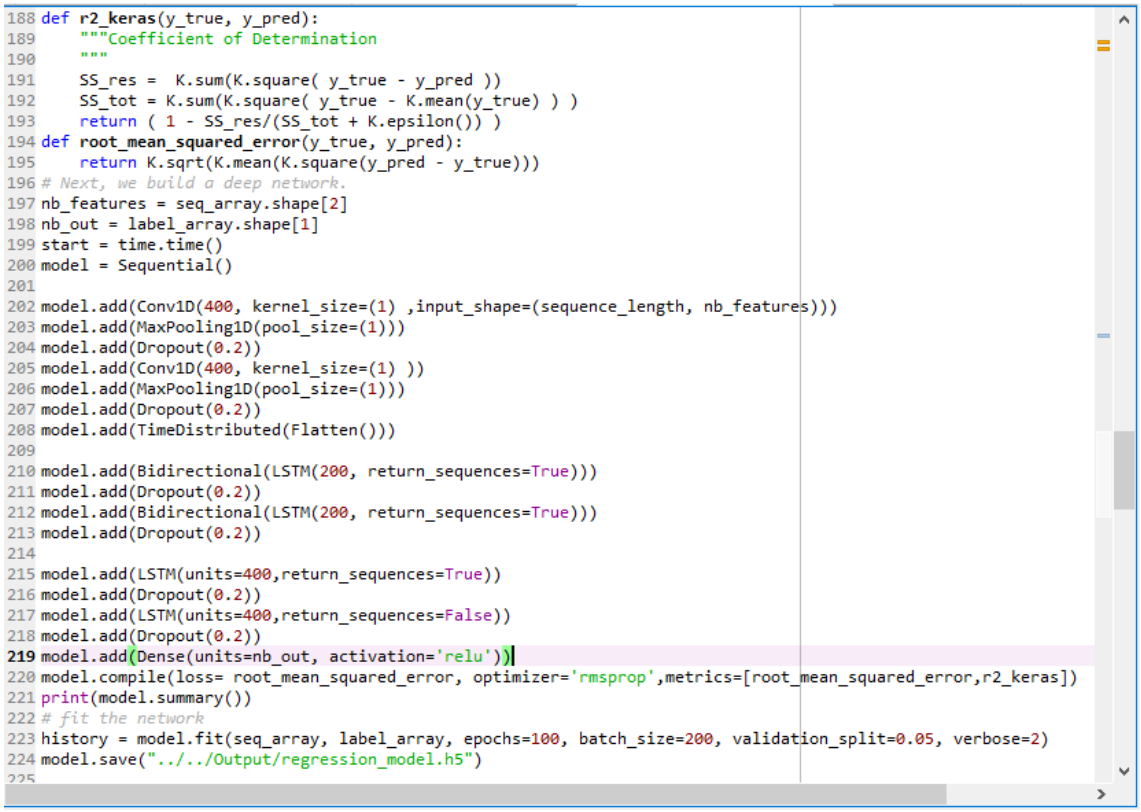
```
# Modeling
#################################
def r2_keras(y_true, y_pred):
    """Coefficient of Determination
    """
    SS_res =  K.sum(K.square( y_true - y_pred ))
    SS_tot = K.sum(K.square( y_true - K.mean(y_true) ) )
    return ( 1 - SS_res/(SS_tot + K.epsilon()) )
# Next, we build a deep network.
nb_features = seq_array.shape[2]
nb_out = label_array.shape[1]

model = Sequential()

model.add(Conv1D(400, kernel_size=(1) ,input_shape=(sequence_length, nb_features)))
model.add(MaxPooling1D(pool_size=(1)))
model.add(Dropout(0.2))
model.add(Conv1D(400, kernel_size=(1)))
model.add(MaxPooling1D(pool_size=(1)))
model.add(Dropout(0.2))
model.add(TimeDistributed(Flatten()))
model.add(LSTM(
         units=400,
         return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(
          units=400,
          return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(units=nb_out, activation='relu'))

model.compile(loss='mean_squared_error', optimizer='rmsprop',metrics=['mse',r2_keras])
print(model.summary())

# fit the network
history = model.fit(seq_array, label_array, epochs=100, batch_size=200, validation_split=0.05, verbose=2)
model.save("../../Output/regression_model.h5")
# list all data in history
```

