الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Research

Registration n°:
...../...../...../...../.....

جامعة غرداية
University of Ghardaia
والتكنولوجيا العلوم كلية
Faculty of Science and Technology
قسم الرياضيات والإعلام الآلي
Department of Mathematics and Computer Science
التطبيقية والعلوم الرياضيات مخبر
Mathematics and Applied Sciences Laboratory

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master

Domain: Mathematics and Computer Science
Field: Computer Science
Specialty: Intelligent Systems for Knowledge Extraction

Topic

Deep Learning for Real-Time Predictive Maintenance and Adaptive Control in Industrial IoT

Presented by:

BENSAHA Aya & KELLOU Ikram

Publicly defended on 06 24, 2026

Jury members:

Mrs. Brahim Nacera	MAA	Univ. Ghardaia	President
Mr. Guerbouz Tahar		Univ. Ouargla	Examiner
Mr. Fekair Mohamed El Amine	MCB	Univ. Ghardaia	Supervisor
Mr. Si-Mohammed Samir	Assoc. Prof	Univ. Lorraine, France	Co-Supervisor

Academic Year: 2025/2026

Acknowledgment

*First and foremost, we thank **God** for granting us the strength and determination to complete this research. The journey of writing this thesis has been a profound learning experience, and we are deeply grateful to those who guided us.*

*We extend our sincerest gratitude to our supervisor, **Dr. Fekair Mohamed El Amine**, for his expert guidance, unwavering support, and immense patience. We deeply appreciate his continuous availability, valuable advice, and the careful review of our work. His profound knowledge and dedication were fundamental to the success of this research.*

*We equally express our deep appreciation to our co-supervisor, **Dr. Si-Mohammed Samir** from the University of Lorraine, France, for his exceptional mentorship and continuous encouragement. We are incredibly thankful for the generous time he dedicated to providing us with highly valuable remarks. His shared expertise and insightful feedback were instrumental in refining this research.*

We also wish to express our sincere appreciation to the distinguished members of the jury for honoring us by reviewing and evaluating this thesis.

Finally, we acknowledge our department for providing the general academic framework during our studies.

Dedication

*First of all, I thank **ALLAH** the Almighty for giving me the strength, the patience, and the perseverance to complete this work and to reach this stage of my studies.*

To my dearest parents,
No words will ever be enough to express my love and gratitude for you. Your endless sacrifices, unconditional love, and constant prayers are the reason I stand where I am today. Everything I have achieved is a reflection of your patience, your guidance, and your unwavering belief in me. This success belongs to you before it belongs to me.

To my dear brothers and sisters, Houda, Amir, Anes, and Ines,
You are my strength, my comfort, and my safe place. Thank you for your love, your support, and for always being present in every step of my journey, especially during moments of fatigue and exhaustion. Your presence in my life makes every achievement more meaningful and every difficulty easier to overcome.

To my dear friends,
Thank you for your encouragement, your kindness, and the beautiful memories we shared. You made this long journey lighter and filled it with moments of joy, support, and motivation.

To my dear partner , Ikram Kellou,
Thank you for your dedication, patience, and true collaboration throughout this project. Working with you was a truly valuable and enriching experience. Your support, consistency, and teamwork played an important role in the success of this work, and I am sincerely grateful for everything we shared during this journey.

Finally, I dedicate this work to all those who believed in me, supported me, and encouraged me during this journey, in one way or another.

Aya Bensaha

Dedication

*First of all, I thank **ALLAH** the Almighty for giving me the strength, the will, and the patience to pursue my studies and to accomplish this modest work.*

To my dearest parents,
No words could ever express my gratitude for your endless sacrifices, unconditional love, and constant prayers. Everything I am, and everything I have achieved today, I owe entirely to your patience and your unwavering belief in me. You are my guiding light and my greatest pride, and I dedicate this success entirely to you.

To my dear brothers,
*Thank you for your protective love, constant presence, and for always cheering me on. Your belief gives me daily confidence. A special, profound thank you to **Brahim**, the one who always has my back. Thank you for stepping in with crucial support and guidance when things got challenging; your endless motivation and presence made all the difference in this journey.*

To my dear sisters, Fairouz, Nahla,
My great support. Thank you for your advice, your love, and your permanent encouragement.

To my dear friends,
Thank you for your presence, your encouragement, and all the beautiful shared moments that made this heavy journey feel lighter.

To my dear partner, Aya Bensaha,
Through every twist and triumph of this journey, you stood by my side. Thank you for being the best teammate, your hard work, resilience, and true partnership are reflected in every part of this project.

Finally, my deepest gratitude goes to all the individuals, near or far, whose belief, guidance, and subtle encouragement helped turn this vision into reality.

Ikram Kellou

مـــلـــخـــص

في سياق الثورة الصناعية الرابعة وإنترنت الأشياء الصناعي ،(IIoT) أصبحت الصيانة التنبؤية تحدياً رئيسياً نتيجة التعقيد المتزايد للأنظمة الصناعية والتوليد المستمر للبيانات اللحظية من المستشعرات. تقدم هذه المذكرة إطاراً ذكياً يجمع بين تقنيات التعلم العميق والتعلم المعزز العميق للتنبؤ بالعمر التشغيلي المتبقي (RUL) واتخاذ قرارات صيانة تكيفية. يعتمد الإطار المقترح على مرحلتين متكاملتين. في المرحلة الأولى، تم تطوير نموذج هجين CNN-LSTM لتعلم خصائص التدهور والتبعيات الزمنية انطلاقاً من الإشارات متعددة المتغيرات لمحركات الطائرات. وفي المرحلة الثانية، تُستخدم قيم العمر التشغيلي المتبقي المتوقعة من قبل عامل تعلم معزز من نوع Soft Actor Critic (SAC) من أجل تحسين قرارات الصيانة بشكل تكيفي. تم تقييم الإطار المقترح باستخدام مجموعات بيانات NASA C-MAPSS في ظروف تشغيل متعددة. وأظهرت النتائج التجريبية دقة في التنبؤ بالعمر التشغيلي المتبقي وموثوقية في اتخاذ قرارات الصيانة، مما ساهم في تقليل الأعطال غير المتوقعة والحد من عمليات الصيانة غير الضرورية. وتبرز النتائج المتحصل عليها إمكانات دمج التحليلات التنبؤية مع التحكم التكيفي لتحقيق إدارة ذكية للصيانة في بيئات إنترنت الأشياء الصناعي.

كلمات مفتاحية: الصيانة التنبؤية، التعلم العميق، التعلم المعزز العميق، الشبكات العصبية الالتفافية والمتكررة (CNN-LSTM)، خوارزمية Soft Actor Critic (SAC)، العمر التشغيلي المتبقي (RUL)، إنترنت الأشياء الصناعي (IIoT).

Abstract

In the context of Industry 4.0 and the Industrial Internet of Things (IIoT), predictive maintenance has become a major challenge due to the increasing complexity of industrial systems and the continuous generation of real-time sensor data. This thesis presents an intelligent framework that combines Deep Learning and Deep Reinforcement Learning techniques for Remaining Useful Life (RUL) prediction and adaptive maintenance decision-making.The proposed framework consists of two complementary stages. In the first stage, a hybrid CNN-LSTM model is developed to learn degradation features and temporal dependencies from multivariate aircraft engine sensor signals. In the second stage, the predicted RUL is used by a Soft Actor-Critic (SAC) reinforcement learning agent to support adaptive maintenance optimization and decision-making.The framework is evaluated using the NASA C-MAPSS datasets under multiple operating conditions. Experimental results demonstrate accurate RUL prediction and reliable maintenance decision-making, leading to a reduction in unexpected failures and unnecessary maintenance interventions. The obtained results highlight the potential of integrating predictive analytics and adaptive control to support intelligent maintenance management in Industrial IoT environments.

Keywords : Predictive Maintenance, Deep Learning, Deep Reinforcement Learning, CNN-LSTM, Soft Actor Critic (SAC), Remaining Useful Life (RUL), Industrial Internet of Things (IIoT).

Résumé

Dans le contexte de l'Industrie 4.0 et de l'Internet des Objets Industriels (IIoT), la maintenance prédictive est un défi crucial en raison de la complexité croissante des systèmes industriels et de la génération continue de données de capteurs en temps réel. Ce mémoire propose un cadre intelligent combinant les techniques d'Apprentissage Profond et d'Apprentissage par Renforcement Profond pour la prédiction de la Durée de Vie Utile Restante (RUL) et la prise de décision adaptative en maintenance.Notre cadre se compose de deux étapes complémentaires. Premièrement, un modèle hybride CNN-LSTM est développé afin d'extraire les caractéristiques de dégradation et les dépendances temporelles à partir de signaux multivariés provenant de moteurs aéronautiques. Deuxièmement, les valeurs prédites de la RUL sont utilisées par un agent d'apprentissage par renforcement Soft Actor-Critic (SAC) pour optimiser les décisions de maintenance de manière adaptative. Le cadre proposé est évalué à l'aide des jeux de données NASA C-MAPSS dans différentes conditions de fonctionnement. Les résultats expérimentaux montrent une prédiction précise de la RUL ainsi qu'une prise de décision fiable, permettant de réduire les défaillances imprévues et les interventions de maintenance inutiles. Les résultats obtenus mettent en évidence le potentiel de l'intégration de l'analyse prédictive et du contrôle adaptatif pour une gestion intelligente de la maintenance dans les environnements IIoT.

Mots clés : Maintenance prédictive, Apprentissage profond, Apprentissage par renforcement profond, CNN-LSTM, Soft Actor Critic (SAC), Durée de Vie Utile Restante (RUL), Internet des Objets Industriels (IIoT).

Contents

List of Figures x

List of Tables xi

List of Acronyms xii

General Introduction 1

1 Background 2

1.1 Introduction . . . 2

1.2 Evolution of Industrial Systems . . . 2

1.2.1 Industry 1.0: Mechanization . . . 2

1.2.2 Industry 2.0: Electrification . . . 3

1.2.3 Industry 3.0: Automation and IT . . . 3

1.2.4 Industry 4.0: Smart Manufacturing . . . 3

1.2.5 Industry 5.0: Human-Centric and Sustainable Manufacturing 4

1.3 Enabling Technologies for Smart Maintenance . . . 4

1.3.1 Industrial Internet of Things (IIoT) . . . 5

1.3.2 Big Data Analytics . . . 5

1.3.3 Cloud, Edge, and Fog Computing . . . 5

1.3.4 Digital Twin Systems . . . 6

1.4 Maintenance Strategies in Industry 4.0 . . . 6

1.4.1 Reactive Maintenance . . . 6

1.4.2 Preventive Maintenance . . . 7

1.4.3 Predictive Maintenance . . . 7

1.4.4 Prescriptive Maintenance . . . 7

1.5 Artificial Intelligence in Industrial Maintenance . . . 8

1.5.1 Machine Learning . . . 9
1.5.2 Deep Learning . . . 9
1.5.3 Reinforcement Learning for Decision Making . . . 10
1.6 Challenges and Limitations of Current Systems . . . 11
1.7 Conclusion . . . 12

2 Literature Review and State of the Art 13
2.1 Introduction . . . 13
2.2 Taxonomy of Intelligent Maintenance Systems . . . 13
2.2.1 Traditional Maintenance and Model Based Approaches . . . 14
2.2.2 Deep Learning for RUL Prediction . . . 15
2.2.3 Uncertainty Quantification in Deep Learning . . . 16
2.2.4 From Prediction to Dynamic Action: Evolution of Adaptive Control Frameworks . . . 17
2.3 Summary of the literature review . . . 18
2.4 Research Gaps and Thesis Contributions . . . 18
2.4.1 Discussion of Research Gaps . . . 18
2.4.2 Research Objectives . . . 19
2.4.3 Main Contributions . . . 19
2.5 Conclusion . . . 19

3 Proposed Methodology 21
3.1 Introduction . . . 21
3.2 Problem Statement . . . 21
3.2.1 Data Description . . . 21
3.2.2 Target RUL Formulation (Piecewise Linear) . . . 22
3.2.3 Problem Formulation and Objectives . . . 23
3.3 Overall Framework Architecture . . . 23
3.4 Data Preprocessing Pipeline . . . 24
3.5 CNN-LSTM Predictive Model . . . 25
3.6 Probabilistic RUL Estimation . . . 29
3.7 Adaptive Control Using SAC . . . 30
3.8 Conclusion . . . 34

4 Experiments and Results **35**

4.1 Introduction . . . 35

4.2 Experimental Setup . . . 35

4.2.1 Hardware and Software Environment . . . 35

4.3 Evaluation Metrics . . . 36

4.3.1 Predictive Performance Metrics . . . 36

4.3.2 Reinforcement Learning Decision Metrics . . . 37

4.4 Evaluation of the CNN-LSTM Predictive Model . . . 37

4.4.1 Visual Convergence and Profile Tracking . . . 38

4.4.2 Comparison with Recent State of the Art Approaches . . . 39

4.5 Adaptive Maintenance Policy Evaluation . . . 40

4.5.1 SAC Agent Training Convergence . . . 41

4.5.2 Operational Testing Results . . . 41

4.6 Discussion . . . 43

4.7 Conclusion . . . 43

Conclusion and Perspectives **44**

Bibliography **45**

List of Figures

1.1 Evolution of industrial revolutions from Industry 1.0 to 5.0 4

1.2 The Pillars of Industry 4.0 5

1.3 Different types of maintenance 6

1.4 Standard architecture of CNN for time-series processing 9

1.5 LSTM network structure 10

1.6 Predictive Maintenance pipeline 11

2.1 Taxonomy of Intelligent Maintenance Systems. 14

3.1 Overall Framework Architecture. 24

3.2 Architecture of the Proposed Hybrid CNN-LSTM model for RUL prediction. 26

4.1 Training vs. validation loss convergence trajectories. 38

4.2 Actual RUL vs Predicted RUL. 39

4.3 Learning curve of the SAC agent. 41

List of Tables

1.1 Comparison of Maintenance Strategies 8

2.1 Summary of the literature review. 18

3.1 Parameters of the C-MAPSS dataset. 22

3.2 Structure and purpose of the provided files in each C-MAPSS sub-dataset. 22

3.3 Sensor evaluation and selection 24

3.4 Hyperparameters and setup configurations for the proposed CNN-LSTM framework. 28

4.1 Hardware specifications and software environment. 36

4.2 Deterministic RUL prediction results across NASA C-MAPSS subsets. 38

4.3 Performance comparison with recent State of the Art methods across all C-MAPSS datasets. 39

4.4 FD002 Dataset Strict Partitioning Strategy. 40

4.5 SAC Adaptive Maintenance Evaluation Results on FD002 Test Set. 42

4.6 Performance comparison between the baseline study and our proposed framework. 42

List of Acronyms

AE	Autoencoder
AI	Artificial Intelligence
ARIMA	AutoRegressive Integrated Moving Average
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
C-MAPSS	Commercial Modular Aero-Propulsion System Simulation
CNN	Convolutional Neural Network
CPS	Cyber-Physical Systems
DBNs	Deep Belief Networks
DDQN	Double Deep Q-Networks
DL	Deep Learning
DRL	Deep Reinforcement Learning
ERP	Enterprise Resource Planning
EMA	Exponential Moving Average
GRU	Gated Recurrent Unit
IIoT	Industrial Internet of Things
IoT	Internet of Things
LSTM	Long Short-Term Memory
MC	Monte Carlo
MDP	Markov Decision Process
ML	Machine Learning
NASA	National Aeronautics and Space Administration
PdM	Predictive Maintenance

PHM Prognostics and Health Management

PLC Programmable Logic Controller

PSO Particle Swarm Optimization

PvM Preventive Maintenance

RL Reinforcement Learning

RMSE Root Mean Square Error

RNN Recurrent Neural Network

RUL Remaining Useful Life

SAC Soft Actor-Critic

SAEs Stacked Autoencoders

SVR Support Vector Regression

UQ Uncertainty Quantification

Introduction

The industrial landscape is rapidly evolving with the integration of the Industrial Internet of Things (IIoT), which continuously generates vast amounts of multi sensor data. Industrial equipment reliability is key to both manufacturing performance and operational safety. Traditional maintenance methods rely on static schedules or reactive strategies that are financially costly.

