TÉCNICO LISBOA

Predicting Remaining Useful Life with Machine Learning Algorithms for Predictive Maintenance

Gonçalo António Gonçalves Gago Felício

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisors: Profª Susana Margarida da Silva Vieira
Prof. João Miguel da Costa Sousa

Examination Committee

Chairperson: Prof. Carlos Baptista Cardeira
Supervisor: Profª Susana Margarida da Silva Vieira
Member of the Committee: Prof. Luís Manuel Fernandes Mendonça

December 2021

Acknowledgments

This work is supported by my supervising team, at IDMEC, Instituto Superior Técnico (IST), Universidade de Lisboa (ULisboa), and my supervising team at Axians, Portugal. Their contribution was indispensable for this work, as I found myself tackling an unfamiliar and challenging field, struggling to find my way through this forest of knowledge.

I would like to personally thank my advisors, Professor João Sousa from IST, Daniela Moniz, and Margarida Solas from Axians for all the time, patience and support they gave to this work. Looking back half a year ago, I can hardly believe at all I've learned and achieved in this journey, it wouldn't have been possible without your guidance.

I am grateful for all my friends at IST who accompanied me for the past 6 years, these were the hardest and most rewarding years of my life that I will never forget. A special mention to José Lopes, Pedro Esteves and Diogo Nunes who, through their own efforts and drive, pushed me farther than I could have gone alone. I also thank my friends from high school, who have been by my side the longest. Even though you took different paths, I have found a constant source of joy and support from you.

I am also deeply grateful to my family, my greatest pillar. Even though I don't always express it, I feel great gratitude and pride for all you've taught and gave me. When I excelled, you were there, and when I faltered, you were also there. I find myself not wanting nor needing more thanks to you.

I am grateful to my girlfriend, Mariana Fernandes. Even though you joined me half way through this adventure, you have been my greatest motivation and source of strength. I hope to learn from your dedication, focus and perseverance so that we may achieve ever higher goals. Thank you for your unwavering love and trust, even when the whole world seemed to be against us.

Last and most important, I thank my mother for everything, now and forever.

Lisboa, December 15, 2021

Gonçalo Felício

Abstract

Asset failures are hard to foresee and incur significant losses in the manufacturing industry. The present work explores the application of Machine Learning (ML) techniques to predict the Remaining-Useful-Life (RUL) of assets from monitoring data, in the context of Predictive Maintenance (PdM). A successful PdM strategy can not only reduce breakdowns and downtime of assets, but also maximize production, improve product quality, and safety of workers. The current environment of Industry 4.0 (I4.0) has provided the data and tools necessary to enable ML techniques in accurately predicting RUL. We propose three benchmark models using the Random Forest (RF), Extreme Gradient Boosted Trees (XGBoost), and Long-Short-Term-Memory (LSTM), and a Bidirectional LSTM (BiLSTM) model, with sliding time window, to further optimize through parameter and features selection.

The proposed models are applied to the N-CMAPSS dataset provided by NASA and the results obtained show the effectiveness of the models. The best performance was given by the BiLSTM model, followed by the LSTM, XGBoost and RF model. Both the parameters and the features selection increased the performance of the BiLSTM model, in specific, the layer topology parameter was identified as having the greatest contribution. Regarding the feature selection, we identify the Pearson correlation method as having degraded the performance and the importance analysis as having improved the model. Although the BiLSTM model presents excellent performance with the subset of data used to perform the optimization, when transferring the model to the remaining subsets of data the results degrade significantly. Several reasons for this behaviour were presented and discussed to better understand how it can be solved.

Limitations and challenges were identified with the chosen dataset and proposed framework, however the main objectives were achieved and the results obtained show promise for further research and in providing critical information to support decision making for predictive maintenance strategies.

Resumo

Falhas de ativos são difíceis de prever e incorrem perdas significativas na indústria de manufatura. O presente trabalho explora a aplicação de técnicas de Machine Learning (ML) para prever o tempo de vida útil restante de ativos a partir de dados de monitoração, no contexto da manutenção preditiva. Uma estratégia de manutenção preditiva bem-sucedida pode, não apenas reduzir as interrupções e o tempo de inatividade dos ativos, mas também maximizar a produção, melhorar a qualidade de produto e a segurança dos trabalhadores. O ambiente atual de Indústria 4.0 forneceu os dados e ferramentas necessárias para potenciar técnicas de ML na previsão de tempo de vida útil. Neste trabalho propomos três modelos de base, o modelo Random Forest (RF), Extreme Gradient Boosted Trees (XGBoost), e Long-Short-Term-Memory (LSTM), e um modelo Bidirectional LSTM (BiLSTM), com janela de tempo deslizante, para otimizar através da seleção de parâmetros e features.

Os modelos propostos são aplicados ao conjunto de dados N-CMAPSS fornecido pela NASA e os resultados obtidos mostram a eficácia dos modelos. O melhor desempenho foi dado pelo modelo BiLSTM, seguido do modelo LSTM, XGBoost e RF. Tanto a seleção dos parâmetros quanto a seleção de features aumentaram o desempenho do modelo BiLSTM, em específico, o parâmetro da topologia da rede foi identificado como tendo a maior contribuição. Em relação à seleção de features, identificamos o método de correlação de Pearson como tendo degradado o desempenho e a análise de Importância como tendo melhorado o modelo. Embora o modelo BiLSTM apresente excelente desempenho com o subconjunto de dados utilizado para realizar a otimização, ao transferir o modelo para os restantes subconjuntos de dados, os resultados degradam significativamente. Vários motivos para esse comportamento foram apresentados e discutidos para entender melhor como resolver este problema.

Limitações e desafios foram identificados com o conjunto de dados escolhido e a estrutura proposta, no entanto, os principais objetivos foram alcançados e os resultados obtidos mostram-se promissores para pesquisa futura e no fornecimento de informações críticas para apoiar a tomada de decisão em estratégias de manutenção preditiva.

Keywords

Remaining useful life prediction
Predictive maintenance
Machine Learning
Bi-directional long short-term memory
N-CMAPSS

Palavras Chave

Previsão do tempo de vida útil restante
Manutenção preditiva
Machine Learning
Bi-directional long short-term memory
N-CMAPSS

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Networks
AUC	area under the receiver operating characteristic curve
BiLSTM	Bidirectional LSTM
CNC	Computational Numerical Control
CNN	Convolutional Neural Networks
DT	Decision Tree
FP	false positive
FN	false negative
GBoost	Gradient Boosting Trees
IoT	Internet of Things
I4.0	Industry 4.0
KNN	K-Nearest Neighbors
K-Means	K-Means Clustering
LSTM	Long-Short-Term-Memory
LR	Linear Regression
ML	Machine Learning
MTTF	mean time to failure
MLP	Multi-Layer Perceptron
MSE	mean squared error
PM	Preventive Maintenance
POF	probability of failure

PRC	precision recall curve
PdM	Predictive Maintenance
RTF	Run-to-Failure Maintenance
RNN	Recurrent Neural Network
RF	Random Forest
RT	Regression Tree
RMSE	Root Mean Squared Error
ROC	receiver operating characteristic curve
RUL	Remaining-Useful-Life
SVM	Support Vector Machine
SVR	Support Vector Machine - Regression
SVM	Support Vector Machine
TP	true positive
TN	true negative
XGBoost	Extreme Gradient Boosted Trees

1

Introduction

Contents

In the manufacturing industry, an asset failure can be described as the asset performing its intended function defectively, for example, a product is manufactured by the asset with quality standards below those predefined, or the asset completely halting its operation, known as breakdown. The latter of the situations has more severe consequences, but both require maintenance actions to be taken. The consequences of a failure can be low, i.e., an office printing machine breaking down. However, in safety-critical systems, such as the aircraft industry, failures can have extremely high economic impacts and even loss of life. For all intended purposes, if no action is taken, asset failures are inevitable. The more an asset is used, the more it will suffer degradation, leading to failure. The only way to avoid failures is by properly designing, installing, operating, and maintaining an asset [6].

Maintenance can be described as the management of assets and control of costs, however asset monitoring and management are arguably complex tasks. Maintenance costs represent a significant share of all the costs associated with manufacturing and production. Depending on the industry, maintenance costs can account for 15% up to 70% of the total production costs [7]. Additionally, maintenance management often times is an ineffective process, with up to one third of all investment being wasted in improper or unnecessary maintenance actions. The main cause for this is the lack of knowledge regarding the actual health condition of assets and the need for maintenance actions [6]. No two assets are equal and no two assets undergo the exact same conditions, therefore, quantifying the actual need for maintenance is heavily dependant on the specific situation of each asset. Relying on statistical trend data from manufacturers to predict failures will lead to ineffective maintenance management, and waiting for an asset to breakdown is also not an option in safety-critical situations [8].

Figure 1.1: Illustrative figure of Predictive Maintenance

Predictive Maintenance (PdM) is a maintenance method that, through the monitoring of the actual operating condition of an asset and other important indicators is able to predict an impending failure

and allow enough time for maintenance actions to be taken and prevent the future failure. The main advantages of this maintenance method is the increased reliability and availability of assets, improved environmental and worker safety, and reduced costs in parts inventory and maintenance labor [3]. By ensuring that maintenance actions are taken only when they are truly required, the time between repairs is maximized and the occurrence of failures minimized. Successful implementations of PdM strategies have shown the effects of these advantages, namely [8]: reduction in maintenance costs of 25% to 30%; elimination of breakdowns of 15% to 70%; and reduction of downtime of 35% to 45%. The technological advancements from Industry 4.0 (I4.0) and Internet of Things (IoT), specifically sensor technology, as well as developments in Artificial Intelligence (AI), in particular Machine Learning (ML), with great success in prediction algorithms, has converged in new predictive maintenance techniques through the use of ML algorithms [9–18].

1.1 Objectives

The Remaining-Useful-Life (RUL) of an asset is arguably the most important information for a successful maintenance management program. Accurately predicting theRULof assets enables effective maintenance management which in turn affects the quality of products manufactured and reduces maintenance costs. A PdM program's main objective is to detect early signs of failure and allow corrective actions to be taken at the right time, preventing the failure altogether, while reducing unnecessary maintenance. The premise for this is that monitoring the actual operation condition, and other important indicators of an asset gives enough data to accurately make this prediction with enough time to perform the maintenance repairs required [3].

The hypothesis of this approach is that the operation conditions of assets relate to the failures of these assets and that ML algorithms are naturally suited to find these relations and accurately predict the RUL , an insight that is fundamental for a successful PdM strategy.

In detail, the objective of the present thesis is two-fold. First, to develop a framework for predictingRULusing ML algorithms and a suitable dataset of sensor measurements and operation conditions. Second, to determine the success of the chosen ML algorithms by analysing appropriate evaluation metrics and comparing with relevant benchmark results from the literature.

1.2 Contributions

This work contributed to the development and implementation of a framework forRULestimation of an asset using Machine Learning algorithms, in the context of Predictive Maintenance. To the extent of our knowledge, the proposed ML models have not been applied to the chosen dataset previously and so the

work also contributes with the comparison between the proposed models, through analysing the root mean square error, to find the best approach for the analyzed context.

1.3 Document layout

The document is organized as follows:

Chapter 2 briefly discusses maintenance management and its challenges, detailing traditional maintenance strategies such as Preventive Maintenance and compares them to Predictive Maintenance. Traditional PdM techniques that require expert knowledge are then explained, such as vibration analysis, and shortcomings that they present. The role of ML for PdM is discussed, regarding the current environment ofI4.0, and state-of-the art ML algorithms applied for PdM are further studied.

Chapter 3 presents the proposed ML algorithms. The algorithms applied are detailed and the reasoning for its structure presented, along with the evaluation metrics used to compare performance, as well as the methods proposed to optimize the models. These include the parameters selection and the feature analysis.

In Chapter 4, the dataset chosen is extensively studied and analysed. It's source, characteristics, benefits as well as challenges to using it are explored. This chapter also explains the preparation pipeline that produces the baseline dataset used in the experiments, with the processing of data and feature engineering performed.

Chapter 5 presents the results obtained and corresponding discussion of the main objectives. Specifically, the success of ML algorithms in accurately predictingRULand comparison with other ML models.

Lastly, Chapter 6 discusses final considerations on the complete thesis and finalizes with a study of related future work.

2

Predictive Maintenance

Contents

Historically, in the period before World War II, maintenance was seen as an added cost to production, without the corresponding value to the company. The most common form of maintenance was to repair assets after they broke down, considering this the cheapest solution. This maintenance strategy is known as Run-to-Failure Maintenance (RTF) and, today, is considered to be the most costly type of maintenance. Currently, high-level management understand the benefits of a proper maintenance strategy, which were overlooked in the past. Maintenance can, not only prevent asset failures but also [6]:

- maximize production;
- optimize useful life of equipment;
- reduce breakdowns and downtime;
- improve product quality and inventory control.

These advantages are crucial to succeed in the highly competitive environment that manufacturing companies face these days, as well as meet the increasing costumer demands in areas such as environmental and public safety, product quality and product reliability.

2.1 Maintenance strategies

The focus of this thesis is RUL estimation for Predictive Maintenance. There are many definitions of PdM in the industry with small differences between them [3, 8], therefore, for this work, it was considered that PdM is a type of planned maintenance where signs of impending failure are monitored and detected in order to predict when the failure of assets will occur, and carry out the appropriate maintenance work to prevent the failure altogether. Before further explaining PdM it is important to consider traditional maintenance strategies such as RTF and Preventive Maintenance (PM) to understand their benefits and limitations.

2.1.1 Run-to-Failure maintenance

Run-to-Failure maintenance, as the name suggests, is a maintenance strategy when action is only taken after a failure has occurred. This is a reactive maintenance strategy, as opposed to a planned one, where no cost is suffered until an asset fails to operate correctly.

Even though this method sounds reasonable, it is the most expensive method used today, and has many drawbacks associated. The biggest costs come from the need for a large spare parts inventory, the high overtime labor, high machine downtime, and low production availability. Additionally, in safety-critical systems like the aircraft industry mentioned earlier, it is impossible to use RTF approach due to the extremely high costs associated with catastrophic failures. Even if the costs associated with failure

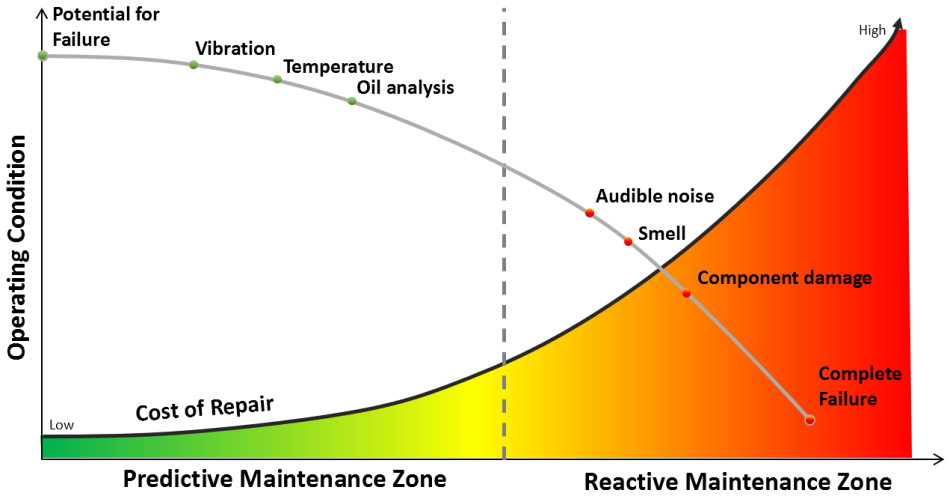


Figure 2.1: Cost of Repair over Time based on [1]

are bearable, analyses of maintenance costs show that repairing an asset in a reactive manner has an average cost three times higher than the same repair performed in a planned manner [6]. Analyses also show that the sooner signs of a future failure are detected and maintenance actions are taken the lower the cost of repairing is [1], as illustrated in Figure 2.1.

2.1.2 Preventive maintenance

Preventive maintenance also has many definitions, but in general, all definitions agree that PM is a method based on time. Specifically, this means that maintenance action is scheduled based on the time of operation of the assets, or another time-based metric. The premise behind this method is that assets behave in known failure modes, i.e., bathtub curve in Figure 2.2, derived from statistical data. With this information the mean time to failure (MTTF) can be accurately calculated and used to schedule maintenance actions.

Analysing Figure 2.2 we can determine that assets have a high probability of failure (POF) in the beginning of its operation. This is mostly due to manufacturing or assembly mistakes, and this phase is called infant mortality phase. Afterwards, the POF decreases to its lowest value and the asset enters its normal life or useful life phase, where the POF is constant. After a period of use, the asset enters the wear-out or end-of-life phase, and the POF increases with time. Under a preventive maintenance program, the time to reach the wear out phase is estimated and the repair or replacement of the asset occurs at this moment.

Where PM can fail is in the assumption that an asset will always follow the same failure mode given its particular classification. In practice, the degradation of the asset is directly dependent on the mode of operation and system variables that affect the normal operating life of the asset. This means that the

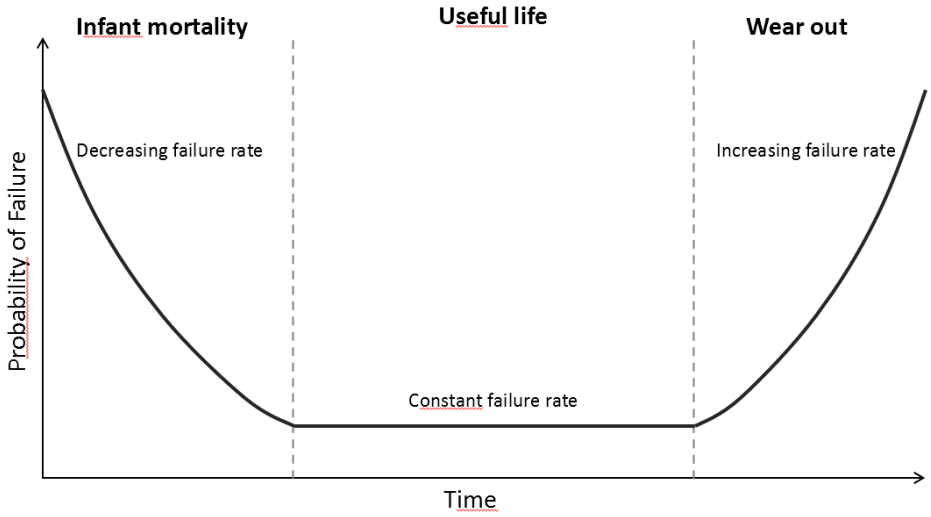


Figure 2.2: Bathtub curve based on [2]

real MTTF is different from estimates based on statistical data and preventive maintenance often leads to maintenance actions being taken too early, or too late. Failure analyses show that the period of useful life cannot be accurately determined from trend data as it is heavily dependent on the specific situation of each asset and failures should be expected to occur at any time, i.e random failure curve [3].

Figure 2.3 shows six common failure curves for industrial equipment and the percentage of time that failure characteristics follow the respective failure curve. The bathtub curve, despite being a very intuitive and widely used method, only describes 4% of the failures, while infant mortality curve describes the majority of the failures with 68%. The bathtub, wear-out, and fatigue curves are failure modes based on time but they only represent 11% of all failures. Figure 2.3 indicates that, in 89% of the situations, using a time-based maintenance strategy like preventive maintenance would lead to ineffective maintenance.

2.1.3 Predictive maintenance

Predictive Maintenance has many definitions in the literature [3, 8], the one adopted for this work is the more general concept thatPdM is a maintenance method where maintenance is scheduled based on the prediction of an impending failure of the asset, derived from the current condition of the asset [6]. PdM uses the most effective technique to determine the current condition of assets and, based on this, schedules maintenance actions when they are needed. Successfully implementing a PdM strategy optimizes the availability of assets and greatly reduces costs of maintenance since asset failures are prevented altogether and the time between repairs is maximized, such that maintenance actions are taken only when necessary. Product quality, reliability, productivity, and profitability are all increased as a direct consequence.

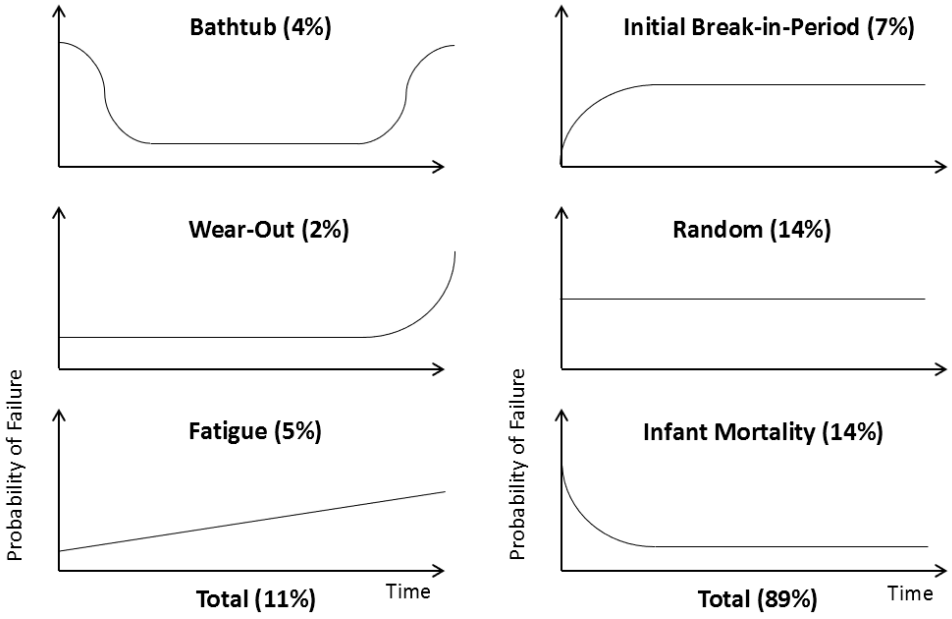


Figure 2.3: Common Failure Curves based on [3]

This type of maintenance was chosen because it shows the highest efficiency and is the most appropriate to use in the significant majority of cases, as suggested in Figure 2.3. PdM is also specially helpful when reliability is most important, situations where breakdowns have severe consequences, such as nuclear power plants, emergency systems, and aircraft industry. A third reason for choosing PdM is the current situation of Industry 4.0. Developments on technologies of Internet of Things, sensor technology, Cloud Services and ML have created space for new and promising solutions in PdM [9, 19]. Both PdM and ML algorithms are heavily dependent on data, in terms of quality and quantity. It is the combination of all these technologies that has provided the data required to achieve better results than traditional PdM techniques [20].