FIGURE 5.3 – Le modèle CNN_LSTM.



```
196 ##################################
197 # Modeling
198 ##################################
199 def r2_keras(y_true, y_pred):
200     """Coefficient of Determination
201     """
202     SS_res =  K.sum(K.square( y_true - y_pred ))
203     SS_tot = K.sum(K.square( y_true - K.mean(y_true) ) )
204     return ( 1 - SS_res/(SS_tot + K.epsilon()) )
205
206 # Next, we build a deep network.
207 nb_features = seq_array.shape[2]
208 nb_out = label_array.shape[1]
209 start = time.time()
210 model = Sequential()
211
212 model.add(Conv1D(400, kernel_size=(1) ,input_shape=(sequence_length, nb_features)))
213 model.add(MaxPooling1D(pool_size=(1)))
214 model.add(Dropout(0.2))
215
216 model.add(Conv1D(300, kernel_size=(1)))
217 model.add(MaxPooling1D(pool_size=(1)))
218 model.add(Dropout(0.2))
219
220 model.add(TimeDistributed(Flatten()))
221
222 model.add(Bidirectional(LSTM(200, return_sequences=True)))
223 model.add(Dropout(0.2))
224 model.add(Bidirectional(LSTM(200, return_sequences=False)))
225 model.add(Dropout(0.2))
226 model.add(Dense(units=nb_out, activation='relu'))
227
228 model.compile(loss='mean_absolute_error', optimizer='rmsprop',metrics=['mae',r2_keras])
229 print(model.summary())
230
231 # fit the network
232 history = model.fit(seq_array, label_array, epochs=100, batch_size=200, validation_split=0.05, verbose=2)
233 model.save("../../Output/regression_model.h5")
    # list all data in history
```

FIGURE 5.4 – Le modèle CNN_Bidirectionnel.



```
188 def r2_keras(y_true, y_pred):
189     """Coefficient of Determination
190     """
191     SS_res =  K.sum(K.square( y_true - y_pred ))
192     SS_tot = K.sum(K.square( y_true - K.mean(y_true) ) )
193     return ( 1 - SS_res/(SS_tot + K.epsilon()) )
194 def root_mean_squared_error(y_true, y_pred):
195     return K.sqrt(K.mean(K.square(y_pred - y_true)))
196 # Next, we build a deep network.
197 nb_features = seq_array.shape[2]
198 nb_out = label_array.shape[1]
199 start = time.time()
200 model = Sequential()
201
202 model.add(Conv1D(400, kernel_size=(1) ,input_shape=(sequence_length, nb_features)))
203 model.add(MaxPooling1D(pool_size=(1)))
204 model.add(Dropout(0.2))
205 model.add(Conv1D(400, kernel_size=(1) ))
206 model.add(MaxPooling1D(pool_size=(1)))
207 model.add(Dropout(0.2))
208 model.add(TimeDistributed(Flatten()))
209
210 model.add(Bidirectional(LSTM(200, return_sequences=True)))
211 model.add(Dropout(0.2))
212 model.add(Bidirectional(LSTM(200, return_sequences=True)))
213 model.add(Dropout(0.2))
214
215 model.add(LSTM(units=400,return_sequences=True))
216 model.add(Dropout(0.2))
217 model.add(LSTM(units=400,return_sequences=False))
218 model.add(Dropout(0.2))
219 model.add(Dense(units=nb_out, activation='relu'))
220 model.compile(loss= root_mean_squared_error, optimizer='rmsprop',metrics=[root_mean_squared_error,r2_keras])
221 print(model.summary())
222 # fit the network
223 history = model.fit(seq_array, label_array, epochs=100, batch_size=200, validation_split=0.05, verbose=2)
224 model.save("../../Output/regression_model.h5")
225
```