As a result, Predictive Maintenance (PdM) has emerged as a crucial strategy centered on estimating the Remaining Useful Life (RUL), defined as the expected operational cycles before a machine reaches a catastrophic failure threshold. Accurately predicting RUL is challenging since industrial sensor data are complex, noisy, and non-linear under varying conditions. Furthermore, merely anticipating failure does not instruct operators on necessary actions. Thus, practical solutions integrating these predictive alarms with autonomous decision-making remain rare.

To solve this problem we propose a combination of Convolutional Neural Network and Long Short Term Memory (CNN-LSTM) architecture to predict machine degradation from multivariate sensor data. Evaluated on the globally recognized NASA C-MAPSS dataset[1], the model efficiently extracts spatio-temporal features to provide highly accurate RUL predictions under complex operational conditions. Building on this, we propose a two stage artificial intelligence pipeline. In the first stage, the CNN-LSTM perception module extracts features from noisy data and predicts the RUL. In stage two, a Soft Actor-Critic (SAC) Deep Reinforcement Learning (DRL) agent's state space is defined by these outputs. This agent recommends the optimum maintenance interventions by autonomously optimizing decisions with a focus on risk reduction and improving overall system reliability. The system is designed with an advanced predictive maintenance framework.

The rest of this thesis is organized as follows: **Chapter** 1 "Background" introduces the concepts of Predictive Maintenance and the algorithms used, **Chapter** 2 "Related Work" looks at work and existing frameworks, **Chapter** 3 "Proposed Methodology" details the experimental method, system design, and proposed architecture, and **Chapter** 4 "Experiments and Results" presents the implementation, testing, and performance results of the proposed solution.

[1]Dataset available at: https://www.kaggle.com/datasets/behrad3d/nasa-cmaps

Chapter 1

Background

1.1 Introduction

Establishing the theoretical foundations of our proposed system is crucial before diving into the core methodology of this thesis. This chapter provides a comprehensive background by introducing the fundamental concepts and algorithms that power modern intelligent industrial systems.

We start by examining the evolution of maintenance strategies, which led to the Predictive Maintenance (PdM) paradigm and the critical role of Remaining Useful Life (RUL) estimation. The fundamental Artificial Intelligence (AI) methods used in our perception module are then presented. In particular, we focus on the use of Deep Learning architectures in multivariate time series forecasting, such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks.

Finally, the chapter covers the principles of Reinforcement Learning (RL). We detail the transition from predictive forecasting to active decision-making by explaining the fundamental concepts of RL, such as Markov Decision Processes, states, actions, and rewards, which form the theoretical basis for our autonomous intervention framework. By the end of this chapter, the reader will have a clear understanding of the theoretical tools utilized to construct our pipeline.

1.2 Evolution of Industrial Systems

The development of industrial production can be categorized into four major revolutions, each representing a fundamental transformation in manufacturing processes and, thus, maintenance practices [1].

1.2.1 Industry 1.0: Mechanization

The first industrial revolution began in the 18th century. It introduced machines powered by water and steam. During this time, people started to use machines instead of doing things by hand. This change led to factories replac-

ing farming work. The spinning jenny and steam engine were innovations, which helped manufacture products more efficiently and faster [1]. Maintenance then was mostly fixing things after they broke, and people did not do much to prevent problems. As industries grew and electricity replaced steam, factories became more complex. This made it clear that better maintenance was needed, which set the stage for the industrial era.

1.2.2 Industry 2.0: Electrification

The Second Industrial Revolution, spanning the late 19th and early 20th centuries, was driven by the introduction of electrical energy and mass production techniques. The assembly line, pioneered by Henry Ford, enabled the standardization and division of labor. Electrical power replaced steam, allowing factories to operate more efficiently and at greater scale [1]. In parallel with these technological advancements, maintenance practices evolved modestly. Organizations began to adopt time based preventive schedules for critical equipment to avoid halting the newly established assembly lines. As these electrical systems became more complex, the introduction of electronics and computers provided the tools necessary to move beyond rigid schedules toward automated control.

1.2.3 Industry 3.0: Automation and IT

The Third Industrial Revolution started in the second half of the 20th century with the introduction of electronics, programmable logic controllers (PLCs), and information technology. This led to automation in manufacturing, which reduced the need for humans to do tasks [1]. In addition, technologies such as Computer Aided Design (CAD), Computer Aided Manufacturing (CAM), and Enterprise Resource Planning (ERP) systems greatly improved production planning and industrial process management. With the growth of digital technologies, condition based monitoring approaches also started to develop, allowing maintenance activities to rely on the actual condition of equipment instead of only following fixed maintenance schedules. While Industry 3.0 digitized the factory floor, it set the stage for the current era where these isolated automated systems are now fully interconnected and intelligent.

1.2.4 Industry 4.0: Smart Manufacturing

The Fourth Industrial Revolution, Industry 4.0, represents the current era of smart manufacturing, characterized by the fusion of cyber physical systems (CPS), the Internet of Things (IoT), cloud computing, and artificial intelligence [2]. In Industry 4.0 environments, machines are interconnected through IIoT networks and continuously generate vast volumes of operational data. This data is processed using advanced analytics and machine learning algorithms to enable real-time monitoring, autonomous decision making, and self optimizing production systems.

1.2.5 Industry 5.0: Human-Centric and Sustainable Manufacturing

Building on the automation foundations of Industry 4.0, Industry 5.0 is increasingly focusing on the human element, although it largely remains an emerging vision for the future. It brings together artificial intelligence and human expertise [3]. This ensures that manufacturing processes can resolve problems, adapt easily to changes, and minimize environmental impact. Industry 5.0 focuses on the effective collaboration between machines and human oversight. This approach enables human operators to safely interact with, monitor, and validate the actions of intelligent agents, ensuring both safety and operational reliability.

To visually summarize this historical progression, **Figure 1.1** illustrates the continuous evolution of industrial systems, from the early days of mechanization (Industry 1.0) to the emerging human-centric paradigm of Industry 5.0.

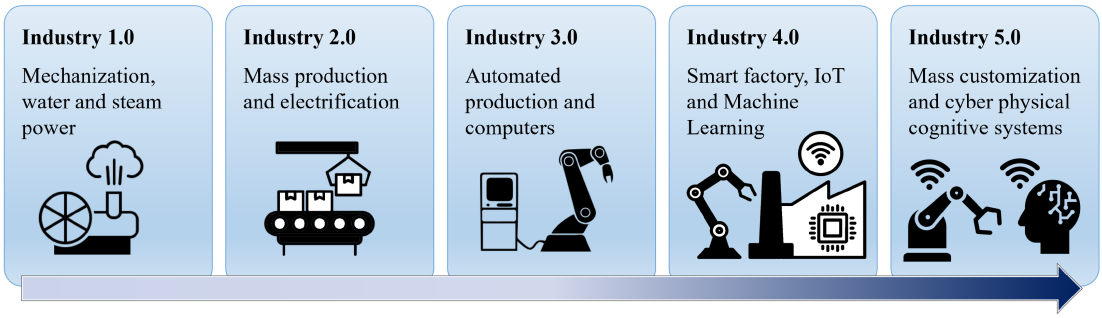

Figure 1.1: Evolution of industrial revolutions [4].

The technologies that distinguish Industry 4.0 from its predecessors, namely IIoT, big data, cloud computing, and digital twins, are not merely enablers of production; they constitute the analytical infrastructure upon which modern intelligent maintenance systems are built. While industrial environments rely on this broad spectrum of technologies, the following section focuses specifically on the elements utilized in our proposed framework. We will examine the role of Industrial IoT as the primary source of operational data, and the application of advanced Artificial Intelligence algorithms to process this data for real-time predictive maintenance.

1.3 Enabling Technologies for Smart Maintenance

As illustrated in Figure 1.2, Industry 4.0 is supported by multiple technological pillars. Specifically, the successful implementation of intelligent predictive maintenance depends on the seamless integration of robust data acquisition frameworks with advanced analytical models. This section focuses on the four core pillars that construct this analytical infrastructure: the Industrial Internet of Things (IIoT), Big Data Analytics, Edge/Cloud Computing, and Digital Twins.

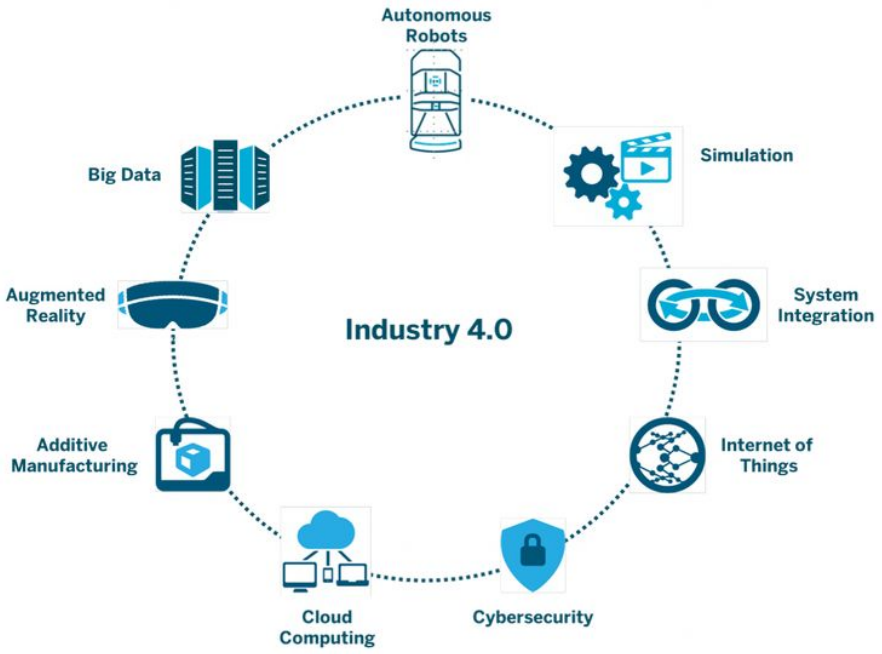

Figure 1.2: The core technological pillars of Industry 4.0 [5].

1.3.1 Industrial Internet of Things (IIoT)

The IIoT acts as the "sensory nervous system" of the smart factory, transforming physical mechanical stress into digital intelligence [6]. By offering high quality data which is essential for identifying early warning indicators of degradation it tackles the problem of machine observability [7]. This reliable information flow is essential for our autonomous decision making models, ultimately supporting risk mitigation and improving overall system reliability [8, 9].

1.3.2 Big Data Analytics

To handle the unprecedented scale of IIoT telemetry, Big Data platforms provide the necessary computational foundation. They improve raw noise into structured intelligence by enabling deep learning models to automatically discover latent features without the bottleneck of manual feature engineering [2]. Additionally, these frameworks facilitate the critical "History to Real Time" loop, simultaneously processing historical logs for model training and real-time streams for live inference.

1.3.3 Cloud, Edge, and Fog Computing

Processing massive industrial datasets requires a distributed hierarchy to balance computational power with latency [2]. Cloud computing offers the scale needed to train complex artificial intelligence models, while Edge computing shifts inference directly to the machine level. This distributed strategy ensures the sub second local responses critical for real-time industrial safety.

1.3.4 Digital Twin Systems

A Digital Twin is a dynamic synchronization layer that bridges physical reality and computational simulation. In smart maintenance, it serves two functional necessities: generating high variance simulated fault data to train robust diagnostic models, and providing a risk free, high fidelity environment to optimize autonomous control policies virtually before physical deployment [10].

1.4 Maintenance Strategies in Industry 4.0

The evolution of maintenance strategies represents a fundamental shift from treating industrial assets as "black boxes" to managing them as "intelligent, transparent systems." As highlighted by recent studies on Industry 4.0 implementation, this transition introduces both unprecedented opportunities for autonomous control and new operational challenges [11]. Figure 1.3 illustrates the four primary maintenance paradigms implemented in modern industrial environments.

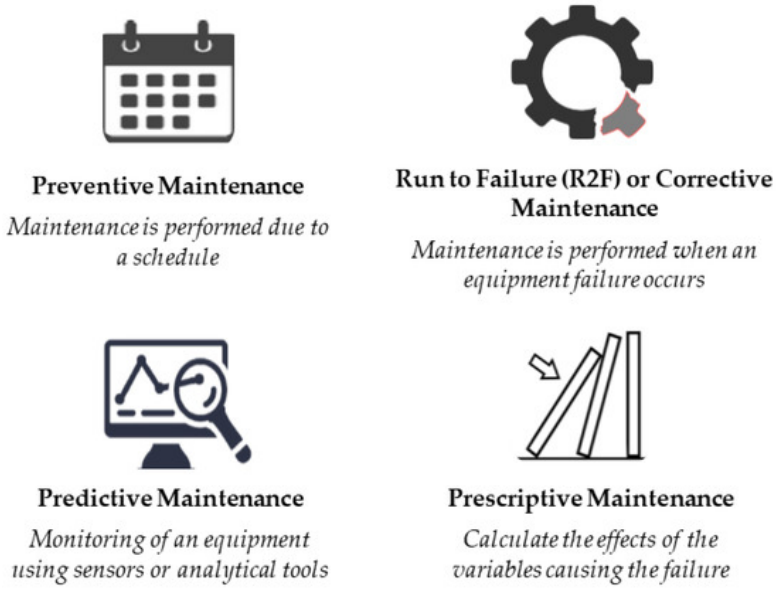

Figure 1.3: Different types of maintenance [12].

Each subsequent strategy was developed to resolve the practical limitations of earlier frameworks, enabling the transition toward autonomous control and proactive risk mitigation.

1.4.1 Reactive Maintenance

Reactive maintenance defers all intervention until equipment failure has already occurred. Although it eliminates upfront monitoring costs, this approach is highly unsuitable for high-criticality environments. In such settings, unplanned breakdowns frequently cause secondary damage and unexpected downtime, creating an economic burden and safety risks that far exceed the cost of proactive monitoring [13].

1.4.2 Preventive Maintenance

Using set schedules based on manufacturer guidelines or historical data, preventive maintenance aims to control risk. Despite its widespread use, its incapacity to take into consideration the dynamic wear patterns and real-time operating conditions of specific machines remains a serious shortcoming. This approach frequently results in the "over-maintenance" conundrum, whereby healthy components are eliminated early based on a calendar date or run-time threshold since it depends on static timelines rather than actual equipment health. In addition to wasting resources, this may introduce human mistake during needless disassembly [13].