2.2 Traditional predictive maintenance

Despite the recent technological advancements helping PdM increase its popularity in the maintenance industry, PdM has been used since 1940, with less developed techniques such as visual inspection [21]. Visual inspection is a PdM technique where a dedicated maintenance expert monitors the condition of assets directly by sight, sound, touch, or smell to detect signs of failures. This technique is still used today as it is very valuable in certain situations, but the necessity for a maintenance expert of the specific asset being monitored coupled with the need to physically interact with the asset makes this technique of limited use in practice. Other commonly used PdM techniques include vibration monitoring, thermography, and tribology [6].

2.2.1 Vibration monitoring

Vibration monitoring and analysis is a technique where the vibration energy of the system is monitored and analysed to infer the system state. This technique is mostly used in rotating or reciprocating equipment but can be extended to other eletromechanical systems. In simple terms, vibration analysis records the vibration patterns of an optimal operating system and, through monitoring, compares this baseline with the current vibration pattern. Vibration analysis is specifically helpful in detecting problems with imbalance, eccentricity, and misalignment of couplings or bearings however to correctly diagnose faults with this technique still requires expert knowledge [8].

2.2.2 Thermography

Thermography is a technique that monitors the emission of infrared energy, i.e., temperature of assets. Similar to vibration analysis, a healthy temperature baseline is obtained first and then compared to the current temperature of the asset to detect thermal anomalies [8]. This technique also requires expert knowledge from a maintenance expert to interpret, but is capable of being used in a larger variety of systems, not just electrical systems. One of the challenges of this technique is to accurately collect thermal data as assets not only emit thermal energy themselves but also reflect and transmit energy from the environment [6].

2.2.3 Tribology

Tribology refers to PdM techniques related to the lubrication system of machinery. Oil analysis has been widely used, usually alongside vibration analysis as these techniques complement each other very well [6]. Oil analysis may be physical or chemical analyses of the oils and lubricants used in the asset, and can give information not only on the condition of the oil, but also the asset condition. Wear particle analysis on the oil can indicate the level of wear of the internal components of the asset and alert to impending failures [8].

2.3 Machine Learning for predictive maintenance

The PdM techniques described above have been used in PdM strategies extensively with varying degrees of success [6]. Even though PdM has great potential, successfully implementing these strategies has proved challenging. One thing that is common in all of the previous techniques is the requirement of expert knowledge of maintenance and machine dynamics in order to operate and interpret the results. With the technological developments of I4.0, using ML techniques for prognostic and diagnostic of the condition of assets has gained increased attention [20]. One of the reasons is that ML techniques do

not require dedicated expert knowledge of the asset being studied. This means ML algorithms are not restrained to a single domain and can be adapted to suit many different situations. Another reason is the success of ML algorithms in developing tools for forecasting that has been applied in a wide variety of industries and situations, including image processing, robotics and speech recognition.

In the context of PdM, ML algorithms have been adopted more and more both in fault detection and in RUL prediction. Another contributing factor to this trend is the advances in sensor and monitoring technology, alongside big data and cloud systems. The success of ML algorithms are heavily dependent on the quantity and quality of data used to train the prediction models. Fortunately, modern companies can easily collect large amounts of data from their systems that contain important insights, by combining the key technologies mentioned. Therefore, the rise of ML techniques in the PdM circles can be attributed to these factors, additionally the results and success of such implementations has cemented their position. The ML techniques explored further include Artificial Neural Networks (ANN), the Random Forest (RF) algorithm, the Extreme Gradient Boosted Trees (XGBoost) algorithm, the Long-Short-Term-Memory (LSTM) algorithm and the Bidirectional LSTM (BiLSTM)

2.3.1 Artificial Neural Networks

ANN is a field of study that is part of ML. ANN are inspired in the biological neurons and neural networks inside our brains. Similar to our biological networks, ANN are composed of a defined number of neurons, or units, interconnected and organized in layers. An ANN has at least three layers, an input layer, where the data to be studied is inserted, an hidden layer, and an output layer, where the result of the ANN is obtained. An attractive ability of ANN is that these models can learn from data and establish the rules and relations between data, without the need to define these basic rules beforehand. This means that expert knowledge of maintenance and machine dynamics is not required in order to predict RUL .

This ability to learn from examples has led ANN models to be applied in many different industries. Hesser, D.F. and Markert, B [22] proposed an ANN model trained with acceleration data in order to classify the tool wear of older Computational Numerical Control (CNC) milling machines, in an approach to retrofit outdated production plants. This study successfully shows the feasibility in adapting older machines towards I4.0 with its key technologies of computational intelligence and big data analysis. Sampaio, G.S. et al. [23] uses an ANN model to predict motor failure. The authors propose a methodology to treat and build a dataset from a vibration system data that simulates a motor, in order to then train the ANN moxdel and predict future failures. The ANN class chosen is a Multi-Layer Perceptron (MLP) due to its easy implementation and good generalization. The proposed model is compared in terms of the Root Mean Squared Error (RMSE) performance index with other common ML algorithms used in the literature, which include Regression Tree (RT), RF and Support Vector Machine (SVM), outperforming the mentioned ML techniques.

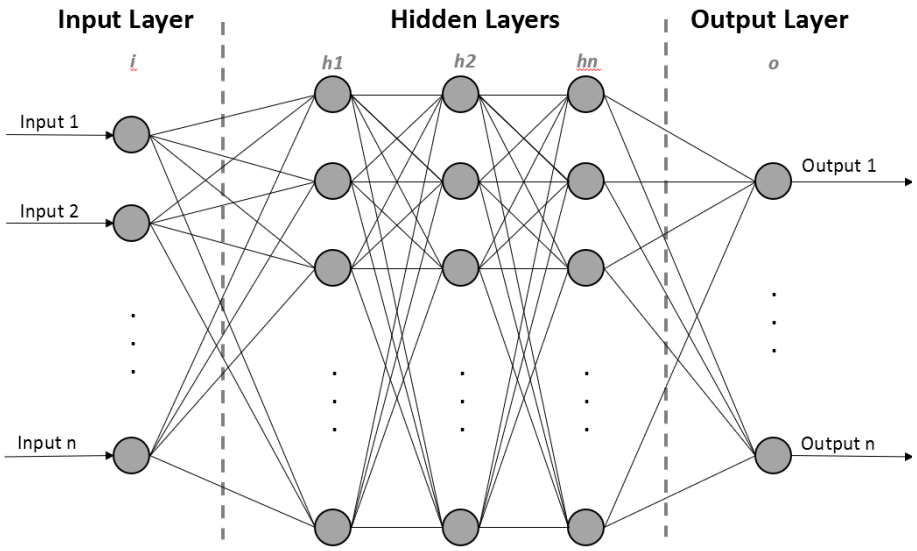


Figure 2.4: Artificial neural network structure

2.3.2 Random Forest

Random Forest is a ML algorithm that combines several randomized Decision Tree (DT), a ML technique with great versatility capable of both classification and regression tasks. The output of the RF model is calculated by averaging the output of the Decision Trees [24]. The premise is that an aggregated answer from many DT is better than a single answer from a single DT. A group of predictors, i.e.,DT, is called an ensemble and therefore this technique is called Ensemble Learning. Despite the simplicity of the RF algorithm it is one of the most powerful ML algorithms available [4]. Benefits of RF include the ability to manage high-dimensional data, a generally fast training speed and good generalization. Additionally, RF is capable of returning a measure of variable importance. RF is also robust against over fitting. Over fitting is when a model is trained and shows great performance on the training data, but when shown new data, performs poorly. When training RF model this can occur if too much data is used in training. Another disadvantage of RF is the need to manually select the features used in training, if the selected features are not suitable, it will negatively affect the results.

Wu, D. et al. [25] use a RF model to predict tool wear of milling machines, comparing the performance with ANN and Support Vector Machine - Regression (SVR) models. The evaluation metrics for comparison are RMSE, R-squared and training time. The features selected are extracted from cutting forces, vibrations and acoustic emissions, using a dataset collected from 315 milling tests. The proposed RF model outperforms the other ML techniques in this study.

Mathew, V. et al. [26] train and compare several ML algorithms to predict RUL of turbofan engine, using the famous CMAPSS dataset from the Prognostics Data Repository of NASA. In this study the authors conclude that RF shows the best results among 10 other ML techniques, including, Linear

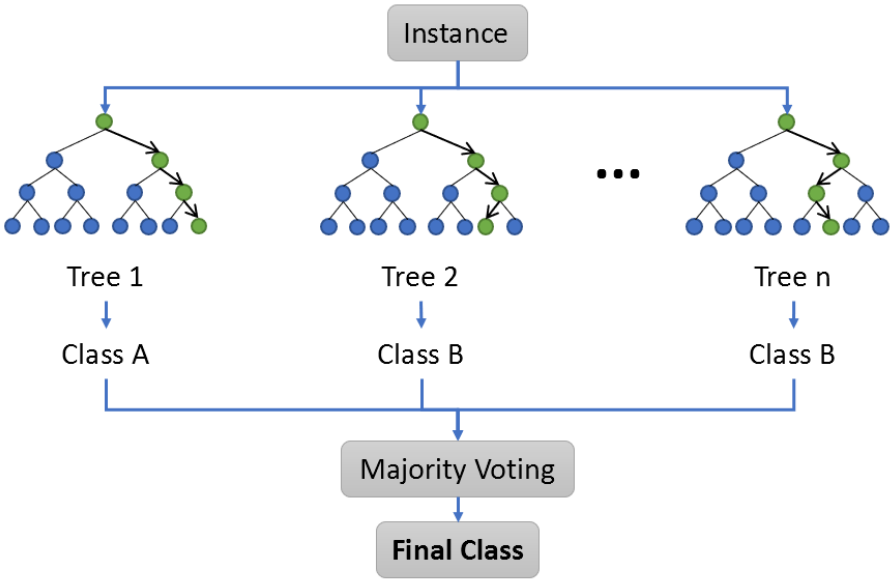


Figure 2.5: Architecture of the Random Forest algorithm

Regression (LR), DT, SVM, K-Nearest Neighbors (KNN), K-Means Clustering (K-Means) and Gradient Boosting Trees (GBoost). The performance index used is once again RMSE. The authors conclude that RF outperforms the other techniques as it better captures the variance of the input variables at the same time and enables a high number of observations to take part in the final prediction, which is in accordance with its framework.

Janssens, O. et al. [27] propose a multisensor system that uses infrared thermal energy data and vibration data. Afterwards features are extracted and used in a RF model for fault detection in rotating machinery. The authors conclude that by fusing the two sensors data they can placate the shortcomings from each data, if used independently, and improve fault detection performance up to 35%.

Quiroz, J.C. et al. use a RF model to diagnose broken rotor bar in line start-permanent magnet synchronous motor. 13 statistical time domain features are extracted from the startup transient current signal an used in the RF model to determine faulty or healthy motors. Additionally, feature importance from the RF model is used to reduce the number of features to just two, and the results are compared between both cases, i.e., using all features or using only two. The results are highly successful with accuracy of 98.8% using all features and 98.4% using the reduced features. The results of these models are also compared with other ML techniques, including DT, Naive Bayes classifier, LR, linear ridge and SVM, with the proposed RF models consistently outperforming. In conclusion RF has been extensively used in the prognostics field with great success due to being easy to apply, robust to over fitting, and ability to identify the most important features. On the other hand, RF requires authors to perform feature engineering to reach these results.

2.3.3 Extreme Gradient Boosted Trees

XGBoost is a scalable tree boosting system that is similar to the RF algorithm but uses a majority voting technique to define the final class. XGBoost has appeared in the prognostics field often alongside RF and LR models and has shown similar results.

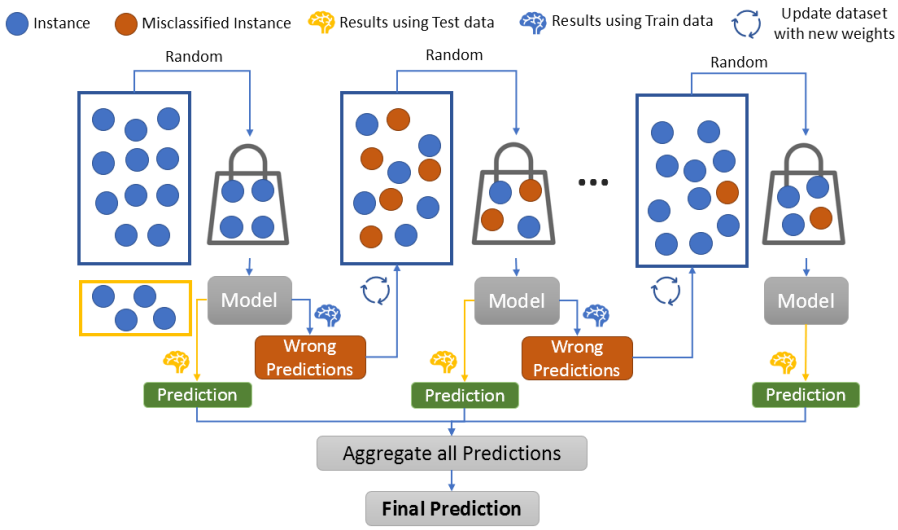


Figure 2.6: Architecture of the XGBoost algorithm

Binding, A. et al. [27] use these three ML techniques in forecasting machine downtime of printing machines based on real-time predictions of future failures. The authors use not only machine-related data but also unstructured data in the form of operator notes, which improved the performance of prediction models. The metrics used to evaluate the models include the area under the receiver operating characteristic curve (AUC), receiver operating characteristic curve (ROC), precision recall curve (PRC) and number of true positive (TP), false positive (FP), true negative (TN), false negative (FN) at different decision thresholds. The results show that RF and XGBoost show similar results and outperform LR.

Calabrese, M. et al. [28] use tree-based classification models, including XGBoost, RF, and GBoost to predict failure in woodworking industrial machines. This is performed by applying temporal feature engineering to event-based IoT data, i.e., log files event-based errors. The proposed framework takes advantage of the distributed Big Data environment that generates historical log data to build the prediction models. Proposed models are optimized by grid searching the hyper-parameters and the evaluation metrics include accuracy,AUC, ROC, PRC for a 30-day RUL target. All three models perform show similar results with GBoost slightly outperforming in overall metrics with 98.9%, 99.6% and 99.1% to accuracy, recall and precision respectively. Despite XGBoost showing very similar results to RF and other similar ML techniques it has seen less presence in the ML prognostics field.

2.3.4 Long-Short-Term-Memory

LSTM algorithm is a neural network composed of LSTM cells. The LSTM cell was proposed in 1997 by Hochreiter, S. and Schmidhuber, J. [29]. The greatest advantage of the LSTM cell over other ML techniques is the ability to identify long-term dependencies in the data which makes this ML technique specifically suitable to study time-series data, long text, audio recordings and more. The architecture of a LSTM cell is shown in Figure 2.7.

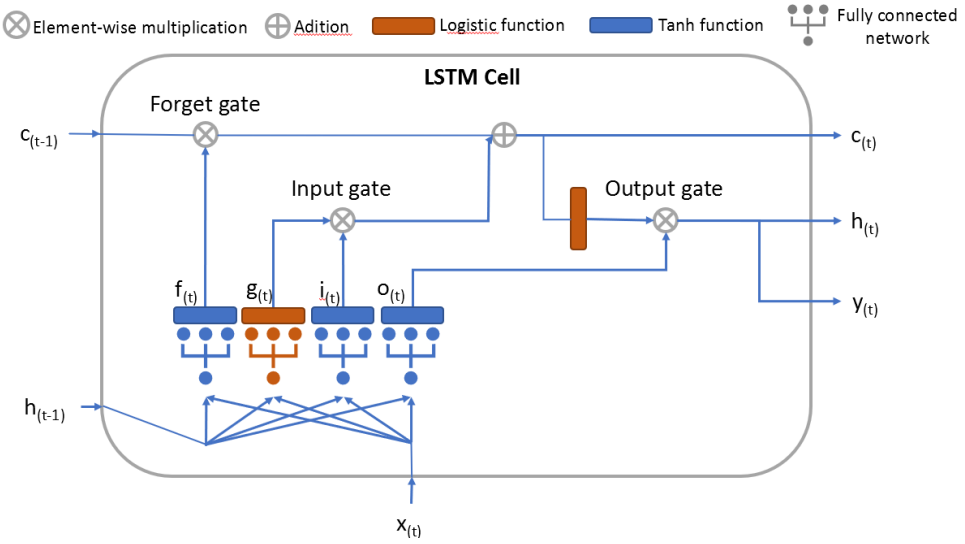


Figure 2.7: Architecture of a basic LSTM cell based on [4]

The ability to recognize long-term patterns comes from the network being able to save in the long-term state information that is relevant, and discard information that is not useful. The long-term state, identified in Figure 2.7 by $c_{(t-1)}$ goes first through a *forget gate*, where the useless information is discarded, then new information is added with the addition operation. The new information to be added is chosen in the *input gate*. The result of the operation is the variable $c_{(t)}$ which is sent out without further operations to become the long-term state of the next step. The long-term state is also copied at this point and goes through the *output gate*, which filters the long-term state to obtain the short-term state $h_{(t)}$ and the output of this time step t, $y_{(t)}$, which are equal.

The input vector $x_{(t)}$ and the short-term state of the previous step $h_{(t-1)}$ relate to the process previously described through 4 different fully connected layers and are the source of new information. These layers control the mentioned gates, *forget, input* and *output gates* through activation functions. The activation function used to control the gates is the sigmoid function that outputs values in the range [0, 1] and the tanh function that outputs values in the range [-1, 1]. With the sigmoid activation function, the gates are closed if the output value is 0, and open if the output value is 1.

The network analyses the input vector and the previous short-term state in the main layer using the

tanh activation function, with output vector $g_{(t)}$. $f_{(t)}$ controls the *forget gate*, selecting which part of the long-term state is discarded. $i_{(t)}$ controls the input gate, selecting which part of $g_{(t)}$ is added to the long-term state of the previous step, generating the long-term state of the current step. Lastly, $o_{(t)}$ controls which part of $c_{(t)}$ is read and becomes the output of the current step $y_{(t)}$.

With this architecture the LSTM cell has the ability to learn which information or memory to retain in the long-term state (with the input gate), how long should it hold on to this memory (with the forget gate) and what memory should be extracted when needed (with the output gate). This gives it the desired ability of capturing long-term patterns in complex time-series data to use for RUL Prediction [4].

The LSTM network computations are shown in Equations (2.1) to (2.6).

$$i_{(t)} = \sigma\left(W_{ix} \cdot x_{(t)} + W_{ih} \cdot h_{(t-1)} + b_i\right) \tag{2.1}$$

$$f_{(t)} = \sigma\left(W_{fx} \cdot x_{(t)} + W_{fh} \cdot h_{(t-1)} + b_f\right) \tag{2.2}$$

$$o_{(t)} = \sigma\left(W_{ox} \cdot x_{(t)} + W_{oh} \cdot h_{(t-1)} + b_o\right) \tag{2.3}$$

$$g_{(t)} = tanh\left(W_{gx} \cdot x_{(t)} + W_{gh} \cdot h_{(t-1)} + b_g\right) \tag{2.4}$$

$$c_{(t)} = f_{(t)} \otimes c_{(t-1)} + i_{(t)} \otimes g_{(t)} \tag{2.5}$$

$$h_{(t)} = o_{(t)} \otimes tanh\left(c_{(t)}\right) \tag{2.6}$$

Where W_{ix}, W_{fx}, W_{ox} and W_{gx} are the weights of each of the four layers connected to the respective gates with respect to the input vector $x_{(t)}$, W_{ih}, W_{fh}, W_{oh} and W_{gh} are the weights of the same four layer with respect to the short-term state $h_{(t-1)}$ and b_i, b_f, b_o and b_g are the bias of each layer. The $\otimes$ symbol represents element-wise multiplication. The output of the layer at step t is the short-term state $h_{(t)}$.

Another attractive feature of the LSTM networks is that they can avoid vanishing/exploding gradients when training the model. The vanishing/exploding gradients problem is when the error gradient that is propagated during training gets smaller and smaller with each layer and the changes to the weighs becomes negligent, making convergence to a good solution impossible. Or, on the opposite case, the gradient grows larger and larger and *explodes*, causing the solution to diverge [4].

LSTM models have been used in RUL prediction given its advantages with time series data. de Oliveira da Costa, P.R. et al. [30] propose a LSTM network combined with global Attention mechanisms in order to learn the relations between RUL and sensor data, and applied the model to the CMAPSS dataset, obtaining competitive results with other state-of-the-art methods. The combination with attention mechanisms allowed the authors to visualize temporal relationships between the target RUL and sensor data, the performance index metrics used are the RMSE and a scoring function *Score* proposed by Saxena, A. et al. [31].

Wu, J. et al. [32] propose a deep LSTM network for RUL prediction, also testing the proposed model on the CMAPSS dataset. The key idea being the deep learning structure having the ability to detect long-term dependencies among the sensors and the target RUL. The model structure and parameters are optimized through grid searching and the results compared with state-of-the-arts reported in the literature, with the same performance metrics mentioned before, RMSE and the *Score* function. With the parameter tuning performed, the authors reach highly competitive results with the deep LSTM model proposed and conclude that deep LSTM algorithm outperforms other recurrent NN models.