FIGURE 5.5 – Le modèle CNN_Bid_lstm.

```python
def r2_keras(y_true, y_pred):
    """Coefficient of Determination
    """
    SS_res =  K.sum(K.square( y_true - y_pred ))
    SS_tot = K.sum(K.square( y_true - K.mean(y_true) ) )
    return ( 1 - SS_res/(SS_tot + K.epsilon()) )
def root_mean_squared_error(y_true, y_pred):
    return K.sqrt(K.mean(K.square(y_pred - y_true)))
# Next, we build a deep network.
nb_features = seq_array.shape[2]
nb_out = label_array.shape[1]
start = time.time()

model = Sequential()
model.add(LSTM(
         input_shape=(sequence_length, nb_features),
         units=400,
         return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(
          units=400,
          return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(
          units=400,
          return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(units=nb_out))
model.add(Activation("relu"))

model.compile(loss= root_mean_squared_error, optimizer='rmsprop',metrics=[root_mean_squared_error,r2_keras])
print(model.summary())
# fit the network
history = model.fit(seq_array, label_array, epochs=100, batch_size=200, validation_split=0.2, verbose=2)
model.save("../../Output/regression_model.h5")
```

FIGURE 5.6 – Le modèle LSTM.

5.3.1 Exemples d'exécution

```
Epoch 95/100

 - 411s - loss: 19.5322 - mse: 19.5322 - r2_keras: 0.9928 - val_loss: 1012.4494 - val_mse: 1012.4494 -
val_r2_keras: 0.4551

Epoch 96/100

 - 411s - loss: 17.6899 - mse: 17.6899 - r2_keras: 0.9936 - val_loss: 828.0700 - val_mse: 828.0700 -
val_r2_keras: 0.5609

Epoch 97/100

 - 411s - loss: 18.9969 - mse: 18.9969 - r2_keras: 0.9931 - val_loss: 575.0517 - val_mse: 575.0516 -
val_r2_keras: 0.7291

Epoch 98/100

 - 409s - loss: 16.3638 - mse: 16.3638 - r2_keras: 0.9940 - val_loss: 382.3294 - val_mse: 382.3294 -
val_r2_keras: 0.8155

Epoch 99/100

 - 414s - loss: 18.0660 - mse: 18.0660 - r2_keras: 0.9934 - val_loss: 899.1978 - val_mse: 899.1979 -
val_r2_keras: 0.6020

Epoch 100/100

 - 409s - loss: 20.8010 - mse: 20.8010 - r2_keras: 0.9925 - val_loss: 680.6637 - val_mse: 680.6637 -
val_r2_keras: 0.6048

dict_keys(['val_loss', 'val_mse', 'val_r2_keras', 'loss', 'mse', 'r2_keras'])


13631/13631 [==============================] - 128s 9ms/step

MSE: 37.449954986572266

R^2: 0.9641234874725342
```

FIGURE 5.7 – l'exécution de modèle CNN-bid avec MSE.

```
Epoch 95/100
 - 449s - loss: 18.3844 - mse: 18.3844 - r2_keras: 0.9933 - val_loss: 665.7147 - val_mse: 665.7148 -
val_r2_keras: 0.6700
Epoch 96/100
 - 449s - loss: 25.6546 - mse: 25.6546 - r2_keras: 0.9907 - val_loss: 554.8393 - val_mse: 554.8392 -
val_r2_keras: 0.7204
Epoch 97/100
 - 449s - loss: 24.3220 - mse: 24.3220 - r2_keras: 0.9908 - val_loss: 686.2722 - val_mse: 686.2722 -
val_r2_keras: 0.6509
Epoch 98/100
 - 451s - loss: 19.2030 - mse: 19.2030 - r2_keras: 0.9929 - val_loss: 952.8217 - val_mse: 952.8218 -
val_r2_keras: 0.3829
Epoch 99/100
 - 449s - loss: 22.2505 - mse: 22.2505 - r2_keras: 0.9918 - val_loss: 700.3920 - val_mse: 700.3920 -
val_r2_keras: 0.6251
Epoch 100/100
 - 448s - loss: 17.9890 - mse: 17.9890 - r2_keras: 0.9935 - val_loss: 423.0785 - val_mse: 423.0785 -
val_r2_keras: 0.7714
dict_keys(['val_loss', 'val_mse', 'val_r2_keras', 'loss', 'mse', 'r2_keras'])
13631/13631 [==============================] - 149s 11ms/step
MSE: 27.676753997802734
R^2: 0.9806690859794617
```

FIGURE 5.8 – l'exécution de modèle CNN-lstm avec MSE.