1.4.3 Predictive Maintenance

Predictive Maintenance (PdM) addresses the limitations of fixed schedules through real-time condition monitoring to estimate a machine's Remaining Useful Life (RUL). RUL is defined as the expected number of operational cycles a machine can perform before its degradation reaches a threshold of failure. Mathematically, if t represents the current operational cycle and $T_{failure}$ represents the exact cycle at which the system breaks down, the actual RUL at time t is expressed as:

$$RUL(t) = T_{failure} - t \tag{1.1}$$

Accurate RUL estimation is the primary goal of prognostic algorithms. Traditional PdM operates as an open-loop system. It generates a prediction (e.g., estimating an engine failure in 10 cycles) without executing a response. The maintenance decision is decoupled from the prediction. This creates an "Information to Action" gap, meaning the system relies entirely on human operators to mitigate risks [7].

1.4.4 Prescriptive Maintenance

Prescriptive maintenance moves past prediction to offer recommendations and autonomous actions. Predictive maintenance estimates when a component will fail, whereas prescriptive maintenance identifies the necessary response. It recommends specific maintenance actions, dynamic scheduling, and adaptive control adjustments to extend equipment life [14, 15].

To clearly outline the operational and technological distinctions between these approaches, Table 1.1 provides a comprehensive summary of their key characteristics, advantages, and limitations.

Table 1.1: Comparison of Maintenance Strategies

Criteria	**Reactive Maintenance**	**Preventive Maintenance**	**Predictive Maintenance**	**Prescriptive Maintenance**
Definition	Maintenance after failure	Scheduled at fixed intervals	Based on real-time monitoring	Actionable recommendations and control
Approach	Run to failure	Time-based	Condition-based	Analytics and adaptive control
Data Requirement	None	Historical schedule	Sensor data (IIoT)	Real-time data and decision models
Decision Making	Reactive	Predefined	Human-assisted	Autonomous or semi-autonomous
Downtime	High	Moderate	Low	Very low
Cost Efficiency	Low	Moderate	High	Very high
Complexity	Low	Low-Moderate	Moderate-High	High
Technology Used	Basic tools	Scheduling systems	Sensors+ML/DL	Advanced analytics, RL, and Control Systems
Key Advantage	Simple	Reduces failures	Early detection	Active risk mitigation
Key Limitation	High breakdown risk	Over-maintenance	High implementation costs & Sensitive to data quality	Complex implementation

1.5 Artificial Intelligence in Industrial Maintenance

As illustrated in Table 1.1, the transition from reactive to prescriptive maintenance necessitates a significant increase in technological complexity. To achieve this, Artificial Intelligence (AI) serves as the core engine, specifically through the synergy of Deep Learning for perception and Reinforcement Learning for strategic decision making. AI enables the system to move beyond static limits and adapt to the dynamic, non linear nature of industrial degradation. The following sections detail the evolution of these AI paradigms from traditional patterns to autonomous systems.

1.5.1 Machine Learning

Machine learning (ML) provides the foundational algorithms for extracting patterns from sensor data. While supervised methods (e.g., Random Forests, SVMs) and unsupervised methods (e.g., Isolation Forests) are established for RUL estimation and anomaly detection, the efficacy of these models depends heavily on complex, domain specific manual feature engineering [16].

This process traditionally requires domain experts to transform raw signals into informative inputs. Statistical techniques, such as dimensionality reduction and automated time series metrics, assist this extraction and reduce manual effort. These statistical methods typically assume linear relationships or rely on shallow mathematical representations. Assets like turbofan engines generate high dimensional IIoT data with complex, non linear temporal interactions. Standard statistical tools struggle to map the deep latent correlations within this data. This limitation requires a transition to deep learning architectures, which autonomously extract features directly from raw sensor inputs to evaluate equipment health.

1.5.2 Deep Learning

Deep learning overcomes the limitations of manual feature engineering by using multi layered neural networks to automatically extract hierarchical representations directly from raw sensor data [16]. In the domain of predictive maintenance, several architectures have emerged as industry standards, each suited to specific signal characteristics:

- **Autoencoders (AEs):** Primarily used for unsupervised anomaly detection. AEs compress normal operational data into a lower dimensional latent space and reconstruct it [17]. When a machine begins to degrade, the reconstruction error spikes, serving as an early indicator of failure without requiring labeled fault data.
- **Convolutional Neural Networks (CNNs):** CNNs extract local patterns from input data. In the IIoT context, these networks process sequential sensor readings by scanning across time-series channels. This mechanism captures localized fault patterns and filters operational noise [18]. Figure 1.4 illustrates this standard architecture.

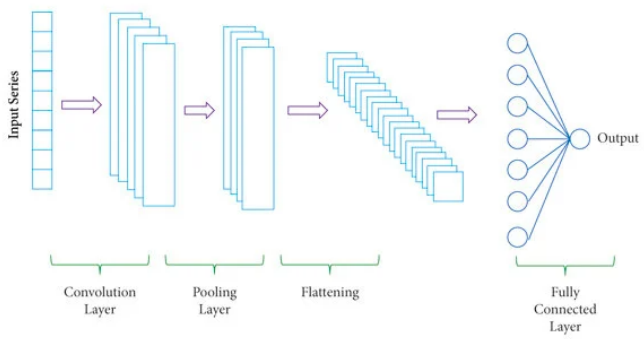

Figure 1.4: Standard architecture of CNN for time-series processing [19].

- **Recurrent Neural Networks (RNNs) and LSTMs:** Industrial degradation is a temporal process that requires sequence modeling. Standard RNNs process time-series data but experience the vanishing gradient problem [20], making them incapable of which limits their ability to track long-term trends. To overcome this limitation, Long Short-Term Memory (LSTM) networks utilize a gating structure to selectively retain or discard historical information. This architecture models the progressive degradation of rotating machinery [21]. Figure 1.5 details the structure of this network.

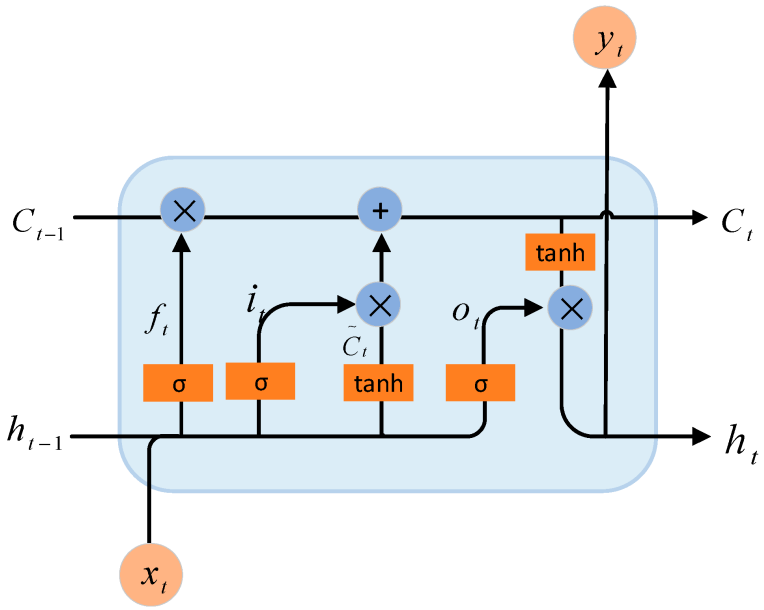

Figure 1.5: LSTM network structure [22].

- **Hybrid Models:** Individual architectures do not address all IIoT data challenges simultaneously. Research frameworks frequently implement hybrid models, such as CNN-LSTM or CNN-BiLSTM networks. These structures merge the local feature extraction of CNNs with the temporal forecasting of LSTMs to process high-dimensional, time-variant data [23].

1.5.3 Reinforcement Learning for Decision Making

Reinforcement Learning (RL) is a distinct paradigm of machine learning in which an autonomous agent learns to make sequential decisions by interacting with an environment. Unlike supervised learning (which maps inputs to static, human labeled data) or unsupervised learning (which finds hidden structures), RL is fundamentally driven by a reward signal. It operates on the principle of trial, error, and delayed gratification, optimizing a cumulative reward over time [24].

In industrial applications, RL provides the mathematical framework required for autonomous decision making. The interaction between the RL agent and the physical system is formalized mathematically as a Markov Decision Process (MDP), which consists of four primary conceptual components [24]:

- **State Space (S):** Represents the condition of the environment at any given

time step. In predictive maintenance, a state typically encapsulates the machine's current operational parameters and its estimated degradation level.

- **Action Space (A):** The set of all possible interventions the agent can execute. These actions can range from discrete choices (e.g., "continue operation" or "stop for maintenance") to continuous adjustments (e.g., reducing operational load or engine RPM).
- **Reward Function (R):** The critical optimization feedback. The agent receives positive reinforcement for favorable outcomes (e.g., safely extending operational life) and incurs massive numerical penalties for negative outcomes (e.g., allowing a machine to run to catastrophic failure).
- **Policy (π):** The underlying strategy the agent develops. The ultimate goal of any RL algorithm is to learn an optimal policy (π^*) that successfully maps every possible state to the best possible action, thereby maximizing the long term value of the system.

Integrating deep learning with reinforcement learning forms Deep Reinforcement Learning (DRL). This architecture maps predictive outputs directly to autonomous maintenance actions, establishing the operational basis for prescriptive systems. Figure 1.6 illustrates this complete progression, from physical sensor data collection to the final decision-making stage.

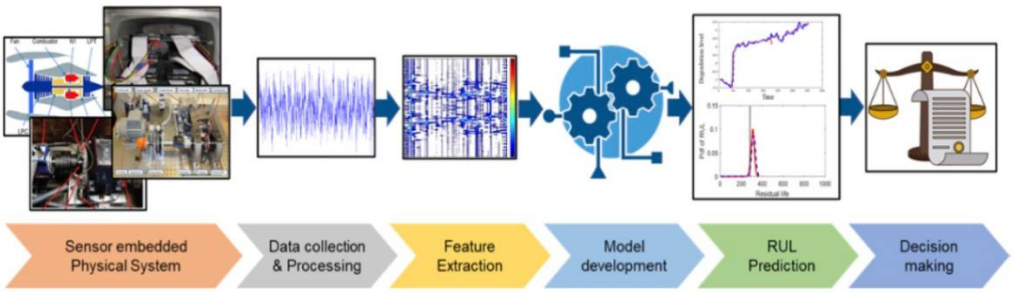

Figure 1.6: Predictive Maintenance pipeline [25].

1.6 Challenges and Limitations of Current Systems

Transitioning Deep Learning prognostic models from controlled experiments to Industrial Internet of Things (IIoT) deployments presents specific engineering challenges:

- **Data Quality Issues:** Industrial sensor data is noisy and imbalanced. Run-to-failure trajectories are rare compared to normal operational data. Training models on skewed datasets reduces accuracy, causing false alarms or missed detections in safety-critical environments [7].

- **Real-Time Constraints:** Deep sequence modeling (e.g., CNN and LSTM layers) requires high computational power, which conflicts with the low-latency requirements of industrial controllers. Deploying these architectures on edge devices is difficult due to hardware limitations, including restricted memory, limited processing power, and the absence of dedicated inference accelerators [2].
- **Lack of Integration (Prediction vs. Control):** Current industrial architectures lack integrated decision-making layers to translate high-dimensional predictions into actionable control policies. Traditional Deep Learning models function only as perception modules to forecast degradation. The reliance on human intervention or static, rule-based controllers creates an "Information to Action" delay. This delay restricts proactive risk mitigation and limits overall operational reliability [14].

Data quality and hardware deployment constraints fall outside the scope of this research. This work focuses on the direct integration of predictive perception with adaptive control. The proposed hybrid methodology links RUL estimation directly to actionable maintenance decisions to improve operational reliability.

1.7 Conclusion

This chapter established the core concepts of predictive maintenance. It traced the evolution of maintenance strategies to AI-driven prognostics and detailed the mechanisms of CNNs, LSTMs, and Reinforcement Learning. The text identified the operational disconnect between predictive perception and autonomous action in industrial systems. The following chapter reviews the State of the Art (SOTA) in recent literature. It analyzes existing methodologies to define the research gap and introduces the proposed hybrid framework as a direct solution to these limitations.

Chapter 2

Literature Review and State of the Art

2.1 Introduction

Chapter 1 established the theoretical concepts of predictive maintenance and deep learning. Building on that foundation, this chapter reviews the academic literature concerning Remaining Useful Life (RUL) estimation and maintenance decision-making. We analyze the progression from traditional physics-based models to advanced deep learning architectures. This review identifies the current limitations in industrial applications, establishing the context for our proposed hybrid methodology to improve operational reliability.

2.2 Taxonomy of Intelligent Maintenance Systems

To categorize the methodologies in the current literature, we propose a taxonomy of Intelligent Maintenance Systems. As illustrated in Figure 2.1, intelligent maintenance operates as a closed-loop framework divided into two main components: perception-based prognostic models (which estimate machine health and RUL) and prescriptive decision-making frameworks (which manage autonomous control and maintenance scheduling). The following subsections review the state of the art within each branch.

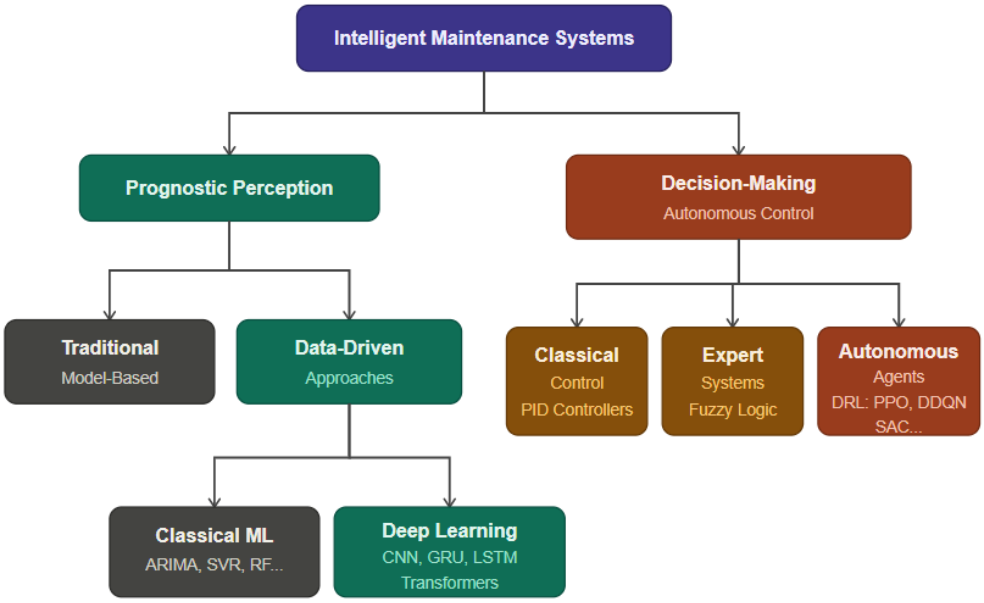

Figure 2.1: Taxonomy of Intelligent Maintenance Systems.

2.2.1 Traditional Maintenance and Model Based Approaches

Historically, machinery prognostics relied on structured physical laws or predefined schedule based maintenance before the mass adoption of data driven paradigms.

Model-Based Methods

Early research utilized mechanistic degradation models. (Paris and Erdogan, 1963) developed mathematical equations to quantify fatigue crack growth [26]. These analytical models provide accurate RUL estimations for isolated components under controlled conditions. However, applying these methods to complex systems, such as turbofan engines, introduces significant limitations. As highlighted by (Lei et al., 2018) achieving a complete understanding of the failure mechanisms in complex mechanical systems is highly difficult, which heavily restricts the application and scalability of purely physical models [27].