Zhang, Y. et al. [33] developed a LSTM model for predicting the RUL of lithium-ion batteries. The proposed model uses the resilient mean squared back-propagation method for optimization and the dropout technique to avoid over fitting. The developed model uses experimental data including temperatures and current rates to train the model and is then compared with SVM and a simple Recurrent Neural Network (RNN) models, outperforming both.

In conclusion LSTM networks have been used both with experimental and simulated data successfully in prediction models due to its ability on analysing large quantities of data, derived from IoT and Bid Data environments, as well as its ability in capturing temporal patterns over the long term. As challenges in using LSTM networks we can identify the need for well behaved data, this is, no missing values. Additionally a large amount of hyper-parameters must be tuned either through techniques like grid search or through literature review and manual testing.

2.3.5 Bidirectional LSTM

BiLSTM is a ML technique composed of two independent and equal LSTM neural networks that are propagated in two opposite directions. One of the LSTM networks has forward propagation while the other has backwards propagation. Both the backward and forward propagation are simultaneous and combine to form to the output unit which is represented in Equation (2.7).

$$y_{(t)} = W_{y\overrightarrow{h_{(t)}}} \cdot \overrightarrow{h_{(t)}} + W_{y\overleftarrow{h_{(t)}}} \overleftarrow{h_{(t)}} + b_y \tag{2.7}$$

Where $\overrightarrow{h_{(t)}}$ and $\overleftarrow{h_{(t)}}$ represent the output of the forward and backward propagation LSTM network, at time step t, respectively. $W_{y\overrightarrow{h_{(t)}}}$ and $W_{y\overleftarrow{h_{(t)}}}$ represent the weights with respect to the forward LSTM layer and the backward LSTM layer, respectively. b_y represent the output layer bias. Finally $y_{(t)}$ is the output of the BiLSTM network at time step t.

The main difference from a basic LSTM neural networks is that the BiLSTM can detect dependencies in the data not only from past information, but also from future information. This is a desirable feature that has been applied in text classification to better capture the preceding and succeeding context that exists in sentences [34]. In the context of time series data, this capability is also desirable to improve the

accuracy of predictions.

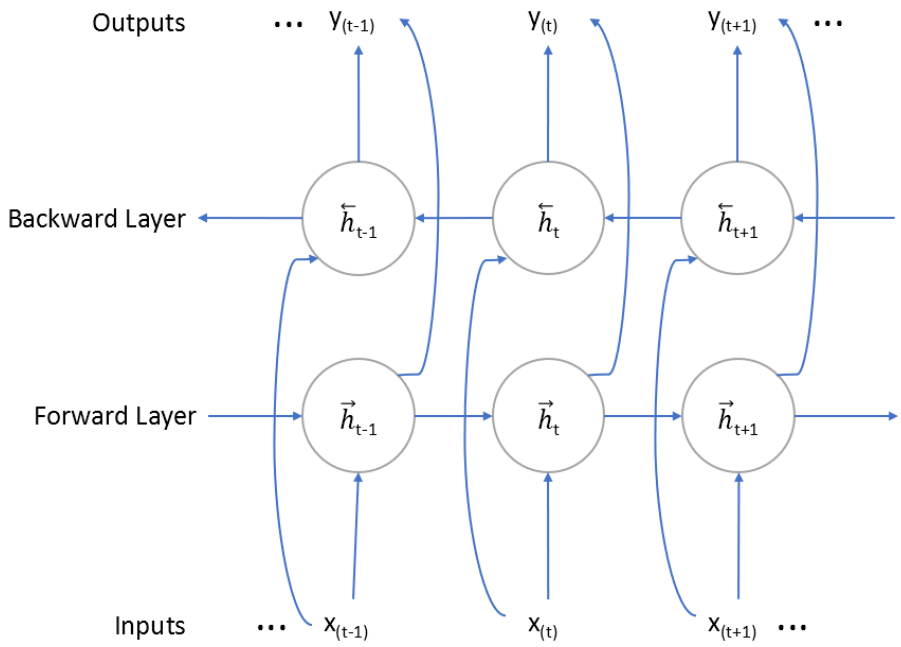


Figure 2.8: Architecture of a BiLSTM algorithm

Zhao, C. et al. [35] proposed a double-channel hybrid deep NN based on Convolutional Neural Networks (CNN) and BiLSTM. The model extracts the relevant features using the CNN and and captures the temporal dependencies of the data with the BiLSTM. The model also uses a sliding time-window for data preprocessing. Testing the model on the CMAPSS dataset the results show a better performance than other state-of-the-art models.

Wang, M. et al. [36] propose a CNN-BiLSTM network with an attention model for RUL prediction and apply the model to experimental data provided by the company iFLYTEK. The data is working data of the core consumable parts of machinery and equipment. The CNN network is used to extract the relevant features and the BiLSTM network is used to capture the forward and backward time features, as was seen in [35]. The attention model is used to further filter the selected features and reduce the impact of noise interference on the RUL prediction. According to the authors the results obtained with the proposed model are satisfactory, proving the effectiveness and efficiency of the proposed model.

The BiLSTM has been used by researchers to predict RUL successfully and is capable of all the advantages of a basic LSTM network while also capturing future information dependencies in the data, improving the accuracy of the models predictions.

2.4 Summary

In short, this chapter starts by describing predictive maintenance and the context surrounding it, explaining other commonly used maintenance strategies, such as PM, as well as traditional PdM techniques, presenting the benefits and the challenges of each. Following this the current situation of ML for PdM is explained. The reasoning of using ML techniques over other techniques is presented, highlighting the relevance of the key technologies of I4.0 that empower ML techniques for PdM. An introduction to relevant ML algorithms as well as use cases for each found in the literature are detailed, showing successful implementations of ML models to predict RUL of assets, with both synthetic, experimental, and real data.

3

Proposed Framework

Contents

In this chapter the proposed ML models are detailed as well as the reasoning of the framework adopted. Next, the evaluation metrics used to measure the performance of models are presented. Afterwards, the process used for parameter optimization of the models hyper parameters and data preparation parameters is presented as well as explanations on the parameters tested and the respective effects on performance. Lastly, the techniques used for analysing and selecting features from the data are explained.

3.1 Proposed Machine Learning models

For this work, we propose a RF, a XGBoost, a LSTM, and a BiLSTM model. The RF, XGBoost and LSTM models will serve to obtain baseline results, from which we can then compare with the results obtained from the BiLSTM model. The RF and the XGBoost models were chosen due to the ease of use of these models as well as their good performance in both RUL prediction and computation time. Both models will be default models, this means no optimization of the models parameters will be performed, this is due to time constraints since properly tuning the parameters of all models is extremely time consuming. Additionally, we believe that, if all the proposed models parameters were optimized, the RF and the XGBoost model would not outperform the LSTM and the BiLSTM model, given the results seen in the literature in the context of predicting RUL [26, 37]. The LSTM neural network is used as it often shows better performance than the RF and the XGBoost models, specially with time series data and regression problems, as is the case of this work. Also, the LSTM model is very similar to the BiLSTM model, which will allow for a better comparison of the results obtained. The LSTM model parameters are not optimized, due to the same reason as mentioned for the RF and the XGBoost models, but, the parameters are chosen based on previous relevant literature [38]. We expect that, through the parameters chosen, the LSTM network, will achieve better performance than both the RF and the XGBoost models.

Finally, we propose a BiLSTM network, due to its ability to capture information from past and future information in the dataset. We expect results to be even better than the results of LSTM network after appropriate parameter tuning. The base model parameters, similar to the case with the LSTM model, are chosen based on [37]. As a limitation to using the BiLSTM model we estimate that training and testing will take a considerably longer time. To counter this limitation the parameter optimization process will be a manual process, this is less time consuming than an automatic method, but is also less precise and does not guarantee that an optimal solution is found. The parameter optimization process will follow the flowchart seen in Figure 3.1 until good RUL predictions are obtained. Additionally, the use of cloud services is employed, through the use of the Kaggle platform[1] for the training and testing of all the models. This platform is chosen given its ease of use regarding ML implementations, has

[1] https://www.kaggle.com/

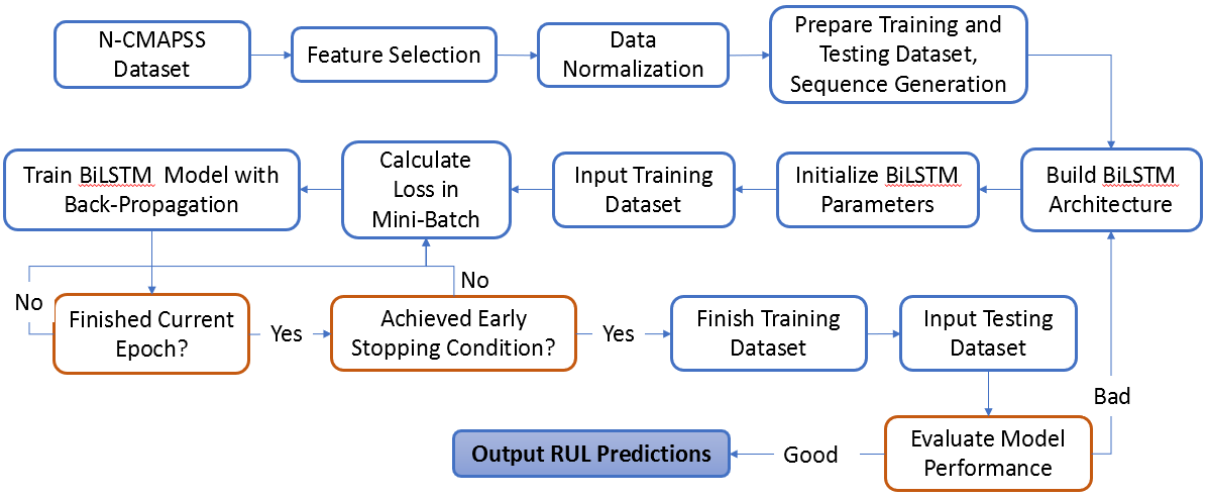


Figure 3.1: Flowchart of the proposed framework for RUL prediction

extensive documentation for support and, most important, provides a graphics processing unit (GPU) for the intensive computations performed in training and testing complex models, such as the BiLSTM model.

One of the limitations of ML prediction algorithms is the generalization and transferability to new, unseen data. To measure this the BiLSTM model will be trained and tested with a subset of the full dataset available. Afterwards the remaining subsets of data will be evaluated with the final BiLSTM model proposed. Given that the remaining subsets of data contain completely new data, it is expected that the BiLSTM model, optimized with first subset, will present better performance for the first subset of data, and worse performance for the remaining subsets.

Lastly, the full dataset is evaluated on the BiLSTM model proposed, and the results compared with previous ones.

3.2 Evaluation metrics

The evaluation metrics used are the ones suggested in [31]. This is the RMSE and NASA's scoring function, denominated $Score$. This metrics have been used extensively by researchers in the context of RUL prediction and make it easier for comparison of results between models. The formula for each performance metric can be seen in Equation (3.1) and Equation (3.2).

$$RMSE = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\Delta^{(j)})^2} \tag{3.1}$$

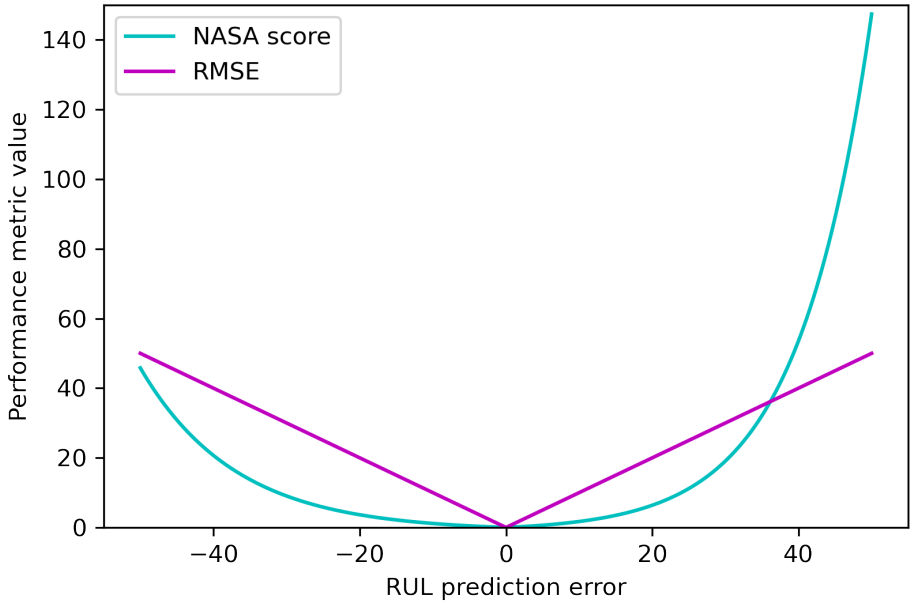


Figure 3.2: Behaviour of the performance metrics used in evaluation

$$s = \sum_{j=1}^{m} exp(\alpha \left|\Delta^{(j)}\right|) \tag{3.2}$$

For both equations m is the total number of data samples, $\Delta^{(j)}$ is the error in the RUL prediction of sample j, that is, the RUL predicted by the model minus the real RUL at sample j, α is $\frac{1}{10}$ if the RUL prediction is over-estimated and $\frac{1}{13}$ if the RUL prediction is under-estimated. This means that, for the Score function, if the ML model predicts a failure earlier than the real value, under-estimation, the penalty is lower than if the model over-estimates, predicting a RUL larger than the real RUL. The reasoning for this imbalance is that, in maintenance management, an under-estimation of RUL would lead to sub-optimal maintenance, but an over-estimation would lead to asset failure, which is a situation much more costly for companies than the alternative. The RMSE is a performance metric that has been commonly used in RUL estimation research that accurately represents the overall error of estimation and is symmetric, unlike the Score function.

The behaviour of both performance metrics are represented in Figure 3.2, which clearly shows the symmetric and asymmetric behaviour of each metric. It is also worth mentioning that the Score metric is a sum of values, while the RMSE is a mean of values. This means that, in order to fairly compare the Score metric between models, the number of samples used in both methods must be the same. This concern does not apply to the RMSE metric as it computes the mean of the error.

3.3 Model parameters selection

Optimizing the parameters of the models is an important step in order to achieve optimal solutions, however, it is also very hard to optimize a model's parameters as the parameters influence each other significantly. What parameters to choose, and how will it affect other parameters in the model is not always obvious and expert experience is required to properly optimizing the model parameters [39]. Because of this, automatic methods are often applied in parameter optimization such as grid search cross fold validation. Given a certain number of parameters and certain values for each parameter the grid search cross fold validation evaluates the performance of all possible combinations of parameters a certain number of times, depending on the number of folds in the cross fold validation. This automatic method reduces the effect of the parameters influencing each other and the effect of variance of the results, as a disadvantage, methods such as this are extremely time consuming when testing several hyper parameters at once, specially with large complex networks and large datasets, as is the case of this work. Due to time constraints the parameter optimization was performed was manually, inspired from the literature [35, 40].

The layer topology, dropout, learning rate, batch size, training/validation split and time-window were all tested, in this order due the influence of each parameter on the model's results. The number of epochs was determined by use of the early stopping technique, an automatic stopping technique to find an optimal number of epochs. During the parameter optimization, the parameter being tested changes while the remaining parameters are fixed. At each iteration the loss function and the RUL prediction graphs are analysed to help guide the choice of parameters. The loss function chosen is the mean squared error (MSE) function, a common function for regression problems. While this does not guarantee an optimal solution, this process is not too time consuming and improves the performance of the model.

3.3.1 Layer topology

The layer topology refers to, in fact, two parameters. The number of layers of the network and the number of neurons of each network. In general, a larger and deeper network can model more complex data. Larger refers to the number of neurons of each layer and deeper refers to the number of layers. As a drawback, these layers take longer to train and test, while not always presenting better performance. Several combinations of values were tested starting with the network proposed in [37]. The network has 4 layers with 64, 32, 16, and 8 neurons, for the first, second, third, and fourth layer respectively. The notation used is, as an example, B(64,32,16,8), in respect to the previous network mentioned. The output layer has the same structure for all networks, being a regression layer with one neuron that outputs the RUL prediction. After a relevant number of combinations is tried and a good performance is obtained the optimization of the layer topology is stopped and the next parameter is tested.

3.3.2 Dropout

Dropout is an powerful technique that helps a model avoid over fitting when training a network and increases generalization and robustness. It causes the network to randomly drop a subset of neurons and their respective connections from the NN. The units are removed at each training iteration and the probability of removing a neuron is the parameter to optimize. A dropout of 0.5 means there is a 50% probability of dropping each neuron in a specific layer. By removing neurons from the network, we force the network to not become overly dependent on any specific neuron [41]. The dropout technique can also be understood as a model ensemble technique, as, in practice, the training is combining the results from different subsets of the original network. Also, by reducing the number of neurons in the network, the time for training is reduced. The dropout is applied uniformly to all the hidden layers and input layer, and the values tested are $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7]$. The base BiLSTM model has a dropout of 0.5.

3.3.3 Learning rate

Learning rate is a parameter that influences the size of the update of the weights after each mini-batch iteration. This influences how fast and how well a network converges to an optimal solution. A larger learning rate will cause the training to converge faster than a smaller learning rate, but might cause the solution to become sub-optimal, trapped in a local minimum. A smaller learning rate can find a optimal solutions more consistently, at the cost of an increase in training time. The base BiLSTM model uses the default learning rate of 0.001.

3.3.4 Batch Size

The batch size, or mini-batch refers to the size of the data samples used in each training iteration. This has a large influence on the computation time and on the performance. A larger batch size will result in less updates performed while training, and consequently, faster training time. However, less updates while training may lead to sub-optimal results. The base BiLSTM model uses a batch size of 512.

3.3.5 Training/validation split

The training/validation split refers to the separation of the training dataset into actual training data and validation data. This has a large influence in over fitting. A model with very little validation data will over fit and provide poor test results, while a model with too big of a validation set will have less data for training which will hurt the performance. A balance between both situations must be found. The values tested are $[0.1, 0.2, 0.3]$, this means separating 10%, 20%, and 30%, respectively, of the training dataset to validation. The separation of the training dataset in training and validation also allows to draw the

loss function graphs of the training and validation data which are extremely useful to understand how the model is behaving to the tested parameters, and give insights on how to choose better parameters. The base BiLSTM model has a dropout training/validation split of 0.1.

3.3.6 Early stopping

The early stopping technique is technique that automatically stops the training of the model when the performance starts degrading. This is a very useful technique used to choose the number of epochs of training. The early stopping technique used monitors the loss function values of the validation data at each epoch and stops the training if the loss function value does not improve over a period of 20 consecutive epochs. Using this technique not only helps in selecting an optimal number of epochs to optimize performance, it also reduces the training time. If the number of epochs is too large, the model will start to over fit the training data.

3.3.7 Time window size

The time window size refers to the number of samples that the model uses for each prediction. This influences how much information is accessed for each prediction. This is a parameter used when preparing the data before training the model. In general, the larger the time window size, the more information is used, and the better are the model predictions. The base BiLSTM model uses a sliding time window with size 50 and step size 1. This means that, during the preprocessing of data, sequences of size 50 are generated, if m is the number of samples in the input data, then $m - 50 + 1$ sequences are generated. Increasing the time window size can improve performance but also increases the training time significantly. The time window sizes tested were 25, 50 and 100.

3.4 Feature selection and analysis

3.4.1 Feature selection

The features selection was performed during the preparation of data and before training the models. The selection of features is very important as it can reduce the computation time of models and even improve the performance. Feature selection is an important step to improve the quality of the data, therefore, before training the models the dataset is analysed to find features that have missing values and features that have only constant values. The features with these characteristics are then removed from the dataset. In the case of the LSTM model and the BiLSTM model, features with missing values make it impossible to run the algorithm. Features with constant values have no useful information and present no relation to the target data, the RUL,and, therefore are also removed.

3.4.2 Feature analysis

Feature analysis was performed after the parameters selection on the optimized BiLSTM model, and is based on the results of two different techniques. For the feature analysis two techniques were chosen, the Pearson correlation matrix and the importance analysis from the RF model. Once the optimized BiLSTM model was obtained, the performance of the models is evaluated for three different feature selections. One model has all the features, another has features selected based on the Pearson correlation coefficients and the last model has features selected based on the importance analysis from the RF base model. The results are then compared to determine the effects on performance based on the chosen techniques, regarding the proposed BiLSTM model and the chosen dataset.

3.5 Summary

In this chapter the framework proposed is explained. We start by presenting the models chosen, which are a RF model, a XGBoost model, a LSTM model, and a BiLSTM model. The RF, XGBoost and LSTM models are used as benchmark models and the parameters chosen are either default parameters or, for the LSTM model, parameters based on the literature review. We explain the benefits and challenges of each model.

Then the initial BiLSTM model is presented with its parameters also based on the literature to create the base BiLSTM model. The benefits and challenges in applying this model are explained followed by an explanation of the framework applied through a flowchart.

The process of improving the base BiLSTM model through parameter optimization is explained. The parameters are briefly explained as well as their effects on the model, and the base model parameters as well as the parameters tested are presented when possible.

Lastly the process of feature analysis and feature selection performed is explained. Feature selection was performed before testing the models to remove undesirable features, and the feature analysis proposed is performed on the final BiLSTM model to better understand the effects of feature selection on model performance.

4

Turbofan engine monitoring Dataset

Contents

In this chapter the chosen dataset will be discussed along with an extensive analysis of its characteristics. The processing performed on the dataset to prepare it for the training of models is also explained.