5.4 L'interface

Nous avons créé une interface graphique qui nous permet de communiquer avec nos modèles, elle est représenté dans la figure 5.8.

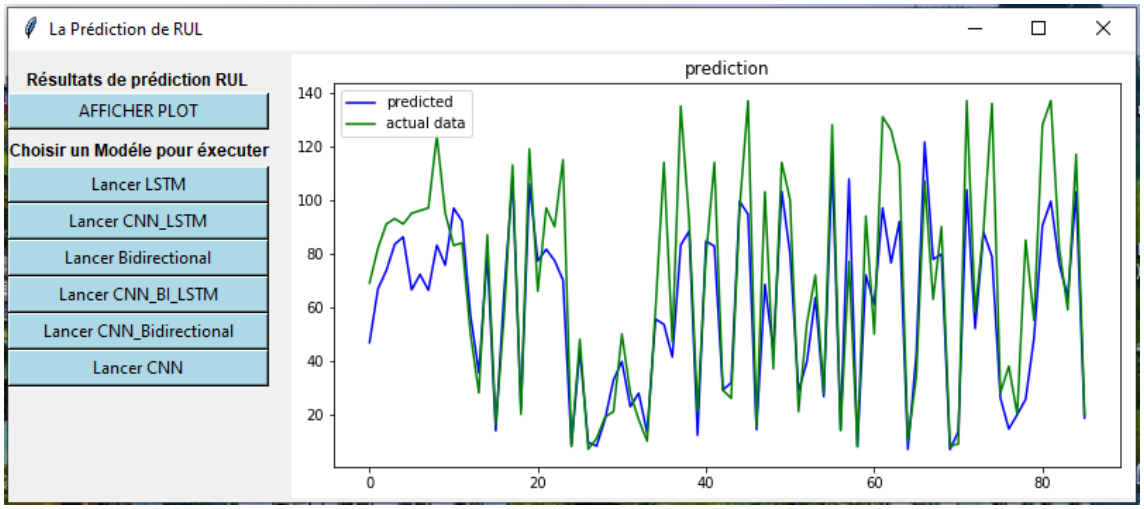


FIGURE 5.9 – L'interface.