Preventive and Classical Methods

Parallel to physical modeling, traditional industrial systems utilize Preventive Maintenance (PvM). This strategy schedules interventions based on average life statistics rather than real-time component health. As noted by (Khelif et al., 2016) [28], these schedule-based methods can result in either "over-maintenance" (replacing healthy parts) or "under-maintenance" (unexpected failures).

To address these limitations, researchers applied statistical mapping and classical machine learning techniques, such as ARIMA and Support Vector Regression

(SVR) [29]. Unlike deep neural networks, these conventional models typically require manual feature engineering to extract temporal patterns from raw sensor data. This requirement for domain-specific feature extraction presents scalability challenges when applied to complex industrial systems, motivating the adoption of automated data-driven methods.

2.2.2 Deep Learning for RUL Prediction

The transition from classical machine learning to Deep Learning (DL) reduced the reliance on manual feature engineering through end-to-end learnable representations. Early DL applications for RUL estimation primarily relied on unsupervised generative models. For instance, (Zhang et al., 2016) [30] employed Deep Belief Networks (DBNs) to extract underlying health indicators from raw turbofan data, while other early studies utilized Stacked Autoencoders (SAEs) to learn compressed representations of normal operating states [31].These early architectures treated multivariate sensor inputs as flat vectors, which limited their ability to explicitly model spatial or temporal correlations.

To address the spatial complexity of multi-sensor data, researchers applied Convolutional Neural Networks (CNNs) to the prognostics domain. (Babu et al., 2016) [18] pioneered the application of deep CNNs for RUL prediction, demonstrating that convolutional filters could successfully extract robust spatial features while filtering high frequency noise. While effective for feature extraction, standard CNNs process each sliding window independently. Lacking internal memory, they face challenges in tracking the progressive, cumulative degradation of a machine over its full operational lifetime.

To overcome the stateless nature of CNNs, sequence modeling using Long Short Term Memory(LSTM) networks. (Zheng et al., 2017) [21] and (Wu et al., 2020) [32] successfully demonstrated that LSTMs, applying their complex internal gating mechanisms, could effectively retain very long term degradation trends across full engine life cycles without losing critical historical context. As a computationally lighter alternative, Gated Recurrent Units (GRUs) were also explored by (zhou et al., 2022) [33], and even combined with manual feature extraction by (Zhao et al., 2017) [34], to achieve similar temporal modeling. Nevertheless, a practical limitation remains: when pure LSTMs or GRUs are applied directly to raw, unfiltered industrial signals, their sequential processing makes them highly susceptible to high-frequency sensor noise. This not only affects model convergence but also significantly slows down the training process due to the computational burden of modeling extremely long, noisy sequences.

This inherent noise sensitivity drove the literature toward Hybrid Architectures, combining the spatial filtering of CNNs with the temporal memory of recurrent layers. (Al-Dulaimi et al., 2019) [23] established a robust benchmark by applying the CNN-LSTM framework to machinery RUL prediction tasks, where CNNs act as a front-end to clean signals before temporal processing.The necessity of refining the input space prior to sequence modeling is further supported by researchers such as (Zhu et al., 2023) [35] and (Sun et al., 2022) [36], who demonstrated that fusing CNNs with recurrent layers not only improves deterministic prediction accuracy but also significantly enhances the model's resilience against

multi scale sensor noise.

Seeking further improvements, recent studies have explored more complex hybrid configurations. For example, (Zhao et al., 2020) [37] developed a CNN-BiLSTM network to capture bidirectional temporal context, while other works investigated CNN-GRU models [38] to balance execution speed and sequence retention. However, these alternatives present distinct disadvantages in prognostics. BiLSTM architectures often introduce unnecessary computational overhead by processing future data points that do not influence the current health state. Conversely, simplified CNN-GRU hybrids often struggle to retain the dependencies of exceptionally long operational sequences. as a result , the forward facing CNN-LSTM architecture remains a suitable and computationally stable choice for machinery RUL prediction.

Recently, research has advanced toward self attention mechanisms and Transformers. For instance, (Yu et al., 2025) [39] utilized a hybrid CNN-BiLSTM model with a dual attention mechanism to dynamically weigh spatial and temporal features, while fully Transformer based architectures, such as the two-stage hierarchical model proposed by (Fan et al., 2024) [40], have been developed to achieve high prognostic accuracy. While these attention based models represent a state of the art approach for deterministic RUL forecasting, their quadratic computational complexity and high dimensional latent outputs present challenges for real-time execution in low dimensional state spaces, such as those required by Deep Reinforcement Learning (DRL) agents. Consequently, the standard CNN-LSTM architecture provides a favorable balance, delivering robust, noise resilient temporal memory while maintaining the compact state representation necessary for integrating an intelligent, closed loop decision engine.

2.2.3 Uncertainty Quantification in Deep Learning

While the evolution from standard RNNs to hybrid architectures has significantly improved the deterministic accuracy of RUL predictions, a critical gap remains. The vast majority of the aforementioned models operate as deterministic function approximators; they output a single, absolute point estimate for the remaining useful life. In safety critical applications such as aerospace propulsion, making irreversible maintenance decisions based on blind point estimates increases the risk of catastrophic failure if the model is overconfident in a noisy data region.

To address this, the cutting edge of prognostic research has shifted toward Uncertainty Quantification (UQ). The integration of Monte Carlo simulations with deep learning architectures is increasingly recognized as a notably strategy for robust risk mitigation. For instance, recent studies by (Tong et al., 2021) [41] and (Wang et al., 2023) [42] demonstrated that coupling Monte Carlo methods with recurrent networks effectively mitigates overfitting and provides reliable uncertainty bounds. Similarly, in the context of aero engines, researchers like (Huang et al., 2021) [43] explored Bayesian Neural Networks combined with LSTM autoencoders to achieve high confidence RUL estimations.

As an efficient alternative to heavy Bayesian updating, Monte Carlo (MC) Dropout transforms standard networks into risk-aware probabilistic estimators [15, 44]. To understand how AI agents leverage these predictions for decision making,

the next section explores the evolution of Deep Reinforcement Learning (DRL) in adaptive maintenance controll.

2.2.4 From Prediction to Dynamic Action: Evolution of Adaptive Control Frameworks

Although perception based prognostic models can accurately estimate the Remaining Useful Life (RUL) of an asset, forecasting alone does not automatically resolve the operational dilemma of determining exactly when to perform maintenance. Addressing this specific challenge translating predictive insights into actionable, dynamic prescriptive policies constitutes the primary objective of this thesis.

Historically, translating predictions into maintenance actions relied on Expert Systems and Fuzzy Logic [45]. These systems operate by mapping condition monitoring data to specific maintenance decisions using predefined "IF-THEN" rule bases. While highly transparent, these rule-based frameworks are inherently static and rely heavily on manual tuning, making them difficult to scale. To automate this process, researchers explored mathematical optimization techniques, treating maintenance scheduling as a cost minimization problem using metaheuristics like Particle Swarm Optimization (PSO) [46]. While effective for static optimization snapshots, PSO requires significant computational resources to operate as a continuous control loop and faces challenges in adapting dynamically to unforeseen stochastic failures.

To achieve autonomous, real-time adaptability, the research frontier shifted toward Deep Reinforcement Learning (DRL). Formulating maintenance scheduling as a Markov Decision Process (MDP) allows an agent to learn policies through continuous environmental interaction. Early DRL applications utilized value based methods. For instance, (Huang et al., 2020) [47] applied Double Deep Q-Networks (DDQN) to optimize preventive maintenance in a serial production line. However, value based algorithms like DDQN are restricted to discrete action spaces. As the complexity of the industrial system grows, evaluating every possible discrete action introduces computational challenges associated with the curse of dimensionality, making these methods difficult to scale for continuous control settings.

To overcome the limitations of discrete value-based methods, the literature adopted Actor-Critic architectures, such as Proximal Policy Optimization (PPO), which directly optimize the decision policy. While PPO provides stable learning in continuous spaces, its on-policy nature typically requires extensive environmental interaction. In industrial maintenance scenarios, where machine failures are rare and reward signals are sparse, on-policy methods often face challenges with sample efficiency and may prematurely converge to sub optimal policies due to insufficient exploration [48].

To address these sample efficiency and exploration challenges, research has explored the Soft Actor-Critic (SAC) algorithm [48, 49]. SAC operates off-policy to improve data efficiency and incorporates entropy maximization into its objective function. Rather than strictly preventing local optima, this entropy regularization encourages broader exploration of the state space, which helps mitigate the risk of premature convergence and assists the agent in finding a more robust maintenance

policy.

More critically, SAC is well suited to handle probabilistic state spaces. In a study on aerospace prescriptive maintenance, (Lee and Mitici.,2023) [15] demonstrated that feeding probabilistic RUL estimates generated via MC Dropout into an Actor-Critic agent provides advantages over relying solely on deterministic predictions. By exposing the agent to the model's epistemic uncertainty, the DRL agent can learn a highly adaptive policy focused explicitly on reducing risks and improving reliability. When the prediction is highly uncertain, the agent can schedule early maintenance to mitigate risk; when the prediction is confident, it extends the operational exploitation of the asset.

2.3 Summary of the literature review

Table 2.1 summarizes the reviewed literature, outlining the various prediction techniques and decision making strategies employed in recent studies.

Table 2.1: Summary of the literature review.

Reference	**Dataset**	**Prediction Technique**	**Decision Making**	**Limitations**
Zio and Di Maio [50]	LBE-XADS (Nuclear) [51]	Traditional Data Driven	Fuzzy Logic	Static rule base; heavily relies on manual tuning by domain experts; lacks dynamic adaptability.
Zhang et al. [30]	C-MAPSS [52]	Deep Belief Networks (DBN)	None (Passive)	Deterministic outputs only; no actionable control or uncertainty estimation.
Al-Dulaimi et al. [23]	C-MAPSS [52]	Hybrid CNN-LSTM	None (Passive)	Blind to prediction confidence; unable to dynamically adapt maintenance schedules.
Huang et al. [47]	Simulated Serial Line [47]	Raw System States	DDQN (Value Based)	Restricted to discrete action spaces; suffers from the curse of dimensionality.
Aglogallos et al. [48]	2010 PHM CNC [53]	RL Observation State	PPO / A2C	Prone to sub-optimal local minima due to on-policy sample inefficiency and sparse rewards.
Lee and Mitici [15]	C-MAPSS [52]	CNN	Soft Actor-Critic (SAC)	CNN only; lacks explicit long-term temporal modeling.
Proposed Framework	**C-MAPSS** [52]	**CNN-LSTM**	**Soft Actor-Critic (SAC)**	**Overcomes prior limitations by separating multi-condition states via K-Means clustering prior to probabilistic control.**

2.4 Research Gaps and Thesis Contributions

2.4.1 Discussion of Research Gaps

The critical analysis of the literature reveals a research gap in intelligent maintenance. Despite the use of Deep Learning in estimating the Remaining Useful Life (RUL) of industrial assets, a "perception action gap" remains. Current hybrid models, such as CNN-LSTM architectures, provide accurate RUL predictions but

generally function as "open-loop" systems focused solely on condition estimation. They forecast *when* a failure will occur but do not prescribe *what to do* about it, lacking autonomous adaptive control capabilities.

Conversely, while Reinforcement Learning (RL) has shown potential for closed-loop control within Industrial IoT environments, applying RL directly to raw, noisy IIoT sensor data often leads to large state spaces and poor training convergence. Therefore, there is a need for a framework that bridges this functional disconnect by combining feature extraction with autonomous maintenance scheduling.

2.4.2 Research Objectives

To address these identified gaps, this thesis aims to shift the industrial maintenance approach from passive prediction to risk-aware adaptive control. The main objectives are defined as follows:

1. To design and evaluate a model capable of filtering noise and extracting spatiotemporal features from multivariate sensor data for RUL prediction.
2. To develop an agent utilizing the Soft Actor-Critic (SAC) algorithm that leverages the perception outputs to autonomously prescribe maintenance actions, balancing equipment exploitation with safety constraints.

2.4.3 Main Contributions

The primary contribution of this work is the integration of perception (Deep Learning) and control (Deep Reinforcement Learning) into a Prescriptive Maintenance framework. By substituting raw sensor data with uncertainty-aware degradation states generated by the CNN-LSTM and Monte Carlo Dropout, this research reduces the state-space complexity for the RL agent. This two-stage architecture aims to improve training convergence and produce effective, cost-sensitive maintenance policies compared to isolated RL or standard prognostic approaches.

2.5 Conclusion

In this chapter, we have conducted a detailed review of the related work and existing literature concerning RUL estimation and autonomous maintenance decision making. We have critically analyzed the progression of prognostic models, evaluating the transition from traditional model based and fuzzy logic approaches to state of theart Deep Learning architectures. In addition, we have explored the strengths and limitations of advanced networks, including CNNs and LSTMs, in handling complex industrial sensor data.

The review also highlighted the recent shift towards adaptive control using Deep Reinforcement Learning. We examined the evolution from value based methods like DDQN to policy gradient approaches like PPO and SAC, identifying the critical need for uncertainty quantification and temporal memory in the decision making process.

Having established this critical evaluation of the current state of the art, we are now poised to move forward into the proposed methodology in the following chapter. By addressing the identified gap between accurate RUL perception and automated decision making, we will introduce an integrated framework designed to push the boundaries of intelligent, risk-aware, and adaptive industrial maintenance.

Chapter 3

Proposed Methodology

3.1 Introduction

This chapter presents the core contribution of this thesis: an integrated Prescriptive Maintenance framework designed to bridge the gap between failure prediction and autonomous decision making. We propose a two phase methodology to achieve this. First, a hybrid CNN-LSTM architecture is developed to extract complex spatio-temporal features from raw industrial telemetry, providing robust RUL estimations. Second, a DRL agent is introduced within a simulated environment. By using the perception outputs of the CNN-LSTM, the DRL agent learns to autonomously prescribe optimal control actions, effectively extending the machine's safe operational life while balancing production demands.

3.2 Problem Statement

Before detailing our proposed methodology, it is essential to formally define the dataset utilized in this study and the exact computational challenges our framework is designed to resolve.

3.2.1 Data Description

The dataset utilized in our research is the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [52], provided by NASA. It is a widely recognized benchmark dataset for predictive maintenance and RUL estimation. The dataset contains simulated run-to-failure degradation trajectories of turbofan engines under various operating conditions and fault modes. It is divided into four distinct sub-datasets (FD001 to FD004), the characteristics of which are detailed in Table 3.1.

Table 3.1: Parameters of the C-MAPSS dataset.

	C-MAPSS			
Dataset	**FD001**	**FD002**	**FD003**	**FD004**
Train trajectories	100	260	100	249
Test trajectories	100	259	100	248
Operating conditions	1	6	1	6
Fault modes	1	1	2	2
Training samples	20,631	53,759	24,720	61,249
Testing samples	100	259	100	248

The multiple operating conditions in FD002 and FD004 introduce significant signal variance, which obscures the subtle degradation signatures of the engine. Consequently, this necessitates the implementation of a condition-aware preprocessing layer within our framework to isolate the degradation trends from the operational variations. Structurally, each of the four sub-datasets is provided in three distinct files, as detailed in Table 3.2.