The dataset chosen was obtained from the Prognostics Data Repository[1], designed for the development of prognostic algorithms and provided by NASA. This dataset is referred to as N-CMAPSS, due to it being a new and improved version of an older dataset named CMAPSS [31], released in 2008 in the same data repository.

Both datasets were generated using the Commercial Modular Aero-Propulsion System Simulation (CMAPSS) developed at NASA. The CMAPSS dataset garnered much attention since 2008, becoming one of the most used datasets in prognostics literature [5], with recent publications showing the best results of prognostic algorithms for the dataset [35], [42]. Having been released in January 2021, and being an improvement on the original, the N-CMAPSS dataset is expected to attract as much, if not more, attention than its parent work. With a great degree of quality, the N-CMAPSS is expected to lead to even better performing prognostic models.

The main advantages in using this dataset is not only its quality, but its quantity as well. The N-CMAPSS dataset provides full run-to-failure trajectories of a fleet of turbofan engines. Having the time-to-failure is crucial for the development of ML models and it's not typically available from real life applications [43]. This is both due to the rarity in failures occurring from excessive maintenance, as well as from the inherent sensitive nature of failures that inhibit companies from publicly sharing their assets' data. This situation has led to most literature turning to synthetic datasets, as was the case with the current work, even though prognostic models developed with this type of datasets will face challenges when transferring to real-life applications. The mentioned challenges arise, mainly, from the fact that generated datasets derive from models that lack the complexity of real systems [5].

4.1 Data description

In order to mitigate the challenges of transferring prognostic algorithms to real life applications the N-CMAPSS has two main improvements when comparing with the original CMAPSS dataset. First, the full flights were simulated, including climb, cruise and descent with real flight operation conditions coming from recordings on board of a commercial jet from NASA DASHlink-Flight Data. This has increased the number of samples in the data and the size of the available data significantly. Second, the degradation model was improved by relating the degradation onset with the operation history of the engine. These changes are meant to increase the fidelity of the model and allow predictive maintenance algorithms to more easily be transferred to real life situations. The N-CMAPSS dataset is divided into 10 sub-datasets

[1] https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan-2

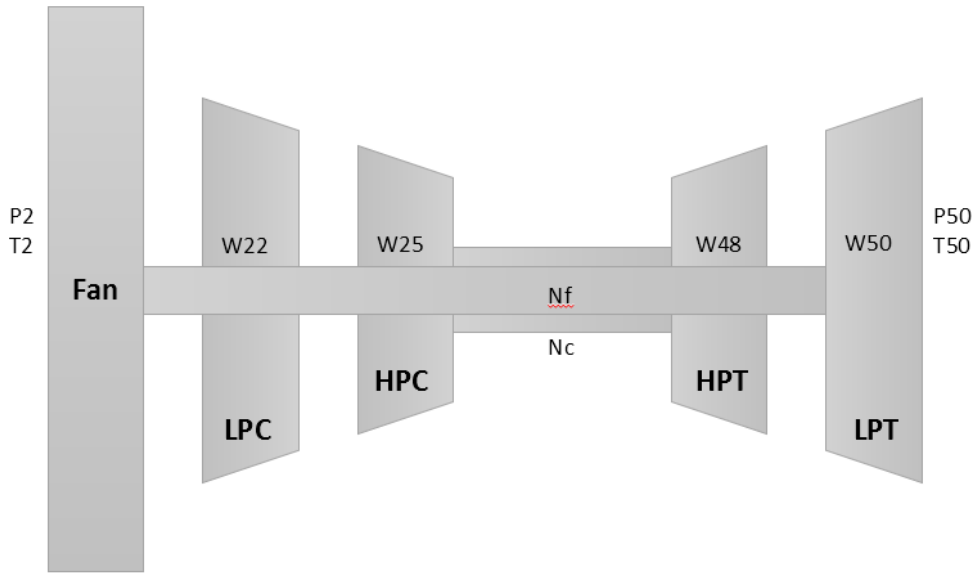


Figure 4.1: Schematic representation of the CMAPSS model depicting the 5 components and a few sensor variables based on [5]

with differences between each other, mainly in regards to the number of engines and the failure modes in each sub-dataset.

4.1.1 Engine Model

The simulation model simplifies a commercial turbofan aircraft engine to having 5 rotating components: fan, low-pressure compressor (LPC), high-pressure compressor (HPC), high-pressure turbine (HPT) and low-pressure turbine (LPT). Figure 4.1 shows a representation of the CMAPSS engine model as well as the sensor measurement variables generated by the model.

These 5 components may be subject to degradation in flow and efficiency. The number of components that are affected changes with the sub-dataset. An engine is considered to reach failure when the overall Health Index (HI), derived from the degradation model, reaches a value of zero or when the total operating cycles of the engine reaches one hundred. A cycle is a full flight, from climb to descent. The length of this cycle depends on the flight class of the engines. In each data file the engines are classified into 3 flight classes.Table 4.1 gives an overview of the three different flight classes present in the dataset, showing the flight length range and the total number of flights for each class. Table 4.2 gives an overview of the sub-datasets with the number of engines, flight classes, and failure modes affecting the efficiency (E) and/or flow (F), of each dataset . This table is slightly different from the one seen in [5] due to the updates of the dataset since its first release, from new data being generated.

Each data file is divided into a development dataset and a test dataset. This is the splitting of data used when training and testing the ML algorithms. Both datasets contain 6 types of variables: the

Table 4.1: Overview of the flight classes in the N-CMAPSS dataset

Flight Class	Flight Length[h]	Number of Engines [#]
1	1 to 3	27
2	3 to 5	33
3	> 5	39

Table 4.2: Overview of the sub-datasets

Name	# Units	Fligth Classes	Fan		LPC		HPC		HPT		LPT		Size
			E	F	E	F	E	F	E	F	E	F	
DS01	10	1,2,3							X				2.8 Gb
DS02	9	1,2,3							X		X	X	2.4 Gb
DS03	15	1,2,3							X		X	X	3.6 Gb
DS04	10	2,3	X	X									3.7 Gb
DS05	10	1,2,3					X	X					2.5 Gb
DS06	10	1,2,3			X	X	X	X					2.5 Gb
DS07	10	1,2,3									X	X	2.7 Gb
DS08a	15	1,2,3	X	X	X	X	X	X	X	X	X	X	3.2 Gb
DS08c	10	2,3	X	X	X	X	X	X	X	X	X	X	2.4 Gb

operative conditions, the measured sensors signals, the virtual sensors, the engine health parameters, the RUL label, and auxiliary data. For the purpose of predicting RUL, the health parameters are not used.

Operative conditions are the real flight conditions as recorded on board of commercial jets, taken from the NASA DASHlink repository. Because of it, this data already has noise present in it. The operation conditions include the altitude (alt), flight Mach number (Mach), throttle-resolver angle (TRA) and total temperature at the fan inlet (T2).Table 4.3 shows these variables' identifier, description, and units. For demonstration purposes, the operative conditions of the first cycle of Unit 1 in DS01 is shown in Figure 4.2. It is clear to see, from this figure, in particular the altitude variable, the full flight of an engine with climb, cruise, and descent clearly demarcated.

Similarly, Tables 4.4 to 4.6 provide the same information for the remaining types of variables in the dataset, namely the id, description, and units. The RUL label is taken from the number of cycles that each unit performs until failure. Figures 4.3 and 4.4 show the respective time series values of the sensor measurements, and the virtual sensor measurements variables. Figure 4.5 shows the flight class of each unit in the DS01, taken from the auxiliary data.

The measured signals are estimates of the measured physical properties at different points along the engine. These measurements were obtained from the CMAPSS model. There are a total of 14 different

Table 4.3: Operative conditions - ω

Id	Description	Unit
alt	Altitude	ft
Mach	Flight Mach Number	-
TRA	Throttle-resolver angle	%
T2	Total temperature at fan inlet	°R

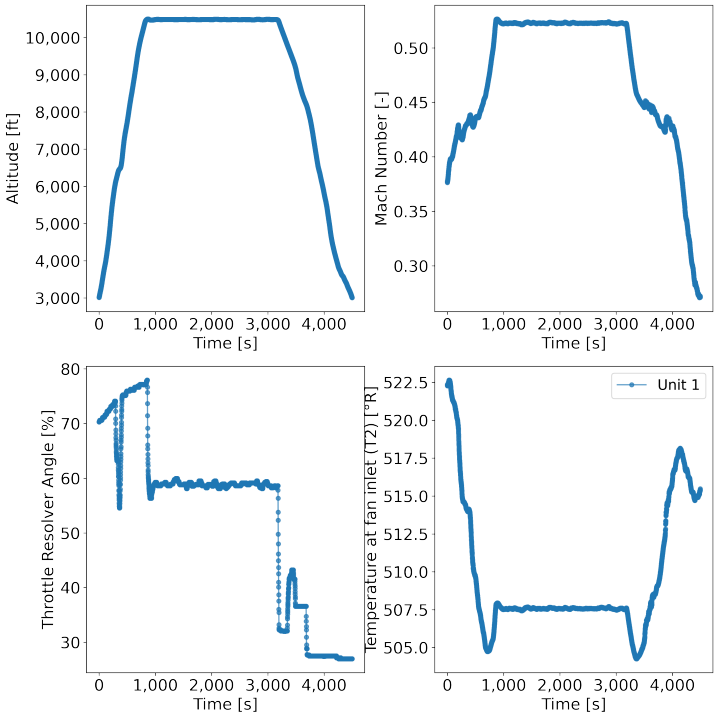


Figure 4.2: Operative conditions of the first cycle of Unit 1 of dataset DS01

variables, including physical properties such as flow, speed, temperature and pressure. These variables simulate the condition monitoring measurements that real life applications perform.

The virtual sensors are estimates of the unobservable properties that are not part of the condition monitoring signals, calculated from the measured signals. There are also a total of 14 variables, including properties such as temperature, pressure, flow, and stall margin.

Auxiliary data contains the unit number, flight cycle number, flight class (Fc), and health state (Hs) for each engine. Figure 4.5 gives the flight class of each unit in dataset DS01 and Figure 4.7 shows the degradation of the efficiency of the HPT component over time, in cycles, of each unit in DS01.

The engine health parameters are unobservable parameters that are used to simulate the deteriorating behaviour of the system components. Figure 4.6 shows the failure modes of DS01. We can clearly see that only the HPT efficiency is deteriorating, in accordance with table 4.2. Figure 4.7 shows the degradation of the HPT efficiency in dataset DS01, the only failure mode present in this dataset. We can identify that, in general, the units with higher flight class, such as unit 2 and unit 10, show a faster degradation rate, in terms of cycles, than units with flight class 1, like unit 1 and 3. This is due to the fact that the cycles of units with flight class 2 and 3 are longer than the flights of units with flight class 1, and so the degradation rate is faster, in terms of cycles, and the RUL is lower as a consequence, also in terms of cycles.

Table 4.4: Sensor measurements - x_s

Id	Description	Unit
Wf	Fuel flow	pps
Nf	Physical fan speed	rpm
Nc	Physical core speed	rpm
T24	Total temperature at LPC outlet	°R
T30	Total temperature at HPC outlet	°R
T48	Total temperature at HPT outlet	°R
T50	Total temperature at LPT outlet	°R
P15	Total pressure in bypass-product	psia
P2	Total pressure at fan inlet	psia
P21	Total pressure at fan outlet	psia
P24	Total pressure at LPC outlet	psia
Ps30	Static pressure at HPC outlet	psia
P40	Total pressure at burner outlet	psia
P50	Total pressure at LPT outlet	psia

Table 4.5: Virtual sensor measurements - x_v

Id	Description	Unit
T40	Total temperature at burner outlet	°R
P30	Total pressure at HPC outlet	psia
P45	Total pressure at HPT outlet	psia
W21	Fan flow	lbm/s
W22	Flow out of LPC	lbm/s
W25	Flow into HPC	lbm/s
W31	HPT coolant bleed	lbm/s
W32	HPT coolant bleed	lbm/s
W48	Flow out of HPT	lbm/s
W50	Flow out of LPT	lbm/s
SmFan	Fan stall margin	-
SmLPC	LPC stall margin	-
SmHPC	HPC stall margin	-
phi	Ratio of fuel flow to Ps30	pps/psi

Table 4.6: Auxiliary data

Id	Description	Unit
unit	Unit number	-
cycle	Flight cycle number	-
Fc	Flight class	-
h_s	health state	-

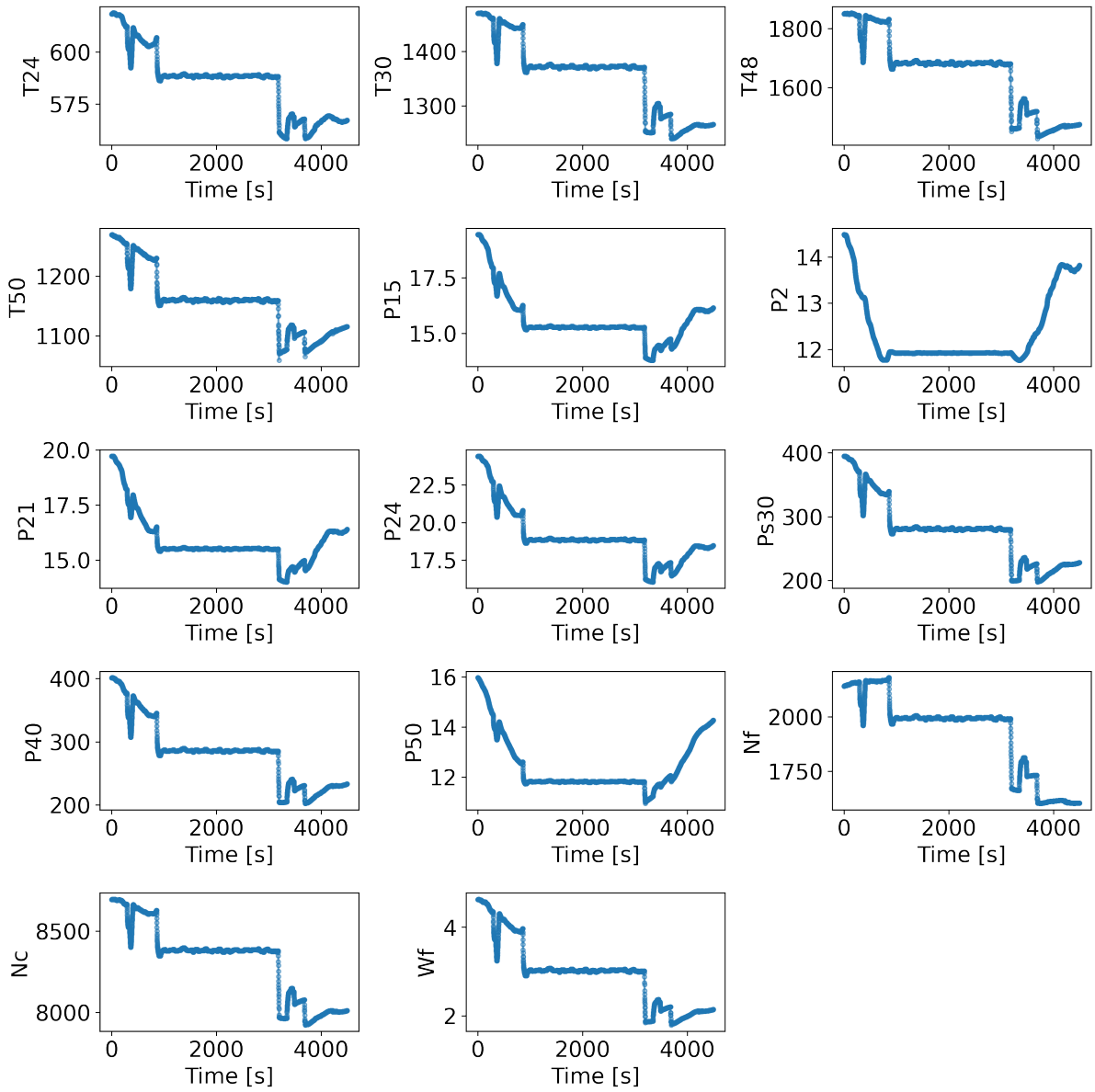


Figure 4.3: Sensor measurements of the first cycle of Unit 1 of dataset DS01

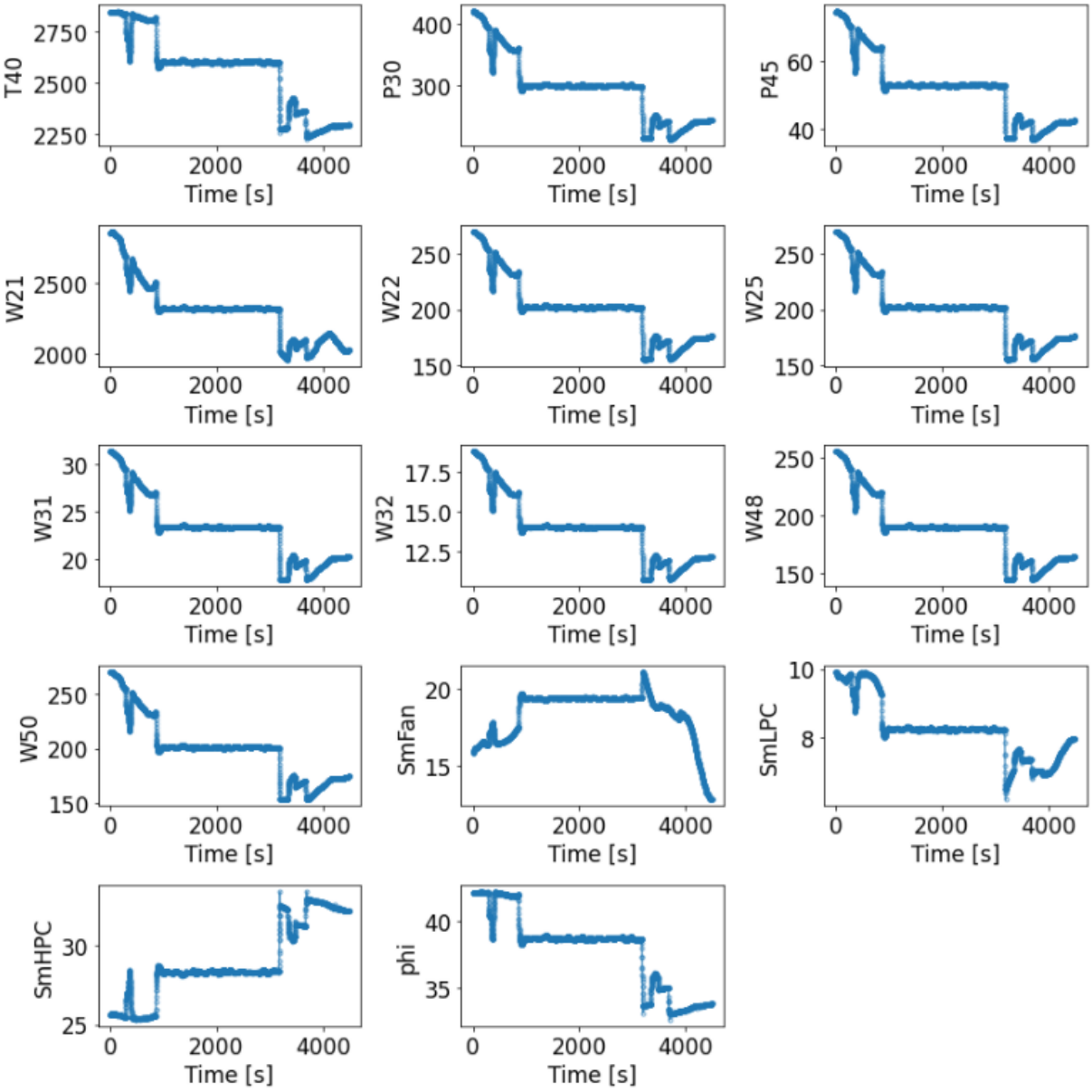


Figure 4.4: Virtual sensor measurements of the first cycle of Unit 1 of dataset DS01

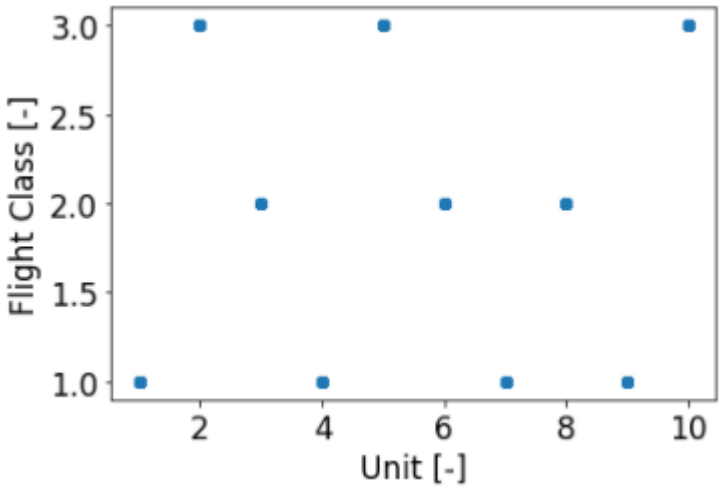


Figure 4.5: Flight class of all units of dataset DS01

4.2 Data analysis

Aside from understanding the behaviour of the data in the time domain, it is very important to understand the relations between the data itself, and between the data and the target variable, RUL. For this, correlation matrices for the operative conditions, sensor measurements, virtual sensor measurements, and the target data RUL were calculated using the Pearson method. The Pearson correlation coefficient gives a measure of the linear correlation between sets of data, its formula can be seen in Equation (4.1), where r is the correlation coefficient, x_i is the value of the x-variable in a sample, $\bar{x}$ is the mean of the values of the x-variable, y_i is the value of the y-variable in a sample and $\bar{y}$ is the mean of the values of the y-variable. Calculating this correlation is important as it can allow one to identify which data has a bigger relation to the target RUL and, therefore is important to use, and also remove data that has very little relation with the RUL, or even datasets that are extremely similar between each other, and so one of them can be removed without losing information.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}} \tag{4.1}$$

4.2.1 Features quality

The results of the correlation matrices are shown in Figures 4.8 and 4.9. Due to size limitations the correlation matrices were separated. Figure 4.8 gives the correlation matrix of the auxiliary data, operation conditions, sensor measurements, and RUL while Figure 4.9 gives the same information but instead of the sensor measurements, it has the virtual sensor measurements.