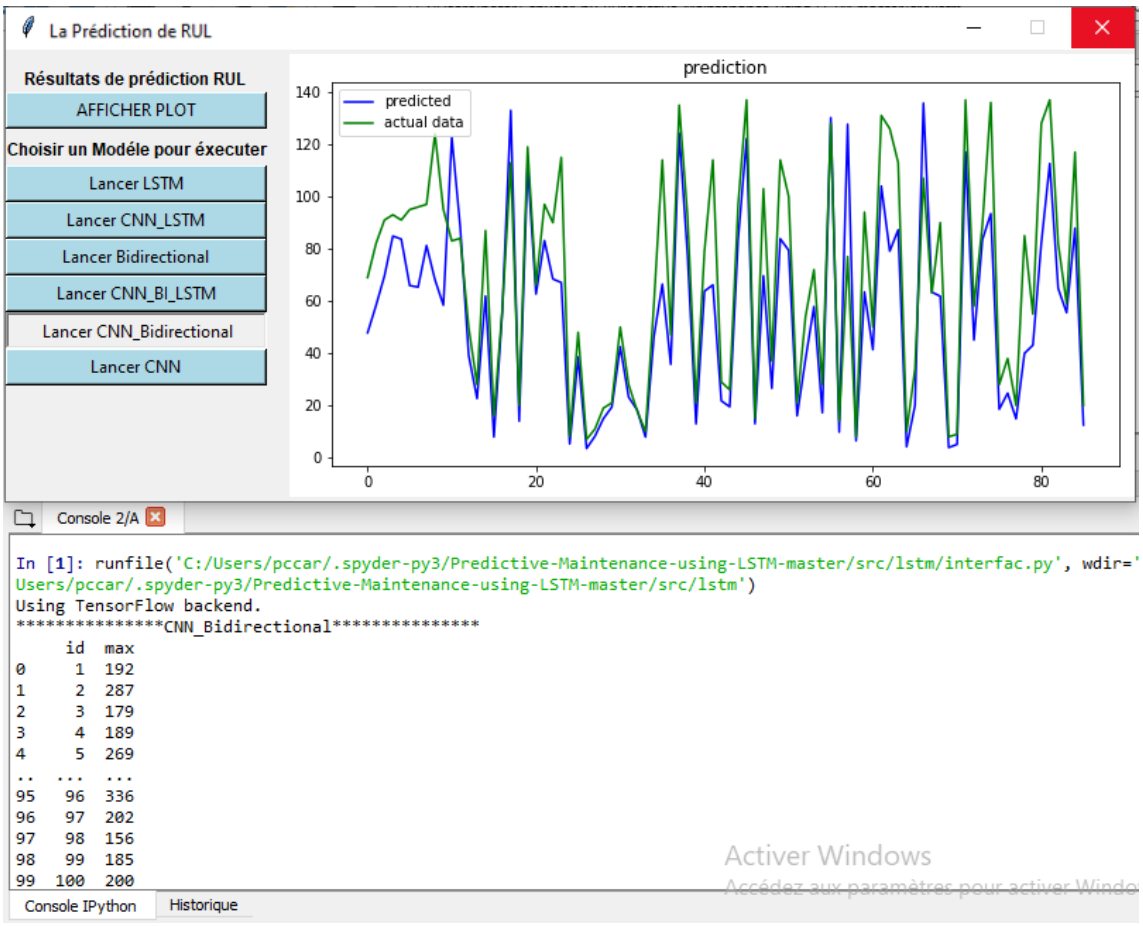


FIGURE 5.10 – L'interface pendant l'exécution.

Cette interface a été créée en utilisant le langage python avec le tkinter gui- graphics. Il contient des boutons qui nous permettent d'exécuter diverses fonctions, qui nous permettent :

- De lancer chaque modèle pour la phase d'entrainement.
- D'afficher le proocessus d'entrainement pour lchaque modèle dans l'environrment.
- D'afficher les plots de prédiction de chaque modèle .

Conclusion

Dans ce chapitre, nous avons vu notre part de contribution au problème de prédiction de RUL , représentant les outils de développement utilisés, les étapes que nous avons suivies pour construire nos modèles ainsi que quelques morceaux de code associés. Et enfin nous avons présenté l'interface créée pour accéder aux différentes fonctions de notre application.

CONCLUSION GÉNÉRALE

Notre projet de fin d'études s'intéresse à l'estimation de temps restant utile (RUL) pour la maintenance conditionnelle. L'objectif principale était de proposer des variantes de LSTM, puis comparer et analyser les résultats afin d' étudier l'efficacité de la méthode par rapport au problème et par rapport aux autres méthodes proposées dans le domaine.

Pour réaliser notre travail, nous avons utilisé le deep learning, la méthode d'apprentissage qui a montré ses performances ces dernières années. Nous avons choisi la méthode LSTM comme méthode de prédiction, ce choix est justifié par l'éfficacité de la méthode pour les problèmes de prédiction de séries chronologiques (time series) qui sont un type difficile de problème de modélisation prédictive.

Ce projet s'est avéré bénéfique sur plusieurs plans. Il nous a permis d'acquérir de nouvelles connaissances et de renforcer d'autres. Il nous a permis de découvrir notamment le domaine de la maintenance, la maintenance conditionnelle et le deep learning qui nous en font nécessiter une recherche bibliographique profonde. Il nous a permis aussi d'apprendre un nouveau langage de programmation qui est Python et ses différentes technologies.