Table 3.2: Structure and purpose of the provided files in each C-MAPSS sub-dataset.

File Type	**Description and Purpose**
Training Set	Contains complete run-to-failure trajectories for a set of engines.
Testing Set	Contains trajectories that are truncated at an arbitrary point prior to failure.
Ground Truth RUL	Provides the true RUL values corresponding to the last recorded cycle of each engine in the testing set, serving as the absolute evaluation baseline.

3.2.2 Target RUL Formulation (Piecewise Linear)

In physical turbofan engines, degradation is negligible during the early stages of operation. Assuming a strictly linear decreasing RUL from the first cycle is physically inaccurate and forces the model to learn non-existent degradation patterns.

To address this, our framework employs a **Piecewise Linear RUL** target function, capping the maximum RUL at a constant threshold of 125 cycles. This threshold is a well-established standard in prognostics literature to prevent the overestimation of early-stage predictions [21]:

$$RUL_{target}(t) = \begin{cases} 125 & \text{if } RUL_{actual}(t) > 125 \\ RUL_{actual}(t) & \text{if } RUL_{actual}(t) \leq 125 \end{cases} \tag{3.1}$$

This formulation ensures that the deep learning model focuses its capacity strictly on the actual degradation phase rather than the healthy operational cycles.

3.2.3 Problem Formulation and Objectives

Based on the operational context and the provided C-MAPSS dataset, the core problem is formulated around continuous monitoring and prescriptive decision-making for degrading turbofan engines. Specifically, our framework is designed to achieve the following two primary objectives:

1. **RUL Prediction:** To accurately predict the RUL of the engines by effectively extracting spatiotemporal degradation features from complex, multi-regime sensor data.
2. **Adaptive Maintenance Scheduling:** To leverage these predictions within a RL environment to autonomously prescribe optimal maintenance actions, mitigating the risk of sudden equipment failures while minimizing unnecessary early component replacements.

The subsequent sections detail our proposed methodology and describe how our integrated architecture is designed to solve these specific challenges.

3.3 Overall Framework Architecture

The proposed prescriptive maintenance framework integrates Deep Learning and Deep Reinforcement Learning to achieve accurate RUL prediction and intelligent decision making. As illustrated in Figure 3.1, the complete system is designed as a two-phase architecture:

Phase 1: Perception Module
This initial phase is responsible for processing raw data and estimating the health state of the equipment. It consists of the first three steps:

- **Step 1: Preprocessing:** Raw multivariate sensor data collected from industrial equipment is cleaned, clustered, and smoothed to filter out severe operational noise and extract true degradation trends.
- **Step 2: Deep Perception (CNN-LSTM):** A hybrid neural network automatically extracts local spatial features and long-term temporal dependencies from the preprocessed sequences to estimate the deterministic Remaining Useful Life.
- **Step 3: Uncertainty Quantification (MC Dropout):** To improve reliability, stochastic forward passes generate a probabilistic distribution of engine failure, providing a richer characterization of the system's health state and risk level.

Phase 2: Deep Reinforcement Learning (DRL)
The second phase focuses entirely on autonomous decision-making based on the perceptions gathered.

- **Step 4: Adaptive Control (SAC Agent):** The generated probabilistic degradation state is fed into a Deep Reinforcement Learning environment. A Soft Actor-Critic agent learns to dynamically prescribe effective maintenance actions, balancing operational safety with maintenance costs.

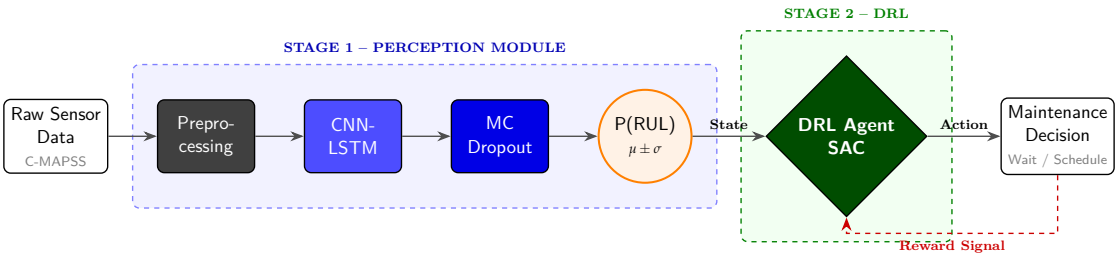

Figure 3.1: Overall Framework Architecture.

3.4 Data Preprocessing Pipeline

Once the target RUL is formulated, the raw sensor data undergoes a rigorous, multi step preprocessing pipeline. Because the subsets vary in complexity, a dual strategy approach is applied to improve the raw sequences into a robust format suitable for the perception layer. The pipeline is executed through the following sequential steps:

1. Data Cleaning and Sensor Selection
The initial step focuses on dataset purification to reduce computational overhead and prevent overfitting. The C-MAPSS dataset provides 21 sensor measurements. However, a rigorous variability and correlation analysis revealed that 7 sensors exhibit near constant values or are dominated by irregular noise. These uninformative features were systematically discarded. Thus, exactly **14 sensors** demonstrating moderate to high variability and strong monotonic correlation with degradation were selected, as detailed in Table 3.3.

Table 3.3: Sensor physical classification and selection rationale based on the official NASA C-MAPSS documentation [54].

Sensor Group	Sensors	Variability Over Time	Correlation with RUL	Selection Rationale
Temperature	s2, s3, s4,s11	High	Strong (monotonic)	Clear thermodynamic degradation trends.
Pressure	s7,s8,s13	Moderate	Moderate	Regime-sensitive behavior.
Speed related	s12,s15,s16	High	Strong	Directly reflect mechanical wear and tear.
Flow & Fuel related	s10,s14,s20,	High	Strong	Highly correlated with engine health deterioration.
Near constant	s1, s5, s6, s18, s19	Very Low	Weak / None	Excluded due to constant variance over time.
Noise dominated	s17, s21	Irregular	Weak	Excluded due to redundancy and lack of fluctuation.

2. Operating Condition Clustering (K-Means)
To address the severe operational noise inherent in the complex datasets (FD002

and FD004), the **K-Means clustering algorithm** is applied to the three operational settings (Altitude, Mach Number, and Throttle). The number of clusters is explicitly set to $K = 6$. The algorithm groups similar operational states, assigning a discrete condition label ($c \in \{1, ..., 6\}$) to every time step. *Note: This step is bypassed for FD001 and FD003, as they operate under a single condition ($K = 1$).*

3. Adaptive Data Normalization
With the operating regimes identified, the 14 cleaned sensor signals are standardized. To handle the differing noise profiles, an adaptive scaling approach was adopted:

- **Min Max Scaling (FD001/FD003):** Applied globally to improve the single condition sensor readings into a $[0, 1]$ range.
- **Robust Scaling (FD002/FD004):** A Robust Scaler is employed to handle extreme signal outliers during regime transitions. It centers data using the median and scales according to the Interquartile Range (IQR). Notably, this is applied independently within each of the six K-Means clusters.

4. Temporal Smoothing via Exponential Moving Average (EMA)
To extract the pure monotonic degradation trend from high frequency noise, an **EMA** filter is applied:

$$s_t = \alpha \cdot x'_t + (1 - \alpha) \cdot s_{t-1} \tag{3.2}$$

An EMA smoothing factor of $\alpha = 0.3$ is utilized for the stable FD001/FD003 datasets, whereas a heavier smoothing factor of $\alpha = 0.1$ is applied to the highly volatile FD002/FD004 subsets to aggressively filter out transition noise.

5. Sequential Data Windowing (Sliding Window)
Because the subsequent perception layer utilizes recurrent architectures designed for sequential data, the preprocessed continuous readings are restructured into discrete sequences using a Sliding Window technique. A time window of length T_w slides over the data with a step size of 1. The input sequence X_t is defined as:

$$X_t = [s_{t-T_w+1}, s_{t-T_w+2}, \ldots, s_t] \in \mathbb{R}^{T_w \times 14} \tag{3.3}$$

The sequence length was specifically configured based on dataset complexity: $T_w = 30$ for FD001/FD003, and expanded to $T_w = 50$ for FD002/FD004 to provide a broader temporal context for decoding multi-condition patterns.

Having established the preprocessing pipeline that transforms raw sensor data into clean, structured sequences, the following section details the CNN-LSTM architecture that processes these sequences to estimate the RUL.

3.5 CNN-LSTM Predictive Model

The perception module is engineered to process high-dimensional sensor data and accurately estimate the RUL. The exact architecture developed and deployed in this study is illustrated in Figure 3.2. The model is sequentially structured,

beginning with a 1D-CNN for spatial feature extraction, followed by LSTM layers for temporal sequence modeling, and concluding with a deep fully connected network.

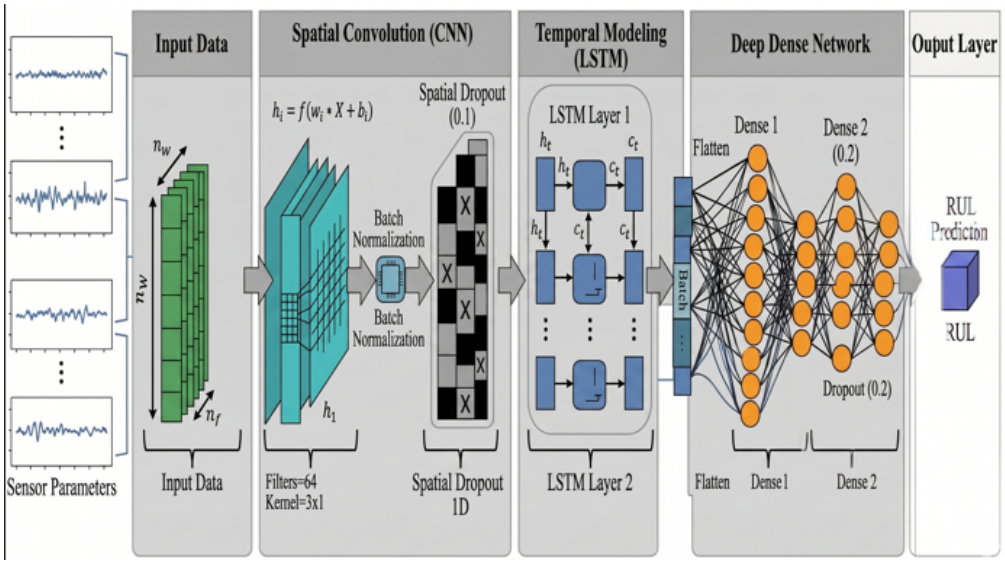

Figure 3.2: Architecture of the Proposed Hybrid CNN-LSTM model for RUL prediction.

Following the architectural design, the model must be systematically trained to learn the complex degradation patterns. The complete training procedure for this hybrid architecture is outlined in Algorithm 1. The process begins by loading the raw C-MAPSS dataset and applying the rigorous, multi step preprocessing pipeline established in the previous section. Once the data is normalized, smoothed, and segmented into discrete time windows, the sequences are fed into the CNN-LSTM networks. The model maps these spatiotemporal features to a deterministic RUL value, optimizing its weights iteratively using backpropagation and dataset specific loss functions.

Algorithm 1 CNN-LSTM Model Training for RUL Prediction

Require: Training dataset D_{train}, Sequence length L, Selected sensor features F

1: Load C-MAPSS dataset and compute target RUL
2: Perform data preprocessing:
- Feature selection
- Operating condition clustering
- Adaptive data normalization
- Signal smoothing (EMA)

3: Generate time-series sequences using sliding windows of length L
4: **for** each engine sequence in training set **do**
5: Extract spatial degradation features using 1D-CNN layers
6: Learn temporal degradation dependencies using Stacked LSTM layers
7: Output deterministic RUL prediction
8: **end for**
9: Calculate Training Loss
10: Train the CNN-LSTM model using backpropagation
11: Optimize model parameters using Adam optimizer
12: **return** Trained CNN-LSTM Model

Spatial Feature Extraction (CNN)

The initial layer is a 1D Convolutional layer equipped with 64 filters and a kernel size of 3, using 'same' padding to preserve the temporal length of the input sequence. Given an input sequence X, the feature map h_i generated by the i-th filter is mathematically defined as:

$$h_i = f(w_i * X + b_i) \tag{3.4}$$

To accelerate convergence and ensure training stability, the CNN output is immediately processed by a Batch Normalization layer. Subsequently, rather than using traditional pooling which might disrupt temporal resolution, a Spatial Dropout 1D layer (rate = 0.1) is applied. This layer drops entire 1D feature maps, promoting independence between channels and effectively preventing overfitting in sequential data.

Temporal Sequence Modeling (Stacked LSTMs)

The spatially extracted sequences are fed into a temporal block of two stacked LSTM layers to overcome the vanishing gradient problem and capture long term degradation dependencies:

- **First LSTM Layer:** Contains 100 hidden units (*return_sequences=True*) with an internal dropout rate of 0.2 to regularize the recurrent connections.
- **Second LSTM Layer:** Contains 50 hidden units (*return_sequences=False*), effectively summarizing the entire degradation history into a single high density vector.

Fully Connected Network and RUL Prediction

The temporal vector is passed through a dense feed forward network to map the extracted features to a continuous RUL value. This consists of:

1. A Dense layer with 64 units (ReLU activation), followed by Batch Normalization and Dropout (rate 0.2).
2. A subsequent Dense layer with 32 units (ReLU activation) to further refine the representations.
3. A final Output Dense layer consisting of a single neuron with a linear activation function, which outputs the predicted RUL.

Dual Configuration Strategy and Model Optimization

To handle varying operational complexities, the training strategy was dynamically adapted based on the target dataset. The full setup is detailed in Table 3.4.

Table 3.4: Hyperparameters and setup configurations for the proposed CNN-LSTM framework.

Category	Parameter	Description	FD001 / FD003	FD002 / FD004
Data Input	Clustering (K)	Number of K-Means clusters for operational settings	None ($K = 1$)	6 Clusters
	Normalization	Type of scaling applied to sensors	Min Max Scaler	Robust Scaler
	Smoothing (α)	Exponential Moving Average (EMA) factor	0.3	0.1
	Sequence Length	Length of the sliding time window (*seq_length*)	30	50
Model Structure	Conv1D Filters	Number of filters and kernel size	64 (Kernel=3)	64 (Kernel=3)
	LSTM Units	Hidden units in LSTM layers (Layer 1, Layer 2)	100, 50	100, 50
	Dense Units	Hidden units in Fully Connected layers	64, 32	64, 32
	Dropout Rate	Regularization dropout rate	0.2	0.2
Training Setup	Batch Size	Number of samples per training batch	32	128
	Loss Function	Optimization objective function	MSE	Huber ($\delta = 15.0$)
	Optimizer	Algorithm and learning rate	Adam ($lr = 10^{-3}$)	Adam ($lr = 10^{-3}$)
	Max Epochs	Maximum number of training iterations	100	100
	Patience	Early stopping patience (epochs)	20	20

As shown in the training setup, a primary distinction in this dual strategy is the adaptation of the optimization objective to account for dataset complexity.

Loss Function Selection

We utilize Mean Squared Error (MSE) for the stable, single-condition subsets (FD001 and FD003). However, the multi-condition subsets (FD002 and FD004) contain extreme noise and outliers that destabilize MSE. To address this, we employ the Huber Loss function, which applies a quadratic penalty for small errors and a linear penalty for large ones:

$$L_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq \delta \\ \delta(|y - \hat{y}| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \tag{3.5}$$

Here, y is the actual RUL, $\hat{y}$ is the prediction, and $\delta = 15.0$ is the transition threshold. This specific configuration tolerates early-stage noise while enforcing strict precision within a 15-cycle error margin.