From these results we can see that several variables have a very low correlation coefficient, less

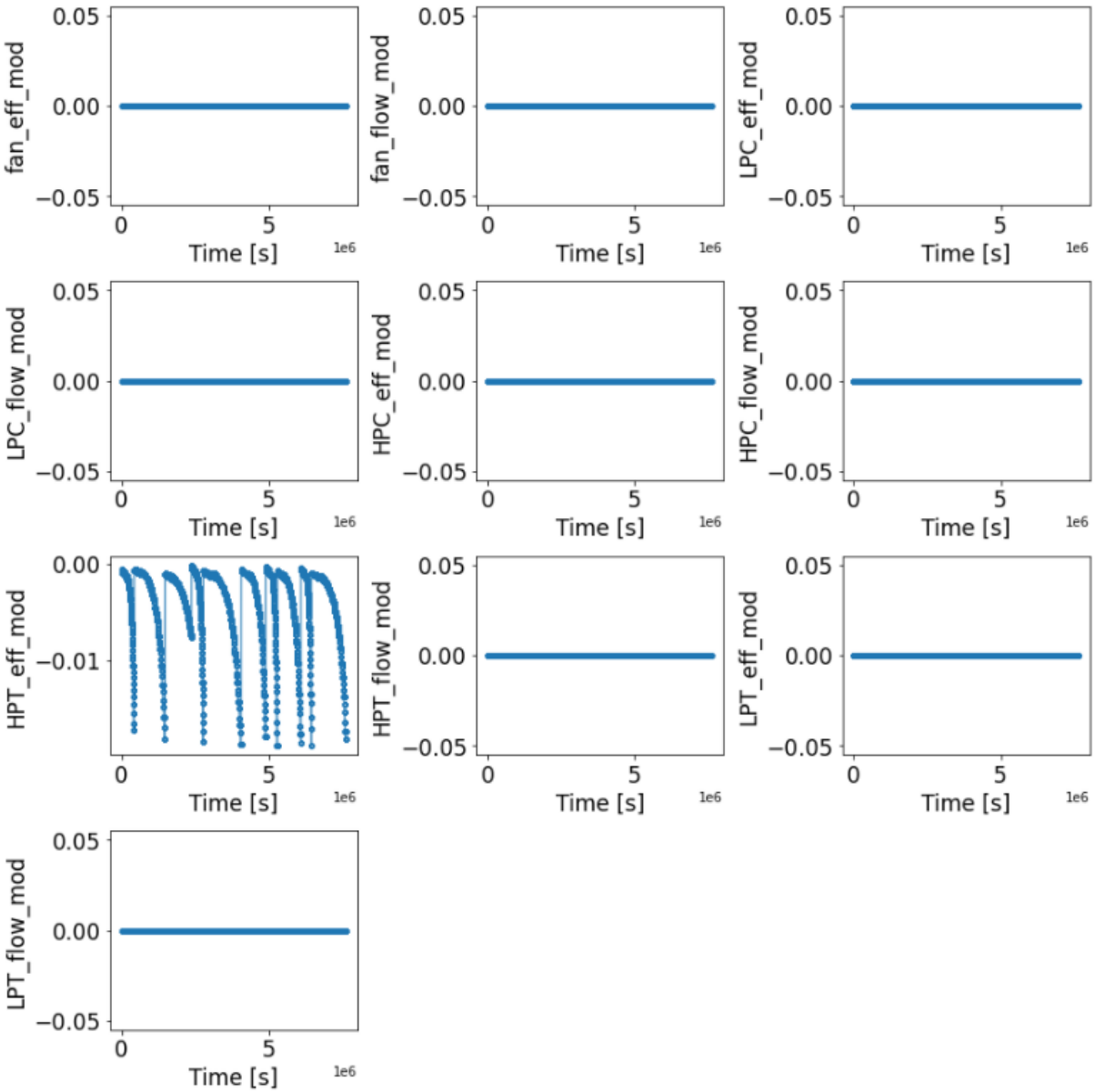


Figure 4.6: Failure modes of all units of dataset DS01

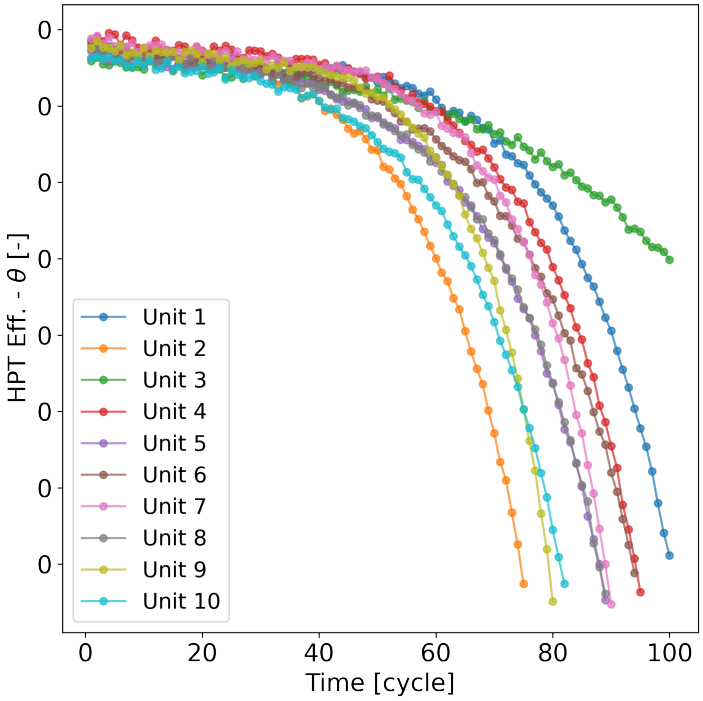


Figure 4.7: Degradation of the HPT efficiency in all units of dataset DS01

than **0.01** (e.g., Nf and Wf) with the target variable. Also, some variables are highly correlated with each other, higher than **0.95** (e.g., P15, P2, P24, and Ps30). From this information it is possible to hypothesize that removing these variables with low correlation to the RUL variable and, for highly correlated variables, using only one of these variables, we can improve the performance of the models. However, the conclusions obtained for DS01 cannot be transferred to a different dataset, since the failure modes are different and the variables will also have different correlation values. When performing the same correlation method with the data of dataset DS08a, which has all failure modes present, the correlation matrices are significantly different, which can be seen in Figures A.1 and A.2.

4.2.2 Feature selection

In this step, all features are analysed to find missing values and features with only constant values. These characteristics are undesirable when training models. However, since this dataset is synthetically generated, no missing values are present and the only feature with constant values is the *hs* variable, which is unusable for the predictive model, as it is not data obtainable from the sensor measurements. With datasets from real applications it is not uncommon to have missing values in the data, which would require appropriate processing.

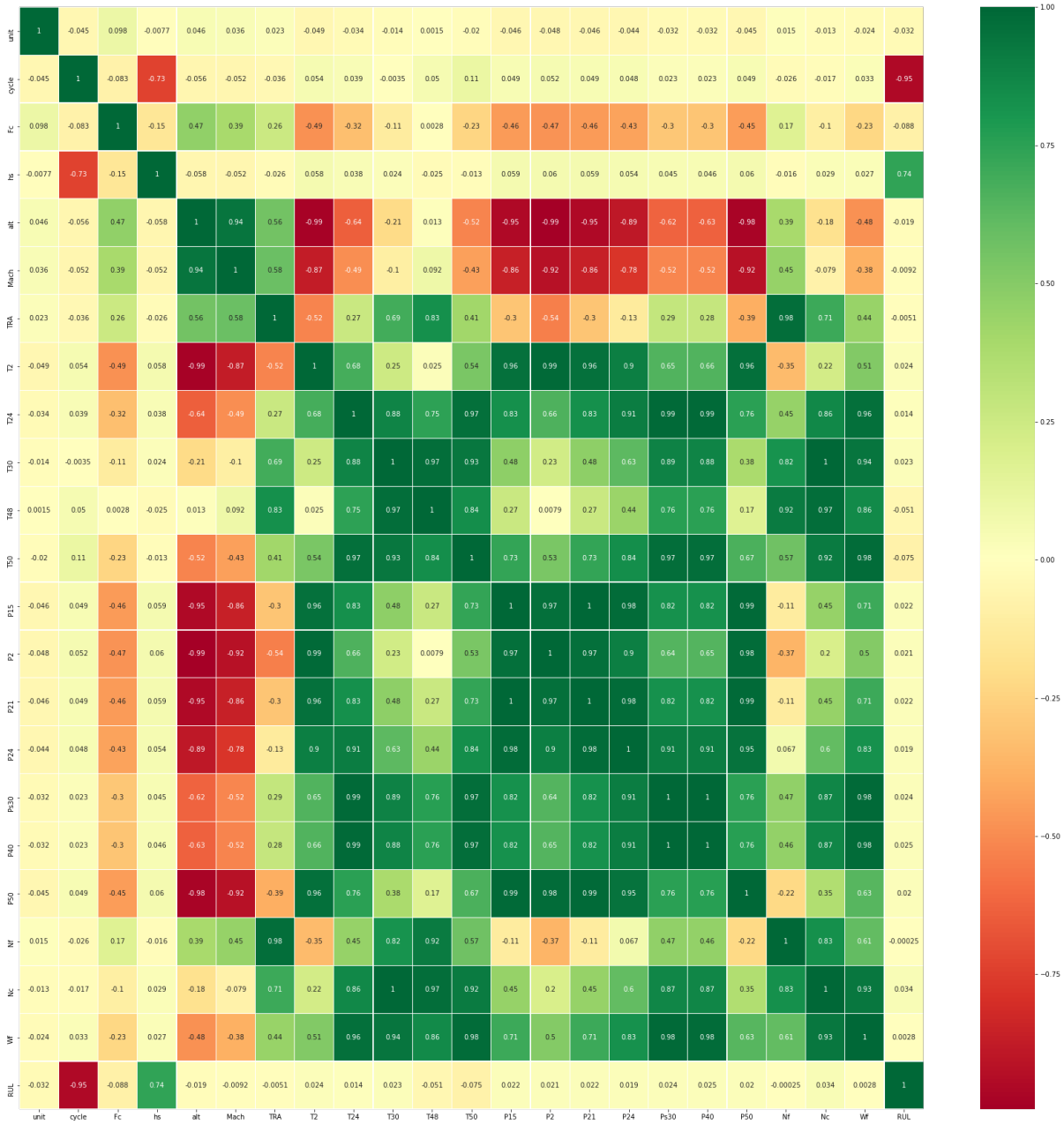


Figure 4.8: Correlation Matrix using the Pearson method of dataset DS01 - auxiliary data, operation conditions, sensor measurements and RUL

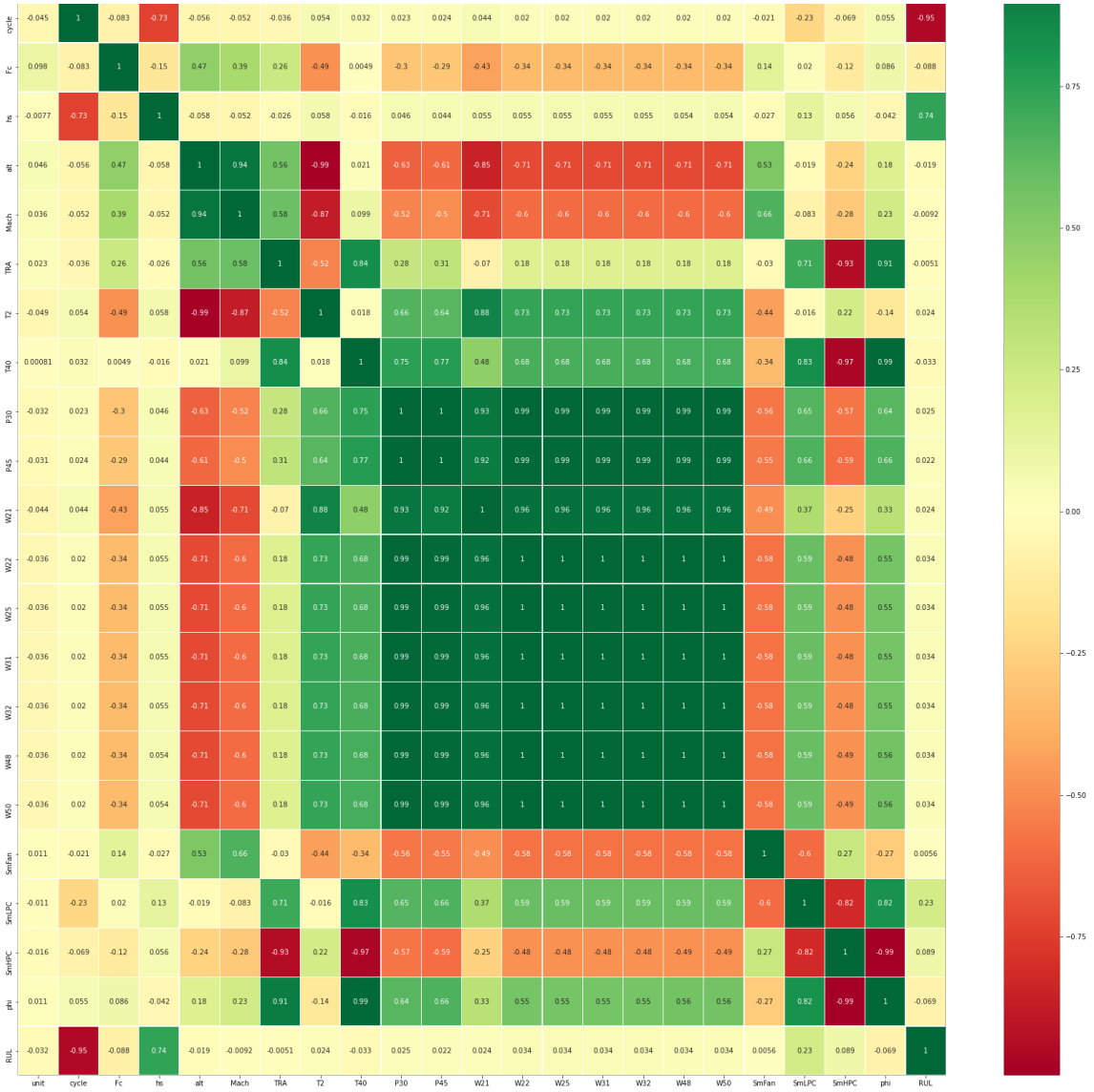


Figure 4.9: Correlation Matrix using the Pearson method of dataset DS01 - auxiliary data, operation conditions, virtual sensor measurements and RUL

4.3 Data preparation

4.3.1 Data sampling

The raw data is sampled in seconds, as is visible in Figure 4.2, Figure 4.3, and Figure 4.4. This means that the model has very a fine degree of information for training. However, a common problem, specially with complex models, is that the bigger the size of the data file, the longer it takes to train and test the models. If the model takes too long to train in practice, any benefit from using more data is lost, as improving the model with parameter optimization methods like grid search becomes impossible to perform in practice. Also, the premise that more data implies better models is, in general, true, but not for all models and all situations, therefore, when increasing the data size, one needs to be critical with its decisions.

Observing our specific dataset and the behaviour of the features over time, we can conclude that increasing the sampling size of the data is beneficial to reduce the size of data while still capturing the features of the data. For this purpose a sampling size of ten minutes was chosen.

4.3.2 Feature scaling

An important operation in the processing of data is feature scaling. This can be achieved through normalization of the data. Normalization is a transformation of data from its original range of values to a specific desired range of values. The method applied in this step is the min-max normalization. This method re-scales the range of the data to the range of values of [0, 1]. Min-max normalization was chosen as it is a simple method of feature scaling that has been previously used to great success in the literature [40] [35]. Feature scaling is important as the dataset used contains widely different range of values between variables, for example the variables of altitude (alt) and Mach number (Mach). Altitude has an approximate range of values between zero and ten thousand while Mach number has an approximate range between 0 and 0.5, which can be seen in Figure 4.2. This difference in order of magnitude can have detrimental effects to the training of ML models. Additionally, some ML algorithms, i.e., LSTM require the input data to be normalized, or cannot be used to train models at all. The formula for the min-max normalization method is show in Equation (4.2).

$$x' = \frac{x - min\,(x)}{max\,(x) - min\,(x)} \tag{4.2}$$

4.4 Summary

In this chapter the dataset chosen is described and analysed. First the origins of the dataset as well as benefits and limitations of using the dataset are explained. Then the model that generates the dataset is briefly explained and the data is described and visualized to better understand its behaviour over time.

Afterwards the data is analysed to understand the quality of the data regarding its features.

Lastly the processing of data performed to prepare it for the ML models is explained highlighting the sampling and the scaling of data applied.

5

Results and Discussion

Contents

In this chapter, the proposed models are evaluated on the N-CMAPSS datasets. First, the benchmark models results are shown, applied to the subset of data DS01. Then, the BiLSTM network parameters are optimized, and the results of each parameter are presented, in specific, the layer topology, dropout, learning rate, batch size, training/validation split, and time window size. Afterwards, the feature analysis results are shown. This includes the selection of features based on the correlation matrices and based on the importance analysis from the RF base model. Then, the optimized BiLSTM model structure is applied to the remaining sub datasets and the evaluation metrics compared. Finally, the optimized BiLSTM model is trained with all the datasets and the results on the test data presented.

5.1 Benchmark models results

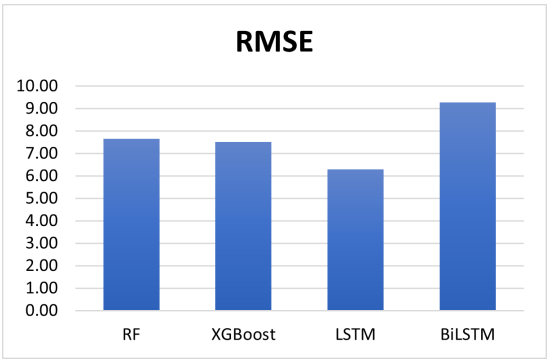
Table 5.1 show the results of the base models for both the evaluation metrics,RMSE and the NASA Score, the same results are seen in Figure 5.1(a) and Figure 5.1(b) through bar plots for better visualization and comparison. The LSTM model shows the best performance for both evaluation metrics, however, looking at the RUL predictions on the test dataset of each model in Figure 5.2 we can see that the BiLSTM model is sub-optimal. This means that better suited parameters are required to improve performance.

Analysing the RUL predictions of the BiLSTM model we see a desirable behaviour, that is not seen in the RF and the XGBoost model, but can be seen also in the LSTM model. The initial RUL prediction of the BiLSTM model is poor, but the model is capable of quickly adjusting and approximating the real RUL curve very accurately. Also when the RUL is close to zero, both the LSTM and BiLSTM model predictions seem to drift farther away from the real RUL, reacting to the discontinuity in the data. Given that the BiLSTM model is capable of using past and future information, we can see the effect of this capability in accurately predicting the RUL in the area close to this discontinuity.

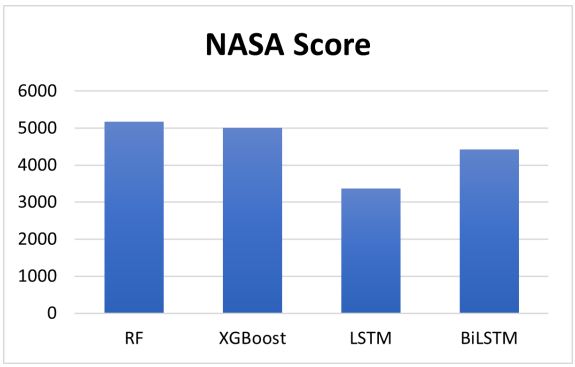
It is worth mentioning that, even though the BiLSTM model has worst performance on the RMSE metric, it has a better performance on the Score metric, compared with the RF and the XGBoost model. This is a consequence of the asymmetry of the Score metric. The RF model and the XGBoost model have an overall error lower, but the error is predominantly over estimations of the real RUL, when compared to the BiLSTM model that has predominantly under estimations, and, therefore, a lower Score metric value.