Durant la réalisation de ce travail, nous avons rencontré des difficultés, mentionnant :

1. La difficulté d'accès aux papiers et les ressources qui ne sont pas toujours gratuits.
2. Dans la majorité des ressources disponibles on constate une incohérence dans la sélection des modèles de comparaison et les paramètres d'évaluation, ce qui rend difficile de valider les résultats.
3. Plus important encore, le manque d'équipements physiques suffisamment puissants pour faciliter l'apprentissage et le test des méthodes appliquées. Ce problème rend également le temps d'exécution sur les autres ensembles (FD002, FD003, FD004) de la base des données CMAPSS plus long (plus de 24h pour une exécution).

A la fin nous pouvons dire que notre travail a atteint ses objectifs fixés au début, nous avons arrivé à proposer cinq variantes de LSTM avec des meilleures configurations, les

implémenter, comparer les résultats et les analyser. Les résultats de notre proposition ont justifié l'efficacité de modèle LSTM et ses différents types.

Mais comme n'importe quel travail scientifique il reste ouvert à d'autres améliorations. Et comme perspectives nous pouvons citer :

- Tester notre modèle sur les autres sous ensembles du dataset CMAPSS.
- Tester notre modèle sur d'autres ensembles de données.
- Essayer notre modèle avec plus de valeurs de Windows size.
- Essayer d'appliquer une méthode d'optimisation sur les paramètres y comprie le nombre de couches les portes de réseau afin d'obtenir la configuration optimale au lieu de changer à chaque fois arbitrairement la configuration.
- Développer notre méthode avec le modèle CuDNNLSTM puisqu'il est plus rapide que le LSTM mais il nécessite une carte graphique puissante pour qu' il s'exécute sur GPU.
- Essayer d'appliquer d'autres travaux de comparaison et/ou d'hybridation avec d'autres méthodes de régression.

BIBLIOGRAPHIE

[1] B.Fahem,O.Lyes, "Mise en place d'un plan de maintenance préventive pourl'amélioration de la disponibilité de la ligne « TAMBOUR » ausein de SAMHA", Construction Mécanique, Sétif, 64 p.

[2] AFNOR, la normalisation française FDX_60-000,"Maintenance industrielle et Fonction maintenance", publié par afnor mai 2002 ;http ://www.ehpadneuilly.com/cariboost_files/FDX_60-000.pdf.

[3] Ahmed Mosallam,"Remaining useful life estimation of critical components based on bayesian approaches" ,these de doctorat en informatique automatique , 2014, 146 p.

[4] Olivier Legrand," introduction de la norme nf en 62353 au centre hospitalier de boulogne sur mer",Certification Professionnelle ABIH, UTC, 2012,disponible sur "https ://www.utc.fr/tsibh/public/3abih/12/stage/legrand/index.html".

[5] Estelle Deloux. ,"Politiques de maintenance conditionnelle pour un système à dégradation continue soumis a un environnement stréssant ", Sciences de l'ingénieur 'physics',Université de Nantes, 2008. Français. fftel-00348191.

[6] Abdallah Kabouche,"Techniques de Maintenance Prédictive pour l'Amélioration de la disponibilité des Installations",thèse de doctorat d'état ,189p.

[7] Rosmaini Ahmad , Shahrul Kamaruddin ,"An overview of time-based and condition-based maintenance in industrial application", Computers & Industrial Engineering 63 (2012) 135–149.

[8] R. C. M. Yam et al.,"Intelligent Predictive Decision Support System for Condition-Based Maintenance ", Int J Adv Manuf Technol (2001) 17 :383–391 .

[9] Alexandre Sarazin et al.,"Système de systèmes dans les architectures PHM", MOSIM'18 - 12ème Conférence internationale de Modélisation, Optimisation. et SIMulation, ISAE ; IMT Mines Albi, Jun 2018, Toulouse, France. 7 p. ffhal-01852185f

[10] Racha Khelif," Estimation du RUL par des approches basées sur ´ l'expérience : de la donnée vers la connaissance ", thèse de doctorat :automatique/Robotique,Université de Franche-Comté :2015,127p.