While the CNN-LSTM provides accurate deterministic RUL estimates, a single point prediction is insufficient for safety-critical decision-making. The following section introduces the probabilistic estimation layer that quantifies prediction uncertainty.

3.6 Probabilistic RUL Estimation

Traditional deterministic RUL predictions fail to capture the inherent uncertainty of industrial environments caused by operational variability and sensor noise. To address this, the proposed framework employs a probabilistic approach. Rather than relying on a single point estimate, it generates a complete probability distribution of future degradation behavior, directly capturing both degradation trends and uncertainty levels.

The resulting distribution serves as the observation state for the SAC reinforcement learning agent. By analyzing this complete profile instead of a single value, the agent makes more adaptive and reliable maintenance decisions under uncertain conditions, as detailed in Algorithm 2.

Algorithm 2 Probabilistic State Generation via Monte Carlo Dropout

Require: Trained CNN-LSTM Model
Require: Input sensor sequence X_{input}
Require: Number of MC iterations N
for each continuous input sequence X_{input} **do**
  for $i = 1$ to N **do**
    Maintain Dropout layers active during inference
    Perform stochastic forward pass
    Store predicted RUL $\hat{y}_i$
  end for
  Compute probabilistic failure distribution over prediction horizon:

$$p_{k,t} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}(\hat{y}_i \leq k)$$

  return Probabilistic degradation state vector $p_{k,t}$
end for

Having established a probabilistic representation of engine health, the framework is now equipped to feed this uncertainty-aware state into the autonomous decision-making agent, detailed in the following section.

3.7 Adaptive Control Using SAC

After estimating the probabilistic RUL distribution, the next objective is to identify the optimal maintenance action for each engine condition. Traditional maintenance strategies generally rely on fixed thresholds or predefined maintenance schedules. However, such approaches are not adaptive and may lead either to unnecessary early maintenance or unexpected engine failures.

To overcome this limitation, the proposed framework integrates DRL for adaptive maintenance decision making. In this work, the Soft Actor-Critic (SAC) [49] algorithm is adopted because of its ability to learn continuous control policies while balancing exploration and exploitation. The SAC agent learns maintenance strategies directly from engine degradation behavior and dynamically adjusts maintenance decisions based on the predicted risk of failure.

Reinforcement Learning Environment

The adaptive maintenance problem is modeled as a reinforcement learning environment where the SAC agent continuously interacts with engine degradation states.

At each decision step:

- The environment provides the probabilistic degradation state generated by the CNN-LSTM model.

- The SAC agent analyzes this state and selects a maintenance action.
- The environment evaluates the action and returns a reward.
- Based on this feedback, the agent updates its policy to improve future decisions and maximize long term rewards.

Through this continuous interaction process, the agent progressively learns effective maintenance strategies using a trial and error learning mechanism.

State Representation

The observation state provided to the SAC agent corresponds to the probabilistic degradation distribution generated during the probabilistic RUL estimation phase.

The state is represented by the probability vector:

$$p_{k,t} = [p_{1,t}, p_{2,t}, \ldots, p_{D,t}]$$

where each element represents the probability that the engine fails within future operating cycles.

Instead of relying on a single predicted RUL value, the SAC agent receives the complete degradation probability distribution, allowing it to better evaluate uncertainty and future failure risk.

Action Space Formulation

The Soft Actor-Critic (SAC) agent, inspired by the formulation proposed by (Lee and Mitici.,2023) [15], is designed to produce continuous maintenance scheduling actions, which is highly suitable for estimating time to maintenance under uncertainty. At each decision step, the agent outputs a continuous scalar action:

$$a_t \in [1.0, D_{cycles} + 1] \tag{3.6}$$

where $D_{cycles} = 30$ represents the maximum predictive horizon defined by the probabilistic state space.

Rather than a rigid binary choice (e.g., maintain now vs. do not maintain), this continuous output provides a flexible and explicit scheduling window. The continuous action a_t is dynamically evaluated against the operational threshold D_{cycles}:

- **Proactive Maintenance ($a_t \leq D_{cycles}$):** The scalar action is rounded to the nearest integer $k = \text{round}(a_t)$, triggering a scheduled maintenance intervention exactly k cycles in the future.
- **Safe Continuation ($a_t > D_{cycles}$):** The engine is deemed healthy within the current prediction window. No immediate maintenance is scheduled, and the operational cycle advances.

Reward Function

The reward function r_t is the principal driving force that guides the SAC agent toward prescriptive behavioral optimality. The objective is to strike a delicate economic and safety balance: maximizing the engine's useful life while strictly penalizing unexpected systemic failures and unnecessary premature interventions.

To precisely align with real world industrial constraints, the environment computes the reward based on a cost sensitive mathematical formulation. The agent's prescribed maintenance point k is evaluated against the actual ground truth Remaining Useful Life (RUL_{true}):

$$r_t = \begin{cases} -C_{uns} - C_{slot}, & \text{if } a_t \leq D_{cycles} \text{ and } RUL_{true} < k \text{ (Late Intervention)} \\ -(C_0 - C_1 \cdot k) - C_{slot}, & \text{if } a_t \leq D_{cycles} \text{ and } RUL_{true} \geq k \text{ (Scheduled Maintenance)} \\ -C_{uns} - C_{slot}, & \text{if } a_t > D_{cycles} \text{ and } RUL_{true} \leq D_{cycles} \text{ (Missed Failure)} \\ 0, & \text{if } a_t > D_{cycles} \text{ and } RUL_{true} > D_{cycles} \text{ (Safe Continuation)} \end{cases} \tag{3.7}$$

Notice that the agent receives an identical severe penalty ($-C_{uns} - C_{slot}$) for both a Late Intervention and a Missed Failure. This mathematically forces the policy to learn that whether it schedules maintenance too late or completely ignores the danger, the catastrophic cost to the system remains the same.

The specific cost parameters governing this environment are explicitly calibrated as:

- $C_{uns} = 2.0$: Penalty for an unscheduled, catastrophic in service failure.
- $C_{slot} = 0.5$: Fixed logistical cost for allocating a maintenance slot.
- $C_0 = 1.0$: Base penalty for triggering a planned maintenance intervention.
- $C_1 = 0.01$: Degradation reward factor, incentivizing the agent to safely maximize the component's lifespan.

Through continuous interaction with this cost environment, the SAC agent progressively learns policies that maximize the cumulative reward, effectively optimizing long term operational efficiency.

Soft Actor-Critic Algorithm

Soft Actor-Critic (SAC) is an off-policy Deep Reinforcement Learning algorithm designed for continuous action spaces. SAC combines policy optimization with entropy maximization in order to improve exploration stability during training.

The algorithm is composed of:

- An Actor network responsible for generating maintenance actions.

- Two Critic networks responsible for evaluating action quality.
- A replay buffer used to store past experiences.

Unlike traditional reinforcement learning methods, SAC incorporates an entropy term into its optimization process. This mechanism promotes exploration and reduces the risk of the agent converging prematurely to non-optimal maintenance strategies.

During training, the SAC agent continuously updates its policy using interactions collected from the maintenance environment until an optimal adaptive maintenance strategy is learned. The complete procedure, encompassing both the training phase and the operational evaluation phase, is systematically outlined in Algorithm 3.

Algorithm 3 Adaptive Maintenance Policy via Soft Actor-Critic (SAC)

Require: Probabilistic RUL state vector $p_{k,t}$, Look-ahead horizon D_{cycles}, Environment $\mathcal{E}$

Initialize Actor π_ϕ, Critics $Q_{\theta_{1,2}}$, Replay Buffer $\mathcal{D}$, and temperature α

Phase 1: SAC Agent Training
for each training episode **do**
    Reset environment $\mathcal{E}$ and observe initial state $s_0 = p_{k,t}$
    while episode not terminated **do**
        Sample action $a_t \sim \pi_\phi(\cdot|s_t)$, execute in $\mathcal{E}$, and observe r_t, s_{t+1}
        Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$; update networks using mini-batch
        $s_t \leftarrow s_{t+1}$
    end while
end for

Phase 2: Operational Policy Evaluation (On Test Engines)
for each unseen evaluation engine **do**
    Reset testing environment $\mathcal{E}_{test}$; set Done $\leftarrow$ False
    while not Done **do**
        Compute state $s_t = p_{k,t}$ via MC Dropout; predict action $a_t = \mu_\phi(s_t)$
        if $a_t \leq D_{cycles}$ **then**
            Schedule proactive maintenance after $k = \text{Round}(a_t)$ cycles
            Record Wasted Useful Life; Done $\leftarrow$ True
        else
            Safe to continue; advance environment by D_{cycles}
            if Engine fails **then** Record Unscheduled Failure; Done $\leftarrow$ True
            end if
        end if
    end while
end for

Adaptive Maintenance Decision Process

Once training is completed, the SAC agent is capable of making adaptive maintenance decisions for unseen engine conditions.

For each engine snapshot:

1. Sensor data is processed by the CNN-LSTM model.
2. Monte Carlo Dropout generates probabilistic RUL estimates.
3. The probability distribution is transformed into the state vector $p_{k,t}$.
4. The SAC agent receives the state and outputs a maintenance action.
5. The final maintenance decision is executed.

This integration between probabilistic prediction and reinforcement learning enables intelligent predictive maintenance capable of operating under uncertain Industrial IoT environments.

3.8 Conclusion

This chapter presented our integrated prescriptive maintenance framework, designed to bridge the gap between failure prediction and autonomous decision-making. Our methodology begins with condition-aware data preprocessing, followed by a hybrid CNN-LSTM architecture to predict the deterministic RUL from sensor data. To account for industrial noise, MC Dropout is integrated to quantify predictive uncertainty, transforming single-point predictions into probabilistic degradation states. Finally, these probabilistic states are utilized by a SAC agent to learn adaptive maintenance policies, providing a risk mitigation approach that overcomes the limitations of static threshold strategies. The next chapter will experimentally validate our framework using the NASA C-MAPSS dataset.

Chapter 4

Experiments and Results

4.1 Introduction

This chapter presents the experimental framework and results derived from the proposed integrated architecture for predictive maintenance and adaptive control. The primary objective is to evaluate the system's performance across its two core operational phases: (1) the accuracy of RUL estimation by the CNN-LSTM perception module, and (2) the effectiveness of the maintenance policies formulated by the SAC DRL agent. Subsequent sections detail the hardware and software environment, evaluation metrics, and algorithmic training configurations. This is followed by a thorough discussion of the experimental results, including RMSE, the asymmetric scoring function, and cumulative DRL rewards.

4.2 Experimental Setup

To ensure the reproducibility of the proposed framework and to handle the computational complexity of training deep neural networks and reinforcement learning agents, a robust experimental environment was established.

4.2.1 Hardware and Software Environment

All experiments, including data preprocessing, model training, and adaptive policy learning, were conducted using the **Kaggle** cloud computing platform. Applying this cloud based environment provided robust hardware acceleration, specifically dual GPUs, which was critical for accelerating the convergence of the CNN-LSTM architecture and managing the extensive interactions required by the SAC agent.

The detailed hardware specifications of the deployed Kaggle instance and the software frameworks utilized in this research are summarized in Table 4.1. To ensure full transparency and reproducibility, the complete source code developed for this study is publicly available online[1].

[1] https://www.kaggle.com/code/ikramkellou/ayabensaha-c-mapss-dl-drl

Table 4.1: Hardware specifications and software environment.

Component	**Specification / Framework**
Hardware Specifications (Kaggle Cloud)	
Processor (CPU)	Intel Xeon CPU (4 Cores)
Graphics Card (GPU)	NVIDIA Tesla P100 (16 GB HBM2)
Memory (RAM)	16 GB
Storage	73.1 GB Available Cloud Disk Space
Software Environment	
Operating System	Kaggle Linux Container (Ubuntu 22.04 LTS)
Programming Language	Python 3.12
Deep Learning (Perception)	TensorFlow 2.x / Keras
Deep RL (Decision Making)	Stable Baselines3
RL Environment	Gymnasium (Custom maintenance simulation)
Data Processing	NumPy, Pandas, Scikit learn

The Python ecosystem was chosen as the primary programming language. Based on the specific computational needs of each phase, a hybrid framework approach was adopted: the deep learning perception module (CNN-LSTM and MC Dropout) was constructed using **TensorFlow/Keras**, while the SAC reinforcement learning agent was implemented using **Stable Baselines3**. Also, the custom maintenance environment was built using the **Gymnasium** API, allowing seamless integration between the probabilistic state generator and the RL agent.

4.3 Evaluation Metrics

To evaluate the effectiveness of the proposed predictive maintenance framework, several performance metrics were used for both the CNN-LSTM prediction model and the SAC adaptive maintenance agent.

4.3.1 Predictive Performance Metrics

The accuracy of the proposed CNN-LSTM model for Remaining Useful Life (RUL) prediction is evaluated using several widely adopted regression metrics, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient of Determination (R^2), and the official NASA asymmetric scoring function (S-score). These metrics are used to assess the prediction accuracy, robustness, and reliability of the proposed predictive maintenance framework under different

degradation conditions.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \quad MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{4.1}$$

The coefficient of determination (R^2 score) is also used to evaluate how well the predicted values fit the real degradation behavior of the engines.

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

In addition, the official NASA asymmetric scoring function (S-Score) is adopted. Unlike traditional regression metrics, this score penalizes late predictions more severely than early predictions because delayed maintenance decisions may lead to catastrophic engine failures.

$$S = \sum_{i=1}^{N} s_i, \quad \text{where} \quad s_i = \begin{cases} e^{-\frac{(\hat{y}_i - y_i)}{13}} - 1, & \text{if } (\hat{y}_i - y_i) < 0 \\ e^{\frac{(\hat{y}_i - y_i)}{10}} - 1, & \text{if } (\hat{y}_i - y_i) \geq 0 \end{cases} \tag{4.3}$$

Finally, an additional prediction accuracy metric is computed to provide an interpretable percentage based evaluation of the model performance:

$$Accuracy = \left(1 - \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{125}\right) \times 100 \tag{4.4}$$

4.3.2 Reinforcement Learning Decision Metrics

The operational efficiency of the prescriptive maintenance policy governed by the SAC agent is evaluated using the following metrics:

- **Wasted Life:** Represents the number of healthy operating cycles lost due to premature maintenance actions:

$$\text{Wasted Life} = \max(0, RUL_{true} - k) \tag{4.5}$$

- **Unscheduled Maintenance Prevented:** Percentage of test engines where failure was avoided.
- **Proximity Precision Interval ($Precision_{<20}$):** Ratio of decisions made within a safe 20 cycle window before actual failure.

4.4 Evaluation of the CNN-LSTM Predictive Model

The proposed hybrid CNN-LSTM network was systematically trained across the four benchmark subsets of the NASA C-MAPSS dataset. The performance

results recorded on the independent test trajectories are compiled in Table 4.2.

Table 4.2: Deterministic RUL prediction results across NASA C-MAPSS subsets.