5.2 Model parameters optimization

The results for the parameters optimization are shown in this section. The parameter optimization was performed with the dataset DS01, which makes the final BiLSTM optimized for this dataset, but not

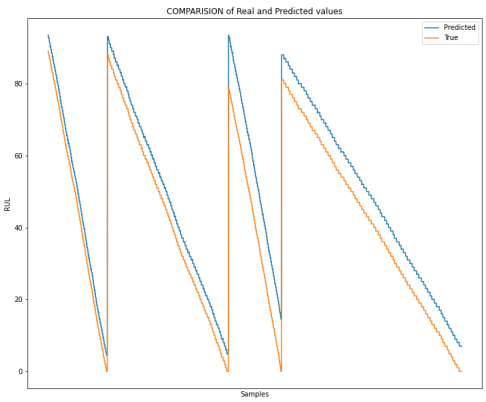


(a) RMSE of the base models: RF, XGBoost, LSTM and BiLSTM

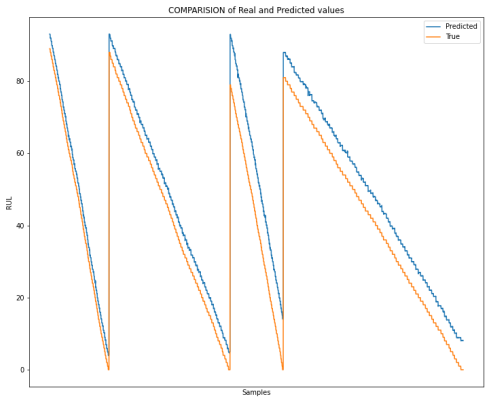


(b) NASA score of the base models: RF, XGBoost, LSTM and BiLSTM

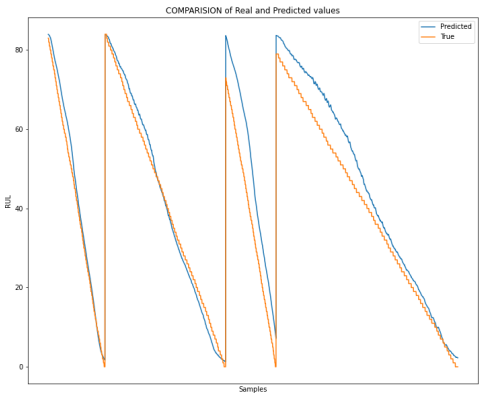
Figure 5.1: Comparison between the performance of the base models on test data DS01



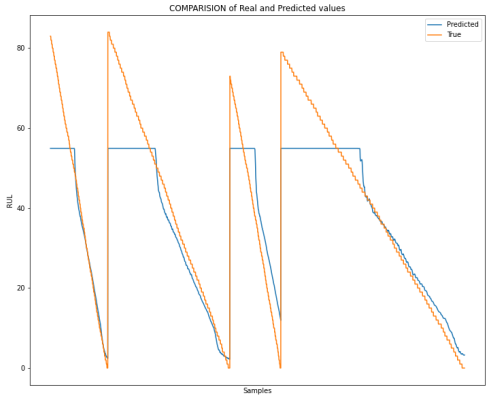
(a) RUL prediction of the base model RF



(b) RUL prediction of the base model XGBoost



(c) RUL prediction of the base model LSTM



(d) RUL prediction of the base model BiLSTM

Figure 5.2: RUL prediction of the base models on test data DS01

Table 5.1: Base models results on test data DS01: RF, XGBoost, LSTM, BiLSTM

Base Models	Score	RMSE
RF	5170	7.657
XGBoost	5004	7.505
LSTM	**3368**	**6.294**
Bi-LSTM	4421	9.275

necessarily for the remaining datasets, given that the datasets are different, specifically in the failure modes. The optimization of a BiLSTM model for each dataset was not performed as it is too time consuming, and given the time constraints. This is something that can be improved upon to obtain an optimized model for each subset of the data, however, given completely new data, the model will most likely still show worse performance than seen in our results.

5.2.1 Layer topology

To increase the complexity of the BiLSTM model, both the number of hidden layers or the number of neurons in each layer were increased. Several network architectures were tried starting with the base model. The results of the evaluation metrics on the test dataset DS01 for each network is shown in Table 5.2. The best performance is given by the network B(256,256,64) with a RMSE of 5.015 and a Score of 2307. This network has a contracting form, this is, the initial layers have a larger number of neurons and the last layers have less neurons. This structure seems to work better for our situation than a constant form or an expanding one, as the results indicate.

Table 5.2: Layer topology tuning of BiLSTM network

BiLSTM Network	Score	RMSE
B(64,32,16,8)	4421	9.275
B(64,64,64,64)	2701	5.446
B(256,256,256,256)	8556	10.75
B(64,256,32,32)	9755	11.61
B(128,128,64,64)	2886	5.744
B(256,256,64,64)	2923	5.676
B(256,256,64)	**2307**	**5.015**
B(128,128,64)	3750	6.632

5.2.2 Dropout

The dropout is applied uniformly to all the network layers, and the values tested are $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7]$. The base BiLSTM model has a Dropout of 0.5. Given the random nature of this technique the results presented in Table 5.3 change slightly when trying to reproduce them. The best performance was seen

with a dropout of 0.5. In order to counteract this limitation, training can be performed with the same parameter several times and the results visualized through a box plot. This would give a better understanding on the variability of each parameter, and more descriptive results overall. This process was not performed due to time constraints, and is a place for improvement.

Table 5.3: Dropout parameter tuning of BiLSTM network

Dropout	0.1	0.2	0.3	0.4	**0.5**	0.6	0.7
Score	2530	3833	3615	2678	**2307**	2859	3873
RMSE	5.298	6.742	6.636	5.439	**5.015**	5.666	6.738

5.2.3 Learning rate

The learning rates tested were $[0.0001, 0.001, 0.01, 0.1]$. The best performance was provided by a learning rate of 0.001, seen in Table 5.4. This is expected, as is not unexpected as it is also the best learning rate found in the literature. The effects of choosing incorrectly the learning rate are demonstrated by the significantly worse performance of the remaining learning rates. Both in RMSE and in Score.

Table 5.4: Learning rate parameter tuning of BiLSTM network

Learning Rate	0.0001	**0.001**	0.01	0.1
Score	6185	**2307**	17228	59878
RMSE	10.48	**5.015**	15.3	24.1

5.2.4 Batch size

The batch size has a significant impact on the training time, specially given the large dataset used. Several batch sizes were tested and a batch size of 512 presents the best performance, seen in Table 5.5. Smaller or larger batch sizes do not improve the performance of the proposed model, but they also do not show a great difference in performance, as was seen with the learning rate. The worst performance is given by a batch size of 128, with a result of 7.412 for the RMSE.

Table 5.5: Batch size parameter tuning of BiLSTM network

Batch size	128	256	**512**	768	1024
Score	4588	3507	**2307**	3015	3550
RMSE	7.412	6.393	**5.015**	5.941	6.65

5.2.5 Training/validation split

The training/validation split parameter refers to the percentage of data of the training data that is used for validation, during training. The values chosen were 10%, 20%, and 30% of the training dataset to be used as validation data, with the notation of 0.1, 0.2, and 0.3 respectively. The best result was provided a training/validation split of 0.1, seen in Table 5.6. The higher the percentage of validation data, the worse the results, as is expected when reducing the actual training data. Given our training data, the model is capable of using only 10% of the data for validation while not over fitting.

Table 5.6: Validation split parameter tuning of BiLSTM network

Validation split	**0.1**	0.2	0.3
Score	**2307**	3129	3387
RMSE	**5.015**	5.962	6.252

5.2.6 Time-window size

Time window size determines the amount of information *seen* by the model for each RUL prediction. The larger the window size, the more information available. However, too large of a window size can lead to over fitting. The time-window size parameter is tested for a size of 25, 50 and 100. Table 5.7 presents the results obtained. The results indicate that the smaller window size of 25 does not contain enough information but a larger window size of 100 is causing over fitting of the training data. A time window size of 50, given the sampling time used for the data processing, is the best option. If the sampling size was changed, it would indirectly affect the time window size. For example, changing the sampling size from 10 minutes to 5 minutes, while keeping the same time window size, would mean that the model would have available the same number of samples, but it would only *see* half the information in terms of time. In other words, instead of using the information from the past 500 minutes, it would use information from the past 250 minutes which can significantly impact the performance of the model in detecting long-term dependencies.

Table 5.7: Time-window size parameter tuning of BiLSTM network

Time-window size	25	**50**	100
Score	3893	**2307**	5214
RMSE	6.893	**5.015**	8.249

5.3 Feature analysis results

The hypothesis of reducing the number of features based on the Pearson correlation coefficients of the features was applied to the optimized model of BiLSTM. Also, using the base RF model, an importance analysis was performed, in order to select the most important features. This is a feature extraction capability of the RF model and the results are shown in Figure 5.3.

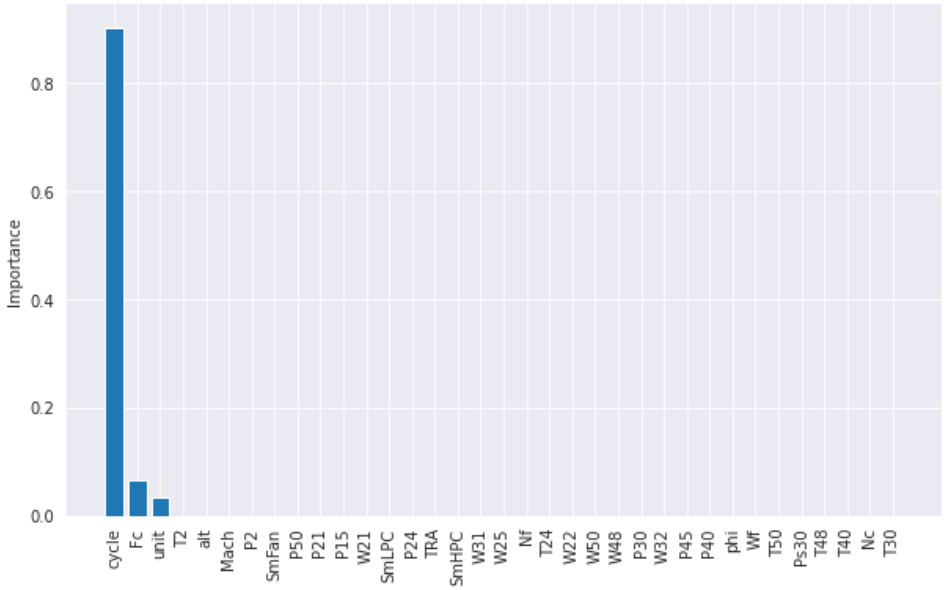


Figure 5.3: Importance analysis results from the RF base model

The three possible models were tested on the DS01 dataset, one model was trained with all the features, another was trained with the reduced features based on the Pearson correlation matrix, the last model was trained with the reduced features based on the importance analysis from the RF base model. According to the correlation matrices in Section 4.2.1, the features to remove, for DS01, are *'Mach', 'TRA', 'Nf', 'Wf', 'P2', 'P40', 'Ps30', 'T48', 'Nc', 'T50', 'P50', 'P24', 'P21', 'phi', 'SmFan', 'P45', 'W22', 'W25', 'W31', 'W32', 'W48', and 'W50'*. Based on the importance analysis the feature to remove are *'Wf', 'P40', 'Ps30', 'T48', 'Nc', 'T50', 'phi', 'P45', 'W22', 'W32', 'W48', AND 'W50'*. The results are presented in table 5.8. The best performance was given by the reduced features based on the importance analysis. Also the network with reduced features based on the correlation matrix performed worse than using all the features. This means that the dependencies of the dataset cannot be fully represented through the Pearson correlation coefficient, which led to worse performance.

Table 5.8: Results of the feature selection based on the feature analysis performed

Feature Selection	All feat.	Correlation matrix feat.	**Importance feat.**
Score	2307	3851	**1852**
RMSE	5.015	6.91	**4.398**

5.4 Performance of final BiLSTM model on all sub datasets

The optimized BiLSTM network was applied to the remaining datasets and the results are shown in Table 5.9. For these results all features are used to better compare the results between models, as the feature analysis was performed only for DS01. Since the model was optimized using the DS01 dataset, it has a poorer performance on the remaining sub datasets, as was expected.

Even though the datasets are similar in structure, they have significant differences, specially the failure modes. This not only affects which features are more relevant for each dataset, it also affects the average RUL of each sub dataset. Datasets with all failure modes present, such as DS08a and DS08c, have an average RUL significantly lower than datasets such as DS01, which only have one failure mode present.

It is clear that the BiLSTM network is capable of providing very good RUL predicitions, as seen from the RUL prediction of DS01, but the model is not directly transferable to the remaining datasets. A new hyper parameter optimization would need to be performed to achieve better results on the remaining datasets. Looking at the loss function graph of each model in Figures A.5 to A.12 we can see that the models for the sub datasets DS04 and DS06 did not run for enough epochs to converge to a solution due to the Early stopping technique implemented. For these models, increasing the number of epochs from 20 to, for example 40, would most likely solve this problem and allow the convergence of the solution, which would improve the results.

Surprisingly, the performance of the model on the sub dataset DS08a is extremely good in comparison with the other models, with a RMSE of 5.999. However when looking at the loss function and the RUL predictions of this model in Figure A.11, we see that the model is, in fact, over fitting the data, as the loss from the validation data is much higher than the loss from the training data, and the RUL predictions are all very similar. The good performance presented seems to be a fortunate instance, as the test data of the sub dataset DS08a is also very similar between engines.

Lastly the results of the BiLSTM model applied to the complete dataset simultaneously. The performance evaluation is shown in Table 5.10 and the loss function graph can be seen in fig. A.3. In regards to the RMSE the result of the full dataset and the mean of the results of each dataset is not too different, with a RMSE of 14.08 and 13.67 respectively. The Score metric can be compared by summing all the Score values of each sub dataset, which gives 221751. This is a better Score value than the one obtained from the model applied to the full dataset of 284079. We can conclude that the model applied to

the full dataset predicts more over estimations than when applying the BiLSTM model on each dataset.

Table 5.9: Results of the BiLSTM network on all sub datasets

Dataset	DS01	DS02	DS03	DS04	DS05	DS06	DS07	DS08a	DS08c	Mean
Score	2307	5657	7369	89725	24988	19317	50292	4356	3665	23075
RMSE	5.015	13.63	8.202	23.58	17.77	17.64	21.19	5.999	10	13.67

Table 5.10: Results of the BiLSTM network on the complete N-CMAPSS dataset

Dataset	Full N-CMAPSS dataset
Score	284079
RMSE	14.08

5.5 Comparison with results of other models applied to the CMAPSS dataset

As far we know there are no other models in the literature applied to the N-CMAPSS dataset to compare with the current results. There are, however, results from other models applied to the CMAPSS dataset, the precursor to the N-CMAPSS dataset used in this work. Even though the datasets have significant differences they have the same origin.

Focusing on the results of the proposed model for the dataset DS01, as this was the model used for the parameter optimization, we can see a significant improvement on the RMSE obtained with the proposed model when compared to other models. specifically, an improvement of 63.3% on the best model applied to the CMAPSS dataset. The benchmark models also show better results in terms of RMSE, even with comparatively less complex ML algorithms than those used on the CMAPSS dataset. These results show the improvement in regards to the quality of the N-CMAPSS dataset, when compared to the CMAPSS dataset. It also shows the benefits of using larger and finer dataset on the asset being studied, for ML algorithms, given the significant improvements of the accuracy of the RUL predictions. The Score performance index is worse for, not because the RUL predictions are less accurate, but because the number of samples predicted is much higher. Since the Score function is a sum of the error, without using the same amount of samples in all the models, it is not possible to accurately compare the results.

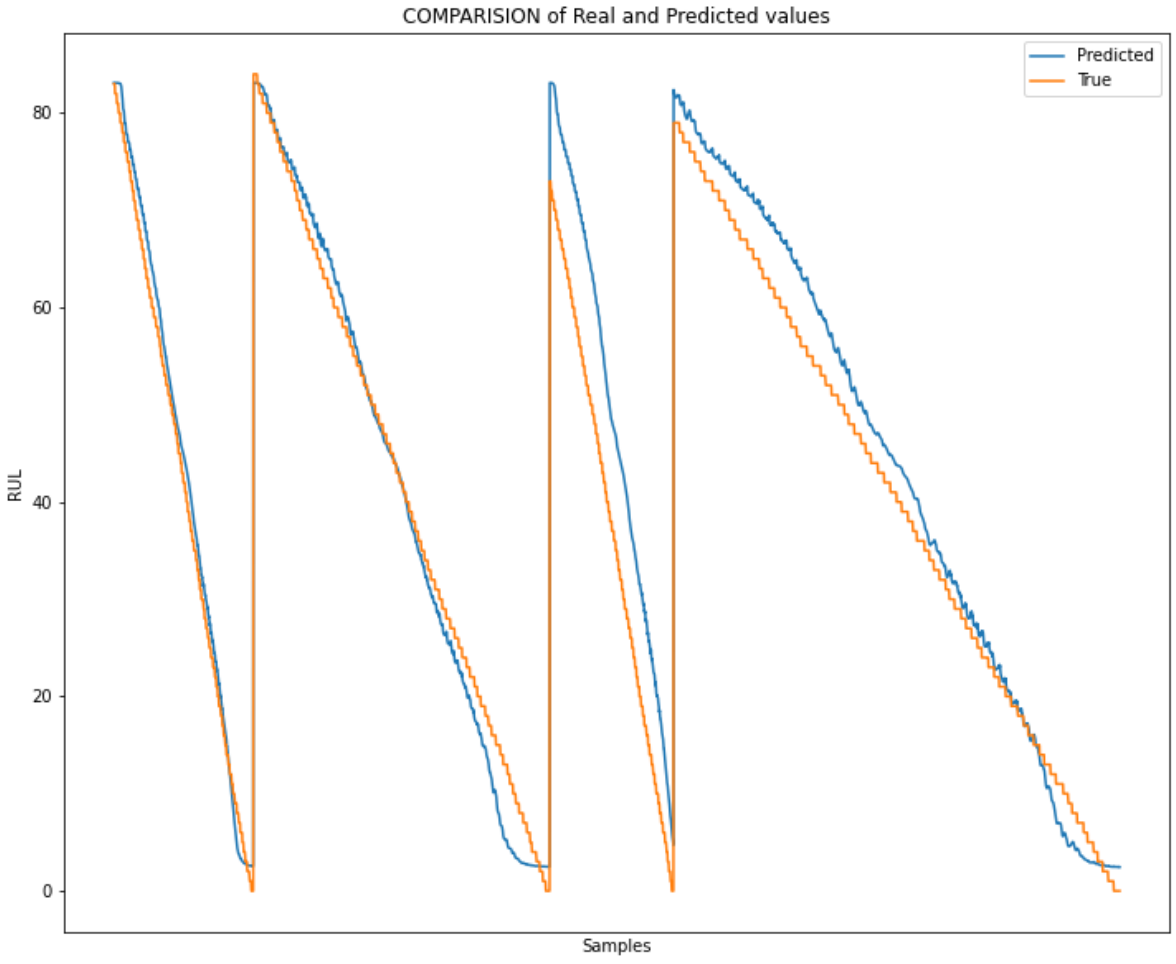


Figure 5.4: RUL prediction of the final BiLSTM model for DS01

Table 5.11: RMSE of other models with the CMAPSS dataset

ML method	FD001	FD002	FD003	FD004
MLP [44]	37.56	80.03	37.39	77.37
SVR [44]	20.93	42.00	21.05	45.32
RVR [44]	23.80	31.30	22.34	34.34
CNN [44]	18.45	30.29	19.82	29.16
MODBNE [45]	15.04	25.05	12.51	58.66
LSTM [38]	16.14	24.49	16.18	28.17
BiLSTM [37]	13.65	23.18	13.74	24.86
DCNN [46]	12.61	22.36	12.64	23.31
CNN-BiLSTM [35]	12.58	**19.34**	**12.18**	**20.03**
DAG [47]	**11.96**	20.34	12.46	22.43

Table 5.12: Score of other models with the CMAPSS dataset

ML method	FD001	FD002	FD003	FD004
MLP [44]	1.80×10^4	7.80×10^6	1.74×10^4	5.62×10^6
SVR [44]	1.38×10^3	5.90×10^5	1.60×10^3	3.71×10^5
RVR [44]	1.50×10^3	1.74×10^4	1.43×10^3	2.65×10^6
CNN [44]	1.29×10^3	1.36×10^4	1.60×10^3	7.89×10^3
MODBNE [45]	3.34×10^2	5.59×10^3	4.21×10^2	6.56×10^3
LSTM [38]	3.38×10^2	4.45×10^3	8.52×10^2	5.55×10^3
BiLSTM [37]	2.95×10^2	4.13×10^3	3.17×10^2	5.43×10^3
DCNN [46]	2.74×10^2	1.04×10^4	2.84×10^2	1.25×10^4
CNN-BiLSTM [35]	2.31×10^2	$\mathbf{2.65 \times 10^3}$	$\mathbf{2.57 \times 10^2}$	3.40×10^3
DAG [47]	$\mathbf{2.29 \times 10^2}$	2.73×10^3	5.53×10^2	$\mathbf{3.37 \times 10^3}$

5.6 Summary

In this chapter the results of the proposed models are presented and discussed. First, the results of the base models used as benchmark are compared. Here we identify LSTM as the best model for both metrics. BiLSTM is, in fact the worst model, in terms of RMSE, due to having incorrect parameters, most importantly, we can infer that the network is not complex enough to model the data. The parameters of the BiLSTM are then optimized manually and one at a time, using the sub dataset DS01. The loss function graph is used to help in the choices of parameters to test. Through the parameter optimization we conclude that, most of the parameters were already optimal, and only the layer topology was changed.

After obtaining good performance with the BiLSTM model the proposed feature analysis is tested and the results presented. We can conclude that, reducing the features based on the Pearson correlation coefficient did not improve performance, meaning that the Pearson coefficients are unable to completely represent the dependencies between the data. Reducing the features based on the importance analysis from the base RF model did improve the performance, achieving the best performance obtained of a Score of 1852 and a RMSE of 4.398.

Next, the remaining subsets of data are tested, to measure the generalization of the model and its

Table 5.13: Proposed models results on the N-CMAPSS dataset DS01: base RF, XGBoost and LSTM model and final BiLSTM model

ML Method	Score	RMSE
RF	5.170×10^3	7.657
XGBoost	5.004×10^3	7.505
LSTM	3.368×10^3	6.294
Bi-LSTM	$\mathbf{1.852 \times 10^3}$	**4.389**

performance with completely new data. With the results obtained we can conclude that, as expected, the performance on the new, unseen data is significantly worse than the performance for the sub dataset DS01. Some sub datasets did not have time to converge, due to the early stopping technique implemented, and other data sets suffered of over fitting.