[11] laloux guillaume, Conseil, formation, "expertise pour le management de la maintenance, Maintenance management conseil coaching" .

[12] Diego Jair Rodriguez Obando, "From Deterioration Modeling to Remaining Useful Life Control : a comprehensive framework for post-prognosis decision-making applied to friction drive systems", Automatic, Université Grenoble Alpes, 2018, English, ffNNT : 2018GREAT086ff. fftel-02019101v2f .

[13] Danh Ngoc Nguyen,"Contribution aux approches probabilistes pour le pronostic et la maintenance des systèmes contrôlés", Optimisation et suréte des système , l'université de technologie de troyes ,2015.

[14] Ali Al-Dulaimi et al.,"Hybrid deep learning neural network model for remaining useful life estimation",Electrical and Computer Engineering, Concordia University, Montreal,978-1-5386-4658-8/18/$31.00 ©2019 IEEE, 10.1109/ICASSP.2019.8683763.

[15] Y. Zhang et al." Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries", IEEE Trans. Vehic. Technol. 67 (7) (2018) 5695–5705.

[16] R. Zhao et al.," Learning to monitor machine health with convolutional bi-directional LSTM networks", Sensors 17 (2) (2017) 273.

[17] Mohammed Msaaf, Fouad Belmajdoub." L'application des réseaux de neurone de type " feedforward" dans le diagnostic statique". Xème Conférence Internationale : Conception et Production Intégrées, Dec 2015, Tanger, Maroc. ffhal-01260830ff

[18] Gregory Gelly. "Reseaux de neurones recurrents pour le traitement automatique de la parole". Réseau de neurones [cs.NE]. Université Paris-Saclay, 2017. Français. ffNNT : 2017SACLS295ff. fftel-01615475f .

[19] Fundamentals of Deep Learning Designing Next-Generation Machine Intelligence Algorithms. Nikhil Buduma. 2017

[20] Sharma, Sagar, "Activation Functions in Neural Networks", https ://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6, Medium, Towards Data Science, feb 2019 ,

[21] Crescenzio Gallo," Artificial Neural Networks : tutorial",2018, ,p.179-189 .

[22] Charu C. Aggarwal, IBM T. J. Watson ."Research Center International Business MachinesYorktown Heights", NY, USA, ISBN 978-3-319-94462-3 ISBN 978-3-319-94463-0 (eBook).

[23] Gilles Burel ,"Une nouvelle approche pour les réseaux de neurones :la représentation scalaire distribuee ",Traitement du Signal, Vol. 10, No. 1, 1993.

[24] El Mahdi Brakni ," Résequx de neurones artificiels appliqués à la méthode électromagnétique transitoire InfiniTEM ", Mémoire présenté à l'université du québec à chicoutimi comme exigence partielle de la maitrise en ingénierie , mai 2011 .

[25] "neural network and introduction to deep learning" ,17p. Format PDF ,https ://www.math.univ-toulouse.fr/ besse/Wikistat/pdf/st-m-hdstat-rnn-deep-learning.pdf.

[26] François Chollet ,"Deep learning with python ", www.manning.com/books/deep-learning-with-python.

[27] Bryan Lim and Stefan Zohren,Time Series Forecasting With Deep Learning : A Survey,1Department of Engineering Science, University of Oxford,arXiv :2004.13408v1 [stat.ML] 28 Apr 2020.

[28] Mohaiminul Islamet al. "An Overview of Neural Network". American Journal of Neural Networks and Applications. Vol. 5, No. 1, 2019, pp. 7-11. doi : 10.11648/j.ajnna.20190501.12 189 .

[29] Rikiya Yamashita et al.,"Convolutional neural networks : an overview and application in radiology", Insights into Imaging (2018) 9 :611–629 ,Published online : 22 June 2018.