Dataset	RMSE	MAE	R^2	*S*-Score	Accuracy (%)
FD001	12.76	9.54	0.90	309.02	92.37
FD002	12.30	8.66	0.92	716.58	93.07
FD003	12.77	8.52	0.89	322.09	93.18
FD004	13.97	8.99	0.89	1193.81	92.81

4.4.1 Visual Convergence and Profile Tracking

To inspect the internal optimization behavior and spatial tracking capabilities, Figure 4.1 displays the training vs. validation loss.

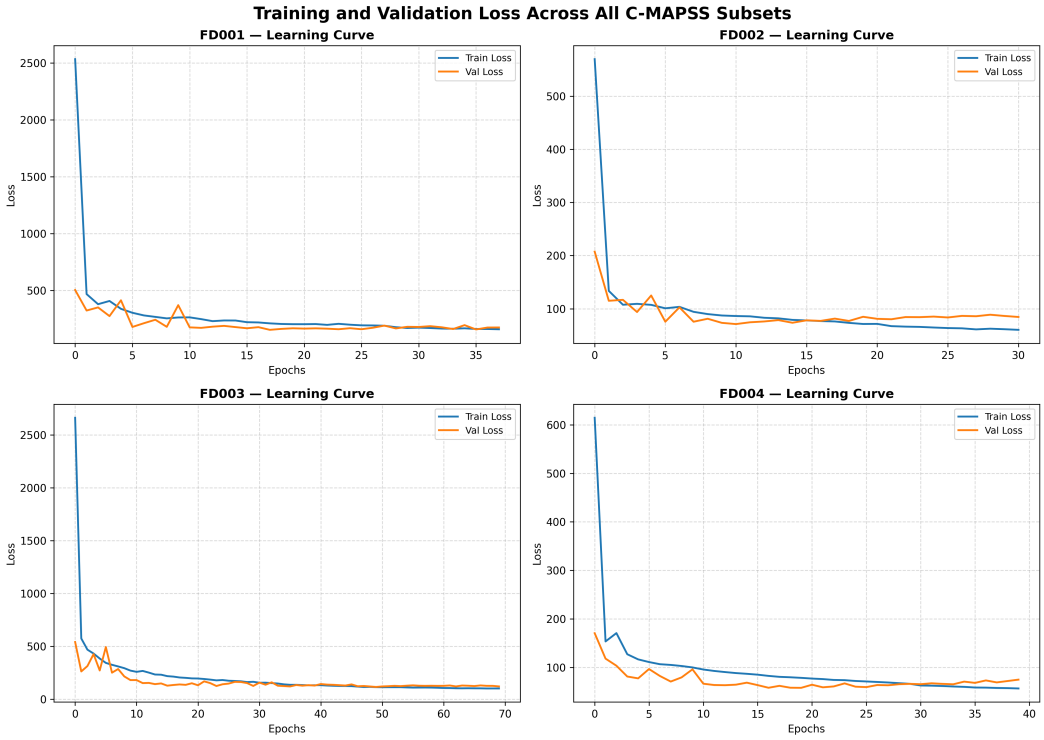

Figure 4.1: Training vs. validation loss convergence trajectories.

The empirical evaluation curves demonstrate excellent convergence across all regimes, with the Huber loss successfully stabilizing gradients in the multi condition datasets (FD002, FD004) and preventing overfitting despite severe sensor noise.

To visually match these metrics with the actual physical test runs, the ground truth test trajectories are contrasted directly against the continuous predictions of our perception module in Figure 4.2.

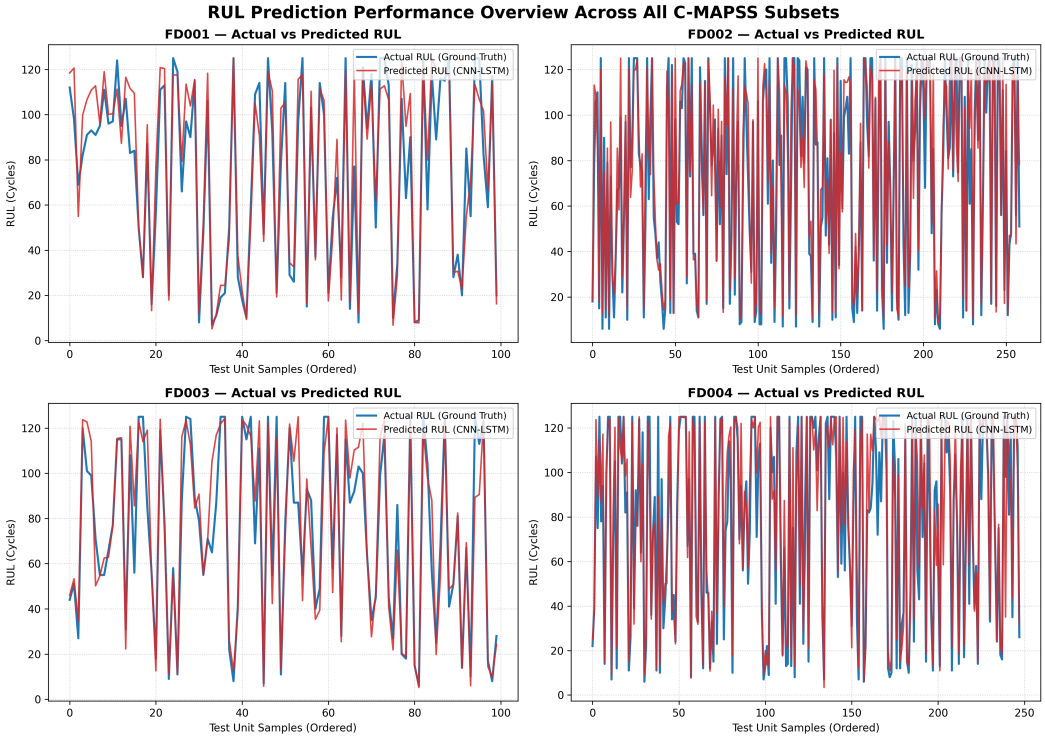

Figure 4.2: Actual RUL vs Predicted RUL.

As exhibited in the tracking comparison, the model's output trajectory (solid red line) matches the actual target slopes (solid blue line) across all test samples. The model successfully adapts to the piecewise linear degradation phases, accurately identifying the start of wear propagation even in complex multi fault modes. This high deterministic tracking precision forms a reliable basis for transitioning to the probabilistic risk phase.

4.4.2 Comparison with Recent State of the Art Approaches

To further validate the robustness and efficiency of the proposed perception module, the predictive performance of our CNN-LSTM architecture was benchmarked against recent state of the art data driven methods, including advanced attention based mechanisms. The comparison, detailed in Table 4.3, focuses on the widely adopted RMSE and S-Score metrics across all four C-MAPSS subsets.

Table 4.3: Performance comparison with recent State of the Art methods across all C-MAPSS datasets.

Model (Reference)	**Year**	**FD001**		**FD002**		**FD003**		**FD004**	
		RMSE	*S*	**RMSE**	*S*	**RMSE**	*S*	**RMSE**	*S*
LSTM (Zheng et al. [21])	2017	16.14	338	24.49	4450	16.18	852	28.17	5550
B-LSTM (Wang et al. [55])	2022	12.45	279	15.36	4250	13.37	356	16.24	5220
Multi-channel CNN (Lee et al. [15])	2023	11.81	-	14.49	-	11.69	-	17.73	-
Transformer (Fan et al. [40])	2024	**10.61**	**169**	13.47	784	**10.71**	**202**	15.87	1449
CNN-BiLSTM-Attention (Bing et al. [39])	2025	12.49	267.08	20.32	1575.88	12.34	225.53	26.54	3905.33
Proposed Method (Ours)	**2026**	12.76	309.02	**12.30**	**716.58**	12.77	322.09	**13.97**	**1193.81**

As shown in Table 4.3, the proposed CNN-LSTM architecture achieves highly competitive results across all subsets. While deeper models may slightly excel on simpler datasets (FD001, FD003), our hybrid approach demonstrates exceptional robustness on the complex, multi condition subsets (FD002, FD004). This balance of high predictive accuracy and computational efficiency confirms that the CNN-LSTM is an ideal, lightweight perception layer for the subsequent Reinforcement Learning phase.

4.5 Adaptive Maintenance Policy Evaluation

While the previous section evaluated the CNN-LSTM perception module using the standard C-MAPSS training and testing subsets, training and evaluating the RL agent requires a different methodological approach. The standard C-MAPSS test trajectories are censored (cut off prior to actual failure). This prevents the SAC agent from experiencing complete run to failure episodes, which are essential for learning terminal states and failure penalties.

Additionally, training both the perception model and the decision making agent on the exact same engines would introduce severe data leakage, as the SAC agent would exploit the neural network's training bias rather than learning to generalize under true uncertainty.

To overcome these limitations and completely eliminate data leakage between the perception and action phases, the adaptive maintenance experiments utilized the full run to failure trajectories of the FD002 subset. The 260 engines were strictly partitioned into three isolated sets. The exact distribution of these engines across the system's layers is detailed in Table 4.4.

Table 4.4: FD002 Dataset Strict Partitioning Strategy.

Framework Phase	**Engines**	**Operational Role**
Perception Training	130	CNN-LSTM baseline training and MC Dropout probability calibration.
Decision Training	100	Soft Actor-Critic (SAC) reinforcement learning policy formulation.
Policy Evaluation	30	Unseen test set for final operational metrics (1,000 episodes).

Importantly, rather than using deterministic point estimates from the CNN-LSTM, MC Dropout was deployed to run 30 stochastic forward passes. This easily transforms the point predictions into a robust, 30 element probabilistic state vector $P(RUL \leq k)$. This vector serves as the direct uncertainty aware observation space for the adaptive control layer, allowing the SAC agent to evaluate complete progressive degradation paths instead of arbitrary fixed thresholds.

4.5.1 SAC Agent Training Convergence

The agent's capacity to optimize its maintenance scheduling policy and maximize cumulative rewards within the stochastic environment is visually demonstrated through its training learning curve in Figure 4.3.

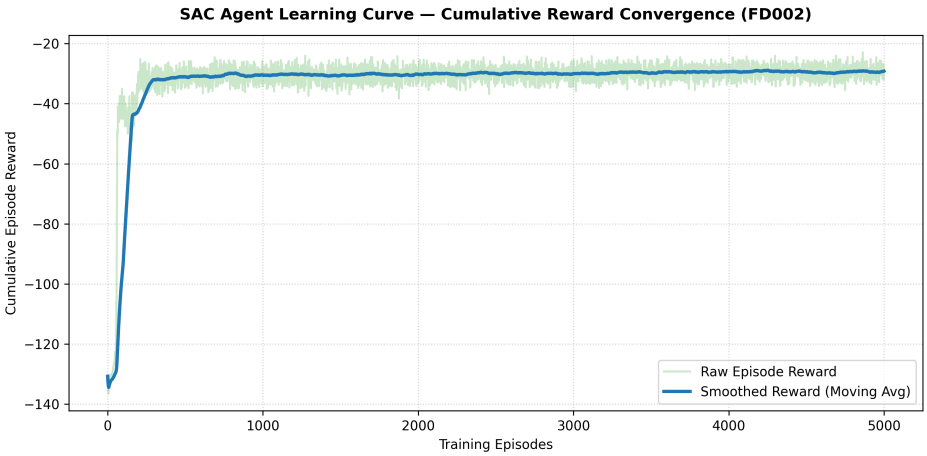

Figure 4.3: Learning curve of the SAC agent.

The empirical learning trajectory illustrates stable asymptotic convergence over the 500,000 continuous timesteps. In the early training episodes, the cumulative reward drops significantly as the agent aggressively explores the environment and incurs severe penalties due to frequent unscheduled component failures ($c_{uns} = 3.0$) and improper alignment with maintenance slots ($c_{slot} = 0.5$). However, by applying the maximum entropy formulation and the experience replay buffer, the SAC policy successfully filters out environment noise, steadily increasing its rewards to converge toward a highly secure and cost effective operational strategy.

4.5.2 Operational Testing Results

Following training convergence, the optimized SAC agent was deployed in a deterministic execution mode on the unseen test engines partition. The operational performance metrics compiled over the 1,000 evaluation episodes are summarized in Table 4.5.

Table 4.5: SAC Adaptive Maintenance Evaluation Results on FD002 Test Set.

Operational Metric	**Recorded Value**	**Analytical Significance**
Total Replacement Decisions	1000	Validates simulation stability across high diagnostic scales.
Unscheduled Failure Instances	18	Reflects near zero tolerance for in-flight breakdowns.
Unscheduled Failure Rate	1.80%	Demonstrates reliable performance under variable conditions.
Unscheduled Failures Prevented	**98.20%**	Confirms the effectiveness of the risk mitigation layer.
Average Wasted Useful Life	**25.75 cycles**	Balances early intervention with component utilization.
Proactive Decision Precision	42.26%	Triggers optimal maintenance tasks close to the real failure boundary.

The empirical results demonstrate that our SAC agent achieves an efficient, risk-aware proactive strategy. Driven by the severe penalties associated with unexpected failures, our framework successfully prevents 98.20% of breakdowns under volatile, multi-operating profiles. Concurrently, the policy maintains an operational balance, minimizing premature interventions to record 25.75 cycles of average wasted useful life. This behavior is further verified by the precision metric, indicating that 42.26% of all replacements were executed within an economically viable 20 cycle window prior to actual failure.To contextualize these findings, Table 4.6 benchmarks our performance against the Deep Reinforcement Learning framework proposed by (Lee and Mitici.,2023) [15]. While their study combines a multi-channel Convolutional Neural Network (CNN) based prognostics model with a Soft Actor-Critic (SAC) agent for maintenance optimization, our framework integrates a hybrid CNN-LSTM architecture for RUL prediction before the decision-making stage. The key methodological difference lies in the inclusion of LSTM layers, which are designed to capture long-term temporal dependencies in engine degradation trajectories, and provide RUL estimates that subsequently guide the SAC maintenance policy, supporting maintenance decision-making under varying operating conditions.

Table 4.6: Performance comparison between the baseline study and our proposed framework.

Metric	**(Lee and Mitici.,2023) [15]**	**Our Framework**
Unscheduled Failures Prevented	95.6%	**98.20%**
Average Wasted Useful Life	**12.81 cycles**	25.75 cycles

As shown in Table 4.6, our methodology demonstrates a distinct prioritization of risk mitigation. While the baseline approach minimizes the average wasted useful life, our policy reduces unexpected breakdowns to a stricter margin (98.20% prevented). This indicates that our integrated architecture adopts a

cautious scheduling policy, which is essential for safety-critical industrial environments where the penalty for in-service failures heavily outweighs the logistical cost of early component replacement.

4.6 Discussion

The experimental results validate the framework's robustness, particularly in multi regime environments. As shown in Table 4.3, while Transformers slightly excel on simpler datasets, our CNN-LSTM model significantly outperforms them on the complex FD002 and FD004 subsets. This superior generalization is directly driven by the K-Means clustering step, which isolates the six operating conditions to eliminate data distribution shifts before sequence modeling, providing a highly accurate yet lightweight perception layer.

Additionally, closing the loop with MC Dropout and the SAC agent highlights the practical value of uncertainty aware control. Guided by an asymmetric, cost sensitive reward function, the SAC agent successfully reduced unexpected failures. It is important to note that this strict safety prioritization inherently resulted in an average Wasted Useful Life of 25.75 cycles per engine. While we acknowledge that this margin is relatively high, it represents the current limit of our proposed architecture's attempts to balance safety and resource utilization. In high stakes aviation environments, mitigating the risk of catastrophic in flight failures remains the utmost priority; however, further refining the policy to minimize this wasted lifespan is a primary objective for future improvements.