Even though the BiLSTM has the capability of extremely accurate RUL predictions it requires proper parameter optimization, for each subset of data available. Additionally, manually optimizing the parameters is very hard and requires expert experience, however we can identify the layer topology as the main parameter to explore, given its influence on the results presented. The BiLSTM model was also applied to the full dataset available, however the results are worse than applying the model on each subset of data. The RMSE of the model applied to the complete dataset is 14.08 while the average of the RMSE of each model applied to the sub datasets separately is 13.67.

Lastly the results of the proposed models are compared with the results from state-of-the-art model applied to the CMAPSS dataset, the predecessor of the N-CMAPSS dataset. In terms of Score values the results cannot be compared, as the number of samples of the models are different. In terms of RMSE the proposed models perform much better in estimating RUL, even when using less complex models like the RF and the XGBoost model. This is a proof of how increasing the quantity of data can improve the performance of ML models. The N-CMPASS dataset has full run-to-failure trajectories, unlike the CMAPSS dataset, which means a larger and finer dataset, and this data allowed for more accurate models.

6

Conclusions and Future Work

Contents

The present thesis explores RUL estimation using ML algorithms in the context of Predictive Maintenance. The overview of PdM in chapter 2 allows for a better understanding of PdM and its position in the maintenance setting. By exploring commonly used maintenance strategies and their limitations as well as the current environment of I4.0, we can understand the reasoning for PdM, specifically how key technologies such as IoT, Big data, and Cloud services enable PdM to excel. We identify the main reason for industries to suffer of inefficient maintenance to be the lack of knowledge regarding the real condition of assets and their need for maintenance. With this in mind, we explore how Machine Learning, a powerful tool for prediction tasks, allied with I4.0 has led to new techniques of PdM and succeeded in accurately measuring the need for maintenance, in specific the RUL of assets. Based on this study, we identify promising ML algorithms, such as Random Forest, XGBoost, LSTM, and BiLSTM, and propose a framework in chapter 3 to achieve robust and reliable ML models for predicting RUL. The framework includes evaluation metrics that accurately describe the overall error and allow for easy comparison between other state-of-the-art models. It also includes the parameter optimization process applied and the feature analysis performed, in order to achieve satisfying results. The RF, XGBoost and LSTM model were chosen to use as benchmark models, while the BiLSTM model was chosen to optimize, due to its advantages identified in the literature review. In specific, the capability of analysing large time series datasets, of detecting long term dependencies in the data using past and future information, and great performance in RUL prediction tasks.

6.1 Conclusions

The following sections discuss final considerations on chapter 3, chapter 4, and chapter 5, which cover the proposed framework, preparation of the dataset chosen, the presentation and discussion on the results obtained from the proposed models and analyses performed.

6.1.1 Proposed framework

The proposed framework consists of the ML models, the evaluation metrics used, the parameters selection process and the features analysis. The RF, XGBoost and LSTM models are appropriate for benchmark given their ease of use and, regarding the LSTM model, its similarity with the BiLSTM model. The BiLSTM model is suitable to explore further given its capabilities and potential for greater performance results. It is, however, a complex model that requires a lot of time to train and test, due to the extensive computations required. From our experience, training the BiLSTM models took, approximately, 20 times longer than training the RF or the XGBoost models. Without the access to the cloud services and the GPU to perform these computations, it would not have been possible to optimize the BiLSTM model, either through an automatic or manual method, and achieve good results.

Regarding limitations on the evaluation metrics chosen, the RMSE was found to be useful to compare between the models proposed in this work and also with other models found in the literature. However, the Score metric is only suitable to compare between the models proposed, as it depends on the number of samples of the input data. It is not possible to compare this performance metric with models outside the present work, unless they use the same number of input samples. A solution to this would be to select additional evaluation metrics common in RUL prediction and time series predictions, such as the mean absolute error and the R squared, to complement the chosen performance metrics.

Concerning the model parameters selection, the main limitation in choosing a manual process is that it does not guarantee that an optimal model was obtained, within the chosen parameters. Using an automatic method, as the grid search cross fold validation would be a solution for this, but, given the time constraints of the present work, it was not applied. Another limitation is that the present framework is susceptible to variance. A solution for this would be to train the models several times during the parameters selection and calculate box plots to understand how variability is affecting the results, in order to be able to choose parameters more confidently.

Lastly, the feature analysis proposed is successful as it allows us to determine, between the chosen techniques, this is, the Pearson correlation coefficients and the importance analyses from the RF model, which is suitable for our dataset to use, and what effects does the feature selection have on the final results. A limitation observed is that the analysis was only applied to the BiLSTM model and to the sub dataset DS01. Applying this analysis to the remaining benchmark models and the remaining sub datasets would provide more information from which to draw conclusions.

In summary, the proposed framework has limitations mainly derived from the limited time available, but it was possible to obtain relevant results and conclusions following the proposed framework.

6.1.2 Dataset

The dataset chosen is the N-CMAPSS. This dataset is very recent and is provided by NASA in the Prognostics Data Repository. The data proved to be of very high quality, in terms of quantity and in terms of features quality. However, too much data can lead to impractical computation times and a resampling technique was performed to reduce the rate of data from the original rate of $1second$ to a sampling rate of $10minutes$. This sampling frequency proved the best equilibrium between capturing the data behaviour and reducing the computation time, in the present work, however, finer sampling rates, given enough computational capacity, may lead to better results.

Regarding limitations of the features analysis performed, we identified that the sub datasets have different failure modes which significantly influence the average RUL of the sub datasets and affects the correlation of the features with the RUL. This makes it difficult to develop a single ML algorithm able to perform well on all sub datasets, with the proposed framework, as the optimal feature selection

on each sub dataset becomes different, which was verified by the results in chapter 5. Given the time constraints of the present work, the feature analysis was performed only for dataset DS01, and not for the remaining sub datasets, which would be an important step in developing optimized models for each of the sub datasets.

A final limitation identified is that, even though this dataset was generated with a high fidelity simulator, it is still a simplified model of its real counterpart. This means that transferring the models developed with this dataset to a real application may prove difficult, moreover it would most certainly require new parameters selection.

Overall even though the chosen dataset has its limitations and challenges, it was possible to perform the intended RUL prediction with relevant results.

6.1.3 Results

In this chapter the results of the proposed models are presented and discussed. The results of the base models used as benchmark are very good, already outperforming the best models applied to the CMAPSS dataset. This is clear evidence in the incresed quality of the N-CMAPSS dataset. Here we can also see the asymmetric behaviour of the Score function when comparing the performance of the BiLSTM model and the RF and XGBoost model. The RF and the XGBoost model have a better performance on the RMSE metric but constantly over estimate the RUL, while the BiLSTM model, even with worse RMSE value, has better Score value, because it has very little instances of over estimation. We can also identify a very desirable behaviour of the BiLSTM algorithm that the RF and the XGBoost do not possess. The BiLSTM model, after a bad initial RUL prediction, is capable of adjusting its predictions to approximate the real RUL curve, while the RF and XGBoost models have better initial RUL predictions, but are unable to adjust their RUL predictions.

The model parameters selection results allow us to conclude that, for the dropout, learning rate, batch size and training/validation split, the initial values chosen based on the literature study, were already the best choices. The layer topology parameter proved to be the hardest one to select, as there are many possible combinations that significantly impact the results. Even with this difficulty present, good results were obtained with the final BiLSTM model, showing an improvement of 63.3% on the best model prediction found in the literature on the RMSE of models applied to the CMAPSS dataset.

The feature analysis proposed showed that, for the dataset DS01, using the Pearson correlation coefficients for feature selection degrades performance, but using the importance analysis from the RF base model improves the results. The analysis achieves the best performance of a Score of 1852 and a RMSE of 4.398, an improvement of 80.3% and 87.7%, respectively, when comparing with the BiLSTM model with no feature selection. Since the importance analysis was applied only to the DS01 dataset, it is not possible to see the effects of feature selection on the remaining datasets, but the Pearson

Correlation matrices of the DS08a, clearly show that each sub dataset requires its own feature analysis for optimal feature selection.

The results of the BiLSTM model applied to the remaining subsets of data show that the performance on the new data is significantly worse than the performance obtained with the sub dataset DS01. Sub datasets DS04 and DS06 did not have time to converge, due to the early stopping technique implemented, and subsets DS08a and DS08c suffered of over fitting. From these results we can conclude that, even though the BiLSTM has the capability of extremely accurate RUL predictions as seen with the results from DS01, it requires proper parameter optimization. Additionally, manually optimizing the parameters requires expert experience, and does not guarantee optimal solutions. We can identify the layer topology as the main parameter to explore, given its influence on the results obtained. The proposed BiLSTM model was also applied to the full dataset, but the results of this approach were not satisfactory.

Lastly, the results of the proposed models are compared with state-of-the-art model applied to the CMAPSS dataset, the predecessor of the N-CMAPSS. It is a limitation of this work that it is not possible to compare the proposed models with other models applied to the N-CMAPSS. To the extent of our knowledge no other ML models applied to the N-CMAPSS were found in the literature, most likely given the recent nature of this dataset. Compared with models applied to the CMAPSS dataset, all the proposed models outperform, when applied to the DS01 sub dataset, even the less complex models. This is a consequence of the quality of the data being higher, even having the same origin. The main contributing factor identified is the N-CMAPSS having significantly more data samples, that increase the models accuracy in RUL prediction.

In conclusion, several limitations and challenges were identified with the chosen dataset and proposed framework, however the main objectives of accurately predicting RUL using ML algorithms, in the context of PdM, and the comparison of the proposed models with other ML models from the literature, were achieved. The results obtained show promise for further research and in providing critical information to support decision making for predictive maintenance strategies.

6.2 Future work

In this section we propose future work regarding the proposed models and framework adopted. Most of the proposals stem from tasks that were intended to be performed but could not be due to time constraints, which also relates to the computation capacity available.

Regarding the feature analysis, we propose an in-depth study of the correlations between the features and the RUL for all subsets of data available. By better understanding this correlation, we can improve on the feature selection process and on the results of RUL prediction for each sub dataset.

Concerning the framework, the future work proposed focuses on two aspects. First, applying an automatic parameter selection method in order to confidently achieve optimal solutions when selecting the parameters of the models. Second, perform the training of models several times in order to calculate average, median, maximum and minimum, as well as the frequency distribution of the performance results in order to reduce the effect of variability on the results presented. With this two improvements we could achieve better models and a good degree of confidence on the results obtained.

Bibliography

[1] G. Anurag and DATTUS, "5 key metrics that affect operational efficiency," accessed October 2, 2021, https://www.pumpsandsystems.com/5-key-metrics-affect-operational-efficiency. [Online]. Available: https://www.pumpsandsystems.com/5-key-metrics-affect-operational-efficiency

[2] Y. Lu, A. A. Miller, R. Hoffmann, and C. W. Johnson, "Towards the automated verification of weibull distributions for system failure rates," in *Critical Systems: Formal Methods and Automated Verification*. Springer, 2016, pp. 81–96.

[3] H. M. Hashemian, "State-of-the-art predictive maintenance techniques," *IEEE Transactions on Instrumentation and measurement*, vol. 60, no. 1, pp. 226–236, 2010.

[4] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[5] M. Arias Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics," *Data*, vol. 6, no. 1, 2021. [Online]. Available: https://www.mdpi.com/2306-5729/6/1/5

[6] R. K. Mobley, *An introduction to predictive maintenance*. Elsevier, 2002.

[7] M.-Y. You, F. Liu, W. Wang, and G. Meng, "Statistically planned and individually improved predictive maintenance management for continuously monitored degrading systems," *IEEE Transactions on Reliability*, vol. 59, no. 4, pp. 744–753, 2010.

[8] S. Selcuk, "Predictive maintenance, its implementation and latest trends," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 231, no. 9, pp. 1670–1679, 2017. [Online]. Available: https://doi.org/10.1177/0954405415601640

[9] Z. M. Çınar, A. Abdussalam Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, and B. Safaei, "Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0," *Sustainability*, vol. 12, no. 19, 2020. [Online]. Available: https://www.mdpi.com/2071-1050/12/19/8211

[10] W. Zhang, D. Yang, and H. Wang, “Data-driven methods for predictive maintenance of industrial equipment: A survey,” *IEEE Systems Journal*, vol. 13, no. 3, pp. 2213–2227, 2019.

[11] Z. Liu, W. Mei, X. Zeng, C. Yang, and X. Zhou, “Remaining useful life estimation of insulated gate biploar transistors (igbts) based on a novel volterra k-nearest neighbor optimally pruned extreme learning machine (vkopp) model using degradation data,” *Sensors*, vol. 17, no. 11, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/11/2524

[12] W. Zhang, X. Li, X.-D. Jia, H. Ma, Z. Luo, and X. Li, “Machinery fault diagnosis with imbalanced data using deep generative adversarial networks,” *Measurement*, vol. 152, p. 107377, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224119312412

[13] H. Yang, F. Zhao, G. Jiang, Z. Sun, and X. Mei, “A novel deep learning approach for machinery prognostics based on time windows,” *Applied Sciences*, vol. 9, no. 22, 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/22/4813

[14] X. Li, Q. Ding, and J.-Q. Sun, “Remaining useful life estimation in prognostics using deep convolution neural networks,” *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[15] C. Zhang, X. Yao, J. Zhang, and H. Jin, “Tool condition monitoring and remaining useful life prognostic based on a wireless sensor in dry milling operations,” *Sensors*, vol. 16, no. 6, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/6/795

[16] S. Ji, X. Han, Y. Hou, Y. Song, and Q. Du, “Remaining useful life prediction of airplane engine based on pca–blstm,” *Sensors*, vol. 20, no. 16, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4537

[17] Z. Shi and A. Chehade, “A dual-lstm framework combining change point detection and remaining useful life prediction,” *Reliability Engineering & System Safety*, vol. 205, p. 107257, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832020307572

[18] M. Calabrese, M. Cimmino, F. Fiume, M. Manfrin, L. Romeo, S. Ceccacci, M. Paolanti, G. Toscano, G. Ciandrini, A. Carrotta, M. Mengoni, E. Frontoni, and D. Kapetis, “Sophia: An event-based iot and machine learning architecture for predictive maintenance in industry 4.0,” *Information*, vol. 11, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/4/202

[19] J. Yan, Y. Meng, L. Lu, and L. Li, “Industrial big data in an industry 4.0 environment: Challenges, schemes, and applications for predictive maintenance,” *IEEE Access*, vol. 5, pp. 23 484–23 491, 2017.

[20] T. Zonta, C. A. da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, “Predictive maintenance in the industry 4.0: A systematic literature

review," *Computers Industrial Engineering*, vol. 150, p. 106889, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835220305787

[21] A. Prajapati, J. Bechtel, and S. Ganesan, "Condition based maintenance: a survey," *Journal of Quality in Maintenance Engineering*, 2012.

[22] D. F. Hesser and B. Markert, "Tool wear monitoring of a retrofitted cnc milling machine using artificial neural networks," *Manufacturing Letters*, vol. 19, pp. 1–4, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2213846318301524

[23] G. Scalabrini Sampaio, A. R. d. A. Vallim Filho, L. Santos da Silva, and L. Augusto da Silva, "Prediction of motor failure time using an artificial neural network," *Sensors*, vol. 19, no. 19, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/19/4342

[24] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, no. 2, pp. 197–227, 2016.

[25] D. Wu, C. Jennings, J. Terpenny, R. X. Gao, and S. Kumara, "A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests," *Journal of Manufacturing Science and Engineering*, vol. 139, no. 7, 04 2017, 071018. [Online]. Available: https://doi.org/10.1115/1.4036350

[26] V. Mathew, T. Toby, V. Singh, B. M. Rao, and M. G. Kumar, "Prediction of remaining useful lifetime (rul) of turbofan engine using machine learning," in *2017 IEEE International Conference on Circuits and Systems (ICCS)*, 2017, pp. 306–311.

[27] O. Janssens, M. Loccufier, and S. Van Hoecke, "Thermal imaging and vibration-based multisensor fault detection for rotating machinery," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 1, pp. 434–444, 2019.

[28] A. Binding, N. Dykeman, and S. Pang, "Machine learning predictive maintenance on data in the wild," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 507–512.

[29] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[30] P. R. d. O. da Costa, A. Akcay, Y. Zhang, and U. Kaymak, "Attention and long short-term memory network for remaining useful lifetime predictions of turbofan engine degradation," *International Journal of Prognostics and Health Management*, vol. 10, no. 4, 2019.

[31] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*, 2008, pp. 1–9.

[32] J. Wu, K. Hu, Y. Cheng, H. Zhu, X. Shao, and Y. Wang, “Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network.” *ISA Trans*, vol. 97, pp. 241–250, Feb 2020.

[33] Y. Zhang, R. Xiong, H. He, and M. G. Pecht, “Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 5695–5705, 2018.

[34] G. Liu and J. Guo, “Bidirectional lstm with attention mechanism and convolutional layer for text classification,” *Neurocomputing*, vol. 337, pp. 325–338, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231219301067

[35] C. Zhao, X. Huang, Y. Li, and M. Yousaf Iqbal, “A double-channel hybrid deep neural network based on cnn and bilstm for remaining useful life prediction,” *Sensors*, vol. 20, no. 24, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/24/7109

[36] M. Wang, J. Cheng, and H. Zhai, “Life prediction for machinery components based on cnn-bilstm network and attention model,” in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 2020, pp. 851–855.

[37] J. Wang, G. Wen, S. Yang, and Y. Liu, “Remaining useful life estimation in prognostics using deep bidirectional lstm neural network,” in *2018 Prognostics and System Health Management Conference (PHM-Chongqing)*, 2018, pp. 1037–1042.

[38] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, “Long short-term memory network for remaining useful life estimation,” in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2017, pp. 88–95.

[39] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” *Advances in neural information processing systems*, vol. 25, 2012.

[40] T. Xia, Y. Song, Y. Zheng, E. Pan, and L. Xi, “An ensemble framework based on convolutional bi-directional lstm with multiple time windows for remaining useful life estimation,” *Computers in Industry*, vol. 115, p. 103182, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361519303987

[41] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[42] Z. Xie, S. Du, J. Lv, Y. Deng, and S. Jia, "A hybrid prognostics deep learning model for remaining useful life prediction," *Electronics*, vol. 10, no. 1, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/1/39

[43] A. Listou Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliability Engineering System Safety*, vol. 183, pp. 240–251, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832018307506

[44] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *International conference on database systems for advanced applications*. Springer, 2016, pp. 214–228.

[45] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2306–2318, 2016.

[46] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering System Safety*, vol. 172, pp. 1–11, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832017307779

[47] J. Li, X. Li, and D. He, "A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction," *IEEE Access*, vol. 7, pp. 75 464–75 475, 2019.

A

Figures

In this appendix figures with the Pearson correlation matrix analysis and results from the training and testing of the proposed BiLSTM model can be seen.