[30] "Réseaux de neurones",9p. Format PDF disponible sur : https ://www.math.univ-toulouse.fr/ besse/Wikistat/pdf/st-m-app-rn.pdf

[31] Paolo Dell'Aversana, "Artificial neural networks and deep learning :A simple overview ", December 2019.https ://www.researchgate.net/publication/337678793.

[32] Boughaba Mohammed et Boukhris Brahim,"L'apprentissage profond (Deep Learning)pour la classification et la recherche d'images par le contenu", 2016/2017,Informatique fondamentale.

[33] Charu C. Aggarwal,"Neural Networks and Deep Learning",IBM T. J. Watson Research Center,ISBN 978-3-319-94462-3,https ://doi.org/10.1007/978-3-319-94463-0

[34] Nicolas Oyharçabal Astorga ,"CONVOLUTIONAL RECURRENT NEURAL NETWORKS FOR REMAINING USEFUL LIFE PREDICTION IN MECHANICAL SYSTEMS", universidad de chille ,19/03/2018.

[35] Jason Brownlee,Long Short-Term Memory Networks With Python,Develop Sequence Prediction Models With Deep Learning,v1.0,2017.

[36] XiangLi et al. ,"Remaining useful life estimation in prognostics using deep convolution neural networks,Reliability Engineering& System Safety ",Volume 172, April 2018, Pages 1-11,https ://www.sciencedirect.com/science/article/abs/pii/S0951832017307779.

[37] https ://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras.

[38] https ://scikit-learn.org/stable/modules/model_evaluation.html.

[39] Staudemeyer, Ralf & Omlin, Christian. (2013). Evaluating performance of long short-term memory recurrent neural networks on intrusion detection data. ACM International Conference Proceeding Series. 218. 10.1145/2513456.2513490.

[40] Azure,"Azure/lstms_for_predictive_maintenance ", Github ,https ://github.com/Azure/lstms_for_predictive_maintenance.

[41] https ://machinelearningmastery.com/cnn-long-short-term-memory-networks/

[42] https ://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/

[43] https ://www.researchgate.net/publication/328826136_Activation_Functions_Comparison_of_tren

[44] penseeartificielle.fr/mish-vs-relu-meilleure-fonction-activation/https ://penseeartificielle.fr/mish-vs-relu-meilleure-fonction-activation/.

[45] Andrej Karpathy (2017). A Peek at Trends in Machine Learning. https ://medium.com/@karpathy/a-peek-at-trends-in-machine-learning-ab8a1085a106.

[46] Kingma, D. P., & Ba, J. L. (2015). Adam : a Method for Stochastic Optimization. International Conference on Learning Representations, 1–13.

[47] https ://towardsdatascience.com/choosing-the-right-hyperparameters-for-a-simple-lstm-using-keras-f8e9ed76f046.

[48] Mishra, Divyanshu," Regression : An Explanation of Regression Metrics And What Can Go Wrong", disponible sur "https ://towardsdatascience.com/regression-an-explanation-of-regression-metrics-and-what-can-go-wrong-a39a9793d914", Medium, Towards Data Science, dec-2019

[49] Jason Brownlee," How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras", June 16, 2017 in Long Short-Term Memory Networks, disponible sur "https ://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/".

[50] Jun Wu,"Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network",ISA Transactions (2019), https ://doi.org/10.1016/j.isatra.2019.07.004.

[51] M1 MIMST ,"Introduction à la Maintenance conditionnelle", Outils de la maintenance conditionnelle,10/6/20,32p.

[52] JDN ,"Python : définition et utilisation de ce langage informatique", 2020 disponible sur "https ://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique".

[53] https ://anaconda.org/anaconda/spyder.

[54] Dzone ,"Anaconda Python Tutorial : Everything You Need to Know", 2020, disponible sur "https ://dzone.com/articles/python-anaconda-tutorial-everything-you-need-to-kn".

[55] Whatls TechTarget ,"Machine learning" ,2018,disponible sur " https ://whatis.techtarget.com/fr/definition/Machine-Learning".