4.7 Conclusion

This chapter presented the experimental evaluation of the proposed predictive maintenance framework using the NASA C-MAPSS dataset. The obtained results demonstrated the effectiveness of the proposed CNN-LSTM model for RUL prediction under varying operating conditions.

The integration of MC Dropout further enhanced the framework by providing probabilistic degradation estimations and uncertainty-aware predictions. These probabilistic outputs were successfully exploited by the SAC reinforcement learning agent to perform adaptive maintenance decision-making.

The experimental results showed that the proposed adaptive maintenance policy was highly capable of mitigating the risk of unexpected engine failures. By establishing a strict safety-first approach, this research provides a robust baseline, paving the way for future algorithmic refinements to further optimize component lifespan.

Overall, the combination of Deep Learning prediction and Deep Reinforcement Learning control proved to be a promising, robust solution for intelligent predictive maintenance in Industrial IoT environments.

Conclusion and Perspectives

This thesis presented a comprehensive, intelligent predictive maintenance framework tailored for Industrial Internet of Things (IIoT) environments, seamlessly integrating Deep Learning (DL) with Deep Reinforcement Learning (DRL). The primary objective of this research was to transcend traditional, static threshold-based maintenance by developing an adaptive, risk-aware policy capable of optimizing component utility while strictly preventing unexpected catastrophic failures.

To achieve this, the proposed architecture was divided into two fundamental phases: perception and action. In the perception phase, a robust hybrid CNN-LSTM model was designed to capture complex spatial-temporal degradation patterns from raw sensor data. To bridge the gap between deterministic prediction and risk-sensitive decision-making, Monte Carlo (MC) Dropout was effectively integrated. This stochastic approach successfully transformed point estimates into continuous probabilistic state vectors, directly quantifying the epistemic prediction uncertainty as the engine degraded over time.

In the action phase, a Soft Actor-Critic (SAC) reinforcement learning agent utilized these uncertainty-aware observation states to optimize maintenance scheduling. The full closed-loop control policy was rigorously evaluated on the highly complex NASA C-MAPSS FD002 dataset under strict data partitioning rules to prevent leakage. Driven by an asymmetric cost formulation, the SAC agent learned a highly proactive policy, successfully preventing 98.20% of unscheduled breakdowns. While this strict safety prioritization inherently led to a relatively high average of 25.75 cycles of wasted residual life per engine, it successfully fulfilled the critical operational mandate of eliminating catastrophic in-flight failures.

While the proposed framework achieved its primary goals of risk mitigation and improved operational reliability, certain limitations provide clear avenues for future enhancements. Given the computational constraints of real-time IIoT networks, future work will focus on deploying this lightweight CNN-LSTM and SAC architecture onto Edge Computing devices (IoT gateways) to minimize inference latency. Additionally, expanding the single-agent framework into a Multi-Agent Reinforcement Learning (MARL) environment would allow for fleet-wide maintenance optimization, where multiple agents coordinate shared maintenance slots and limited resources across an entire fleet of aircraft. Finally, exploring dynamic reward shaping—where the penalty parameters dynamically adjust based on real-time economic conditions rather than fixed scalars—could further refine the agent's adaptability in volatile industrial sectors.

Bibliography

[1] Klaus Schwab. *The Fourth Industrial Revolution*. World Economic Forum, 2016.

[2] Ida Hector and Rukmani Panjanathan. Predictive maintenance in industry 4.0: A survey of planning models and machine learning techniques. *PeerJ Computer Science*, 10:e2016, 2024.

[3] Praveen Kumar Reddy Maddikunta, Quoc-Viet Pham, Natarajan Deepa, Kapal Dev, Thippa Reddy Gadekallu, Rukhsana Ruby, Madhusanka Liyanage, et al. Industry 5.0: A survey on enabling technologies and potential applications. *Journal of industrial information integration*, 26:100257, 2022.

[4] Soroush Korivand, Gustavo Galvani, Arash Ajoudani, Jiaqi Gong, and Nader Jalili. Optimizing human–robot teaming performance through q-learning-based task load adjustment and physiological data analysis. *Sensors*, 24(9): 2817, 2024.

[5] Fernando Salvetti and Barbara Bertagni. Leadership 5.0: an agile mindset for a digital future. *International Journal of Advanced Corporate Learning*, 13(2):57, 2020.

[6] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23, 2015.

[7] Hua Wang, Wenjing Zhang, Dong Yang, and Yansong Xiang. Deep-learning-enabled predictive maintenance in industrial internet of things: methods, applications, and challenges. *IEEE Systems Journal*, 16(3):4351–4365, 2022.

[8] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4): 2347–2376, 2015.

[9] Stefan Profanter, Ayhun Tekat, Kirill Dorofeev, Markus Rickert, and Alois Knoll. Opc ua versus ros, dds, and mqtt: performance evaluation of industry 4.0 protocols. In *2019 IEEE International Conference on Industrial Technology (ICIT)*, pages 955–962. IEEE, 2019.

[10] Mohsen Attaran and Bilge Gokhan Celik. Digital twin: Benefits, use cases, challenges, and opportunities. *Decision Analytics Journal*, 6:100165, 2023.

[11] Gianpaolo Di Bona, Vittorio Cesarotti, Gabriella Arcese, and Tommaso Gallo. Implementation of industry 4.0 technology: New opportunities and challenges for maintenance strategy. *Procedia Computer Science*, 180:424–429, 2021.

[12] Olcay Özge Ersöz, Ali Fırat İnal, Adnan Aktepe, Ahmet Kürşad Türker, and Süleyman Ersöz. A systematic literature review of the predictive maintenance from transportation systems aspect. *Sustainability*, 14(21):14536, 2022.

[13] R Keith Mobley. *An introduction to predictive maintenance*. Elsevier, 2002.

[14] Alessandro Giacotto, Henrique Costa Marques, and Alberto Martinetti. Prescriptive maintenance: a comprehensive review of current research and future directions. *Journal of quality in maintenance engineering*, 31(1):129–173, 2025.

[15] Juseong Lee and Mihaela Mitici. Deep reinforcement learning for predictive aircraft maintenance using probabilistic remaining-useful-life prognostics. *Reliability Engineering & System Safety*, 230:108908, 2023.

[16] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X Gao. Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115:213–237, 2019.

[17] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[18] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications*, pages 214–228. Springer, 2016.

[19] Nihad Brahimi, Huaping Zhang, Lin Dai, and Jianzi Zhang. Modelling on car-sharing serial prediction based on machine learning and deep learning. *Complexity*, 2022(1):8843000, 2022.

[20] S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[21] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *2017 IEEE international conference on prognostics and health management (ICPHM)*, pages 88–95. IEEE, 2017.

[22] Fan Yang, Guangqiu Huang, and Yanan Li. A new combination model for air pollutant concentration prediction: A case study of xi'an, china. *Sustainability*, 15(12):9713, 2023.

[23] Ali Al-Dulaimi, Sima Zabihi, Amir Asif, and Arash Mohammadi. A multimodal and hybrid deep neural network model for remaining useful life estimation. *Computers in Industry*, 108:186–196, 2019.

[24] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2nd edition, 2018.

[25] Yuxin Wen, Md Fashiar Rahman, Honglun Xu, and Tzu-Liang Bill Tseng. Recent advances and trends of predictive maintenance from data-driven machine prognostics perspective. *Measurement*, 187:110276, 2022.

[26] Paul Paris and Fazil Erdogan. A critical analysis of crack propagation laws. *Journal of Basic engineering*, 85(4):528–533, 1963.

[27] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical systems and signal processing*, 104:799–834, 2018.

[28] Racha Khelif, Brigitte Chebel-Morello, Simon Malinowski, Emna Laajili, Farhat Fnaiech, and Noureddine Zerhouni. Direct remaining useful life estimation based on support vector regression. *IEEE Transactions on industrial electronics*, 64(3):2276–2285, 2016.

[29] Celestino Ordóñez, Fernando Sánchez-Lasheras, Javier Roca, and Francisco de Cos Juez. A hybrid arima–svm model for the study of the remaining useful life of aircraft engines. *Journal of Computational and Applied Mathematics*, 346:184–191, 01 2019. doi: 10.1016/j.cam.2018.07.008.

[30] Chong Zhang, Pin Lim, A Kai Qin, and Kay Chen Tan. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE transactions on neural networks and learning systems*, 28(10):2306–2318, 2016.

[31] Lei Ren, Yaqiang Sun, Jin Cui, and Lin Zhang. Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. *Journal of Manufacturing Systems*, 48:71–77, 2018.

[32] Jingyao Wu, Kun Hu, Yiwei Cheng, Haiping Zhu, Xinyu Shao, and Yuhou Wang. Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network. *ISA Transactions*, 97:241–250, 2020. doi: 10.1016/j.isatra.2019.07.004.

[33] Jianghong Zhou, Yi Qin, Dingliang Chen, Fuqiang Liu, and Quan Qian. Remaining useful life prediction of bearings by a new reinforced memory gru network. *Advanced Engineering Informatics*, 53:101682, 2022.

[34] Rui Zhao, Dongzhe Wang, Ruqiang Yan, Kezhi Mao, Fei Shen, and Jinjiang Wang. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Transactions on Industrial Electronics*, 65(2):1539–1548, 2017.

[35] Yongmeng Zhu, Jiechang Wu, Xing Liu, Jun Wu, Kai Chai, Gang Hao, and Shuyong Liu. Hybrid scheme through read-first-lstm encoder-decoder and broad learning system for bearings degradation monitoring and remaining useful life estimation. *Advanced Engineering Informatics*, 56:102014, 2023.

[36] Shuguang Sun, Shuo Wei, Jingqin Wang, Xu Shao, Jinfa Liu, and Hui Gao. Remaining useful life prediction for circuit breaker based on opening-related vibration signal and sa-cnn-gru. *IEEE Sensors Journal*, 22(23):23009–23022, 2022.

[37] Chengying Zhao, Xianzhen Huang, Yuxiong Li, and Muhammad Yousaf Iqbal. A double-channel hybrid deep neural network based on cnn and bilstm for remaining useful life prediction. *Sensors*, 20(24):7109, 2020.

[38] Adryan Fitra Azyus, Sastra Kusuma Wijaya, and Budhy Kurniawan. Remaining useful life prediction of turbofan engines using cnn-gru: A comparative study on c-mapss dataset. *Journal of Mechanical, Civil and Industrial Engineering*, 6(1):19–27, 2025.

[39] Bing Yu, Haonan Guo, and Jianqiang Shi. Remaining useful life prediction based on hybrid cnn-bilstm model with dual attention mechanism. *International Journal of Electrical Power & Energy Systems*, 172:111152, 2025.

[40] Zhengyang Fan, Wanru Li, and Kuo-Chu Chang. A two-stage attention-based hierarchical transformer for turbofan engine remaining useful life prediction. *Sensors*, 24(3):824, 2024.

[41] Zheming Tong, Jiazhi Miao, Shuiguang Tong, and Yingying Lu. Early prediction of remaining useful life for lithium-ion batteries based on a hybrid machine learning method. *Journal of Cleaner Production*, 317:128265, 2021.

[42] Mingxian Wang, Gang Xiang, Langfu Cui, Qingzhen Zhang, and Juan Chen. Remaining useful life distribution prediction framework for lithium-ion battery fused prior knowledge and monitoring data. *Measurement Science and Technology*, 34(12):125108, 2023.

[43] Dengshan Huang, Rui Bai, Shuai Zhao, Pengfei Wen, Jiawei He, Shengyue Wang, and Shaowei Chen. A hybrid bayesian deep learning model for remaining useful life prognostics and uncertainty quantification. In *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–8. IEEE, 2021.

[44] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[45] Marius Băban, Călin Florin Băban, and Marius Darius Şuteu. Maintenance decision-making support for textile machines: A knowledge-based approach using fuzzy logic and vibration monitoring. *Ieee Access*, 7:83504–83514, 2019.

[46] Yusuf Yare, Ganesh K Venayagamoorthy, and UO Aliyu. Optimal generator maintenance scheduling using a modified discrete pso. *IET generation, transmission & distribution*, 2(6):834–846, 2008.

[47] Jing Huang, Qing Chang, and Jorge Arinez. Deep reinforcement learning based preventive maintenance policy for serial production lines. *Expert Systems with Applications*, 160:113701, 2020.

[48] Anastasis Aglogallos, Alexandros Bousdekis, Stefanos Kontos, and Gregoris Mentzas. Health state prediction with reinforcement learning for predictive maintenance. *Frontiers in Artificial Intelligence*, 8:1720140, 2025.

[49] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[50] Enrico Zio and Francesco Di Maio. Fault diagnosis and failure mode estimation by a data-driven fuzzy similarity approach. *International Journal of Performability Engineering*, 8(1):49, 2012.

[51] Antonio Cammi, Lelio Luzzi, Alessandro A Porta, and Marco E Ricotti. Modelling and control strategy of the italian lbe-xads. *Progress in Nuclear Energy*, 48(6):578–589, 2006.

[52] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[53] PHM Society. 2010 phm society data challenge dataset. Prognostics and Health Management Society, 2010. URL https://phmsociety.org/phm_competition/2010-phm-society-conference-data-challenge/.

[54] Dimple Kapoor, Deepali Gupta, Shivani AgarwaI, Mudita Uppal, Sapna Juneja, and Manoj Kumar Sharma. Starnet: Stacked transfer-aware for robust remaining useful life prediction for c-mapss multi-regime engines. *IEEE Access*, 2026.

[55] Xiaojia Wang, Ting Huang, Keyu Zhu, and Xibin Zhao. Lstm-based broad learning system for remaining useful life prediction. *Mathematics*, 10(12): 2066, 2022.

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليـــم العالـــي والبحـــث العلمـــــي
Ministère de l'Enseignement Supérieur et de La Recherche Scientifique

كلية العلوم و التكنولوجيا
قسم الرياضيات و الإعلام الآلي

Faculté des Sciences et de la Technologie

Département Des Mathématiques & de l'Informatique

جامعـــة غردايـــــة

الرقم .....2026/ ق.ر.ا /ك.ع.ت/ج.غ

شهادة الترخيص بالإيداع

أنا الأستاذ(ة):

Mme. BRAHIM Nacera

بصفتي رئيس (ة) والمسؤول(ة) عن تصحيح مذكرة تخرج ماستر الموسومة بـــــــ

Deep Learning for Real-Time Predictive Maintenance and Adaptive Control in Industrial IoT

من انجاز الطالب(ة):

BENSAHA Aya

والطالب(ة):

KELLOU Ikram

الكلية: العلوم والتكنولوجيا.
القسم: الرياضيات والاعلام الالي.
الشعبة: اعلام الي.
التخصص: الأنظمة الذكية لاستخراج المعارف.
تاريخ التقييم/المناقشة: **24/06/2026**
أشهد ان الطالب (الطلبة) قد قام (قاموا) بالتعديلات والتصحيحات المطلوبة من طرف لجنة المناقشة وان المطابقة بين النسخة الورقية والالكترونية استوفت جميع شروطها.

امضاء المسؤول عن التصحيح

مصادقة رئيس القسم

رئيس قسم الرياضيات و الإعلام الآلي
الحاج موسى إلياس

جامعة غرداية
قسم الرياضيات و الاعلام الآلي
كلية العلوم والتكنولوجيا