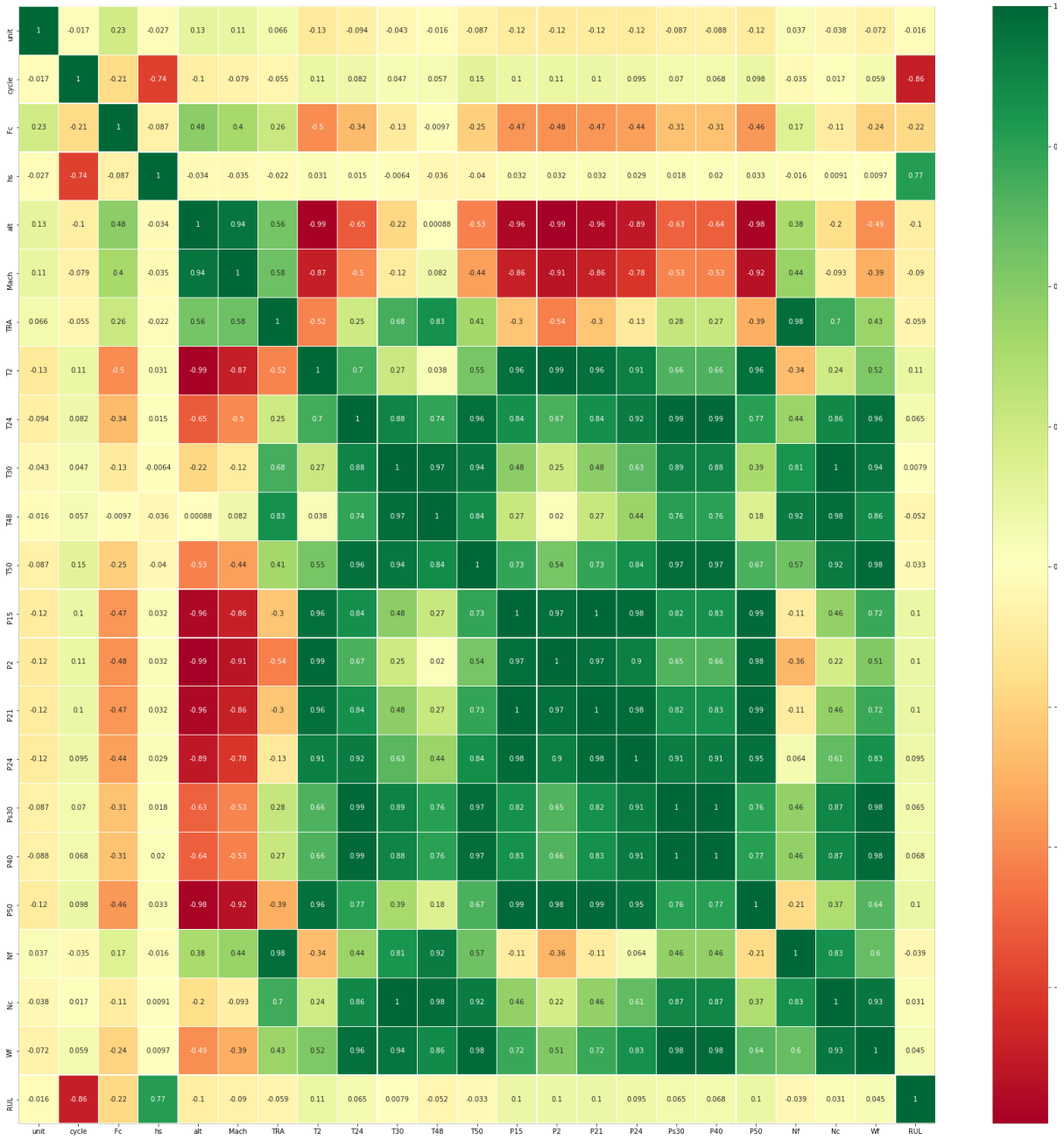


Figure A.1: Correlation Matrix using the Person method of dataset DS08a - auxiliary data, operation conditions, sensor measurements and RUL

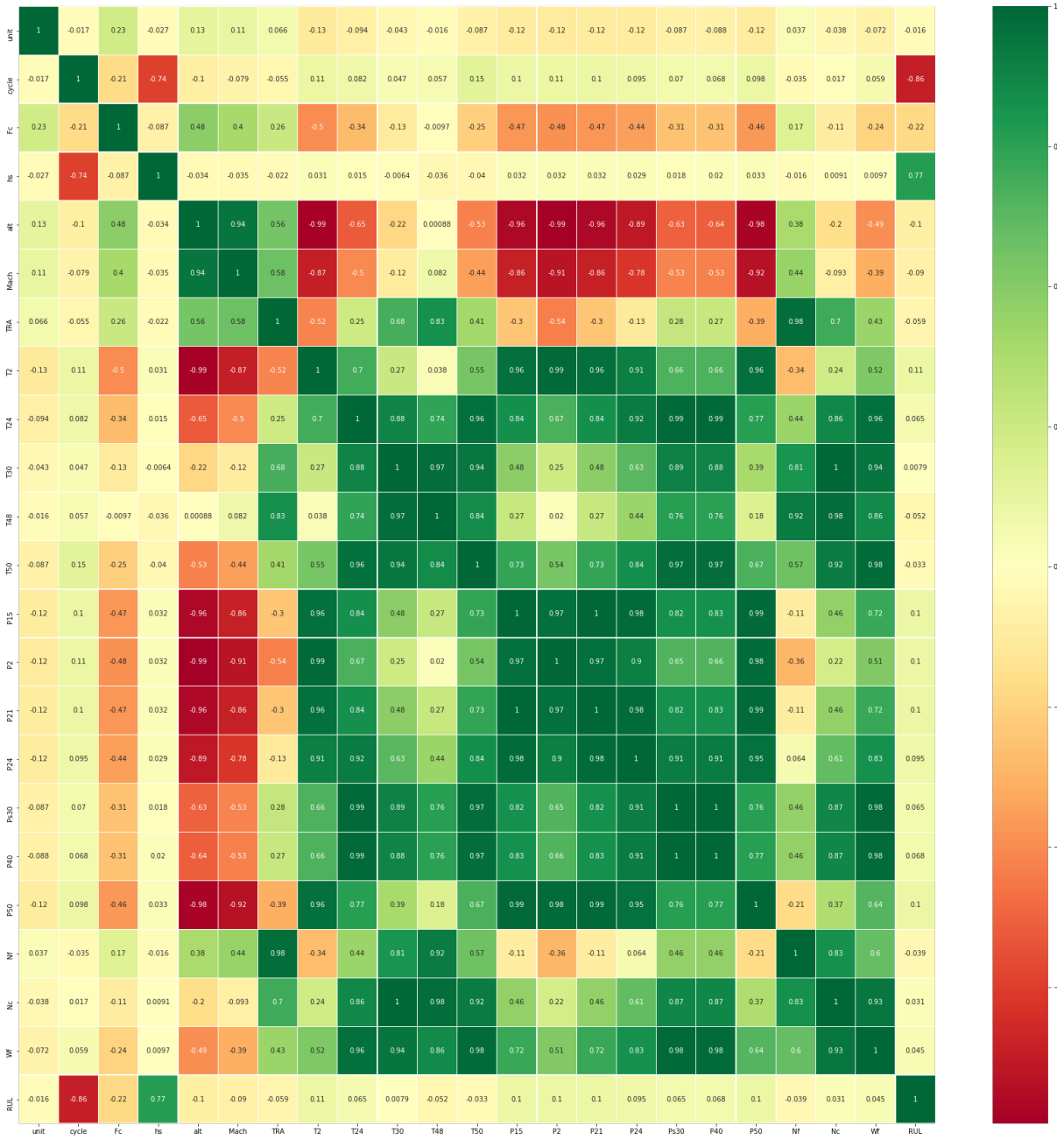


Figure A.2: Correlation Matrix using the Person method of dataset DS08a - auxiliary data, operation conditions, virtual sensor measurements and RUL

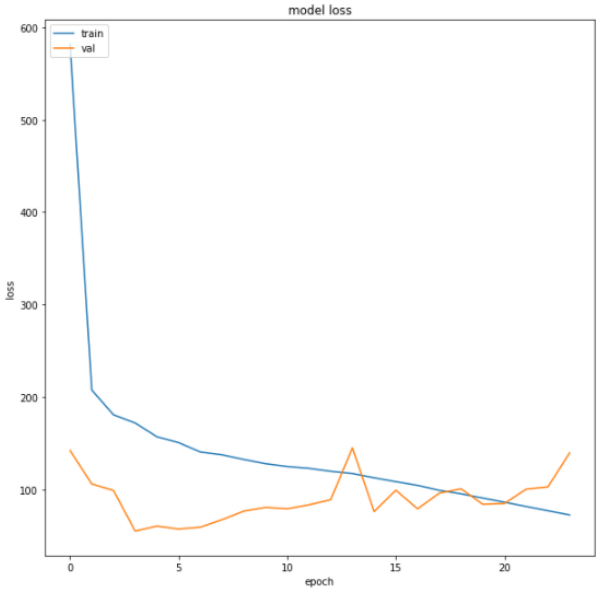
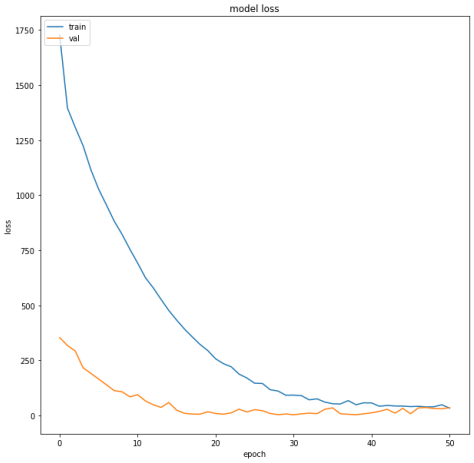
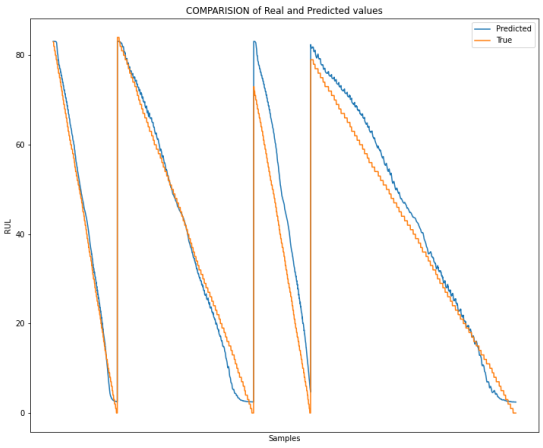


Figure A.3: Loss function of the training of the proposed BiLSTM model for the full N-CMAPSS dataset

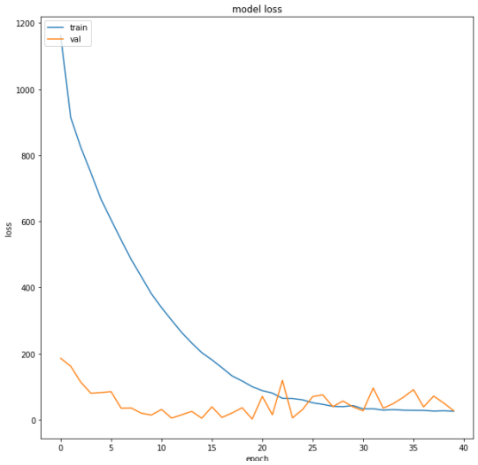


(a) Loss function of the training of the proposed BiLSTM model for DS01

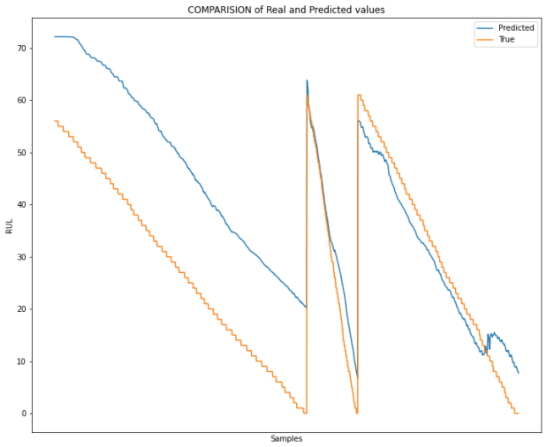


(b) RUL prediction of the proposed BiLSTM model for DS01

Figure A.4: Results of final BiLSTM model applied to DS01

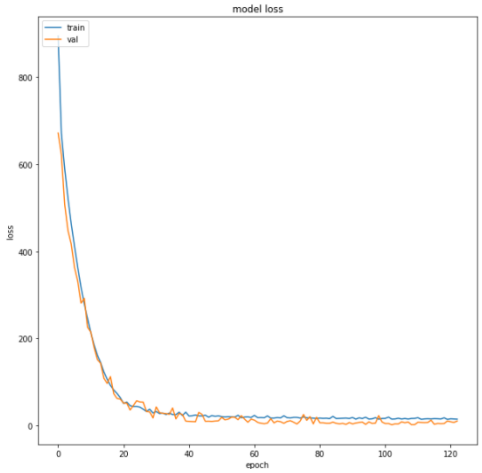


(a) Loss function of the training of the proposed BiLSTM model for DS02

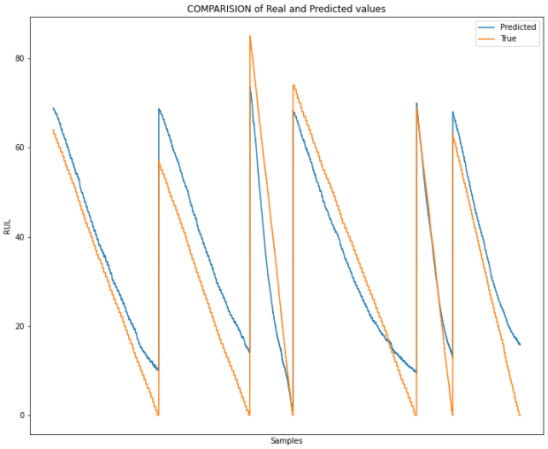


(b) RUL prediction of the proposed BiLSTM model for DS02

Figure A.5: Results of final BiLSTM model applied to DS02

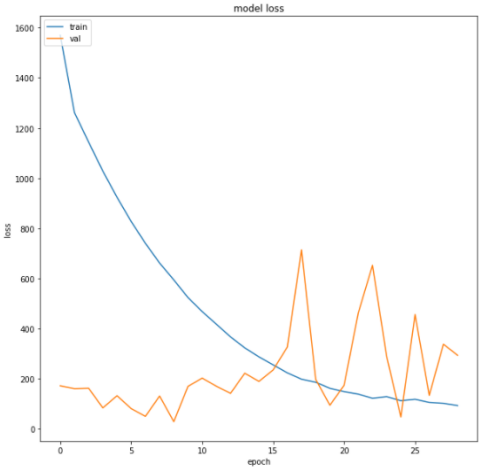


(a) Loss function of the training of the proposed BiLSTM model for DS03

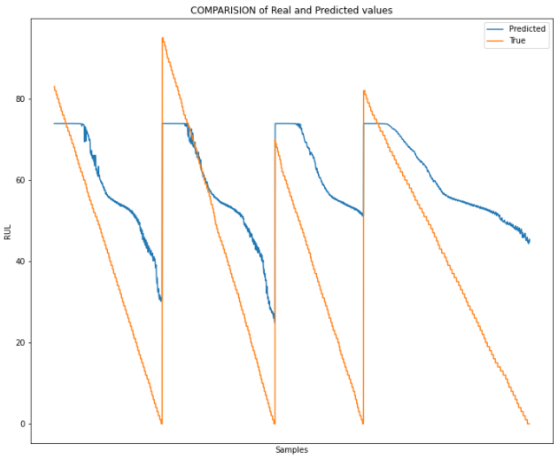


(b) RUL prediction of the proposed BiLSTM model for DS03

Figure A.6: Results of final BiLSTM model applied to DS03

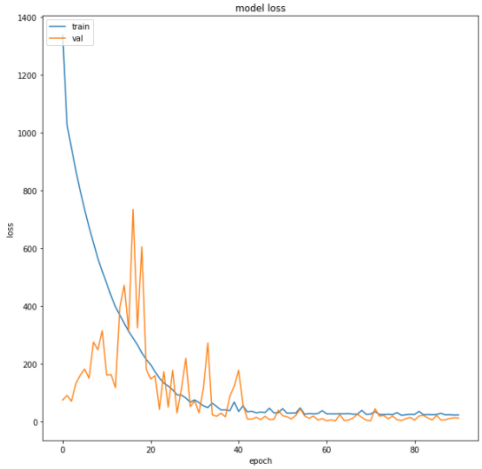


(a) Loss function of the training of the proposed BiLSTM model for DS04

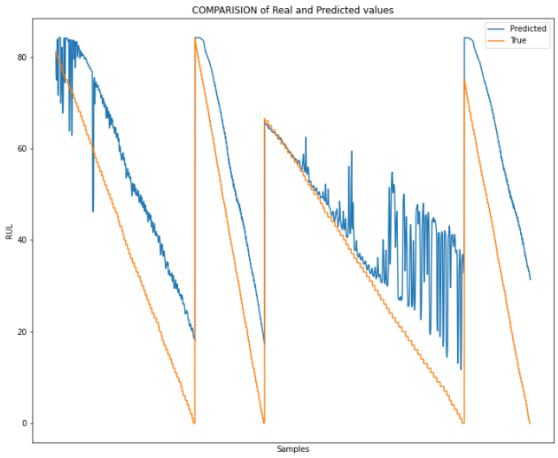


(b) RUL prediction of the proposed BiLSTM model for DS04

Figure A.7: Results of final BiLSTM model applied to DS04

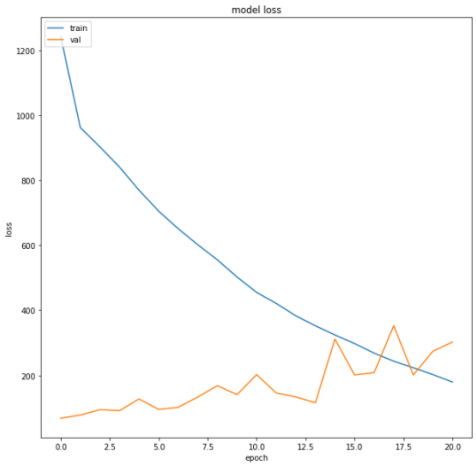


(a) Loss function of the training of the proposed BiLSTM model for DS05

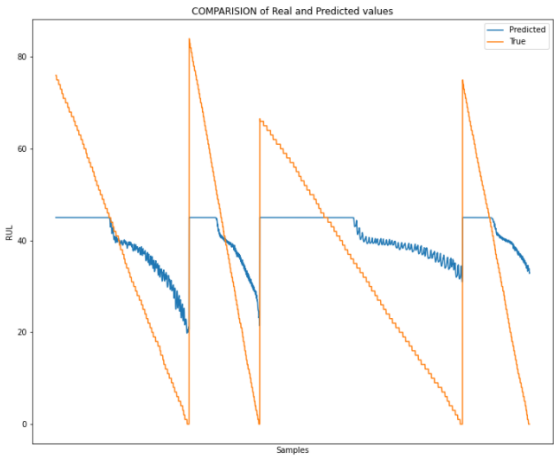


(b) RUL prediction of the proposed BiLSTM model for DS05

Figure A.8: Results of final BiLSTM model applied to DS05

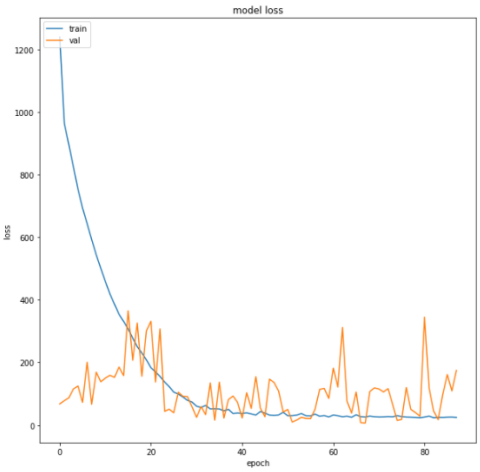


(a) Loss function of the training of the proposed BiLSTM model for DS06

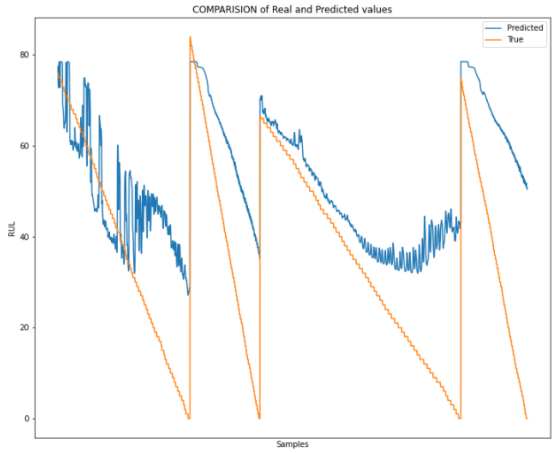


(b) RUL prediction of the proposed BiLSTM model for DS06

Figure A.9: Results of final BiLSTM model applied to DS06

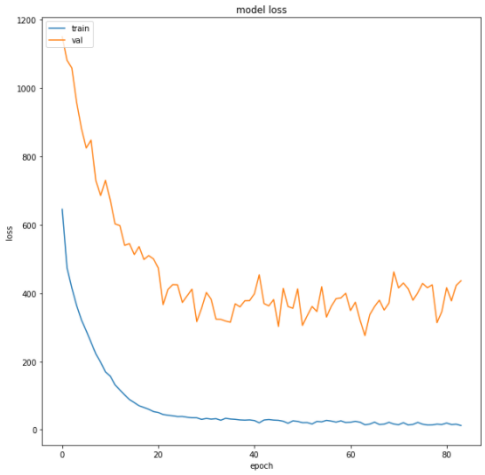


(a) Loss function of the training of the proposed BiLSTM model for DS07

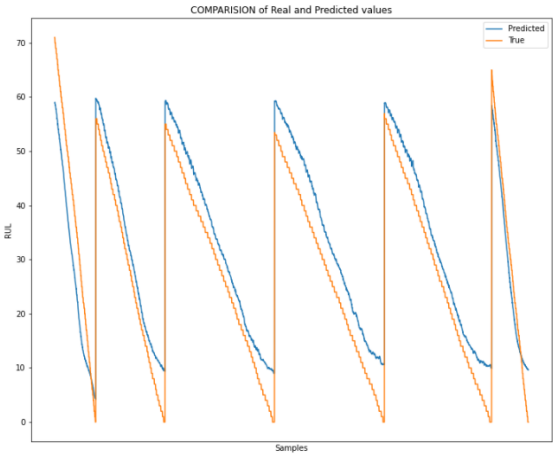


(b) RUL prediction of the proposed BiLSTM model for DS07

Figure A.10: Results of final BiLSTM model applied to DS07

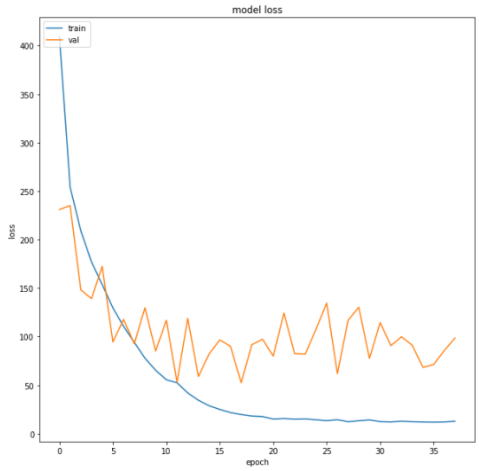


(a) Loss function of the training of the proposed BiLSTM model for DS08a

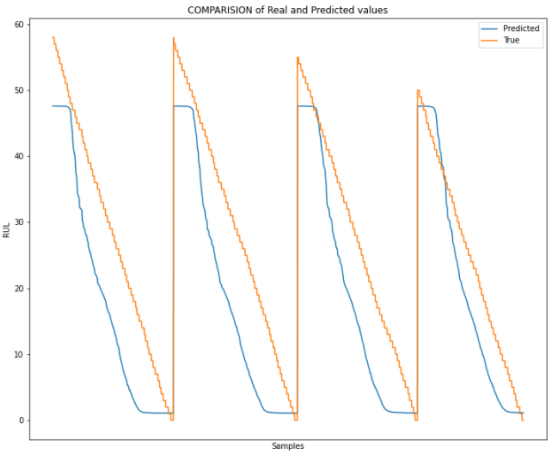


(b) RUL prediction of the proposed BiLSTM model for DS08a

Figure A.11: Results of final BiLSTM model applied to DS08a



(a) Loss function of the training of the proposed BiLSTM model for DS08c



(b) RUL prediction of the proposed BiLSTM model for DS08c

Figure A.12: Results of final BiLSTM model applied to DS08c